Smoothed Particle Hydrodynamics in Complex Shapes

Takahiro Harada* Seiichi Koshizuka† Yoichiro Kawaguchi‡

The University of Tokyo

Abstract

In this paper, we propose an improved computation model of wall boundary in Smoothed Particle Hydrodynamics, a particle method for fluid simulation. Generally, particle methods calculate a wall boundary by converting it to wall particles. The proposed method uses a distance function calculated from a polygon model as a wall boundary. As a result, fluid motion in complex shapes can be simulated easily. Since the method does not use wall particles, it is able to represent a wall boundary without increasing the particle resolution. When a boundary is represented by wall particles, we have to generate a large number of wall particles. The proportion of the number of wall particles in total number of particles is high. However the proposed method does not need wall particles, it can reduce the total number of particles. After the simulation, surface mesh is usually constructed to visualize a simulation result from particles. However, it is difficult to generate smooth surface from them. We also propose a visualization method which can construct smooth fluid surfaces contacting with a wall boundary.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: Physically Based Animation, Fluid Simulation, Particle Method, Smoothed Particle Hydrodynamics

1 Introduction

We are living with many kinds of liquids whose behaviors are complex. When making an animation of liquids in computer graphics, the motion of liquid is calculated by solving the governing equations. There are two main approaches to solve a fluid motion: Eulerian and Lagrangian methods. Eulerian method stores physical values on a grid. In contrast, Lagrangian method does not use a grid but uses particles which are not connected each other and physical values are held on the particles. A method which uses particles is called a particle method. Since particles themselves represent a bunch of fluid, it can be applied to free surface flow easily. Moreover, as the convection term is calculated by convecting particles themselves, there is no numerical dissipation caused by the convection term. Moving Particle Semi-implicit (MPS) method [Koshizuka and Oka 1996] and Smoothed Particle Hydrodynamics (SPH) [Monaghan 1992] are particle methods which are used for fluid simulation. The MPS method which can solve incompressible flow is mainly applied to engineering area[Shibata et al. 2004][Harada et al. 2006]. SPH, which computes compressible flow, is often used in computer graphics because the computation cost of SPH is lower than that of the MPS method[Müller et al. 2003]. Lowering the compressibility, SPH can solve incompressible flow approximately. In particle methods, wall boundaries as well as fluid are represented by particles. To prevent penetration of fluid particles to the wall boundary, several layers of wall particles have to be prepared. The number of wall particles is proportional to the surface area of the boundary. Thus the shape of the boundary get much complex, the number of wall particles increases. There is also a drawback by using wall particles. A boundary geometry can not be represented correctly. Hence it is not easy to compute fluid motion in complex shapes with wall particles.

In this paper, we present a wall boundary computation method which does not use wall particles but a distance function calculated from a polygon model. Since it does not use wall particles, wall boundary is represented correctly. As a result, a fluid motion in complex shapes can be simulated easily. Not using wall particles, the proposed method can reduce the total number of particles and this leads to acceleration of computation. The computation cost of the boundary calculation is low because wall boundary calculation can be done by a few interpolations.

Although, in particle methods, the surface of liquid is extracted from particles, it is not easy to generate smooth surface contacting to the wall boundary as well as the free surface. We also propose a method which generate smooth surface contacting to the wall boundary. Because most of the values needed when the surface is generated are computed in advance, we can use the technique and improve the surface with low computational cost at run time.

2 Related Works

In computer graphics, most of the Eulerian fluid simulation use level set method for tracking the fluid surface[Foster and Fedkiw 2001]. Mass conservation is not guaranteed in the Eulerian methods because of numerical dissipation accompanying with advection calculation. Particle level set method which combines level set method and lagrangian particles is developed to prevent volume loss of fluids[Enright et al. 2002]. To reduce the numerical dissipation caused by advection computation, the constrained interpolation profile (CIP) method and the back and forth error compensation and correction (BFECC) were used for the computation of advection term[Song et al. 2005] [Kim et al. 2005]. Coupling of fluid and rigid bodies[Carlson et al. 2004], viscoelastic fluid[Goktekin et al. 2004], interaction with thin structure[Guendelman et al. 2005] and coupling of two and three-dimensional calculations[Irving et al. 2006] are studied. Since fluid simulation with a structural grid lowers the accuracy of boundary calculation, simulations using octree grids[Losasso et al. 2004] and tetrahedron meshes[Klinger et al. 2006] are also studied.

There are also many studies which use Lagrangian methods. The MPS method is well studied in the area of computation engineering[Shibata et al. 2004][Harada et al. 2006]. The MPS method realizes incompressibility by solving the Poisson equation on particles.

*takahiroharada@iii.u-tokyo.ac.jp

†koshizuka@t.u-tokyo.ac.jp

‡yoichiro@iii.u-tokyo.ac.jp

Premoze *et al.* introduced the MPS method to the graphics community[Premoze et al. 2003]. Another particle method SPH, which is developed in astronomy, is applied for fluid simulation[Müller et al. 2003]. Coupling with deformable bodies[Müller et al. 2004] and multi-phase flow simulation[Müller et al. 2005] are studied. Particle methods are well-suited for real-time simulation because they conserve mass even in a low resolution simulation[Müller et al. 2003].

In general, particle methods calculate a wall boundary by generating wall particles[Morris et al. 1997]. Müller *et al.* developed a method in which fluid particles interact with lagrangian meshes[Müller et al. 2004]. This method computes particle interaction with meshes by generating temporary particles on the surface of the mesh at every time step and momentum is exchanged between them. Since the density of generated particles varies among polygons which do not have a uniform surface area, it does not guarantee that the constant particle density near the wall boundary.

3 Smoothed Particle Hydrodynamics

3.1 Governing Equations

The governing equations for incompressible fluid are the mass and momentum conservation equations as follows.

$$\frac{D\rho}{Dt} = 0 \tag{1}$$

$$\frac{D\mathbf{U}}{Dt} = -\frac{1}{\rho}\nabla P + \nu\nabla^2\mathbf{U} + \mathbf{g} \tag{2}$$

where $\rho, \mathbf{U}, P, \nu, \mathbf{g}$ are the density, velocity, pressure, dynamic coefficient of viscosity and acceleration of gravity, respectively.

3.2 Discretization

A fluid is represented by a set of particles in SPH[Monaghan 1992]. A physical value $\phi(\mathbf{x})$ at $\mathbf{x}$ is interpolated by a weighted sum of physical values ϕ_j of neighbor particles j.

$$\phi(\mathbf{x}) = \sum_j m_j \frac{\phi_j}{\rho_j} W(\mathbf{x} - \mathbf{r}_j) \tag{3}$$

where $m_j, \rho_j, \mathbf{r}_j$ are the mass, density, position of particle j, respectively. W is a weight function. Density of fluid at position $\mathbf{x}$ is calculated by Equation (3) as follows.

$$\begin{aligned} \rho(\mathbf{x}) &= \sum_j m_j \frac{\rho_j}{\rho_j} W(\mathbf{x} - \mathbf{r}_j) \\ &= \sum_j m_j W(\mathbf{x} - \mathbf{r}_j) \end{aligned} \tag{4}$$

The gas equation leads the pressure of fluid p as

$$p = p_0 + k(\rho - \rho_0) \tag{5}$$

where p_0, ρ_0 are the rest pressure and density.

To compute the momentum conservation equation on particles, the gradient and laplacian operators are represented by differentiating Equation (3). These operators affect the weight functions. However, the pressure and viscosity force modeled by the differentiated weight function are not symmetric, i.e., they do not preserve momentum. Thus symmetric pressure force $\mathbf{F}_i^{press}$ and viscosity force $\mathbf{F}_i^{vis}$ on particle i are introduced as follows.

$$\mathbf{F}_i^{press} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W_{press}(\mathbf{r}_{ij}) \tag{6}$$

$$\mathbf{F}_i^{vis} = \mu \sum_j m_j \frac{\mathbf{v}_j - \mathbf{v}_i}{\rho_j} \nabla W_{vis}(\mathbf{r}_{ij}) \tag{7}$$

where $\mathbf{r}_{ij} = \mathbf{r}_j - \mathbf{r}_i$ and $\mathbf{r}_i, \mathbf{r}_j$ are the position vectors of particles i, j.

In this study, we used the weight functions that are employed by Müller *et al.*[Müller et al. 2003]. The weight functions for the pressure, viscosity and other terms are

$$\nabla W_{press}(\mathbf{r}) = \frac{45}{\pi d^6}(d - |\mathbf{r}|)^3 \frac{\mathbf{r}}{|\mathbf{r}|} \tag{8}$$

$$\nabla W_{vis}(\mathbf{r}) = \frac{45}{\pi d^6}(d - |\mathbf{r}|) \tag{9}$$

$$W(\mathbf{r}) = \frac{315}{64\pi d^9}(d^2 - |\mathbf{r}|^2)^3. \tag{10}$$

As a particle outside of the effective radius d does not have an influence, the value of these weight function outside of the area is zero.

4 Wall Weight Functions

In this section, the detail of the proposed wall boundary calculation model which does not use wall particles is described. When the governing equations are solved in SPH, a wall boundary has to be taken into consideration. Hence we have to develop a new calculation model for density of fluid, the viscosity and pressure terms.

4.1 Density Computation

Density of fluid is calculated as Equation (4) when we use wall particles as a wall boundary. We assume that there is a wall boundary inside of the effective radius of particle i. The computation of fluid density is divided into the contribution of fluid particles and that of wall particles as follows.

$$\begin{aligned} \rho_i(\mathbf{r}_i) &= \sum_j m_j W(\mathbf{r}_{ij}) \\ &= \sum_{j \in fluid} m_j W(\mathbf{r}_{ij}) + \sum_{j \in wall} m_j W(\mathbf{r}_{ij}) \end{aligned} \tag{11}$$

The first term in the right side is the contribution of fluid particles and the second term is that of wall particles. To compute the second term, temporary wall particles are lined up perpendicular to the perpendicular line from particle i to the wall boundary as shown in Figure 1. Distribution of wall particles in the effective radius is determined uniquely when the vector from the wall $\mathbf{r}_{iw}$ is determined. Since the distance $|\mathbf{r}_{ij}|$ is used in the calculation of the weight function W, the contribution of wall particles is calculated as a function of the distance to the wall boundary as follows.

$$\rho_i(\mathbf{r}_i) = \sum_{j \in fluid} m_j W(\mathbf{r}_{ij}) + Z_{wall}^{rho}(|\mathbf{r}_{iw}|) \tag{12}$$

We call $Z_{wall}^{rho}(|\mathbf{r}_{iw}|)$ wall weight function of density.

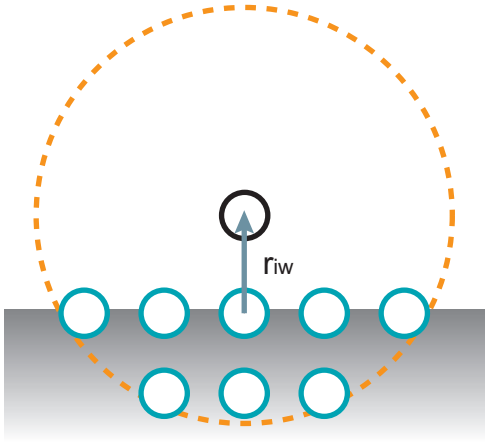

Figure 1: Distribution of wall particles in an effective radius.

4.2 Viscosity Term

The viscosity term is also decomposed into the contribution of fluid particles and that of wall particles. The viscosity force $\mathbf{F}_i^{vis}$ for particle i can be divided into viscosity force from fluid particle $\mathbf{F}_{i,fluid}^{vis}$ and that from wall particle $\mathbf{F}_{i,wall}^{vis}$ as follows.

$$\begin{aligned}\mathbf{F}_i^{vis} &= \mu \sum_j m_j \frac{\mathbf{v}_j - \mathbf{v}_i}{\rho_j} \nabla W_{vis}(\mathbf{r}_{ij}) \\ &= \mathbf{F}_{i,fluid}^{vis} + \mathbf{F}_{i,wall}^{vis}. \end{aligned} \tag{13}$$

Assume that the velocities of all the wall particles are the same. The viscosity force from wall $\mathbf{F}_{i,wall}^{vis}$ is calculated as

$$\mathbf{F}_{i,wall}^{vis} = \mu(\mathbf{v}_j - \mathbf{v}_i) \sum_{j \in wall} m_j \frac{1}{\rho_j} \nabla W_{vis}(\mathbf{r}_{ij}). \tag{14}$$

What we are dealing with is incompressible fluid. Although the mass conservation equation is not solved in SPH, it can compute near-incompressible fluid. This means that the density inside of fluid is approximately uniform. When wall particles are used for boundary condition, the density of wall particles is also approximately uniform. Therefore, in Equation (14), $m_j \frac{1}{\rho_j}$ of all the wall particles is constant by assuming the density of wall particle ρ_j is constant. The distribution of wall particles in the effective radius is also determined uniquely when $\mathbf{r}_{iw}$ is determined. Thus the contribution of wall particles in the viscosity term is calculated as follows by using the distance to the boundary $|\mathbf{r}_{iw}|$ from particle i.

$$\mathbf{F}_{i,wall}^{vis} = \mu(\mathbf{v}_j - \mathbf{v}_i) Z_{wall}^{vis}(|\mathbf{r}_{iw}|) \tag{15}$$

We call $Z_{wall}^{vis}(|\mathbf{r}_{iw}|)$ wall weight function of viscosity.

4.3 Pressure Term

When incompressible fluid is solved by using a particle method, the pressure term works to make the particle number density constant. So the pressure force from wall $\mathbf{F}_{i,wall}^{press}$ to fluid particle i is modeled as a force which makes the distance from the fluid particle to the wall boundary to be the rest distance d. When particle i is located at the position where the distance to the wall boundary is $|\mathbf{r}_{iw}|$ ($|\mathbf{r}_{iw}|$ is smaller than d), the pressure force pushes the particle i distance $d - |\mathbf{r}_{iw}|$ in the direction of $\mathbf{n}(\mathbf{r}_i)$ which is the normal vector of the closest wall boundary point at $\mathbf{r}_i$. Thus the pressure force $\mathbf{F}_{i,wall}^{press}$ from wall is modeled as

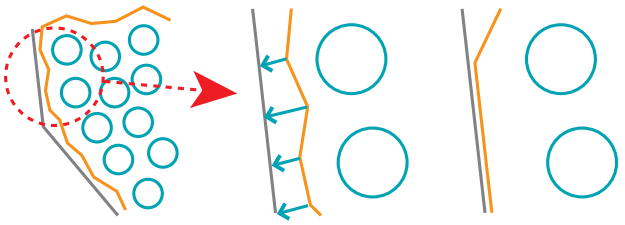

Figure 2: Procedures of surface fitting. Surface polygons are constructed from particles and then the positions of vertices within a certain distance to the wall are corrected.

$$\begin{aligned}\mathbf{F}_{i,wall}^{press} &= m_i \frac{\Delta \mathbf{x}_i}{\Delta t^2} \\ &= m_i \frac{(d - |\mathbf{r}_{iw}|)\mathbf{n}(\mathbf{r}_i)}{\Delta t^2}. \end{aligned} \tag{16}$$

4.4 Computation of Wall Weight Functions

Since the wall weight functions in Equations (15) and (12) depends on the distance to the wall boundary $|\mathbf{r}_{iw}|$, all we have to do is calculation of the distance in order to compute these wall weight functions. The wall weight functions can be computed in advance and the values are referred at the computation of fluid.Particles are placed as shown in Figure 1 and the wall weight functions are calculated by integrating the weight functions numerically because the wall weight functions cannot be computed analytically. The wall weight function is computed at a few points within the effective radius. At the other points, the function is interpolated from the calculated values.

By precomuting the wall weight functions, the contribution of wall is computed easily by referring the value corresponding to the distance. However, to calculate the distance to the boundary, we have to compute the distance to all the polygons comprising the boundary and select the minimum value. When the number of polygons increases, the computation becomes too expensive for real-time applications. Therefore a distance function is introduced to reduce the computational burden[Jones et al. 2006]. As a distance function is the distance to the closest boundary point, the distance from particle i to the boundary $|\mathbf{r}_{iw}|$ is obtained from the distance function by using particle position $\mathbf{r}_i$. When the wall boundary does not deform, the distance function does not change. So the distance function can be also computed beforehand. Consequently, we do not have to compute the distance to the all the polygons but to refer the precomputed distance function at the computation of fluid. When the pressure term is calculated, the normal vector of the boundary point which is the closest to particle i is needed as well as the distance to the boundary. The vector from the closest point on the boundary is the same direction to the normal vector, this vector is called the gradient vector. If the gradient vector is obtained, we can calculate the normal vector. The gradient vector also does not change if the wall boundary does not deform. Hence the gradient vector can be calculated beforehand and referred at fluid computation as well. Also, the distance function is the absolute value of the gradient vector because it is the vector to the closest point on the boundary.

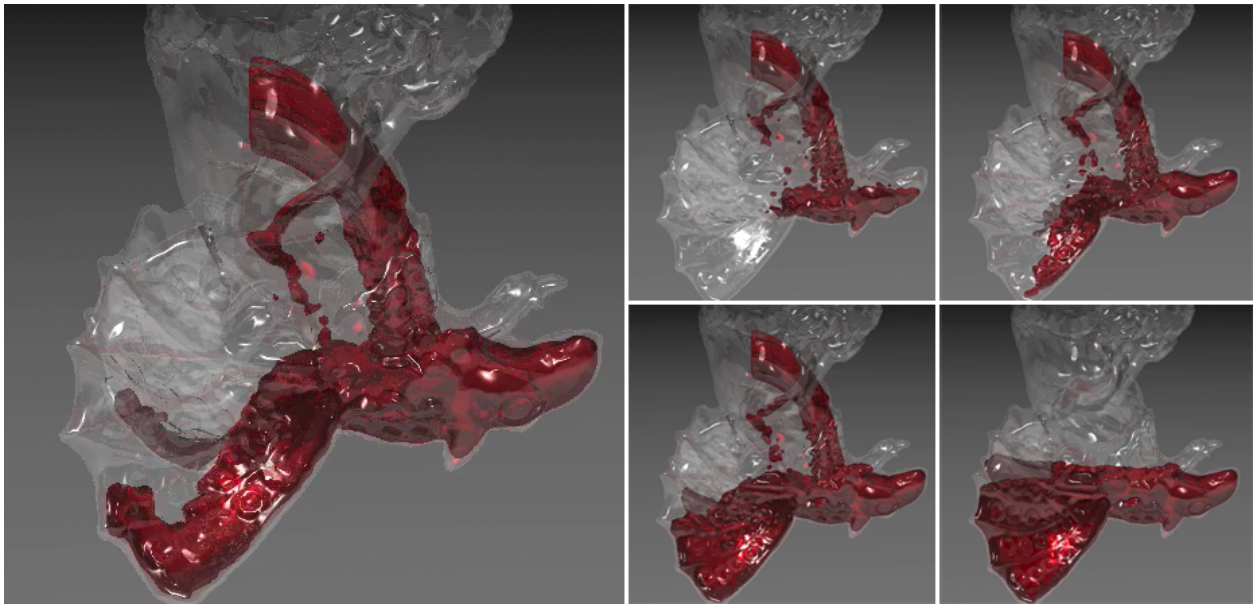

Figure 3: A computation result of free surface flow. A gargoyle polygon model was used as a wall boundary.

4.4.1 Computation of Distance Function

To compute the distance function of a boundary in three-dimension, we have to compute the distance to the faces, edges and vertices of polygons and then select the minimum value[Mauch 2000]. When there are a large number of polygons belonging to a boundary, the computational cost become high. In this study, an approximated distance function is computed by assuming the closest point is always on the faces of a boundary.

The distance function does not need to be computed in all the computational domain but only in a region near a wall boundary because the distance function is used to calculate the contribution of the wall boundary to a particle close to the boundary. Therefore, bounding boxes are generated for each polygon with some margin. A distance function has to be computed in the distance of the effective radius of fluid particles. Therefore, the margin is adjusted by considering the effective radius. The distance to a polygon is calculated in the bounding box. There are some overlapped region of bounding boxes. In the region, the distance to the polygons are compared and the larger value is discarded. In this way, an approximated distance function is computed.

This computation can be accelerated by using graphics processing units (GPUs). At first, a texture corresponding to a three-dimensional grid is prepared as a flat 2D texture [Harris et al. 2003] and then the bounding boxes are rendered. In the pixel shader, the gradient vector and the distance is computed. The gradient vector is output in the color channel and the distance is output in the depth value. The treatment of the overlapped region is done with the depth test.

In all the examples shown in this paper, a three-dimensional grid whose resolution was 128^3 was used. The gradient vector at the position of a particle is calculated by trilinear interpolation of the values of surrounding 8 points and the distance function is calculated as the length of the vector. This procedure can calculate much accurate distance value than direct interpolation of the distance function as shown by [Sud et al. 2006]. The method can compute distance function from a dragon polygon model, which consists of about 200,000 polygons within a second. The size of the precomputed data was about 25MB in this resolution. The distance function calculated by this method has discontinuity at edges because it does not compute the distance from edges. It is not a problem because our fluid simulation does not compute a fluid near edges.

5 Rendering

Particle methods discretize fluid into a set of particles and simulate their motion. If these particles are rendered as spheres, it is difficult to recognize the result as a liquid. To render realistically, the liquid surface has to be extracted from these particles. An implicit function is generated by assigning density distribution for each particles and summing them up. Then an implicit surface is generated. Unfortunately, the extracted surface is not smooth but blobby. This can be said to the surface contacting to wall boundary as well as the free surface. The surface contacting to wall boundary should be fitted to the shape of wall. In this section, a method that makes the liquid surface fit to the wall geometry is described.

5.1 Surface extraction

In the first step, an implicit surface is constructed. a density distribution function f is defined for each particles. Assume a particle i is at position $\mathbf{r}_i$. The density at position $\mathbf{x}$ is calculated as follows.

$$\phi_i(\mathbf{x}) = f(|\mathbf{x} - \mathbf{r}_i|) \tag{17}$$

By integrating all the contribution of particles, the implicit function of fluid is

$$\Phi(\mathbf{x}) = \sum_j \phi_j(\mathbf{x}). \tag{18}$$

Then surface polygons are extracted at $\Phi(\mathbf{x}) = 0$ by using Marching Cubes[Lorensen and Cline 1987].

5.2 Surface Fitting

The extracted surface which is contacting to the wall boundary is not fitted to the wall boundary. To get fitted surface, we assume that a vertex on the extracted surface is on the boundary surface if it is within a certain distance to the wall. Then the position of the vertex is corrected by moving the vertex to the closest point on the wall (see Figure 2).

Whether a vertex is within the distance or not is determined by using the distance function used in simulation. To move the vertex to the closest point, the vector to the point is also needed. Fortunately, the vector is the gradient vector and it is already computed. A vertex at position $\mathbf{x}$ is moved to the corrected position $\mathbf{x}'$ with the distance function $d(\mathbf{x})$ and the gradient vector $\mathbf{n}(\mathbf{x})$ as follows.

$$\mathbf{x}' = \mathbf{x} - d(\mathbf{x})\mathbf{n}(\mathbf{x}) \tag{19}$$

6 Results and Discussions

We implemented the proposed method on a PC with Pentium 4 3.6GHz CPU and 3.0GB RAM. The programs were written in C++. An uniform grid was introduced to reduce the computational cost of neighboring particle search. Several kinds of polygon models were used as a wall boundaries and some results are shown in this paper. The gradient vector was precomputed and interpolated at run time. Since the wall weight functions were also precomputed, the computation cost of the proposed wall boundary calculation method is low. When wall particles are used in the particle method, the particle diameter should be small enough to represent a geometry of wall boundary correctly. As a result, we have to generate a large number of wall particles and it leads to long computation time. However, the proposed method can represent a wall boundary correctly even in a low resolution simulation.

Figure 3 is a result of a simulation of pouring fluid into a gargoyle model (upside down). Surface polygons were extracted and surface fitting was applied to the fluid polygons in rendering. Although the model has thin wings, the fluid flowed into these parts smoothly. In Figures 4 and 5, a liquid was poured into a dragon model. Surface fitting was also applied to these results. To show the effectiveness of the method, the fluid was rendered with a matte shader. We can see that the surface contacting with wall boundary is fitted to the geometry of the wall.

A buddha model was used as a boundary and a fluid was poured in Figure 6. As the model has a fine structure smaller than the diameter of particles, fluid particles cannot enter this part. This is a limitation of particle methods. To put fluid particles into such a fine part, the resolution of particles has to be increased. The computation times with these polygon models as wall boundaries are shown in table 1. One time step is finished within a second.

When the wall particle boundary is used, the number of wall particles which have to be generated increases as the geometry of wall boundary becomes more complicated and the surface area increases. Wall particles are generated from the gargoyle, dragon and buddha models by using surface voxelization method[Dong et al. 2004]. Table 2 shows the number of wall particles, total number of particles and it's ratio. We can see that how large number of wall particles we have to generate. Since the proposed method never need wall particles, up to a half of particles are eliminated.

Although we assume that the curvature of the wall boundary is zero when the wall weight functions are calculated, we can get plausible animation. For more accurate calculation, the curvature of the wall boundary should be taken into consideration. There is another limitation of the method because all the boundaries are assumed to be flat. Therefore, it cannot deal with an edge of a polygon. Adaptation to a curved boundary and an edge is a future work.

Table 1: Total number of fluid particles and computation time for one time step (in milliseconds).

Model	Fluid	Time
Gargoyle	20,000	309.4
Dragon	20,000	281.2
Buddha	20,000	296.8

Table 2: The number of wall particles (1 layer) and its ratio to the total number of particles.

Model	Wall	Total	Ratio
Gargoyle	26,688	46,688	0.572
Dragon	18,582	38,582	0.482
Buddha	12,084	32,084	0.377

The surface fitting technique makes the surface of fluid contacting to the wall boundary to be smooth. Since the free surface is left untouched, the surface is not smooth but blobby. Generation of smooth free surfaces is also an open problem of particle methods.

7 Conclusion and Future Works

In this paper, we presented a wall boundary computation method which uses polygons as a wall boundary. As a wall boundary is represented not by wall particles but wall weight functions and a distance function, the proposed method enables us to simulate fluid in complex shapes. We showed that the proposed method can reduce the total number of particles. We also presented surface fitting method which makes the surface contacting with wall boundary to fit to the geometry.

In this study, a wall boundary is assumed to be fixed. We are going to extend the method to interaction between fluid and rigid and elastic bodies and explore the method which can make smooth free surface from particles in the future.

References

CARLSON, M., MUCHA, P., AND TURK, G. 2004. Rigid fluid: Animating the interplay between rigid bodies and fluid. *ACM Transactions on Graphics 23*, 3, 377–384.

DONG, Z., CHEN, W., BAO, H., ZHANG, H., AND PENG, Q. 2004. Real-time voxelization for complex polygonal models. *Proc. of Pacific Graphics*, 73–78.

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. *ACM Transactions on Graphics 21*, 721–728.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. *Proc. of Siggraph*, 15–22.

GOKTEKIN, T., BARGTEIL, A., AND O'BRIEN, J. 2004. A method for animating viscoelastic fluids. *ACM Transactions on Graphics 23*, 464–467.

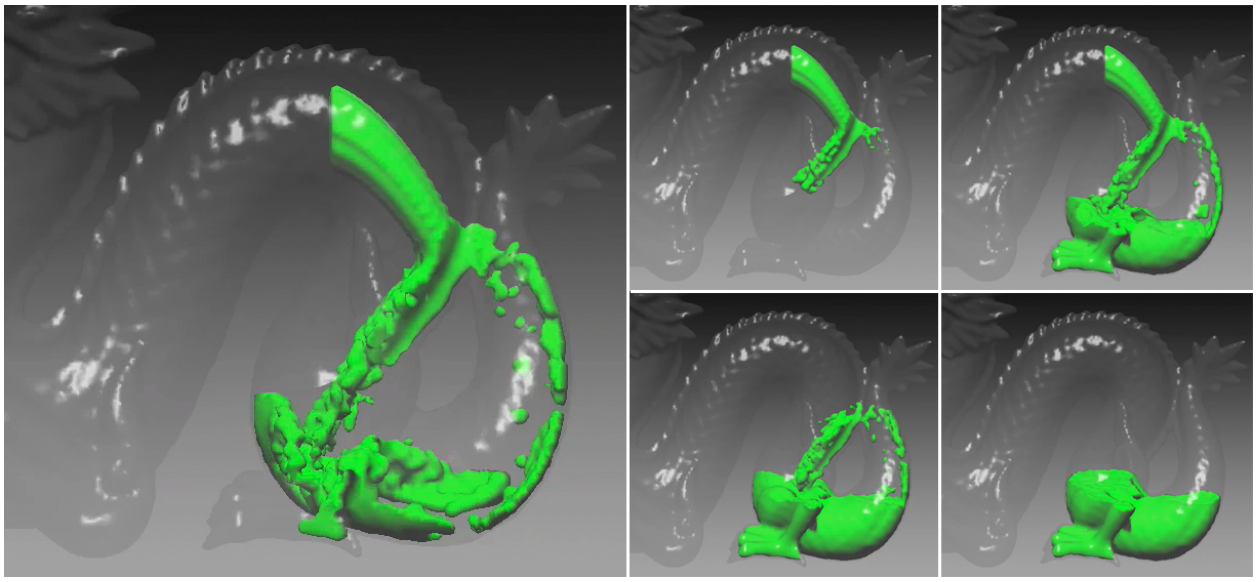

Figure 4: A computation result of free surface flow. A dragon polygon model was used as a wall boundary.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. *ACM Transactions on Graphics 24*, 910–914.

HARADA, T., SUZUKI, Y., KOSHIZUKA, S., ARAKAWA, T., AND SHOJI, S. 2006. Simulation of droplet generation in micro flow using mps method. *JSME International Journal Series B 49*, 3, 731–736.

HARRIS, M., BAXTER, W., SCHEUERMANN, T., AND LASTRA, A. 2003. Simulation of cloud dynamics on graphics hardware. In *Proc. of the SIGGRAPH / Eurographics Workshop on Graphics Hardware*, 92–101.

IRVING, G., GUENDELMAN, E., LOSASSO, F., AND FEDKIW, R. 2006. Efficient simulation of large bodies of water by coupling two and three dimensional techniques. *ACM Transactions on Graphics 25*, 812–819.

JONES, M., BAERENTZEN, J., AND SRAMEK, M. 2006. 3d distance fields: a survey of techniques and applications. *IEEE Transactions on Visualization and Computer Graphics*.

KIM, B., LIU, Y., LLAMAS, I., AND ROSSIGNAC, J. 2005. Flowfixer: Using bfecc for fluid simulation. *In Eurographics Workshop on Natural Phenomena*, 51–56.

KLINGER, B., FELDMAN, B., CHENTANEZ, N., AND O'BRIEN, J. 2006. Fluid animation with dynamic meshes. *ACM Transactions on Graphics 25*, 820–825.

KOSHIZUKA, S., AND OKA, Y. 1996. Moving-particle semi-implicit method for fragmentation of incompressible flow. *Nucl. Sci. Eng. 123*, 421–434.

LORENSEN, W., AND CLINE, H. 1987. Marching cubes: A high resolution 3d surface construction algorithm. *Proc. of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 163–169.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating water and smoke with and octree data structure. *ACM Transactions on Graphics 23*, 457–462.

MAUCH, S. 2000. A fast algorithm for computing the closest point and distance transform. *Technical Report, California Institute of Technology*.

MONAGHAN, J. 1992. Smoothed particle hydrodynamics. *Annu.Rev.Astrophys. 30*, 543–574.

MORRIS, J., FOX, P., AND ZHU, Y. 1997. Modeling low reynolds number incompressible flows using sph. *Journal of Computational Physics 136*, 214–226.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. *Proc. of Siggraph Symposium on Computer Animation*, 154–159.

MÜLLER, M., SCHIRM, S., TESCHNER, M., HEIDELBERGER, B., AND GROSS, M. 2004. Interaction of fluids with deformable solids. *Journal of Computer Animation and Virtual Worlds 15*, 3, 159–171.

MÜLLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-based fluid-fluid interaction. *Proc. of Siggraph Symposium on Computer Animation*, 237–244.

PREMOZE, S., TASDIZEN, T., BIGLER, J., LEFOHN, A., AND WHITAKER, R. 2003. Particle-based simulation of fluids. *Computer Graphics Forum 22*, 3, 401–410.

SHIBATA, K., KOSHIZUKA, S., OKA, Y., AND TANIZAWA, K. 2004. A three-dimensional numerical analysis code for shipping water on deck using a particle method. *Proc. ASME Heat Transfer/Fluid Engineering,HT-FED04-56477*.

SONG, O., SHIN, H., AND KO, H. 2005. Stable but nondissipative water. *ACM Transactions on Graphics 24*, 1, 81–97.

SUD, A., GOVINDARAJU, N., GAYLE, R., AND MANOCHA, D. 2006. Interactive 3d distance field computation using linear factorization. In *Proc. of ACM Symposium on Interactive 3D Graphics and Games*.

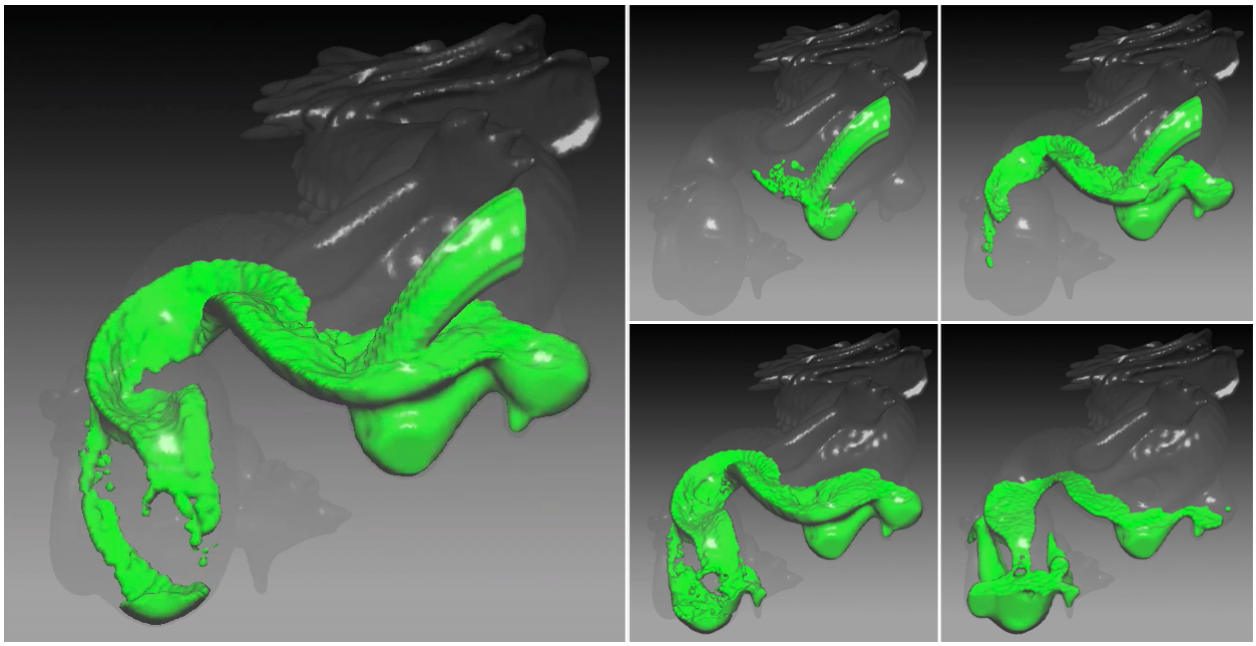

Figure 5: A computation result of free surface flow. A dragon polygon model was used as a wall boundary.

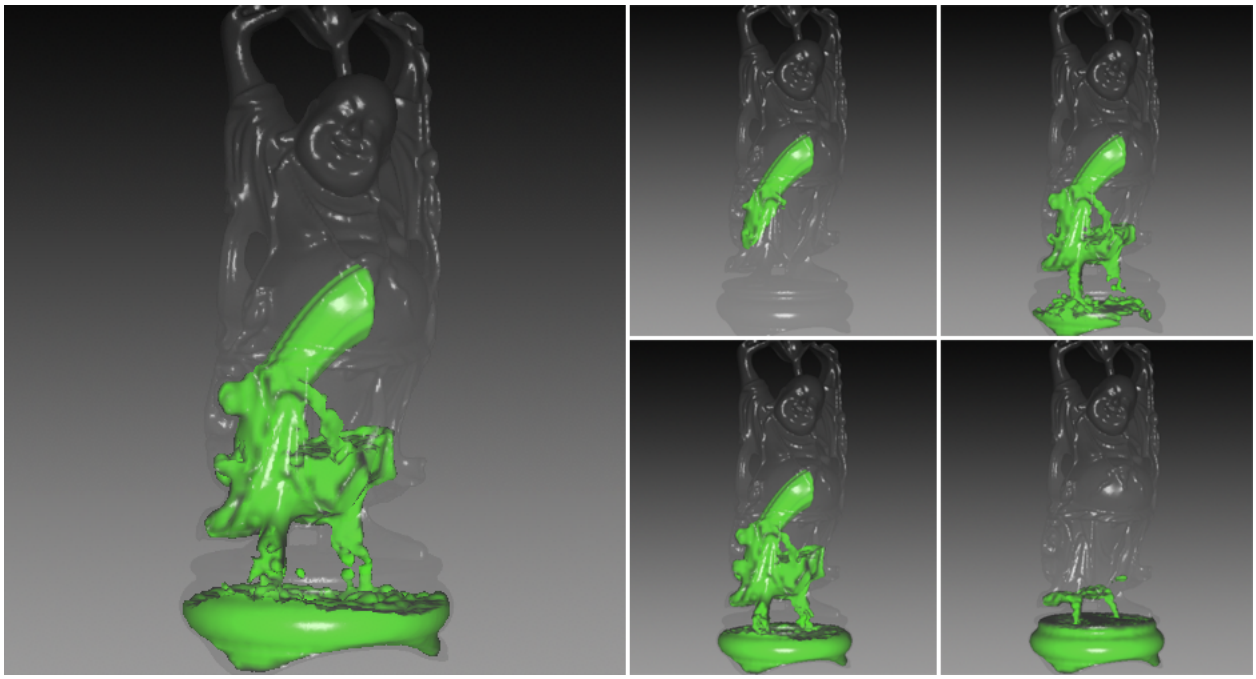

Figure 6: A computation result of free surface flow. A buddha polygon model was used as a wall boundary.