The Journal of the Society for Art and Science Vol. 6 No. 3 pp. 154 - 166

Surface Reconstruction for Particle Simulation Using Level Set Method

Purkhet Abderyim* Tadahiro Fujimoto* Norishige Chiba* Mamtimin Geni**
* *Iwate University*
*{ *purkhet, fujimoto ,nchiba* }*@cis.iwate-u.ac.jp*
** *Xinjiang University*
***mgheni@163.com*

Abstract

A particle-based method is one of the most commonly used methods for fluid simulations. It has certain advantages over other fluid simulation methods; for example, it is easy to simulate irregular and complex fluid motion with topological changes. However, the surface reconstruction for particles remains a challenging problem. In this paper, we propose a surface reconstruction method for moving particles whose motion data are given by a particle-based simulation method. Our method utilizes a level set method to reconstruct a surface because the level set method enables the surface to change its topology free according to the irregular arrangement of particles. Moving particles' data such as positions and velocities given by a gridless particle-based method are effectively used in the level set method executed on a grid space. As a result, a suitable surface is reconstructed so as to follow the moving particles with topological changes. Besides, our method makes isolated splash particles to appropriately contribute to the surface reconstruction. Some experimental results show the useful performance of our method.

Keywords : fluid, surface reconstruction, level set method, particle-based simulation, MPS method

1. Introduction

Simulation of fluid motion in computer graphics has been attempted using a variety of methods. Two main categories of fluid simulation methods exist: the grid-based methods and the particle-based methods. The grid-based method discretizes the simulation space by subdividing it into grid cells, and calculates the fluid flow on each cell. This method readily enforces the incompressibility condition, but does not guarantee mass conservation for small features. This method usually involves calculating a large number of cells that are empty, i.e., they do not contain any fluid [1, 2].

On the other hand, the particle-based method discretizes the fluid mass by using particles as liquid elements, and tracks the particles moving through the space [1]. This method provides a conceptually simple and adaptable simulation framework. However, a particle-based method has the loss of fluid mass as one serious disadvantage. In addition, particularly for computer graphics, one of the major disadvantages of a particle-based method is the difficulty of representing a smooth surface of the fluid [3]. A particle-based method informs us about the position, velocity, and certain other physical properties of each particle. However, this information is not sufficient to determine which particles constitute a surface, where the surface is constructed according to the particles, and how the surface is given a smooth representation.

Several different methods have been used to create a surface from particles. Chiba *et al.* [4] defined a 3D scalar field by assigning a Gaussian density distribution to each particle, and created an implicit equivalued surface on the scalar field by applying the Tetrahedral Cell Subdivision method. The Marching Cube method can also be used for this purpose, but it tends to make a reconstructed surface to have an unrealistic balloon-like shape. Wang *et al.* [5] proposed a method for the case in which particles were given in 3D space by putting 2D simulation slices side by side, each of which contained original particles simulated in 2D space. This method defines a polygonal outline from outermost particles in each slice, and connects the outlines of neighboring slices to generate triangular strips of a 3D surface. However, the outermost particles sometimes fail to represent suitable outlines to define an appropriate 3D surface of the fluid. Mukai *et al.* [6] proposed a smooth rendering method for a fluid surface, which was given as a mesh defined using a Z buffer to which particles were projected.

Premoze *et al.* [3] proposed a method for simulating and representing fluid motion. They used particle-based MPS and

MPS-MAFL methods to solve the Navier-Stokes equations to obtain fluid motion as moving particles, and reconstructed the surface from the particles by using a grid-based level set method [2]. The level set method [7] is an effective tool for tracking a boundary in 2D or 3D space in general and has been used in various fields such as computer graphics, image processing, medical engineering, and so on. One of its distinctive abilities is to enable the topological change of a boundary during its evolution such as split and fusion. Foster *et al.* [8] used the grid-based MAC (Marker-and-Cell) method to simulate fluid motion and mitigated the excessive numerical dissipation of the semi-Lagrangian method [2]. They improved the MAC method by using the grid-based level set approach to reconstruct the fluid surface. The above two surface reconstruction methods [3, 8] used the level set method in completely different approaches. The details are described in Section 2.

In this paper, we propose a method of surface reconstruction for particles using a level set approach. This approach is based on Foster's one and is completely different from Premoze's one. Our method reconstructs a surface representing the boundary between particles and empty regions, and evolves the surface on the velocity field defined by the velocities of particles provided by a particle-based fluid simulation. This simple and direct mechanism results in creating a natural-looking surface flowing on the velocity field. To apply our surface reconstruction method, any particle-based fluid simulation method can be used as long as the positions and velocities of particles are obtained at each simulation time. In the experiments shown in this paper, we used the Moving Particle Semi-implicit (MPS) method for the fluid simulation.

2. Previous Work

2.1 Summary of Level Set Method

Osher and Sethian [7] originally introduced the level set method. It evolves a boundary curve in 2D space, or a boundary surface in 3D space, according to a given force field. In the following, we mainly use the term "interface" to refer to the boundary "curve" or "surface". Unlike most other surface reconstruction methods, the level set method is capable of arbitrarily changing the topology of an interface.

The level set method utilizes a level set function $\phi(\mathbf{x}, t)$ to define an interface, where **x** is the coordinates of a point on the space, and t is time. The function ϕ gives an implicit surface representation; at time t, the interface is given as the set of points **x** that satisfy the following condition:

$$\phi(\mathbf{x}, t) = 0 \tag{1}$$

This representation divides the space into two parts. The sign of ϕ is positive when a point **x** is outside the interface, and the sign is negative when **x** is inside the interface. From Equation (1), the chain rule yields the following level set equation:

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \frac{d\mathbf{x}(t)}{dt} = 0 \tag{2}$$

Here, an outward directed normal **n** to the interface is represented in terms of ϕ as follows:

$$\mathbf{n} = \frac{\nabla \phi}{|\nabla \phi|} \tag{3}$$

The interface consists of the points **x** that have the same level set value ϕ (**x**) = 0. So, the normal **n** to the interface is the direction in which the value ϕ increases and is defined using the gradient of ϕ. Besides, a scalar function F that provides an evolving speed in the outward normal direction is represented as follows:

$$F = \frac{d\mathbf{x}(t)}{dt} \cdot \mathbf{n} \tag{4}$$

Then, Equation (2) is rewritten as follows:

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = 0 \tag{5}$$

This equation is a general form commonly used in many cases. By giving an evolving speed scalar function F arbitrarily, the solution of this equation gives a level set function ϕ that provides an interface by Equation (1).

Usually, Equation (5) is discretized in time t, and the solution of ϕ is obtained at time intervals of Δt. This is achieved as follows. First, an initial interface is given at $t = 0$. Then, the function ϕ is initialized by giving the signed distance d from **x** to the nearest point of the initial interface:

$$\phi(\mathbf{x}, t = 0) = \pm d \tag{6}$$

After the initialization, the function ϕ is updated repeatedly at time intervals of Δt according to a given evolving speed F

using Equation (5). In general implementation, this updating process is achieved in Eulerian grid-based approach. That is, Equation (5) is also discretized on a grid space, and the value ϕ on each grid cell is updated using the values ϕ on neighboring cells.

The function F was originally defined so as to provide an evolving speed in Equation (4). However, when F is used in the above-described updating process, it is considered as a kind of "force field" to evolve an interface. In this case, it can be given an arbitrary scalar, that is, not only an evolving speed but also other scalars to define a desired force field that evolves the interface. For example, when the curvature of an interface is given as F, the interface is evolved so as to smooth out its bumpy shape [9, 10]; in this case, the interface changes its shape by itself. Moreover, the level set method works as a general tool to track a boundary, and can be applied to various applications. For example, some "segmentation" or "boundary detection" techniques in 2D image or 3D volume processing applications utilize the level set approach. In this case, the force field F is given appropriate values so as to stop the evolution of a 2D/3D interface near boundary parts on an image/volume; each pixel/voxel of the image/volume is given a value of F using the gradient of pixel/voxel intensities such that the value is zero on boundary parts and increases with a plus sign, or decreases with a minus sign, toward farther parts apart from the boundary parts [11, 12]. Such an application does not have any velocity information originally, and the force field F is defined for convenience.

2.2 Previous Surface Reconstruction Methods Using Level Set Method

The method proposed by Foster *et al.* [8] is an early work for surface reconstruction using a level set method. This method uses an Eulerian grid-based fluid simulation and calculates a fluid velocity on each grid cell of a grid space. Then, using directly the velocities on the grid cells as a force field F, the level set function ϕ is updated to evolve its interface. At time intervals of Δt, after each calculation of the grid cell velocities, the update calculation of the functionϕ and the interface is executed once. This means that the update of the velocities by the fluid simulation and the update of the interface by the level set method are synchronized at each time step. This results in evolving the interface on the flow caused by the velocity field F.

On the other hand, the surface reconstruction using a level set method by Premoze *et al.* [3] was applied to a particle-based fluid simulation. Their usage of the level set method is completely different from Foster's one; Premoze's level set method is quite similar to the boundary detection techniques mentioned in Section 2.1. Premoze's method creates a level set interface according to the distribution of particles, not the velocities of particles, on each animation frame. The particle distribution is specified as the number of particles existing at each position on the space. Actually, the method uses kernel functions so as to smooth the number over the space, which results in using the following force field:

$$F(\mathbf{x}) = T + \sum_i f_i(\mathbf{x}) \qquad (7)$$

The scalar T is an offset parameter. The kernel function f_i for a particle i is defined as follows:

$$f_i(\mathbf{x}) = \frac{1}{1 + \left| d_i(\mathbf{x}) / r_i \right|^2} \qquad (8)$$

The scalar r_i is the radius of the particle i. The distance function d_i gives the distance from the particle i to a position $\mathbf{x}$. The kernel function f_i gives a kind of "existence value" of the particle i on its surrounding area; the function f_i gives a full existence value at the position of the particle i and reduces the value as the distance d_i increases. In detail, to avoid undesirable gains of existence values in the direction perpendicular to the interface, the distance d_i is defined by separately defining a tangent distance and a perpendicular one using the velocity of the particle i and composing the two distances. In order to use Equation (7) in a similar way to boundary detection techniques, the offset parameter T has to be carefully set such that it becomes zero on the boundaries between outermost particles and empty regions. The values $F(\mathbf{x})$ of Equation (7) are sampled on grid positions $\mathbf{x}$ on a grid space to determine a force field F. Once the force field F is determined for the particle distribution in one animation frame, the level set values $\phi(\mathbf{x})$ on the grid positions $\mathbf{x}$ are updated iteratively until the evolution of the interface stops; the converged interface consists of "detected boundaries" that separate particle regions and empty regions. Before the iterative updating steps for one frame, the level set values $\phi(\mathbf{x})$

are initialized by its previous frame's values. Premoze's method succeeds in creating a suitable interface for given particles. However, the definition of the distance d_i using tangent and perpendicular distances seems to be somewhat ad hoc. Besides, the offset parameter T has to be experimentally chosen so as to fit the interface to the boundaries between outermost particles and empty regions.

2.3 Approach of Our Method

Our method proposed in this paper has the following properties:

(a) The fluid simulation is executed by a Lagrangian particle-based method, which is similar to Premoze's method.

(b) The level set method for surface reconstruction uses a velocity field, which is similar to Foster's method.

Our method constructs a velocity field as the force field F by using the velocities of particles provided by a particle-based fluid simulation. Then, the level set function ϕ is updated and the interface is evolved on the velocity field F in a similar way to Foster's method. After each fluid simulation step, the particle velocities are projected onto grid cells using a Gaussian distribution-like function to yield a smooth velocity field on the grid space. By using the level set method used in Foster's method, our updating process of ϕ spends just one iteration step for one frame. One distinctive technique of our method is how to appropriately treat an isolated particle, in other words, a splash particle. This technique makes an isolated particle to affect the surface reconstruction less than other particles near the surface. An isolated particle can be distinguished using surrounding particles' positions. However, instead, we utilized the "particle number density", which is explained in Section 3, provided by the fluid simulation.

3. Summary of MPS Method

There are several choices of particle-based fluid simulation methods. In this paper, we use the Moving Particle Semi-implicit (MPS) method [13], which solves the Navier-Stokes equation for incompressible fluids, to calculate fluid dynamics.

3.1 Governing Equations of fluid

The motion of incompressible fluid can be described by the following equations.

$$\nabla \bullet u = 0 \tag{9}$$

$$\frac{du}{dt} = -\frac{1}{\rho}\nabla P + \nu\nabla(\nabla u) + f \tag{10}$$

Equation (10) is the Navier-Stokes equation describing the behavior of fluid. Equation (9) states that the mass m is constant. The right side of the Navier-Stokes equation in Equation (10), which is a governing equation of fluid, is made up of a pressure term, a viscosity term and an external force term. Here, t is time, u is velocity, ρ is density, P is pressure, v is dynamic coefficient of viscosity, and f is external force.

3.2 Moving Particle Semi-implicit Method (MPS)

The MPS method used in the proposed method is a Lagrange type simulation method using particles.The MPS method discretizes Equation (10) by using the interaction between particles. The interaction between particles is modeled based on the weight function w below.

$$w(r) = \begin{cases} \dfrac{r_e}{r} - 1 & 0 \le r < r_e \\ 0 & r_e < r \end{cases} \tag{11}$$

Here, r is a distance between two particles, and r_e is a range in which the interaction between particles extends. The gradient model at the position of a particle i is as follows.

$$\langle \nabla \phi \rangle_i = \frac{d}{n^0} \sum_{j \neq i} \frac{\phi_j - \phi_i}{\left| \vec{r_j} - \vec{r_i} \right|^2} \left(\vec{r_j} - \vec{r_i} \right) w\left(\left| \vec{r_j} - \vec{r_i} \right| \right) \tag{12}$$

Here, d is space dimension, and n^0 is a fixed value for particle density. The vector $\vec{r_i}$ is the position of the particle i, and the vector $\vec{r_j}$ is the position of a particle j surrounding the particle i. For a physical quantity ϕ, this equation means

averaging gradient vectors $(\phi_j - \phi_i)(\vec{r_j} - \vec{r_i}) / |\vec{r_j} - \vec{r_i}|^2$ between a particle i and particles j around the particle i using the weight function w. The Laplacian model at the position of a particle i is as follows.

$$\langle \nabla^2 \phi \rangle_i = \frac{2d}{n^0 \lambda} \sum_{j \neq i} (\phi_j - \phi_i) w(|\vec{r_j} - \vec{r_i}|) \qquad (13)$$

This equation means providing physical quantity of particles j around the particle i to the particle i using the weight function w. Here, λ is a coefficient for adjusting the variance of the distribution to analytical solution. The particle number density can be computed as follows:

$$\langle n \rangle_i = \sum_{j \neq i} w|r_j - r_i| \qquad (14)$$

The Poisson equation for pressure is as follows:

$$\langle \nabla^2 \rho^{n+1} \rangle_i = -\frac{\rho}{dt} \frac{\langle n^* \rangle_i - n^0}{n^0} \qquad (15)$$

After the Poisson equation is solved, the correction of velocity **u'** is computed:

$$u' = -\frac{dt}{\rho} \langle \nabla p^{n+1} \rangle \qquad (16)$$

We can then get the velocity and position of next step.
The algorithm of the MPS method can be outlined as follows:

The MPS algorithm
Set the initial value of velocity $\mathbf{u}^0$ and position $\mathbf{r}^0$
for n=1 to Maximal step N
Compute the temporary particle velocities **u*** and positions **r***;
Compute particle number density **n*** using new particle locations **r***;
Set up and solve Poisson pressure equation;
Compute velocity correction **u'** from the pressure equation
Compute new particle positions and velocities:
$\mathbf{u}^{n+1} = \mathbf{u}^* + \mathbf{u}^0$
$\mathbf{r}^{n+1} = \mathbf{r}^* + \mathbf{u}^0 dt$
end for

4. Surface Reconstruction for Particles Using Level Set Method on Velocity Field

In this section, we describe our surface reconstruction method for particles using the level set method and its algorithm. Currently, we have implemented the 2D version and tested the fundamental performance of the method. So, we explain its algorithm in this section and show experimental results for 2D cases in the next section.

Our surface reconstruction method creates a surface covering the outermost particles whose movements are calculated by a fluid simulation. Our method requires the positions and velocities of particles in each animation frame. In addition, our method used particle number densities of the particles, which were given by the MPS method we used in the experiments of this paper. Naturally, other particle-based fluid simulation methods can be used instead; in this case, particle number densities can be calculated from the particles' positions.

In our method, the simulation space is first divided into a grid space. Then, as the initialization for the level set method to evolve an interface to follow moving particles, an initial interface and initial values of the level set function ϕ on the grid cells are determined. To do this, the particles in the initial animation frame obtained from the fluid simulation are projected onto the grid cells. Each particle has a projection radius r_p, and the projection determines which cells intersect with the particle's disk with the radius. If a cell intersects with

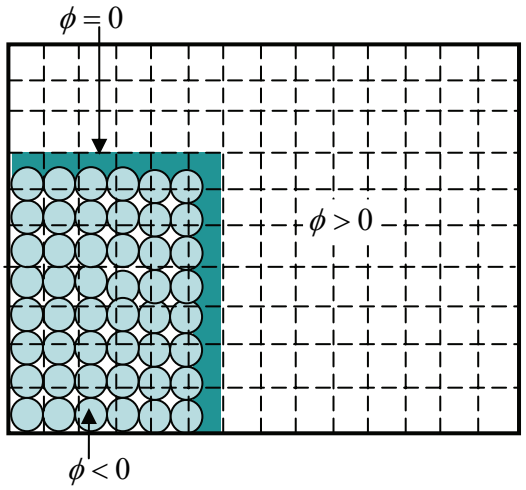


Figure 1: Initialization of level set values ϕ.

more than one particle, the cell is inside the fluid. If a cell intersects with no particle, it is outside the fluid. Then, an image processing technique using a median filter is applied to fill small gaps and determine which cells are inside and which are outside the fluid. Next, an inside cell in contact with outside cells is designated as being a boundary cell. Afterward, every cell is given an initial level set value ϕ with the signed distance from the cell to the nearest boundary cell; an inside cell is given a negative value, an outside cell is given a positive value, and a boundary cell is given 0. This is shown in Figure 1.After the above initialization, the level set method updates the level set values $\phi_{i,j}$ on the grid cells (i, j) and evolves the interface in each time step at intervals of Δt. At each time step, the velocities of the particles from the fluid simulation are projected onto the grid cells. The velocity $F_{i,j}$ on a grid cell (i, j) is determined by the velocities u_k of particles k as follows.

$$F_{i,j} = \sum_{k=1}^{N} \left(\frac{w_k}{\sum_{m=1}^{N} w_m} \right) u_k \qquad \left(\sum_{m=1}^{N} w_m \neq 0 \right) \quad (17)$$

The number N is the number of particles that contribute to $F_{i,j}$. The contribution of u_k to $F_{i,j}$ is determined by a weight w_k according to the distance from the cell position (i, j) to the position of the particle k. The weight w_k is given as follows.

$$w_k = \begin{cases} \left(1 - \left(\frac{d_k}{R}\right)^2\right)^2 (n_k / S) & (0 \le d_k < R,\ n_k < S) \\ \left(1 - \left(\frac{d_k}{R}\right)^2\right)^2 & (0 \le d_k < R,\ S \le n_k) \\ 0 & (R \le d_k) \end{cases} \quad (18)$$

The scalar R represents the influence radius of a particle to a grid cell, as shown in Figure 2.

The scalar d_k is the distance between the position of a particle k and the central position of a grid cell (i, j). The scalar n_k is the particle number density of the particle k, which is obtained from the MPS method using Equation (14). The scalar S is a threshold value for the particle density n_k to judge whether the particle k is treated as an isolated particle, in other words, a splash particle, or not. Equation (18) is a Gaussian

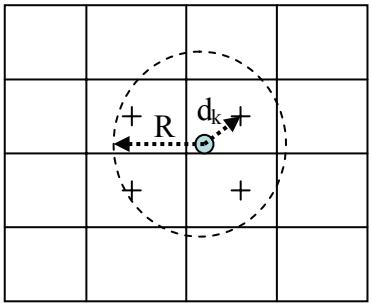


Figure 2: Influence radius of a particle to a grid cell.

distribution-like function; the top has the maximum value of w_k at $d_k = 0$, and w_k decreases smoothly as d_k increases to R. When d_k exceeds R, w_k becomes 0.

Equations (17) and (18) average the particles' velocities into a smoothed velocity field on the grid space. This causes a smooth interface even if neighboring particles' velocities are disturbed in different directions. The threshold value S controls the influence of splash particles. If the density n_k is smaller than S, the particle k is considered to be isolated and treated as a splash particle. In this case, even if the particle k is within the influence radius R from the cell position (i, j), the contribution of u_k to $F_{i,j}$ is reduced linearly by the density n_k. This works to avoid some unnatural motion and shape of the evolving interface, such as a protruding shape caused by following a splash particle. After projecting the particles' velocities on the grid cells, a grid cell on which any particles' velocities have not been projected in empty regions is given a suitable velocity. Such an "empty" grid cell is given the velocity of the nearest grid cell that has the projection of more than one particle's velocity. This results in a smooth velocity field all over the grid space. Then, after determining the velocities $F_{i,j}$ on all grid cells, the level set values $\phi_{i,j}$ are updated by the level set equation of Equation (5), in which the velocities F_{ij} are used for determining F, by using the same scheme as Foster's method described in Section 2.2. This updating process is executed using the Eulerian discretized form of Equation (5) on the grid space and requires only one iteration step for one frame. Finally, the new boundary cells in which $\phi_{i,j} = 0$ provide a new updated interface. In our method, one animation frame is produced by a series of the following processes: the calculation of velocities $F_{i,j}$, the update of level set values $\phi_{i,j}$, and the determination of a

new interface. This is repeated for consecutive frames to yield the evolution of the interface that follows the motion of particles.

The level set method is computationally inefficient if the level set values $\phi_{i,j}$ of all grid cells are computed at every step. To overcome this drawback, we only compute the level set values of grid cells that are near the interface. This is known as the narrow band method [14]. Therefore, in the same way, the velocities $F_{i,j}$ are calculated for only the grid cells near the interface. Figure 3 shows an animation frame obtained using the narrow band method. The small circles represent particles whose motions are computed by the fluid simulation. The red cells are boundary cells and constitute an interface. The green and blue cells are respectively inside and outside cells. The narrow band method computed the velocities $F_{i,j}$ and level set values $\phi_{i,j}$ for only these colored cells.

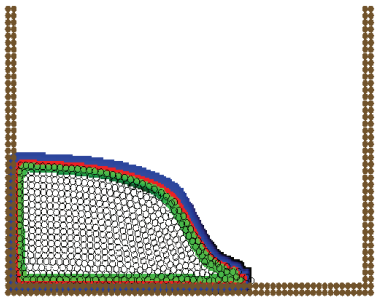


Figure 3: Narrow band method.

The algorithm of our surface reconstruction method is summarized as follows.

Step 1: Divide the simulation space into a grid space.

Step 2: Initialize an interface and level set values $\phi_{i,j}$ on grid cells (i, j) for time step $t = 0$.

Step 3: Input particle data (the positions, velocities, and particle number densities of particles) of time step t from fluid simulation.

Step 4: Determine the necessary cells for the narrow band method.

Step 5: Project the velocities of particles onto the grid space.

Step 6: Calculate velocities $F_{i,j}$ on grid cells.

Step 7: Update level set values $\phi_{i,j}$.

Step 8: Update the interface according to the updated level set values $\phi_{i,j} = 0$.

Step 9: Update time step t to $t+\Delta t$, then go to step 3.

The parameters that we have to adjust in our method are the projection radius r_p, the influence radius R, and the threshold value S .The radius r_p affects an initial interface at time step $t = 0$, and R affects a velocity field to evolve the interface. These two radii have to be carefully given according to the size of a grid cell. As summarized in Section 2.2, Premoze's method [3] has completely the same situation in the definition of the kernel function f_i that uses the radius r_i of a particle i, as shown in Equation (8). If the two radii r_p, R, and the grid cell size are appropriately selected, our method can evolve the interface so as to follow moving particles well. We currently select these parameters experimentally, and gave the same value to the two radii and the grid cell size in the experiments shown in Section 5. An automatic selection of these parameters is an open problem in future. On the other hand, Premoze's method has to determine an offset parameter T to fit the interface to outermost particles in addition to the particles' radii r_i, as shown in Equation (7). The remaining parameter S in our method controls the degree of the influence of an isolated particle to the evolution of the interface. This threshold S has to be adjusted somewhat experimentally for a natural-looking effect. To treat an isolated particle in Premoze's method, the control of the particles' radii r_i and the offset parameter T affect not only the isolated particle's contribution but also other particles' contribution to the shape of the interface, and to balance both of them seems somewhat difficult.

5. Experimental Results

We applied our method to two 2D examples, as shown in Figures 4 and 5. In each figure, some animation frames are selected from the whole frame sequence and arranged in time order.

Figure 4 shows the result of the surface reconstruction for a water drop simulation. The total number of particles is 900, including wall particles. The resolution of a gird space is 140 x 90 .The size of a cell, the projection radius r_p, and the influence radius R are all 0.004. To treat isolated particles using particle number densities, the range r_e in Equation (11) is 0.004, and the threshold value S is 2.0. The narrow-band method used a width of 10 cells inside and outside the

interface. In the figures, the small circles represent particles, which were provided by the fluid simulation. The blue regions represent inside cells, and the red narrow regions represent the interfaces. In (a), both the drop surface in the air and the water surface in the container were initialized. The results (b) to (e) were then obtained by updating the surfaces using our method. In (b), the drop and water surfaces make contact, and we can see that they are about to be fused together. In (c), we can observe that the two surfaces have been merged to make a new water surface. In (c), (d), and (e), there are some splash particles that are isolated from the water body upward on the surface; these particles do not strongly affect the surface evolution.

Figure 5 shows the case of a breaking water flow. In this example, the total number of particles is 805. The parameters are given the same values as those of Figure 4. The figures in the left column show the particles provided by the fluid simulation. In these figures, particles are colored according to their particle number densities; a particle inside the fluid is colored black, and the color gets lighter as a particle gets more isolated. The figures in the right column show the surface reconstruction results by applying our method to the particles in the left column; the two figures in the left and right columns in the same row correspond to each other and have the same distribution of particles at the same simulation time. The figures of (a) show the initial positions of particles and the initial level set surface. The figures from (b) to (h) show that the evolving surface follows the movement of particles provided by the fluid simulation. In the figures from (c) to (h), some particles splash away from the water body but do not strongly affect the surface evolution. From (c) to (g), the evolving tip part of the surface bouncing back from the right side wall is getting in contact with the middle part of the surface, and the two parts are merged to make a small surface inside the fluid as a hole in (g). This means that the topology of the surface in (g) is changed from the initial topology. Such a topological change is a strong advantage caused by the level set method.

The results in Figures 4 and 5 show that on the whole our method reconstructs an appropriate surface that follows moving particles. However, we observe that, in some parts, the surface does not fit outermost particles well, and there are small gaps between the surface and the particles. We consider that this is caused by the reason that our method does not directly fit the surface to the outermost particles of each frame but evolves the surface by using the particle's velocities of each frame. That is, the evolving surface and the moving particles do not have direct relation and are related indirectly through a velocity field. So, after starting with an initial surface, time steps in the level set iterative calculation tend to gradually accumulate the gaps between the surface and outermost particles. Besides, the construction of the velocity field on a grid space by averaging and smoothing the particles' velocities may influence the gaps. The above is an open problem to solve in future. As one solution, the reinitialization of the evolving surface to fit outermost particles may be needed at intervals of several time steps.

In addition, the parts of the reconstructed surface in contact with the wall of the container sometimes move unnaturally; such surface parts sometimes leave and are detached from the wall unnaturally. This behavior is caused by a constructed smooth velocity field. Our method constructs the velocity field by projecting moving particles' velocities onto the grid space and smoothing the projected velocities on the grid cells by weights according to the distances between the particles' positions and the grid cells. Using this approach, even on a grid cell in contact with the wall, the grid cell's velocity perpendicular to the wall does not become completely zero. Then, when the level set method is applied to the non-zero velocity on the grid cell in contact with the wall, as a result, the surface is detached from the wall. It is an open problem in future to obtain a smooth velocity field with zero-velocity perpendicular to the wall.

For these examples, we used a 3.4 GHz Pentium 4 CPU and 1 GByte of memory. The computation time per frame was between 1.3 and 2.0 seconds; the time varied according to the number of the cells within the narrow band width, although the grid resolution was constant.

6. Conclusion and Future works

In this paper, we proposed a surface reconstruction method for moving particles using the level set method. Any particle-based fluid simulation method can be used for our surface

reconstruction method, and we used the MPS method in the experiments in this paper. Our method is based on a simple and natural way in that an interface evolves on the flow of particles' velocities. First, our method divides the simulation space into a grid space, and initializes an interface and a level set function. Then, at each time step, by using the particles' positions, velocities, and particle number densities obtained from the fluid simulation, the level set function is updated, and an evolved interface is obtained. The level set mechanism allows an interface to change its topology free such as split and fusion. Besides, our method controls the contribution of an isolated particle, in other words, a splash particle to the surface reconstruction to prevent the unnatural movement of the interface.

An important open problem to solve in the future is how to prevent the increasing gaps between an evolving interface and outermost particles during the level set iterations, as described in Section 5. Another future work is to extend our 2D method to a 3D surface reconstruction method. This extension has some problems to solve, such as how to create a 2D interface mesh from grid cells in which $\phi_{i,j,k} = 0$ in a 3D grid space. In addition, an efficient calculation method is needed especially for a large-scale fluid animation; to achieve this, GPU implementation is one of promising approaches.

Acknowledgement

This work was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B) 16300021.

REFERENCES

[1] S.Clavet, P.Beaudoin and P.Poulin, Particle-based Viscoelastic Fluid Simulation, Eurographics/ACM SIGGRAPH Symposium on Computer Animation 2005.

[2] Y.Zhu and R.Bridson, Animating Sand as a Fluid, ACM SIGGRAPH 2005.

[3] S.Premoze, T.Tasdizen, J.Bigler, A. Lefohn, and R. Whitaker. Particle-based simulation of fluids, Computer Graphics Forum, Vol.22, I.3, pp.401-411, 2003.

[4] N.Chiba, S.Sanakanishi, K.Yokoyama, I.Ootawara, K.Muraoka and N.Saito, Visual Simulation of Water Currents Using a Particle-based Behavioral Model, The Journal of Visualization and Computer Animation, Vol.6, pp.155-171, 1995.

[5] Q.Wang, J.Bu, C.Chen, T.Fujimoto and N.Chiba, Surface Reconstruction for Animation of Ocean Waves, CAD/Graphics 2005.

[6] N.Mukai, N.Nishimura and M.Kosugi, Real-time Bleeding Simulation Based on Moving Particle Semi-implicit Method, Nicograph2006.

[7] S.Osher and J.Sethian, Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton--Jacobi Formulations, Journal of Computational Physics, 79, pp. 12--49, 1988.

[8] N.Foster and R.Fedkiw, Practical Animation of Liquids, SIGGRAPH 2001.

[9] S.Osher and R.Fed kiw, The Level Set Method and Dynamic Implicit Surfaces, Springer-Verlag, New York, 2002.

[10] Moving Interfaces and Boundaries, from http://math.berkeley.edu/~sethian/

[11] R.Malladi, J,Sethian, and B.Vemuri, Shape Modeling with Front Propagation: A Level Set Approach, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 17, No. 2, February 1995.

[12] S.Osher and N.Paragios, editors. Geometric Level Set Methods in Imaging, Vision and Graphics, Springer,New York, 2003.

[13] S.Koshizuka, H.Tamako, and Y.Oka, A Particle Method for Incompressible Viscous Flow with Fluid Fragmentation, Comput. Fluid Dynamics, J.4, pp.29-46, 1995.

[14] S.Yui, K.Hara, H.Zha and T.Hasegawa: A fast narrow band method and its application in topology -adaptive 3-D modeling, Proc. ICPR02, pp.122-125, Aug, 2002

[15] P.Abderyim, T.Fujimoto, N.Chiba and M.Gheni, Surface Reconstruction for Particle Simulation Using Level Set Method, Nicograph 2006.

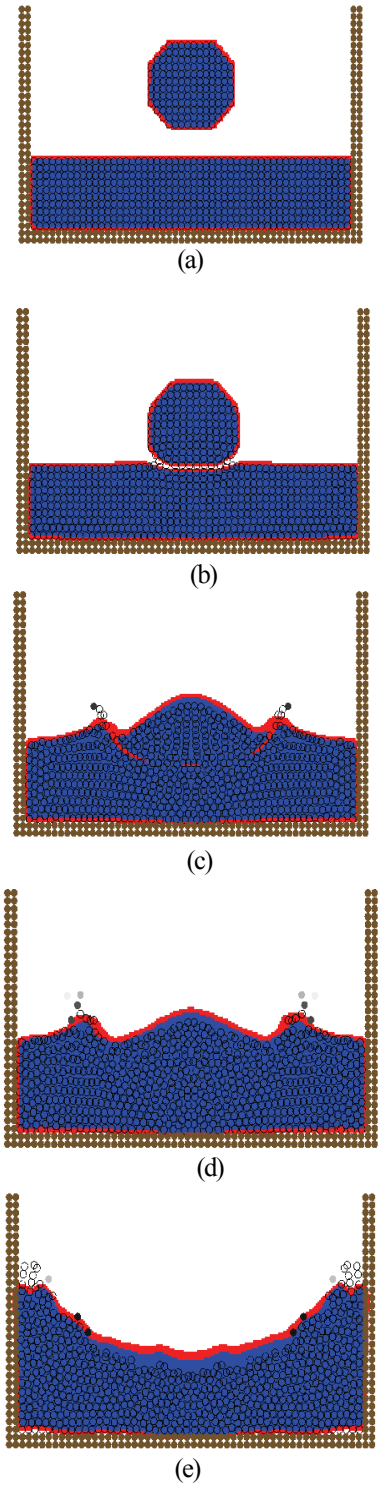
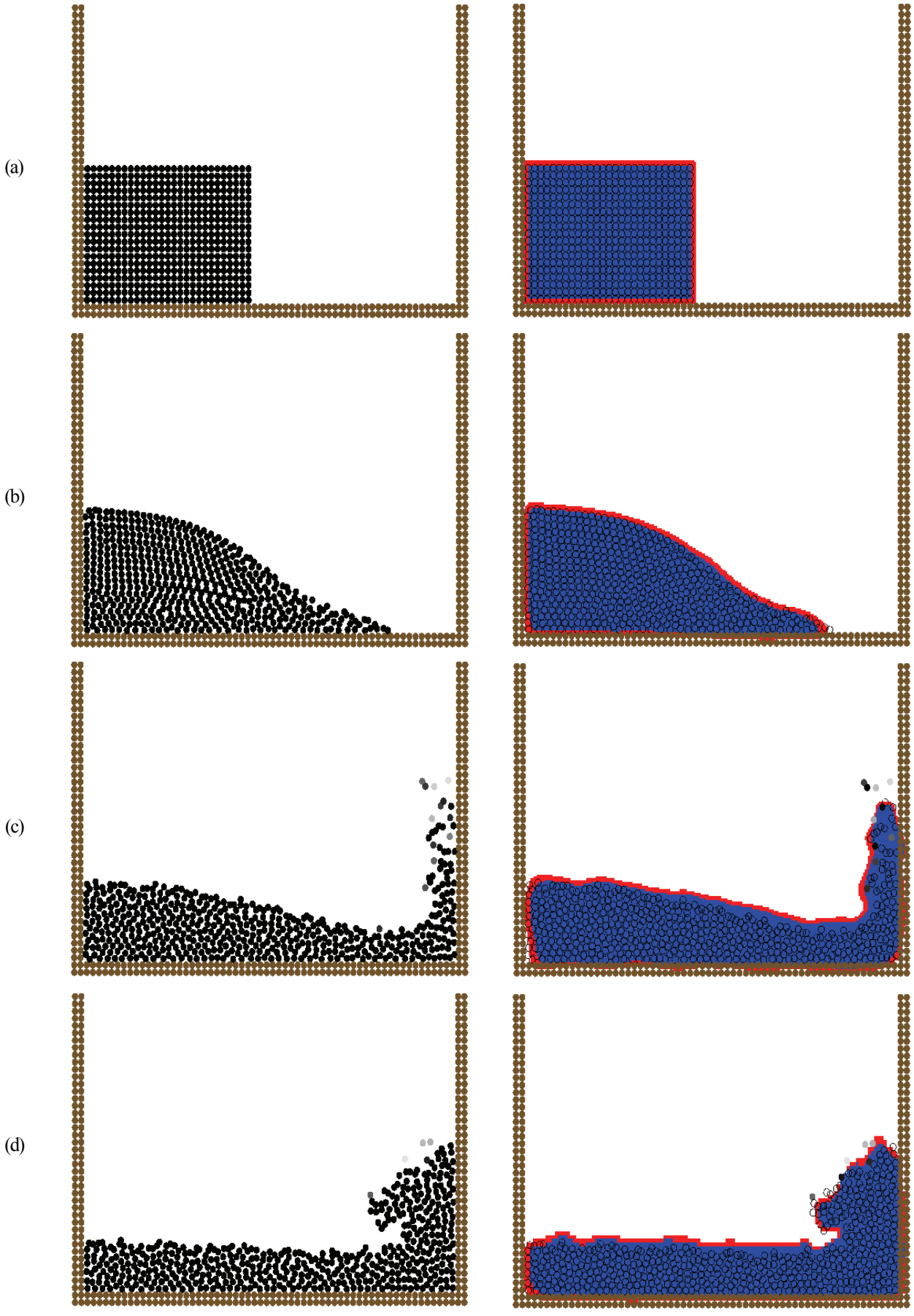


Figure 4 : Surface reconstruction for a falling water drop.


(a)
(b)
(c)
(d)

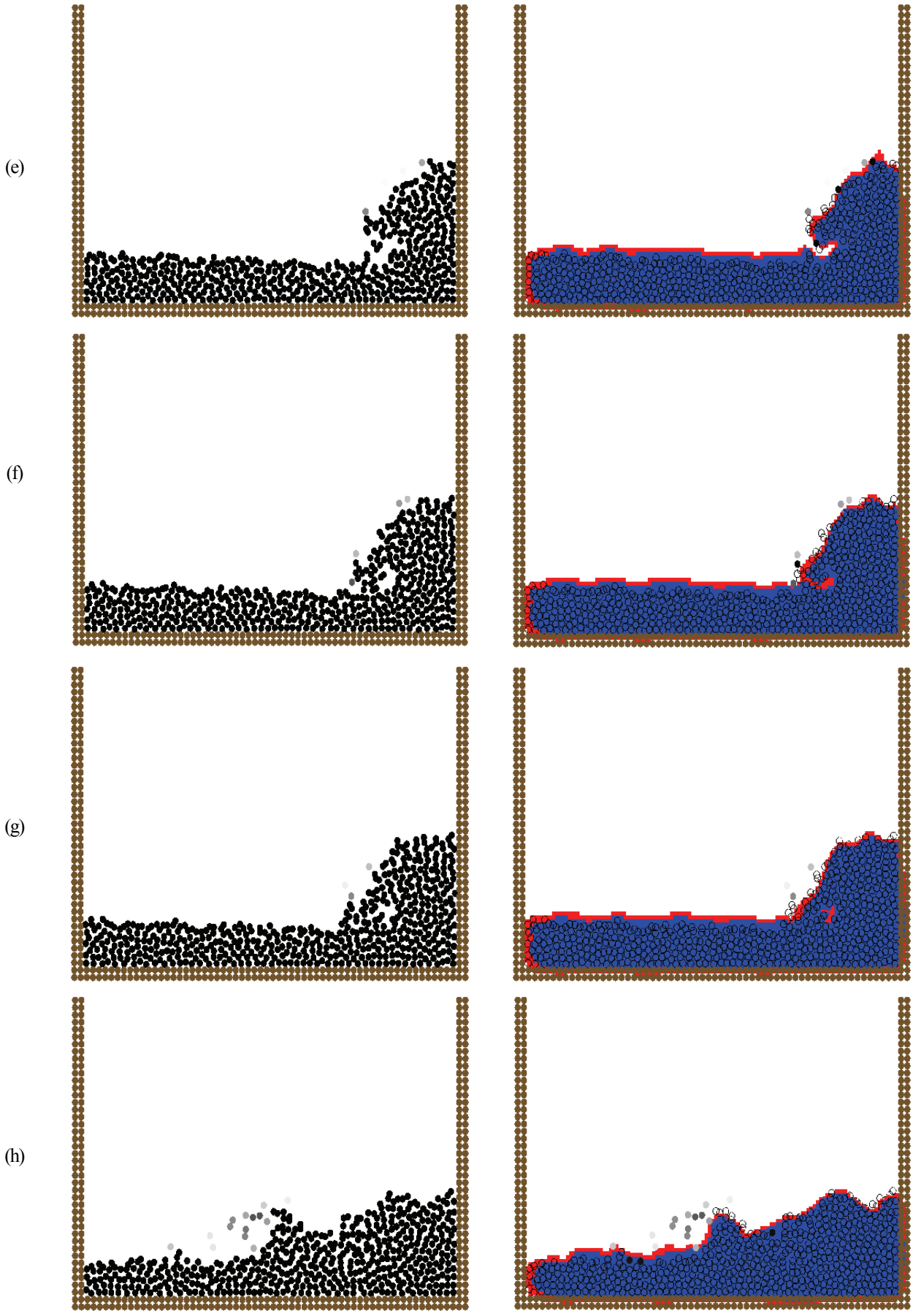


Figure 5: Surface reconstruction for a breaking water flow.

Authors' biographies:

PURKHET ABDERYIM is currently a Ph.D. candidate in computer science at Iwate University. His research interests include computer graphics, computer animation and laser graphics. He received the B.E in computer science from Xinjiang University, China and an ME in computer science from Iwate University in 1999 and 2006, respectively. He worked at Xinjiang University from 1999 to 2003.

TADAHIRO FUJIMOTO is currently an associate professor in the Department of Computer and Information Sciences at Iwate University. His research interests include computer graphics, geometric model, fractal theory, and mathematics for shape description in general. He received a BE in electrical engineering, and an ME and Ph.D. in computer science from Keio University in 1990, 1992, and 2000, respectively. He worked at Mitsubishi Research Institute from 1992 to 1995. He was a research associate in the Department of Computer and Information Sciences at Iwate University from 1999 to 2002, and a lecturer from 2002 to 2005. He is a member of SAS Japan, IEICE Japan, IPS of Japan, IEEE and ACM.

NORISHIGE CHIBA is currently a professor in the Department of Computer and Information Sciences at Iwate University. His research interests include computer graphics, algorithm theory and science on form. He received a BE in electrical engineering from Iwate University and an ME and DE in information engineering from Tohoku University in 1975, 1981 and 1984, respectively. He worked at Nippon Business Consultant Co., Ltd. from 1975 to 1978. He was a research associate in the Department of Communication Engineering at Tohoku University from 1984 to 1986, an associate professor of computer science at Sendai National College of Technology from 1986 to 1987 and an associate professor of the Department of Computer and Information Sciences at Iwate University from 1987 to 1991. He is a member of SAS Japan, IEICE Japan, IPS of Japan, IEEE and ACM.

MAMTIMIN GENI is currently a professor in the Department of Mechanical Engineering, Xinjiang University. His research interests include fracture mechanics, computational science and environmental simulation. He received the B.E in mechanical engineering from Xinjiang University, China and Ph.D. in engineering from Science University of Tokyo, Japan. He was a JSPS Post Doctoral Fellowship at Science University of Tokyo from 1997 to 1999. He is Visiting Associate Professor in the Department of Aircraft Engineering Northwestern Polytechnical University, China.