EUROGRAPHICS 2003 / P. Brunet and D. Fellner
(Guest Editors)

Volume 22 (2003), Number 3

Animation of Bubbles in Liquid

Jeong-Mo Hong and Chang-Hun Kim

Department of Computer Science, Korea University

Abstract

We present a new fluid animation technique in which liquid and gas interact with each other, using the example of bubbles rising in water. In contrast to previous studies which only focused on one fluid, our system considers both the liquid and the gas simultaneously. In addition to the flowing motion, the interactions between liquid and gas cause buoyancy, surface tension, deformation and movement of the bubbles. For the natural manipulation of topological changes and the removal of the numerical diffusion, we combine the volume-of-fluid method and the front-tracking method developed in the field of computational fluid dynamics. Our minimum-stress surface tension method enables this complementary combination. The interfaces are constructed using the marching cubes algorithm. Optical effects are rendered using vertex shader techniques.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation

1. Introduction

Liquids are very attractive substances. As well as having beautiful optical properties, their movements are mysterious, or as an eastern saying goes, "just observing water can provide good meditation." Many studies have been done in an attempt to animate and render liquids in the computer graphics field. And thanks to recent improvements in computing powers and simulation techniques, more phenomena related to liquids have become subjects of animation. In this paper, we are to introduce one more subject pertaining to liquid animation, i.e. bubbles.

Bubbles are pockets of air enclosed by liquid. Bubbles exist everyplace where liquid and air coexist. As opposed to air skimming over liquid surfaces, bubbles are governed by the interactions between air and liquid. There are many factors to be considered when attempting to simulate the deformation and movement of bubbles. There are two flows to consider - i.e. those occurring inside and outside of the bubble bodies. Differences in specific gravity between the two fluids generate buoyancy forces. Surface tension forces are exerted at the interfaces between the two fluids.

In general, the density of liquids is much higher than that of gases. For example, water is as eight hundred times as heavier than air. This fact is one of the reasons for the free surface approximation in which the existence of air is generally ignored in liquid simulations. Many recent studies on liquid animation have referred to the free surface studies which have been done in computational fluid dynamics (CFD). These studies showed very natural results and some

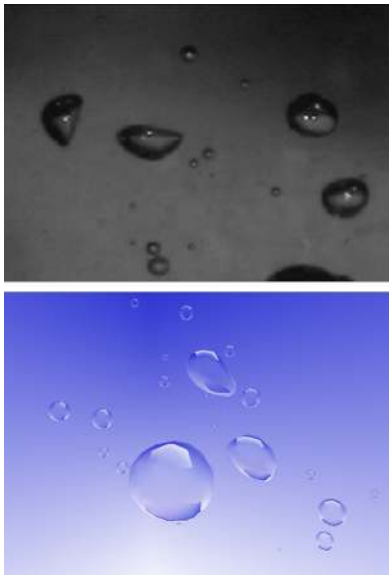

Figure 1: *Rising Bubbles in Liquids. Photo Image (up) and Rendered Image (down)*

air flows were able to be inserted as surface boundary conditions - e.g. wind. However, since enclosed air is an altogether different affair, those studies are not suitable for bubbles. Besides the additional factors described above, one more consideration needs to be taken into account - i.e. two fluids have to be simulated at the same time. This problem is studied in the form of a multiphase flows in CFD with the phasechange problem.

Like other fluid problems, many techniques have been developed for the simulation of multiphase flows in CFD. However, since all techniques have their own characteristic approach, in order to decide which technique to use for computer animation, a set of selection criteria are needed. The criteria that we used were the ease of programming, the numerical stability and the fast simulation, even at the cost of accuracy. However, since the main virtue of CFD is accuracy, no existing technique matched these characteristics exactly, therefore we had to combine and modify various existing techniques for our purposes.

In this paper, we present a new fluid animation technique in which liquid and gas interact with each other, using the example of bubbles rising in water (Figure 1). Our system is based on the complementary combination of the volume-of-fluid (VOF) method and the front-tracking method which were developed in the field of CFD. The VOF method is an efficient and fast scheme for free surface simulation with the inherent capability of topological changes. It can be easily extended for the simulation of multi-phase flows. However, to reduce the effect of numerical diffusion in the VOF scheme, the interfaces between the two fluids in the simulation grid need to be decided exactly, which is simple in 2D, but complicated and computationally expensive in 3D with fluid volume constraints. In contrast to the VOF method, the front-tracking method introduces no numerical diffusion. However, a book keeping process to maintain the front connectivity is needed to handle the topological changes and physically accurate interfacial geometry is required for the calculation of surface tension. Since our minimum-stress surface tension method calculates the surface tension effects not from the interfacial geometry but directly from the simulation data, it was possible to combine these two methods.

Due to the VOF scheme being used, fast interface construction is possible with the marching cubes algorithm. Interfaces composed of polygon meshes are rendered by means of the vertex-shader. Optical effects - refract, reflection and dispersion - are included.

Section 2 presents the previous works on liquid animation and some related CFD techniques, in order to explain the limitations of previous works and the characteristics of our approaches. Section 3 introduces some new concepts for the representation of multiphase fluids and overviews our method. In section 4, the simulation process is discussed in relation to the Navier-Stokes equation. Section 5 discusses the techniques introduced for visualization. In section 6, we present our results. We conclude and discuss ideas for the future research in Section 7. All figures are explained in 2 dimensions and their extension to 3 dimensions should be fairly evident.

2. Previous Work

The characteristics of a physically based model are strongly influenced by the physical and mathematical foundation of that model. Therefore, a combination of both models and CFD techniques is necessary in order to provide a more meaningful explanation. CFD researches include many topics - the accuracy of simulation, numerical techniques, the handling of geometry, and so on. Among them, we will concentrate on only those parts which are directly related to our study.

The governing equation of fluids is known as the momentum or Navier-Stokes equations. Following some initial approaches using simplified versions of the Navier-Stokes equations [10, 14], the animation of complex water was studied [4] using the marker and cell (MAC) method [8] with the full 3D Navier-Stokes equation. In the MAC method, the Navier-Stokes equation is discretized within some fixed uniform Cells and fluids are expressed by Marker particles. Marker particles were able to describe both natural and detailed scenes [4, 8]. This scheme is also applied to melting animations [1]. In order to treat the smooth and detailed surfaces using these marker particles, the implicit surfaces and level-set methods were used [5]. Realistic optical properties were rendered with the physically based ray tracer. In the simulation of very complex scenes, volume loss occurred and to fix this problem the particle level-set method was introduced [3]. This approach enabled the animation of very complex scenes and velocity extrapolation gave us more control with coarse grids. Although the MAC method presents the explicit expresion of liquids with marker particles, it is difficult to estimate the volume of liquid in a cell from marker particles. To represent and simulate two fluids with one grid system, we have to know the volume of each fluid in one grid. Therefore, the animation techqniques based on MAC method [1, 3, 4, 5] are not suitable for our purposes.

For fast animation of liquids [12], the volume of fluid (VOF) method [9] was used, in which the liquid surfaces were constructed using the marching cubes algorithm [13] and rendered with polygonal techniques. The VOF method handles topological changes naturally with the marching cubes algorithm, and basically uses only one scalar value - the volume of fluid - for one cell, through which we can know the total volume of fluid in the simulation space. The VOF method assumes that the liquid in a cell is gathered on one corner. From the volume value of one cell and its adjacent cells, the exact position of the liquids needs to be estimated to eliminate the effects of numerical diffusion. This is a problem involving the intersection of a line and a square in 2D cases [15]. In 3D, these become a plane and a cube

[7], and some numerical iteration is required in order to find a solution. Therefore, it is inefficient to eliminiate numerical diffusion within VOF scheme for computer animation purposes.

Some spherical objects related to fluids such as liquid foams [11], water droplets [6, 21] and soap bubbles [2] have been studied. However for air bubbles enclosed in liquids, the simulation of environmental liquids is unavoidable. This phenomenon can be explained as a kind of multiphase flows. The front-tracking method [19] involves the simulation of multiphase flows without numerical diffusion. The original front-tracking method explicitly discretized the free surface using particles and maintains a connectivity list between these particles [20]. This connectivity list is difficult to maintain when parts of the free surface break apart or merge together as is often seen in complex flows of water and other liquids. To avoid this difficulty, the point-set method was introduced [18]. Although this approach unchains the front tracking method from its dependence on logical interface point connectivity, the point regeneration algorithm is complex and computationally expensive. The level contour reconstruction method [16] is similar to the combination of the VOF method and the marching cubes algorithm used in [12], which possesses the inherent capability of being able to deal with topological changes. The feedback from interfaces to simulation grids still removes numerical diffusion. However, for the calculation of surface tension forces and for numerical accuracy, the physically exact interfaces are needed.

Our minimum-stress surface tension method is implemented independently from the details of interfacial geometry with the sufficient convergency for computer animation. Moreover, the feedback provided by the front-tracking method removes the numerical diffusion and guarantees mass conservation with the benefits coming from the VOF scheme.

In solving the Navier-Stokes equations, the initial approach was based on explicit finite difference scheme [4, 8]. For computer graphics, the stable fluid scheme [17] based on implicit approaches such as semi-Lagrangian method and implicit diffusion was proposed for large time steps and numerical stability. Subsequently, efficient pressure iteration was introduced [5]. Our method utilizes these techniques instead of the standard CFD techniques (see 4.1).

3. Overview

3.1. Representation of Multiphase Fluids

In contrast to previous works which have dealt with the free-surface problem, our study considers two fluids simultaneously. To represent two fluids with one fixed grid system, we define an *indicator function* $I = I(\mathbf{x},t)$. $I(\mathbf{x},t)$ takes the value 1 in one fluid and 0 in the other fluid. A *material field* is defined by the values of I in each cell and Figure 2.a shows an example. As you can see in Figure 2.a, there are some transition zones between 0 and 1, which are at the interfaces between the two fluids.

We can define the interfaces between two fluids with some iso-surface construction algorithm. As is shown in Figure 2.b, we used the marching cubes algorithm with a threshold value of 0.5. Details are discussed in section 5.1.

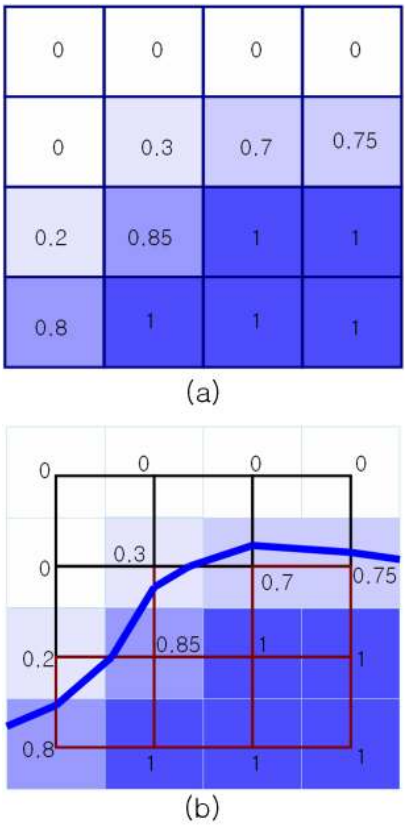

Figure 2: *Material field (a) and interfaces (b) constructed from this material field*

3.2. System Outline

In this section, we divided our animation process into three steps in order to provide a conceptual explanation. They are the velocity field update, material field update and visualization processes. A more detailed explanation of the general processes involved in fluid animation can be found in the literature [4, 5, 8, 9, 17].

Velocity field update

In this step, the velocity field is updated from the initial or previous velocity field by solving the Navier-Stokes equation. Material field data are needed for the calculation of the gravity forces and surface tension forces. The details are discussed in sections 4.1 and 4.2.

Material field update

After updating the velocity field, we should update the material field by evolving the indicator function to reflect the movement of the fluids caused by the velocity field. This involves the flow of materials. The details are discussed in section 4.3.

Visualization

From the updated material field, we construct rendering primitives and render them. As well as the polygonal meshes representing the interfaces, some particles are included for the sake of providing more detailed scenes. The details are discussed in section 5.

4. Simulation of Multiphase Flows

4.1. Navier Stokes Equation

The momentum equation, the so called Navier-Stokes equation for multi-phase flows [19] is

$$\frac{\partial(\rho\mathbf{u})}{\partial t} = \mu\nabla\cdot(\nabla\mathbf{u}) - \nabla\cdot(\rho\mathbf{u}\mathbf{u}) - \nabla P + \rho\mathbf{g} + \int_{\Gamma(t)} \sigma\kappa\mathbf{n}\delta(\mathbf{x}-\mathbf{x}_f)ds \quad (1)$$

where $\mathbf{u}$ is the velocity, ρ is the density, μ is viscosity, P is the pressure and $\mathbf{g}$ is the gravity. The surface integral is a surface tension term. The physical definition and finite difference scheme of surface tension are described in 4.2.

Conservation of mass written for the entire flow field is

$$\nabla\cdot(\rho\mathbf{u}) = -\frac{\partial\rho}{\partial t}. \quad (2)$$

The discrete forms for the finite difference method of (1) and (2) can be written as

$$\frac{\mathbf{w}^{n+1}-\mathbf{w}^n}{\Delta t} = \mathbf{A}^n + \mathbf{F}^{n+1} - \nabla_h P \quad (3)$$

$$\nabla_h\cdot\mathbf{w}^{n+1} = M^{n+1}. \quad (4)$$

Here $\mathbf{w} = \rho\mathbf{u}$ is the fluid mass flux. The advection, diffusion and external forces terms in (1) are lumped into $\mathbf{A}$, the right side of (2) is denoted by M and the surface integral in (1) is denoted by $\mathbf{F}$.

Following the spirit of Chorin's projection method, we split the momentum equation into

$$\frac{\tilde{\mathbf{w}}-\mathbf{w}^n}{\Delta t} = \mathbf{A}^n + \mathbf{F}^{n+1} \quad (5)$$

and

$$\frac{\mathbf{w}^{n+1}-\tilde{\mathbf{w}}}{\Delta t} = -\nabla_h P \quad (6)$$

where we introduce the variable $\tilde{\mathbf{w}}$, which is the new fluid mass flux if the effect of pressure is ignored. The first step is to find this mass flux using (5)

$$\tilde{\mathbf{w}} = \mathbf{w}^n + \Delta t(\mathbf{A}^n + \mathbf{F}^{n+1}). \quad (7)$$

The pressure is found by taking the divergence of (6) and using (4). This leads to a Poisson equation for P

$$\nabla^2 P = \frac{\nabla\cdot\tilde{\mathbf{w}} - M^{n+1}}{\Delta t}, \quad (8)$$

which can be solved using a standard Poisson solver. The updated mass flux is found from (6)

$$\mathbf{w}^{n+1} = \tilde{\mathbf{w}} - \Delta t\nabla P. \quad (9)$$

The updated velocity is $\mathbf{u}^{n+1} = \mathbf{w}^{n+1}/\rho^{n+1}$.

In this paper, the phase change problem coming from heat transfer is not included. In isothermal cases, $\partial\rho/\partial t = 0$, which reduces (2) to

$$\nabla\cdot\mathbf{u} = 0 \quad (10)$$

and (5) to

$$\nabla_h\cdot\mathbf{w}^{n+1} = 0 \quad (11)$$

with $M = 0$. If we consider equation (10) as a *volume* conserving condition, the whole process of finding a solution becomes similar to one involving free-surface conditions.

Since there is no *vacant* space in our simulation, unlike free-surface simulations, all cells should be simulated. Therefore, the free-surface conditioning such as classifying cells and modifying the velocities of surface cells [4, 8], is not needed.

In the first projection step of (5) and (7), we incorporated the stable fluids scheme [17], in which the advection is calculated using the semi-Lagrangian method and the diffusion is calculated with implicit method. The second projection step of (6), (8) and (9) is solved in the form of a mass conservation process [5], in which we use a standard conjugate gradient solver as a Poisson solver. All equations are discretized on the standard staggered MAC grids [8].

4.2. Surface Tension

Surface tension is the apparent interfacial tensile stress (force per unit length of interface) that acts whenever a liquid has a density interface, such as when the liquid is in contacts with a gas, vapor, second liquid, or solid. The mathematical definition of surface tension F in (1) is

$$\mathbf{F} = \int_{\Gamma(t)} \sigma\kappa\mathbf{n}\delta(\mathbf{x}-\mathbf{x}_f)ds \quad (12)$$

where σ is the surface tension coefficient, κ is twice the mean interface curvature, $\mathbf{n}$ is the unit normal to the interface, $\mathbf{x}_f = \mathbf{x}(x,t)$ represents the parameterization of the interface $\Gamma(t)$ and $\delta(\mathbf{x}-\mathbf{x}_f)$ is a three-dimensional delta function that is nonzero only where $\mathbf{x} = \mathbf{x}_f$. Figure 3 visually explains the surface tension forces defined in (12). The black lines refer to a portion of the surfaces. Light blue arrows represent the tension forces being exerting at the interfaces. The

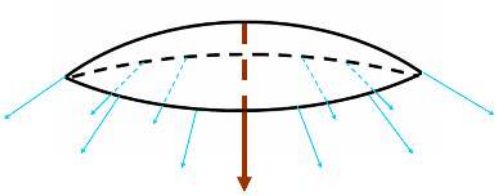

Figure 3: *Surface tension forces*

red arrow, representing the sum of these tension forces, represents the total force being exerting on this portion of the surface.

In front tracking scheme, these forces are calculated using the polygon meshes representing interfaces and distributed to the simulation grids as body forces [16, 19, 20]. Since, the interfaces are constructed from material field descretized on the simulation grids, it is inefficient and unreliable to distribute surface tension forces to the simulation grids estimated from those interfaces. To overcome this inefficiency and remove dependency on interfacial geometry in surface tension calculation as discussed in section 2, our *minimum stress surface tension method* calculates surface tension forces directly from the material field. The physical meaning of (12) is that the surface tension is a tendency to minimize the total stress of interfacial surfaces. So, we defined the stress of material field and let surface tension forces to minimize this stress.

To define the stress of a position on the material field, $S(\mathbf{x})$, first, we define a imaginary stress-zero iso-surface whose value is $I^0(\mathbf{x})$, on which $S(\mathbf{x}) = 0$. Then, $S(\mathbf{x})$ can be defined by the deviation of $I(\mathbf{x})$ from $I^0(\mathbf{x})$. In Cartesian coordinate system, $S(\mathbf{x})$ is defined as

$$S(\mathbf{x}) = c\sum_l (I_l^0(\mathbf{x}) - I(\mathbf{x})) \cdot \mathbf{n}_l, \tag{13}$$

where c is a control coefficient, l is $\{x,y\}$ in 2D and $\{x,y,z\}$ in 3D and $\mathbf{n}_l$ is the unit normal of l direction. In our implementation, we defined $I_l^0(\mathbf{x})$ as

$$I_l^0(\mathbf{x}) = \sum_l \mathbf{n}_l \cdot \{ \sum_{p=m-l} (I(\mathbf{x}_{p+}) + I(\mathbf{x}_{p-}))\}/a, \tag{14}$$

where $m = \{x,y\}$ and $a = 2$ in 2D, and $m = \{x,y,z\}$ and $a = 4$ in 3D. Finally, we can define the material field version of (12) as

$$F(\mathbf{x}) = -\sum_l (S(\mathbf{x}) \cdot \mathbf{n}_l) \nabla_l I(\mathbf{x}), \tag{15}$$

where $\nabla_l I(\mathbf{x}) = (\nabla I(\mathbf{x}) \cdot \mathbf{n}_l)\mathbf{n}_l$.

Figure 4 shows an example of our method. To find the y portion of $F(center)$, first, we assume an imaginary stress-zero iso-surface (red line). In this case, the value of this iso-surface, $I_j^0(center)$, is $(I_{x-} + I_{x+})/2 = (0.3 + 0.5)/2 = 0.4$ using (14). The material value of the center cell 0.9 is bigger

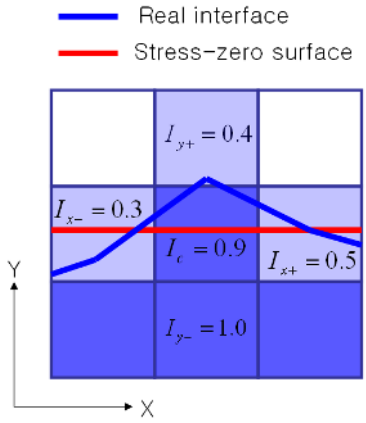

Figure 4: *The minimum-stress surface tension method*

then 0.4 and this implies that the interfaces constructed by marching cubes algorithm (blue line) would not be on the stress-zero surface. Now, we can calculate the direction and magnitude of the y portion of $F(center)$ using (14). The the x portion of $F(center)$ can be calculated in the same way. The extension to 3D cases are fairly evident.

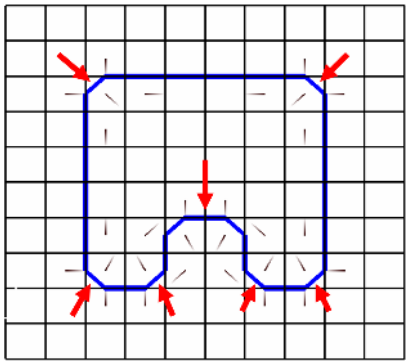

Figure 5: *the surface tension forces inserted as body forces*

The calculated surface tension forces are inserted to (3) as body forces for Navier-Stokes simulation. Figure 5 shows an example. The small lines - the direction is heading from black to white - are normalized surface tension forces inserted as body forces. Red arrows are introduced as visually understandable explanations of the surface tension forces.

4.3. Update of Material Field

The last step in the simulation is the update of the material field. As described in section 3.1, our system describes the positioning of fluids by means of an indicator function. After getting the velocity field as in 4.1, we should evolve the indicator function to reflect the movement of the fluids caused by the velocity field.

The time dependence of indicator function I on a velocity field is governed by the equation [9],

$$\frac{\partial I}{\partial t} + u\frac{\partial I}{\partial x} + v\frac{\partial I}{\partial y} = 0. \tag{16}$$

In the VOF representation, (16) can be solved by trasporting the volume of fluid from one cell to another cell [9]. After some experiments to get the smoothness of animation in the combimation of marching cubes algorithm, we decided our discretized form of (16). In the case of Figure 6, the change of center cell with our discretization is

$$\frac{\triangle I_C}{\triangle t} = -I_C \cdot v_N - I_C \cdot v_W - I_C \cdot v_E + I_S \cdot v_S. \tag{17}$$

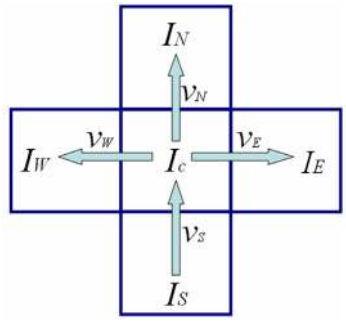

Figure 6: *An example of Indicator function update*

While (17) is easy to implement and shows very smooth animation in combination of marching cubes algorithm (discussed in 5.1), it has the inherent property of numerical diffusion. In ideal simulations, material values of the cells far from interfaces must be 0 or 1. Numerical diffusion occurs when this condition is not fulfilled as shown in Figure 7(b). Numerical diffusion prevents the robust and correct liquid simulation. As discussed in section 2, it is difficult to meet this condition within VOF scheme. However, with the aid of front-feedbacks used in front-tracking method, numerical diffusion can be corrected. In constrast to the MAC representation, we can know the total volume or mass of fluids explicitly with the VOF representation. The total mass of a volume at time t, M^t is

$$M^t = \int_V \rho I^t(\mathbf{x}) dV. \tag{18}$$

Therefore what we have to do to correct the mass loss is just to modify $I^{t+\triangle t}$ to meet $M^{t+\triangle t} = M^t$ by changing some material values.

Since we use marching cubes algorithm for constructing interfaces, it is easy to find the location of interfaces or fronts, and move them by scaling adjacent material values. In this modifying step, we fix the value of the indicator function to 0 or 1 *except for* the cells near interfaces or fronts in order to remove numerical diffusion and maintain the location of the interfaces at the same time. Subsequently, we pulled out or pushed back the interfaces to maintain the total mass by scaling the material values near the interfaces. The scaling factor is decided as

$$SF = \frac{M^t - M_{fixed}}{M^{t+\triangle t} - M_{fixed}}. \tag{19}$$

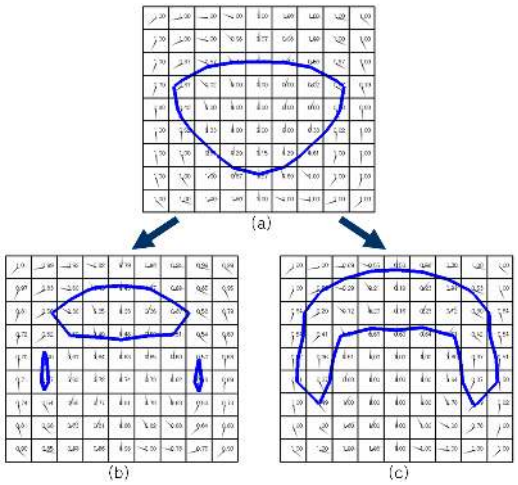

Figure 7: *Restricting the numerical diffusion*

Figure 7 is a rising bubble example. Figures 7.a represents initial configurations. Unlike Figure 7.b, with our correcting step, Figure 7.c shows the numerically perfect mass conservation and no numerical diffusion.

5. Visualization

5.1. Interface Construction

In the front-tracking method, the interfaces between two materials - in our case, water and air - are composed of polygon meshes for the easy calculation of the surface tension. Book-keeping method [20] in the case of polygon meshes is difficult because of the topological changes of the fluids. Recently, the iso-surface construction method for front-tracking [16] was used to solve this problem. This approach handles topology changes in natural way, which is appropriate for the purpose of animation. Since our use of surface tension steps using the minimum-stress tendencies discussed in section 4.2 reduces the need for a detailed expression of the interfaces, we were able to use the marching cubes algorithm for iso-surface construction. In addition to its compatibility with our staggered grid system, the look-up table style of the marching cubes algorithm supports fast animation [12, 13]. In this case, the material field plays a role of the intensity field needed in marching cubes algorithm (see Figure 2.b). The vertex normal was calculated by interpolating the gradient of the material field.

With the marching cubes algorithm, there are many possibilities of discontinuities arising in the animation. Furthermore, since our indicator function is defined by a discontinuous delta function, the continuity of the animation could be damaged. However, though the approach we used to update the indicator function introduces the numerical diffusion without front-tracking steps, it shows smooth animation with the marching cubes algorithm. This is one more benefit which arises from the use of our indicator function update method (discussed in 4.3).

5.2. Particle System

Small bubbles are spherical due to the domination of the surface tension forces. We used the particle system for small bubbles with no deformation. While the computational cost associated with the particles is small, they provided for a lively animation. The velocity of a particle is determined by the linear interpolation of six facial velocities of the cell containing that particle. For natural behavior, buoyant forces are added as body forces, which is similar to the approach taken in the MAC method. Sizes and initial positions are randomly decided. In some cases, the use of the particle system alone could provide for a good animation of bubbles.

5.3. Rendering

Unlike other approaches using implicit surfaces [3, 5], in our system, the interfaces are composed of polygon meshes, which enables fast rendering supported by hardware-acceleration [12]. Some optical effects were able to be implemented by means of a vertex-shader. Reflection, refraction and dispersion effects were applied using conventional vertex-shader codes [22], which results in visually pleasing scenes.

6. Results and Discussion

We have implemented our animation system using the OpenGL APIs and the NVIDIA vertex shader codes. We have tested this system on Windows PC system. Our test machine is a PC with 512 MB of RAM and an Intel Pentium IV processor running at 1.4 GHz. It uses an NVIDIA GeForce 2 MX graphics card with 64 MB of video RAM.

Before simulation process, the properties and the initial conditions of fluids should be given. They are the initial velocity field, viscosity and density of each fluid, gravity, surface tension coefficient between two fluids and initial configuration.

Surface tension

Figure 8 and 9 are examples provided to show the convergency of our minimum-stress surface tension method. The influence of gravity was omitted for clarity. Even though our algorithm calculates the surface-tension forces using material field independently from the details of interface geometry, any arbitrary shapes converged to the spherical ones with no volume loss. In spherical shapes, all surface tension forces are cancelled by each other. Some oscillatory phenomena were also included, which are similar to those observed in nature. We used 13 x 13 x 13 simulation grids and frame rate was 7.6 fps.

A bubble near a free surface

When rising bubbles arrive at a free surfaces, they are absorbed by the atmospheric air leaving violent impacts on the free surfaces. Figure 10 shows this phenomenon. In spite of the severe shape changes, this simulation shows natural animation. The merging of bubble meshes and surface meshes, is done naturally. We used 15 x 15 x 15 simulation grids and frame rate was 5.2 fps.

Although this result proves that it is possible to deal with the free-surface condition within our simulation frameworks, there occurs a problem of volume gain of atmospheric air - i. e. the free-surfaces lowers before meeting the bubble. The reason of this problem is that we conserve *whole air volume or whole liquid volume* in our current implementation as discussed in section 4.3. To fix this problem, we have to check each separated air volumes - i.e. $M^t = \sum_i M_i^t$ - and conserve each of them. With our front-feedbacks, we can easily find the separated volumes and conserve them. We will handle this problem with free-surface conditions as a future work.

Rising bubbles

Figure 11 shows a decorated version of the rising bubbles problem. The bubble rises due to their buoyancy. After merging with other small bubble, it rises with certain fixed shape. The shape constitutes a kind of balance point between buoyancy forces and the surface tension forces. The small bubbles are animated using particle system with no deformation as discussed in section 5.2. Visually pleasing optical effects were included using vertex shader techniques. We used 9 x 9 x 25 simulation grids and frame rate was 1.2 fps.

7. Conclusion and Future Work

In this study, we present a new fluid animation technique in which liquid and gas interact with each other. Our algorithm is based on a complementary combination of various CFD techniques which are selected and modified for computer animation purposes with the aid of our minimum stress surface tension method. We introduced the finite difference scheme for the simulation of the multi-phase Navier-Stokes equation and used appropriate visualization techniques using the marching cubes algorithm and the hardware acceleration.

Since our algorithm can handle topological changes and surface tension fairly easily and with no volume loss or numerical diffusion, we can extend it to the physically based

simulation of water droplet model interacting with static environments or other droplets.

References

1. M. Carlson, P. J. Mucha, R. B. Van Horn II and G. Turk, "Melting and flowing", *ACM SIGGRAPH Symposium on Computer Animation* (2002).
2. R. Durikovic, "Animation of soap bubble dynamics, cluster formation and collision", *Computer Graphics Forum (Eurographics 2001 Proc.)* **20** (3), 67–76 (2001).
3. D. Enright, S. Marschner and R. Fedkiw, "Animation and rendering of complex water surfaces", *SIGGRAPH 2002, ACM TOG*, **21**, 736–744 (2002).
4. N. Foster and D. Metaxas, "Realistic animation of liquids", *Graphical Models and Image Processing*, **58**, 471–483 (1996).
5. N. Foster and R. Fedkiw, "Practical animation of liquids", *In Proceedings of ACM SIGGRAPH 2001*, 23–30 (2001).
6. Fournier, A. Habibi and P. Poulin, "Simulating the flow of liquid droplets", Graphics Interface '98, (1998).
7. D. Gueyffier, J. Li, A. Nadim, R. Scardoveli and S. Zaleski, "Volume-of-Fluid interface tracking with smoothed surface stress method for three-dimensional flows", *J. Comput. Phy.*, **152**, 423–456 (1999).
8. F. H. Harlow and J. E. Welch, "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surfaces", *Phys. Fluids*, **8**, 2182–2189 (1965).
9. C. W. Hirt and B. D. Nichols, "Volume of Fluid (VOF) method for the dynamics of free boundaries", *J. Comput. Phys.*, **39**, 201–255 (1981).
10. M. Kass., and Miller, G., "Rapid, stable fluid dynamics for computer graphics", In *Computer Graphics (Proceedings of ACMSIGGRAPH 90)*, **24**, 49–57 (1990).
11. H. Kuck, C. Vogelgsang and G. Greiner, "Simulation and rendering of liquid foams", Graphics Interface 2002 (2002).
12. A. Kunimatsu, Y. Watanabe, H. Fujii, T. Saito, K. Hiwada, T. Takahashi, H. Ueki, "Fast simulation and rendering techniques for fluid objects", *Computer Graphics Forum (Eurographics 2001 Proc.)*, **20**(3), 357–367 (2001).
13. W. E. Lorensen, and H. E. Cline, "Marching cubes: A high resolution 3D surface consruction algorithm.", In emphComputer Graphics (Proceedings of ACM SIGGRAPH '87), (21) (4), 163–169 (1987).
14. J. O'Brien, and J. Hodgins, "Dynamic simulation of splashing fluids", In *Proceedings of Computer Animation 95*, 198–205 (1995).
15. W. J. Rider and D. B. Kothe, "Reconstructing volume tracking", *J. Comput. Phy.*, **141**, 112–152 (1998).
16. S. Shin and D. Juric, Modeling, "Three-dimensional multiphase flow using a level contour reconstruction method for front tracking without connectivity", *J. Comput. Phys.*, **180**, 427–470 (2002).
17. J. Stam, "Stable fluids", In *Proceedings of ACM SIGGRAPH 1999*, 121-128 (1999).
18. D. J. Torres and J. U. Brackbill, "The point-set method: front-tracking without connectivity", *J. Comput. Phys.*, **165**, 620-644 (2000).
19. G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawashi, W. Tauber, J. Han, S. Nas and Y.-J. Jan, "A front tracking method for the Computations of multiphase flow", *J. Comput. Phys.*, **169**, 708-759 (2001).
20. S. Unverdi and G. Tryggvason, "A front tracking method for viscous, incompressible, multifluid flows", *J. Comput. Phys.*, **100** 25 (1992).
21. Y. Yu, Ho, Jung and H. Cho, "A new water droplet model using metaball in the gravitational field", *Computers and Graphics*, **23** (2), 213-222 (1999).
22. http://developer.nvidia.com/

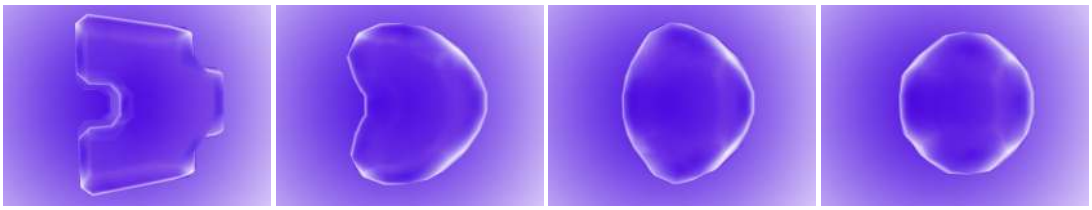

Figure 8: *Surface tension convergency*

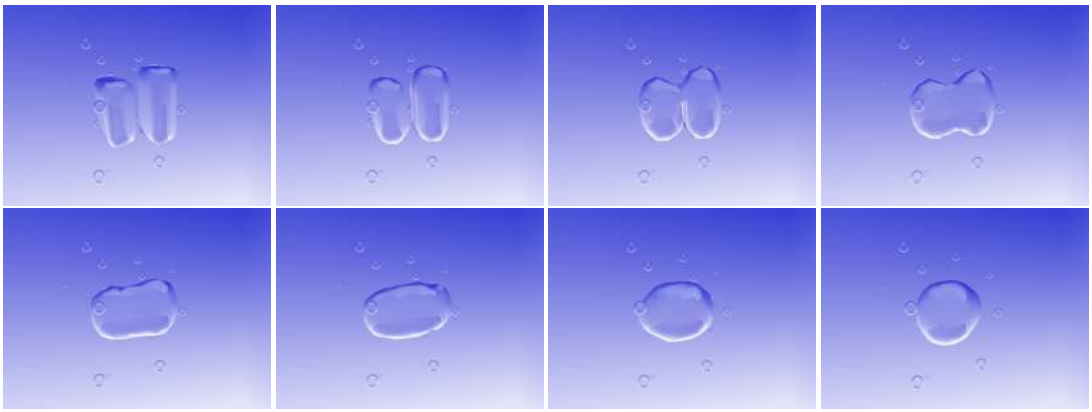

Figure 9: *Deformation and merging caused by surface tension*

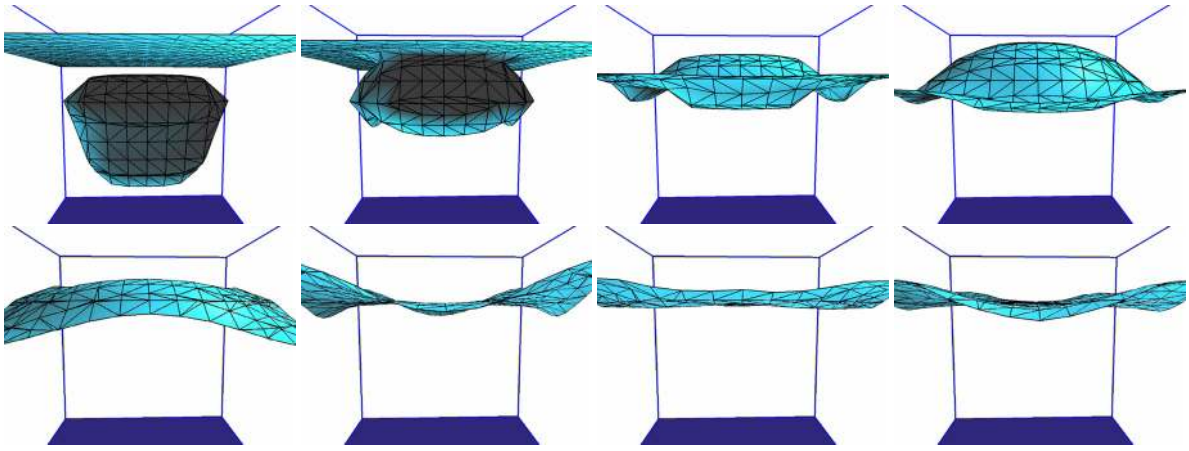

Figure 10: *A bubble merges into a free surface.*

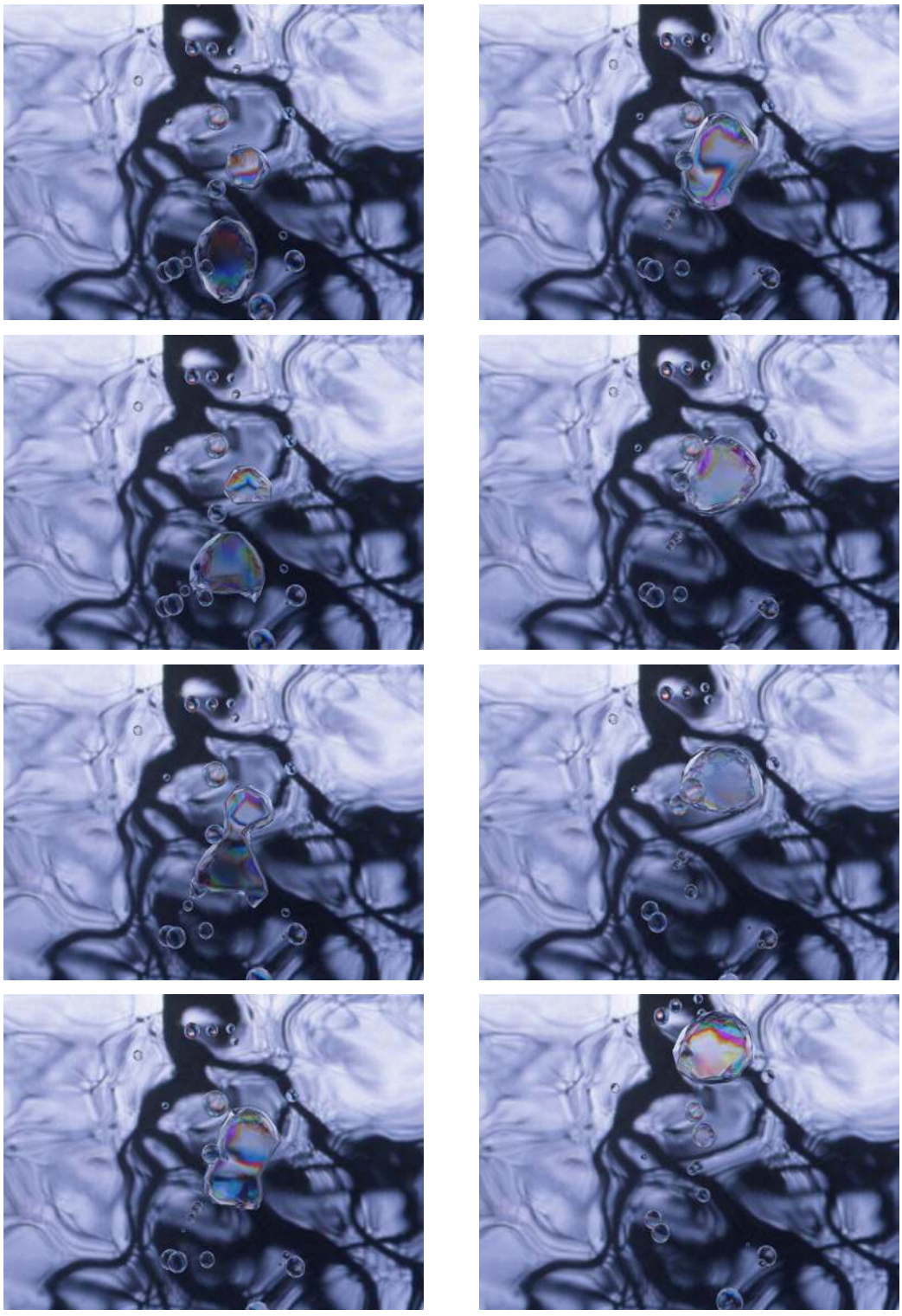

Figure 11: *Rising Bubbles in a Liquid (from left-up to right-down)*