물리적 모델에 기반한 다상 유체 현상 애니메이션 *

송오영°, 신현철, 고형석
그래픽스 및 미디어 연구실, 서울대학교
{song,shin,ko}@graphics.snu.ac.kr

A Physics-Based Modelling of Multiphase Fluid Phenomena

Oh-young Song°, Hyuncheol Shin, and Hyeong-Seok Ko
Graphics & Media Lab., Seoul National University

요 약

This paper presents a physically based technique for simulating complex multiphase fluids. This work is motivated by the "stable fluids" method developed by Stam to handle gaseous fluids. We extend this technique to water, which calls for the development of methods for modeling multiphase fluids and suppressing dissipation. We construct a multiphase fluid formulation by combining the Navier–Stokes equations with the level set method. By adopting constrained interpolation profile (CIP)-based advection, we reduce the numerical dissipation and diffusion significantly. We further reduce the dissipation by converting potentially dissipative cells into droplets or bubbles that undergo Lagrangian motion. Due to the multiphase formulation, the proposed method properly simulates the interaction of water with surrounding air, instead of simulating water in a void space. Moreover, the introduction of the non-dissipative technique means that, in contrast to previous methods, the simulated water does not unnecessarily lose mass and its motion is not damped to an unphysical extent. Experiments showed that the proposed method is stable and runs fast. It is demonstrated that two-dimensional simulation runs in real-time.

1. Introduction

Water, which covers two thirds of the earth, undergoes myriad types of motion in its constant interactions with air, solids, and living creatures. Water has featured prominently in several recent feature animations, including Finding Nemo and Perfect Storm. The success of those movies depended greatly on visual effects in the animation of water. Physically based approaches have been shown to effectively reproduce water movement, with quite impressive results [6, 3].

However, several open challenges remain in this field. One key issue is speeding up the simulation of water. In the case of gaseous phenomena, interactive simulation methods already have been introduced by [23]. The method is called "stable fluids", which allows a large simulation time step to be used without causing instabilities. Unfortunately, this method is known to suffer from large amounts of numerical dissipation, which results in loss of mass. This is not important when simulating dissipative media such as fog or smoke, but it is not tolerable when animating intrinsically non-dissipative substances like water. Another undesirable property of the stable fluids method that must be noted is numerical diffusion, which dampens the fluid motion. Although damping is an inherent property of all fluids, the damping caused by numerical diffusion in the stable fluids method is too severe. Therefore, if we wish to simulate water using an approach based on the stable fluids method, we must modify that method so as to prevent the numerical dissipation and reduce the numerical diffusion.

This paper presents a new physically based method for simulating water. The proposed method, which is based on the semi-Lagrangian methodology, retains the speed and stability of the stable fluids technique while additionally including mechanisms to fix the problems of numerical dissipation and diffusion. To obtain nondissipative water, we adopt the constrained interpolation profile (CIP) method, which has been shown to remarkably reduce dissipation due to the use of coarse grids. To prevent dissipation due to the use of a large time step, we propose a novel particle-based approach, which we show to be quite effective at preventing dissipation of small-scale features. This particle-based approach is also used to simulate droplets and bubbles, which contributes to the overall visual realism. In addition, compared to existing methods, the proposed method simulates water–air interactions more accurately by employing the multiphase dynamic equations that account for the presence of air.

The rest of the paper is organized as follows: Section 2 reviews previous work; Section 3 formulates the multiphase fluid; Section 4 describes the CIP-based fluid solver; Sections 5 and ?? present our particle-based technique for preventing dissipation;

*본 연구는 정보 통신부, 서울대학교 자동화시스템 공동 연구소 및 두뇌 한국 21 프로젝트의 지원으로 수행되었음.

Section 6 reports our experimental results; and finally, Section 7 concludes the paper.

2. Previous Work

Early work on physically based simulation of water for graphics applications concentrated on animating the height-field representation of the water surface. To obtain interactive performance, researchers used the two-dimensional (2D) approximation of the Navier–Stokes equations. Kass and Miller [13] generated the height fields using an approximate version of the 2D shallow water equations. To simulate water–object interactions, Chen and Lobo [2] solved the 2D Navier–Stokes equation that includes pressure. O'Brien and Hodgins [18] proposed a method for simulating splashing liquids by integrating a particle system into a 2D height field model.

Height fields cannot be used to represent water that is undergoing a highly dynamic motion such as pouring. To handle such motions, researchers turned to the 3D Navier–Stokes equations. Foster and Metaxas [7, 8] animated 3D liquids by modifying the Marker and Cell method proposed by Harlow and Welch [12]. In addition, Foster and Metaxas simulated gases by using an explicit finite difference approximation of the Navier–Stokes equations [9]. Stam [23] introduced the unconditionally stable fluid model, which utilizes the semi-Lagrangian method in combination with an implicit solver. This model gave significantly improved simulation speeds, but suffered from numerical dissipation. To reduce the dissipation in simulations of gaseous fluids, Fedkiw et al. [4] proposed the use of vorticity confinement and cubic interpolation. Based on the stable semi-Lagrangian framework, Rasmussen et al. [22] proposed an efficient method for depicting large-scale gaseous phenomena, and Feldman et al. [5] proposed an explosion model that incorporated a particle-based combustion model into the semi-Lagrangian framework. Treuille et al. [29] proposed a constrained optimization technique for keyframe control of fluid simulations. In [16], they improved the optimization speed drastically by adapting the adjoint method.

In order to handle 3D liquids, the semi-Lagrangian scheme must be augmented with a robust and accurate method for tracking the liquid surface. To address this issue, Foster and Fedkiw [6] proposed a novel method for representing a dynamically evolving liquid surface, which was based on combining the level set method with massless marker particles. Enright et al. [3] improved this hybrid scheme by introducing the "particle level set method" which could capture water surface with a remarkable accuracy. The particle level set method was employed in recent studies of fluids by Carlson et al. [1], Goktekin et al. [19], and Losasso et al. [15].

Takahashi et al. [27] simulated multiphase fluids by employing the CIP method coupled with the volume of fluid scheme; their method simulated the water–air interaction properly, instead of simulating water in a void space. When we are to animate water at an interactive rate, as demonstrated by Stam [23] in the case of gas, then the use of large time steps should be allowed. But it can cause dissipation of mass. In [6, 3], the time step size had to be restricted to prevent loss of mass. Although the CIP scheme used by Takahashi et al. [27] lessened the degree of the dissipation, loss of mass was still noticeable when large time steps were used.

Several particle-based methods have been proposed as alternatives to the above grid-based approaches. Miller and Pearce [17] simulated fluid behavior using particles connected with viscous springs. Terzopoulos et al. [28] adopted a molecular dynamics model to simulate particles in the liquid phase. Stam and Fiume [24] introduced "smoothed particle hydrodynamics" (SPH) to depict fire and gaseous phenomena. In SPH, the fluid is modeled as a collection of particles with a smoothed potential field. Premože et al. [21] introduced the use of the moving particle semi-implicit method (MPS) for simulating incompressible multiphase fluids. One drawback of particle-based methods is that, if insufficient particles are used, they tend to produce grainy surfaces. To prevent this, a sufficiently large number of particles must be used, which increases the computational cost.

3. Formulation and Overview

In order to produce realistic movement of water in the presence of air, we base our method on the multiphase Navier–Stokes equations in combination with the level set method. The multi-phase Navier–Stokes equations can simultaneously represent both water and air. The level set method, which can represent the water–air interface as an implicit surface, has been shown to be a robust method for capturing topological changes of water surfaces. Furthermore, the surface curvature can be accurately calculated from the level set values, and hence the surface tension, which is proportional to the curvature, can be easily incorporated into the dynamic simulation.

We start by introducing the incompressible Navier–Stokes equations for a multiphase fluid. Let $\mathbf{u} = (u, v, w)$ denote the velocity field of the fluid. Then, the flow of fluid is described by

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

and

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} + \frac{\mathbf{f}}{\rho} + \frac{\nu}{\rho} \nabla^2 \mathbf{u} - \frac{\nabla p}{\rho}, \tag{2}$$

where p is the pressure, ρ is the density, ν is the kinematic viscosity, and $\mathbf{f}$ represents the external forces per volume. Eqs. (1) and (2) state that mass and momentum, respectively, should be conserved. To treat the immiscible multiphase fluid consisting of water and air within a single formulation, we employ the level set function ϕ [20, 26]. ϕ is an implicit signed distance function defined to be positive for water and negative for air. Thus the sign of ϕ also determines the density and viscosity of the medium.

The water dynamically evolves in space and time according to the underlying fluid velocity field $\mathbf{u}$. The updates in the level set values due to $\mathbf{u}$ are expressed by the level set equation:

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0, \tag{3}$$

The surface of water, which will be a focal point throughout this work, can be obtained by tracking the locations for which $\phi = 0$.

To solve the above equations numerically, we divide the space into a finite number of cells. We evaluate the pressure and level set values at the cell center, but we evaluate the velocity at the center of each cell face. This approach is the classical staggered grid discretization [12], which naturally enforces boundary condition and avoids the checkerboard pattern of the pressure [30]. At each time step, our simulator performs the following three steps:

1. **Advect the level set:** The level set ϕ is advected according to Eq. (3), which causes the density and viscosity fields appearing in Eq. (2) to be updated.
2. **Update the velocity:** Eq. (2) is solved for $\mathbf{u}$ using the following procedure [23]: (1) calculate the advection component $\mathbf{u}\cdot\nabla\mathbf{u}$ using the semi-Lagrangian method; (2) apply the forces $\mathbf{f}/\rho$; (3) add the effect of the viscous term $\nu/\rho\nabla^2\mathbf{u}$ by employing implicit central differencing; and (4) project the velocity field so that the condition $\nabla\cdot\mathbf{u}=0$ is met.
3. **Simulate droplets/bubbles:** Droplets/bubbles are identified and simulated using the particle dynamics until they merge into the body of water/air.

Execution of the above procedure produces the ϕ and $\mathbf{u}$ of the next time step. The method for implementing Steps 1 and 2 is presented in the following section, and the implementation of Step 3 is described in Section 5.

4. CIP-Based Fluid Simulator

The framework of our fluid simulator is based on the semi-Lagrangian scheme, which was briefly introduced in Section 3. A detailed description of the semi-Lagrangian scheme can be found in [23, 25]. In the method proposed here, we make several modifications to the previous semi-Lagrangian scheme to reduce the numerical dissipation and diffusion. This section describes those modifications.

4.1 CIP advection

In the semi-Lagrangian scheme, advection is implemented by referring to the function value at $\mathbf{x}-\mathbf{u}\Delta t$.[1] Since the physical values of ϕ and $\mathbf{u}$ are defined only at discrete points, the function values for $\mathbf{x}-\mathbf{u}\Delta t$ can be obtained by linearly interpolating the neighboring grid points. This approach is computationally efficient and unconditionally stable [23]. However, it may smooth out the subcell features. This problem, referred to as nonphysical numerical diffusion, causes the movement of fluid to be excessively damped, which hampers the generation of turbulent effects such as the formation of water droplets or air bubbles in violently interacting multiphase fluids. It also causes dissipation of the mass.

Fortunately, an anti-diffusive technique called the Constrained Interpolation Profile (CIP) method is proposed by [32, 33]. We adopt the CIP method when solving Eq. (3), resulting in a reduction of mass dissipation, and also when solving the advection term $\mathbf{u}\cdot\nabla\mathbf{u}$ in Eq. (2), leading to a reduction in the degree of damping. Adoption of the CIP method allows us to simulate phenomena such as turbulent flows or swirls with reasonable visual quality.

The key idea of the CIP method is to use not only the function values at the grid points but also the spatial derivatives at those points for constructing the profile inside the grid cell. For example, in the one-dimensional case, the profile corresponding to $[x_i,x_{i+1}]$ now has four conditions ϕ_i, ϕ_i', ϕ_{i+1}, and ϕ_{i+1}', and can be represented by the third order polynomial

$$\Phi(X)=[(aX+b)X+\phi_i']X+\phi_i, \tag{4}$$

where $X=x-x_i$ for $x\in[x_i,x_{i+1}]$. The coefficients a and b can be expressed in terms of the four conditions:

$$\begin{aligned} a &= (\phi_i'+\phi_{i+1}')/\Delta x^2-2\Delta\phi/\Delta x^3,\\ b &= 3\Delta\phi/\Delta x^2-(2\phi_i'+\phi_{i+1}')/\Delta x, \end{aligned}$$

where $\Delta x=x_{i+1}-x_i$ and $\Delta\phi=\phi_{i+1}-\phi_i$. The spatial derivatives used in the CIP method are directly determined from the differentiated version of the original advection equation.

To advect the level set values, we differentiate Eq. (3) with respect to ξ, which gives

$$\frac{\partial\phi_\xi}{\partial t}+\mathbf{u}\cdot\nabla\phi_\xi=-\mathbf{u}_\xi\cdot\nabla\phi, \tag{5}$$

where $\phi_\xi=\partial\phi/\partial\xi$, $\mathbf{u}_\xi=\partial\mathbf{u}/\partial\xi$, and ξ is one of the spatial variables (x,y,z). The task of solving the above equation for ϕ_ξ can be performed in two steps: firstly solving the non-advective part $\partial\phi_\xi/\partial t=-\mathbf{u}_\xi\cdot\nabla\phi$ using finite differencing; then advecting the result according to

$$\frac{\partial\phi_\xi}{\partial t}+\mathbf{u}\cdot\nabla\phi_\xi=0. \tag{6}$$

Noting that the advections of Eqs. (3) and (6) are driven by the same velocity field $\mathbf{u}$, we can advect both ϕ and (ϕ_x,ϕ_y,ϕ_z) by referring to the same point of the profile. For the one-dimensional case, suppose that the grid point x_j is backtracked to $x_r\in[x_i,x_{i+1}]$. Then, the advected results ϕ_j and ϕ_j' can be obtained by evaluating Eq. (4) and its differentiated form at $X_r=x_r-x_i$. Namely, $\phi_j=\Phi(X_r)$ and $\phi_j'=\Phi'(X_r)$.

A problem can arise if we use the profile of Eq. (4) as it stands. In the one-dimensional case, for example, the value $\Phi(X_r)$ may lie outside the range $[\phi_i,\phi_{i+1}]$, which can cause instabilities. One solution to this problem is the rational CIP method proposed by Xiao et al. [31], which suppresses the above oscillation problem by using a rational function instead of a polynomial function. In the present work, we developed an alternative CIP scheme that explicitly modifies the derivatives, with this modification being applied only to the cells in which oscillations are present. This scheme, which is described in detail in the Appendix, guarantees a monotonic profile; hence we call it monotonic CIP. Because it uses polynomials, monotonic CIP runs faster than rational CIP.

[1] In fact, more sophisticated backtracking methods can be used. In this work, we used the second order Runge-Kutta backtracking.

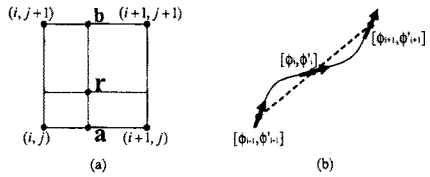

그림 1: (a) Two-dimensional CIP interpolation; the x-axis is along the horizontal direction and the y-axis is along the vertical direction. (b) Schematic of a situation where ϕ_{i-1}, ϕ_i, and ϕ_{i+1} are aligned.

Higher (i.e., two- or three-) dimensional CIPs have been proposed by Yabe et al. [33]. In those methods, however, the derivative constraints are not applied at every grid point, which can result in a non-monotonic profile. Here we implement higher dimensional CIPs based on the one-dimensional monotonic CIP solver. Our implementation of higher dimensional CIPs, which is described below, is always stable. Consider the two-dimensional case shown in Fig. 1 (a), where $\mathbf{r}$ is the backtracked point. For this system, we must determine $[\phi, \phi_x, \phi_y, \phi_{xy}]_{\mathbf{r}}$ from the values at the four corners: (i,j), $(i+1,j)$, $(i,j+1)$, and $(i+1,j+1)$. Here, $[\cdot]_{\mathbf{x}}$ denotes the function and derivative values at point $\mathbf{x}$. The one-dimensional solver can determine $[\phi, \phi_x]_{\mathbf{a}}$ from $[\phi, \phi_x]_{(i,j)}$ and $[\phi, \phi_x]_{(i+1,j)}$, and $[\phi_y, \phi_{xy}]_{\mathbf{a}}$ from $[\phi_y, \phi_{xy}]_{(i,j)}$ and $[\phi_y, \phi_{xy}]_{(i+1,j)}$. Similarly, it can determine $[\phi, \phi_x, \phi_y, \phi_{xy}]_{\mathbf{b}}$. Then, it can determine $[\phi, \phi_y]_{\mathbf{r}}$ from $[\phi, \phi_y]_{\mathbf{a}}$ and $[\phi, \phi_y]_{\mathbf{b}}$, and $[\phi_x, \phi_{xy}]_{\mathbf{r}}$ from $[\phi_x, \phi_{xy}]_{\mathbf{a}}$ and $[\phi_x, \phi_{xy}]_{\mathbf{b}}$.

In the three-dimensional case, the CIP implementation additionally calls for the values of ϕ_{xy}, ϕ_{yz}, ϕ_{xz}, and ϕ_{xyz}. Obtaining those values by analytic differentiation of the original formula involves a large amount of computation. However, our experiments indicated that approximating the second order and higher derivatives by finite differencing of the first order derivatives caused no visually noticeable difference in the result, but significantly reduced the amount of computation.

We close this section by noting the attractive aspects of CIP in comparison with other interpolation methods. In cases where ϕ_{i-1}, ϕ_i, and ϕ_{i+1} are aligned, such as that shown in Fig. 1 (b), spline techniques that do not utilize the derivative information interpret ϕ as being straight. In contrast, because CIP utilizes the spatial derivatives ϕ'_{i-1}, ϕ'_i, and ϕ'_{i+1} of the original equation, it results in more accurate modeling of the real situation. Therefore, the CIP method allows us to use a fairly coarse grid. Another advantageous feature of the CIP method is that, whereas high-order interpolation methods for the semi-Lagrangian scheme such as the cubic spline, quintic Lagrange [25], and monotonic cubic polynomial [4] methods require three stencils (or four grid points) to construct the profile in the one-dimensional case, the profile used in the CIP method can be constructed with information from a single cell. This feature is particularly useful when treating boundaries.

4.2 Non-advection part

We now apply the external forces, $\mathbf{f}$, to the result of the advection according to $\mathbf{u} \cdot \nabla \mathbf{u}$ of Eq. (2). The external forces consist of gravity and user-defined force. The gravitational force is expressed as $\rho \mathbf{g}$, where $\mathbf{g}$ is the gravitational acceleration. The user-defined force is interactively given from mouse or keyboard input.

The result obtained by applying the external forces then goes through the diffusion step, $\nu/\rho \nabla^2 \mathbf{u}$, for which we use the implicit solver following the approach described in [23]. To process the last term of Eq. (2), $-\nabla p/\rho$, we impose the fluid incompressibility condition represented in Eq. (1), which produces the Poisson equation:

$$\nabla \cdot (\frac{\nabla p}{\rho}) = \frac{\nabla \cdot \tilde{\mathbf{u}}}{\Delta t}, \tag{7}$$

where $\tilde{\mathbf{u}}$ is the intermediate velocity field obtained by processing Eq. (2) up to the diffusion term. Eq. (7) can be discretized as

$$\begin{aligned} \sum_{n=\{i,j,k\}} (\rho^{-1}_{n+\frac{1}{2}} + \rho^{-1}_{n-\frac{1}{2}}) p_n - \rho^{-1}_{n+\frac{1}{2}} p_{n+1} - \rho^{-1}_{n-\frac{1}{2}} p_{n-1} \\ = -\frac{1}{\Delta t} \sum_{n=\{i,j,k\}} (\tilde{u}_{n+\frac{1}{2}} - \tilde{u}_{n-\frac{1}{2}}), \end{aligned} \tag{8}$$

where $p_{n\pm1}$ are the pressure values taken from the centers of the neighboring cells, and $\tilde{u}_{n\pm\frac{1}{2}}$ and $\rho_{n\pm\frac{1}{2}}$ are the velocity and density values taken from the cell faces. We can assemble the collection of equations of the form of Eq. (8) covering the domain space into a large linear system

$$\mathbf{A}\mathbf{p} = \mathbf{b}, \tag{9}$$

where $\mathbf{p}$ is the vector of unknown pressures required to make the velocity field divergence free. $\mathbf{A}$ is a positive-definite, symmetric matrix in which density is a variable in space. This system can be efficiently solved using the incomplete-Cholesky preconditioned conjugate gradient method (ICCG) [11]. The solution of Eq. (9) is then used to calculate ∇p at the cell faces. Finally, we obtain the divergence free velocity field by

$$\mathbf{u} = \tilde{\mathbf{u}} - \Delta t \frac{\nabla p}{\rho}. \tag{10}$$

5. Prevention of Dissipation in Small-Scale Features

Although the CIP-based simulator described in the previous section can represent the content of each cell up to the third order, it still suffers from the following two problems: (1) when a large time step is used, it produces non-negligible mass errors; and (2) it cannot represent subcell-level features such as water droplets and air bubbles. In this section, we develop a novel technique to solve these problems. The proposed technique uses particles[2]

[2]Note that our use of particles should not be confused with that proposed by Enright et al. [3]. They use massless marker particles to obtain a more accurate water surface, whereas we use physical particles that have mass as well as volume to represent water droplets and bubbles. Compared to their method, our method requires far fewer particles and, as a consequence, has a negligible computational overhead.

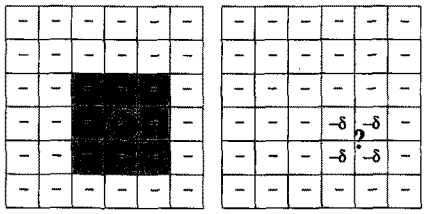

그림 2: Identification of droplets: (a) Apparent droplet, (b) Unapparent droplet. In (b), δ is a small positive value such that $\delta < \varepsilon$

to complement the grid-based framework, thereby overcoming some of the limitations of that framework.

5.1 Identification of Dissipative Cases

If the level set value of a particular cell center is positive whereas those of adjacent cell centers are negative, we can interpret the isolated region to be a droplet as shown in Fig. 2 (a). A potential problem arising from having such a small isolated region is that it may dissipate or even be lost in subsequent simulation. In such cases, therefore, we transform the region into a droplet that undergoes Lagrangian motion from that moment onwards. We determine the volume[3] of the droplet based on the level set values. The approximation we use is

$$V_f = \int_{\Omega_c} H(\phi(\mathbf{x}))d\mathbf{x} \approx \sum_c H_\varepsilon(\phi(\mathbf{x}_c))\Delta x \Delta y \Delta z, \qquad (11)$$

where the set Ω_c represents the droplet (the circular region of Fig. 2 (a)), c is the index ranging over the cells that cover Ω_c (the shaded cells of Fig. 2 (a)), and H_ε is the smeared Heaviside function

$$H_\varepsilon(\phi) = \begin{cases} 0 & : \ \phi < -\varepsilon \\ \frac{1}{2} + \frac{\phi}{2\varepsilon} + \frac{1}{2\pi}\sin(\frac{\pi\phi}{\varepsilon}) & : \ |\phi| \le \varepsilon \\ 1 & : \ \phi > \varepsilon, \end{cases} \qquad (12)$$

where $\varepsilon = 1.5\Delta x$. Since the fluid content of the isolated region is already transformed into a droplet, we reduce the level set values of the region to values that are small enough (e.g., $-\varepsilon$) that the region will not be mistakenly identified as containing water during subsequent processing. Generation of bubbles can be thought of as the dual of that of droplets; identification of isolated regions and approximation of bubble volume can be done with the same procedure and same equations, except that in this case we use the negated version of original level set values.

In addition to the cases in which the isolated region can be identified by simply looking at the signs of the level set values, referred to here as "apparent" cases, there also exist "unapparent" cases in which the isolated region cannot be identified from the sign of the level set values. For example, consider the case in which a 2×2 grid with small negative level set values is surrounded by the cells with large negative level set values, as shown in Fig. 2 (b). Since the level set function we use represents signed distance, small negative values imply that water is nearby. Therefore, the situation considered here can be reconciled only by introducing a droplet somewhere around ?-marked location of Fig. 2 (b). The volume of the droplet is again computed by Eq. (11). During dynamic simulation of a multiphase fluid, the above situation can occur when a body of water is splitting (or two bodies of air are merging). Cases of unapparent bubbles are treated similarly.

There is another source of errors of misinterpreting the phase, which is related to the back-tracking in semi-Lagrangian advection. Specifically, when a large time step is used, back-tracking may leave some cells never referred. In such cases, the masses of those cells are lost in the subsequent time steps. We prevent this type of error by converting the non-referred cells into droplets/bubbles. Again, the volumes of the cells are determined by evaluating Eq. (11) for the non-referred cells. The droplets/bubbles are then advected by the underlying velocities. Droplets/bubbles that are advected into the water/air medium are ignored.

In pure grid-based methods, the above cases are beyond the resolution limit and thus no droplet/bubble is formed. By introducing a small amount of extra computation, however, the procedures described above can generate droplets/bubbles; this not only reduces the mass dissipation but also enhances the visual details of the dynamically evolving water.

The volume approximation of Eq. (11), which uses the smeared Heaviside function given in Eq. (12), theoretically has first-order accuracy. Our experiments showed that it is computationally efficient and gives sufficient visual realism for our purposes. If greater accuracy is needed, contouring methods such as the marching cube algorithm [14] can be used.

5.2 Dynamic Simulation of Droplets/Bubbles

As a droplet/bubble advances with the initial velocity, it experiences gravitational and drag forces, as well as pressure from the surrounding fluid, which cause the fragment to accelerate or decelerate. The forces acting on the fragment can be summarized as

$$\mathbf{f} = m_f\mathbf{g} + \alpha_d r^2(\bar{\mathbf{u}} - \dot{\mathbf{x}})\|\bar{\mathbf{u}} - \dot{\mathbf{x}}\| - V_f\nabla p, \qquad (13)$$

where m_f is the mass, V_f is the volume, α_d is the drag coefficient, r is the radius, $\dot{\mathbf{x}}$ is the current velocity of the fragment, and $\bar{\mathbf{u}}$ is the interpolated velocity of the grid-based fluid measured at the center of the fragment. The third term, which represents the force due to the pressure difference, produces buoyancy. In Eq. (13), the second and third terms model the interaction with the neighboring fluid. Therefore, the action force given in Eq. (13) must be coupled with the reaction force $-\alpha_d r^2(\bar{\mathbf{u}} - \dot{\mathbf{x}})\|\bar{\mathbf{u}} - \dot{\mathbf{x}}\| + V_f\nabla p$ acting on the grid-based fluid model. The movement of fragments over time is obtained by standard numerical integration. If

[3]Since we assume incompressibility of both media, mass-preservation is equivalent to volume-preservation.

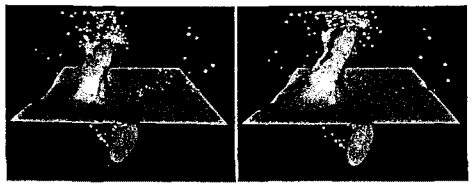

그림 3: Instantaneous hollows and ripples on the water surface created by restitution of Lagrangian droplets/bubbles

two or more fragments overlap during the dynamic simulation, they are merged into a single larger fragment.

5.3 Restitution of Droplets/Bubbles

When either of the following two conditions are met, we restitute the droplets/bubbles to the grid-based fluid model: (1) when the volume of a fragment becomes larger than twice the cell size, or (2) when a fragment hits the surface or moves into the same phase fluid.

In the first case, the fragment has become too big to be represented as a non-deforming particle, and thus its behavior is better modeled within the grid-based framework. Therefore, in such cases we remove the fragment and restitute its mass to the grid-based fluid model. This restitution is executed by updating the level set values and setting the grid point velocities to the fragment velocity. We update the level set values by

$$\phi(\mathbf{x}_i) = s_p(r_p - |\mathbf{x}_i - \mathbf{x}_p|), \tag{14}$$

where $s_p = +1$ for the case of a water droplet and -1 for the case of an air bubble, r_p is the radius of the fragment, $\mathbf{x}_p$ is the center of the fragment, and $\mathbf{x}_i$ is the grid point being updated.

The second case corresponds to the situation in which a droplet/bubble returns to the body of water/air, and therefore we remove the fragment. In the case where the fragment hits a surface cell, the cell velocity is updated by taking the average of the previous cell velocity and the fragment velocity. As for the level set values, we determine the new values for the cells covering the fragment by taking the inverse functions of Eq.s (11) and (12). In the case where the fragment moves into the same phase fluid, we perform the same procedure as described above, pretending that it hit a surface cell.

The above procedure we devised for updating the level set values and cell velocities, interestingly, contributes to creating visual details at water surface. The procedure in fact forms small ripples: A droplet falling into water contributes a downward velocity to the cell, which generates an instantaneous hollow. But soon, the region is pushed back and forms a small bump (See Fig. 3.).

6. Experimental Results

The technique presented in this paper is implemented in two dimensional and three dimensional versions, on a PC with an Intel® Pentium®4 3.2GHz processor, 1GB memory. The innovations introduced in the present work make possible the real-time simulation of the movement of 2D water. Moreover, the fast simulation speed of the proposed method enabled highly interactive control of the water motion through several types of action; for example, by clicking or dragging the mouse, we could add a volume of water, create/remove solid walls, or introduce force fields, all with immediate effect. This section summarizes our experimental results.

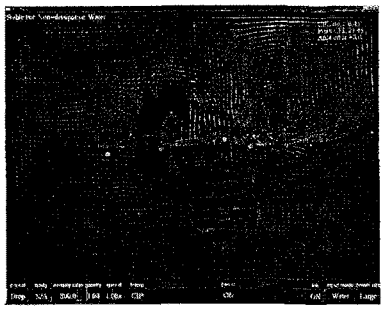

그림 4: Real-time 2D Simulation

The 2D simulator based on a grid of resolution 64×48 ran at $30 \sim 60$ fps, which is sufficiently fast to enable real-time simulation of water that is subjected to interventions such as applying forces or adding water volumes. Clip 1 contains a real-time video capture taken during the simulation (see Fig. 4 for a snapshot). Although the CFL number (i.e. $\mathbf{u}\Delta t/\Delta x$) was 25 or higher, the simulation ran stably. The mass error is not evident, and the movement of water is clearly non-dissipative. We were able to let ink mix over the water medium, as shown in [23], which provided a perceptual cue on the movement of water. The clip shows (1) the movement of water at different gravitational conditions, (2) the generation of droplets/bubbles in turbulent water, (3) the behavior of water when interacting with solids, (4) changes in buoyancy due to changes in the density of the fluid, (5) the result of replacing the CIP advection with linear advection, which noticeably increases dissipation and damping, and (6) the interactions water makes with non-void air due to our multiphase implementation of the fluid.

We ran the 3D version of our simulator to experiment the cases shown in Figs. 5, 6, and 7, and produced the animations shown in Clips 2, 3, and 4, respectively. In the case shown in Fig. 5, a football was thrown into water with a large initial velocity, which realistically produced a violent splash of water and a trail of air bubbles behind the ball. The density ratio of the ball relative to water was 0.75. In the case shown in Fig. 6, a cubic container containing water and an object was rotated. The collision the water made with the wall created highly dynamic water surface and plenty of droplets and bubbles. In the case shown in Fig. 7, an empty cup was drowned into water, which caused the air to be released and to produce a dynamic response with water. If only water-phase was simulated, water would fill the

그림 5: A football thrown into water

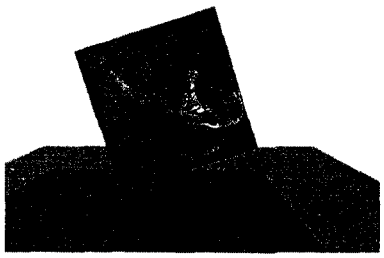

그림 6: Simulation of water in a rotating cubic container

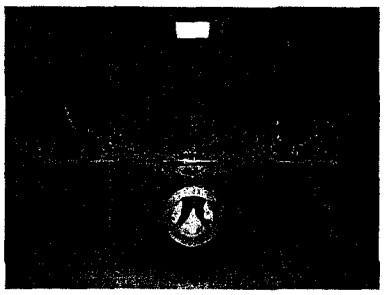

그림 7: A cup drowned into water

cup without experiencing any resistance, thus the cup would submerge smoothly, which is not physically correct. We constrain all fluid fragments to be within the fluid simulation domain. So, the droplets getting out of the vertical boundaries (the walls and the front transparent glass) are stopped and then slide down along the surface of the wall. The simulations were performed on a $80 \times 80 \times 80$ grid. Simulation of one time step took about $30 \sim 40$ seconds. A fixed time step of $\Delta t = \frac{1}{30}$ second was used in all the above 2D and 3D simulations, except for the case of football. As for the football example, in order to obtain a reasonable visual quality, we adjusted Δt so that the CFL number does not exceed 5.0. Extraction of water surface was done using the marching cube algorithm [14], and rendering was done by Mental Ray®.

7. Conclusion

The introduction to the graphics community of the semi-Lagrangian methodology by Stam in 1999 opened the way for the interactive manipulation/control of gaseous fluids. Since then, it has been an open problem to extend the technique to water. In this paper we have proposed a solution to this problem.

The problem mainly consisted of two technical challenges: (1) modeling a multiphase fluid, and (2) finding a dissipation-free technique. In this paper, the problem of modeling a multiphase fluid was solved by combining the Navier–Stokes equations with the level set method. The problem of preventing dissipation was solved by two means: (1) adoption of the CIP method, which could model the subcell-level details with third-order accuracy; (2) development of a particle-based method to prevent dissipation in small-scale features such as droplets/bubbles.

Instead of simulating water in a void space, the multiphase fluid formulation proposed here properly simulates the dynamic movement of water while it engages in complex interactions with surrounding air. Due to the measures taken to prevent dissipation in the proposed method, the simulated water does not unnecessarily lose volume or its motion is not damped to an unphysical extent.

Although this paper proposes makes a significant improvements in modelling non-dissipative behavior of water, the focus has not been put on producing accurate shape of water surface. When an application requires more accurate modeling of water surface, then the particle level set method [3] can be adopted to the framework we propose, but with an increased amount of computation. As for modeling droplets/bubbles, to our knowledge, the present work is the first that proposed a mechanism for two-way exchange of masses between the grid-based framework and particles for the conservation of mass and momentum. More accurate modeling of the the geometrical shape and size of droplets/bubbles needs further study.

Appendix: Monotonic CIP

In this appendix, we develop a method for modifying ϕ_i' and ϕ_{i+1}' such that the profile becomes monotonic. If, for simplicity, the

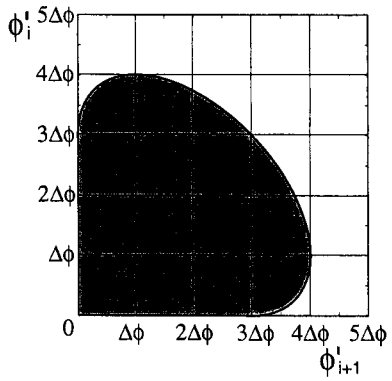

그림 8: Plot of (ϕ'_i, ϕ'_{i+1}): the shaded region guarantees monotonic profiles in the case of $\Delta\phi \geq 0$

grid size $(x_{i+1} - x_i)$ is 1, differentiation of Eq. (4) produces

$$\Phi'(X) = (3X^2 - 4X + 1)\phi'_i + (3X^2 - 2X)\phi'_{i+1} - (6\Delta\phi)X^2 + (6\Delta\phi)X. \quad (15)$$

When $\Delta\phi \geq 0$, the necessary and sufficient condition for the profile to be monotonically increasing inside the cell is $\Phi'(X) \geq 0$. By manipulating Eq. (15), we found that this condition can be reduced to

$$[\ \phi'_i \geq 0,\quad \phi'_{i+1} \geq 0,\quad \phi'_i + \phi'_{i+1} \leq 3\Delta\phi\ ]\ \ OR$$
$$[\ 9\Delta\phi - 6\Delta\phi(\phi'_i + \phi'_{i+1}) + (\phi'_i + \phi'_{i+1})^2 - \phi'_i\phi'_{i+1} \leq 0\ ],$$

which corresponds to the shaded region in Fig. 8 [10]. Similarly, when $\Delta\phi < 0$, the necessary and sufficient condition for the profile to be monotonically decreasing is

$$[\ \phi'_i \leq 0,\quad \phi'_{i+1} \leq 0,\quad \phi'_i + \phi'_{i+1} \geq -3\Delta\phi\ ]\ \ OR$$
$$[\ 9\Delta\phi + 6\Delta\phi(\phi'_i + \phi'_{i+1}) + (\phi'_i + \phi'_{i+1})^2 - \phi'_i\phi'_{i+1} \leq 0\ ].$$

Therefore, the monotonic CIP technique works in the following way: when (ϕ'_i, ϕ'_{i+1}) does not belong to the shaded region, we modify the values so that the tuple goes into the region. Although the modification may create a more diffusive profile than the original one, the result is oscillation-free and still has a third order accuracy in space.

참고 문헌

[1] M. Carlson, R. J. Mucha, and G. Turk. Rigid fluid: Animating the interplay between rigid bodies and fluid. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2004)*, 23(3):377–384, 2004.

[2] J. X. Chen and N. D. V. Lobo. Toward interactive-rate simulation of fluids with moving obstacles using Navier-Stokes equations. *Graphical models and image processing: GMIP*, 57(2):107–116, 1995.

[3] D. Enright, S. Marschner, and R. Fedkiw. Animation and rendering of complex water surfaces. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2002)*, 21(3):736–744, 2002.

[4] R. Fedkiw, J. Stam, and H. W. Jensen. Visual simulation of smoke. *Computer Graphics (Proc. ACM SIGGRAPH 2001)*, 35:15–22, 2001.

[5] B. E. Feldman, J. F. O'Brien, and O. Arikan. Animating suspended particle explosions. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2003)*, 22(3):708–715, 2003.

[6] N. Foster and R. Fedkiw. Practical animation of liquids. *Computer Graphics (Proc. ACM SIGGRAPH 2001)*, 35:23–30, 2001.

[7] N. Foster and D. Metaxas. Realistic animation of liquids. *Graphical models and image processing: GMIP*, 58(5):471–483, 1996.

[8] N. Foster and D. Metaxas. Controlling fluid animation. In *Computer Graphics International 97*, pages 178–188, 1997.

[9] N. Foster and D. Metaxas. Modeling the motion of a hot, turbulent gas. *Computer Graphics (Proc. ACM SIGGRAPH '97)*, 31(Annual Conference Series):181–188, 1997.

[10] F. Fritsch and R. Carlson. Monotone piecewise cubic interpolation. *SIAM J. Numer. Anal.*, 17:238–246, 1980.

[11] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The John Hopkins Univserity Press, 1996.

[12] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, 8(12):2182–2189, 1965.

[13] M. Kass and G. Miller. Rapid, stable fluid dynamics for computer graphics. *Computer Graphics (Proc. ACM SIGGRAPH '90)*, 24(4):49–57, 1990.

[14] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics (Proc. ACM SIGGRAPH '87)*, 21(4):163–169, 1987.

[15] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2004)*, 23(3):457–462, 2004.

[16] A. McNamara, A. Treuille, Z. Popović, and J. Stam. Fluid control using the adjoint method. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2004)*, 23(3):449–456, 2004.

[17] G. Miller and A. Pearce. Globular dynamics: A connected particle system for animating viscous fluids. *Computers and Graphics*, 13(3):305–309, 1989.

[18] J. O'Brien and J. Hodgins. Dynamic simulation of splashing fluids. In *Proceedings of Computer Animation 95*, pages 198–205, 1995.

[19] T. G. G. A. W. B. J. F. O'Brien. A method for animating viscoelastic fluids. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2004)*, 23(3):463–468, 2004.

[20] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based in hamilton-jacobi formulations. *J. Comp. Phys.*, 79:12–49, 1988.

[21] S. Premože, T. Tasdizen, J. Bigler, A. Lefohn, and R. T. Whitaker. Particle-based simulation of fluids. In *Eurographics 2003 proceedings*, pages 401–410. Blackwell Publishers, 2003.

[22] N. Rasmussen, D. Q. Nguyen, W. Geiger, and R. Fedkiw. Smoke simulation for large scale phenomena. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2003)*, 22(3):703–707, 2003.

[23] J. Stam. Stable fluids. *Computer Graphics (Proc. ACM SIGGRAPH '99)*, 33(Annual Conference Series):121–128, 1999.

[24] J. Stam and E. Fiume. Depicting fire and other gaseous phenomena using diffusion processes. *Computer Graphics (Proc. ACM SIGGRAPH '95)*, 29(Annual Conference Series):129–136, 1995.

[25] A. Staniforth and J. Côtè. Semi-lagrangian integration scheme for atmospheric model - a review. *Mon. Weather Rev.*, 119(12):2206–2223, 1991.

[26] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *J. Comp. Phys.*, 114:146–159, 1994.

[27] T. Takahashi, H. Fujii, A. Kunimatsu, K. Hiwada, T. Saito, K. Tanaka, and H. Ueki. Realistic animation of fluid with splash and foam. In *Eurographics 2003 proceedings*, pages 391–400. Blackwell Publishers, 2003.

[28] D. Terzopoulos, J. Platt, and K. Fleischer. Heating and melting deformable models (from goop to glop). In *Proceedings of Graphics Interface '89*, pages 219–226, 1989.

[29] A. Treuille, A. McNamara, Z. Popović, and J. Stam. Keyframe control of smoke simulations. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2003)*, 22(3):716–723, 2003.

[30] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[31] F. Xiao, T. Yabe, and T. Ito. Constructing oscillation preventing scheme for advection equation by rational function. *Comp. Phys. Comm.*, 93:1–12, 1996.

[32] T. Yabe and T. Aoki. A universal solver for hyperbolic equations by cubic-polynomial interpolation i. one-dimensional solver. *Comp. Phys. Comm.*, 66:219–232, 1991.

[33] T. Yabe, F. Xiao, and T. Utsumi. The constrained interpolation profile method for multiphase analysis. *J. Comp. Phys.*, 169:556–593, 2001.