Simulation of Bubbles in Foam With The Volume Control Method

Byungmoon Kim
Georgia Tech, NVIDIA
bmkim@cc.gatech.edu

Yingjie Liu
Georgia Tech
yingjie@math.gatech.edu

Ignacio Llamas
NVIDIA
llamas@nvidia.co m

Xiangmin Jiao
Georgia Tech
jiao@gatech.edu

Jarek Rossignac
Georgia Tech
jarek@cc.gatech.edu

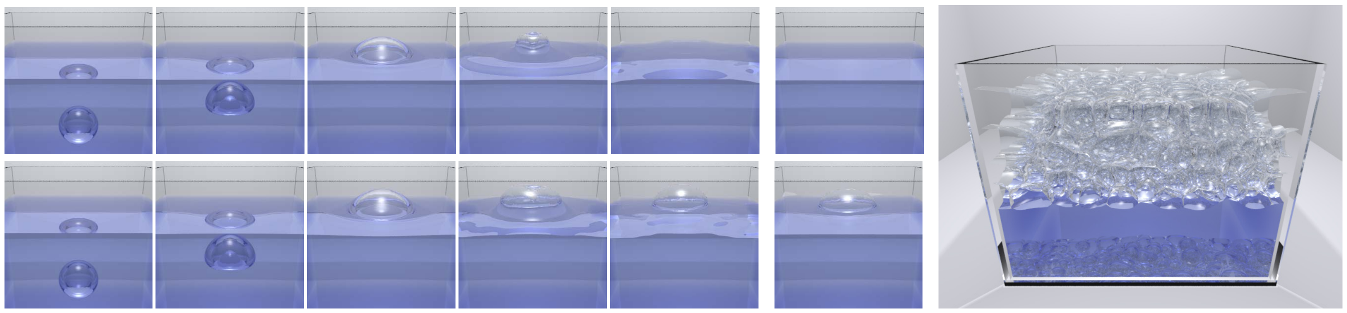

Figure 1: *When the level set is advected by the BFECC [Dupont and Liu 2003] method, the simulation of a rising bubble produces volume loss (top). When the proposed volume control method is used, the volume of bubble is preserved regardless of the length of the simulation (bottom). From left to right, each column shows the bubble at $t = 0$, 0.0625, 0.125, 0.25, 0.5, and 10.0 second. The image on the far right shows a foam structure obtained after raising more than 400 bubbles.*

Abstract

Liquid and gas interactions often produce bubbles that stay for a long time without bursting on the surface, making a dry foam structure. Such long lasting bubbles simulated by the level set method can suffer from a small but steady volume error that accumulates to a visible amount of volume change. We propose to address this problem by using the volume control method. We track the volume change of each connected region, and apply a carefully computed divergence that compensates undesired volume changes. To compute the divergence, we construct a mathematical model of the volume change, choose control strategies that regulate the modeled volume error, and establish methods to compute the control gains that provide robust and fast reduction of the volume error, and (if desired) the control of how the volume changes over time.

1 Introduction

In real fluids, we often observe a bubble rising to the surface, and then floating on the surface. Multiple bubbles interact. If they do not burst, they will stack, forming *wet foam*. The water between those stacked bubbles will drain, leaving a micrometer-thin film of liquid between bubbles. The resulting structure is called *dry foam*. Simulation of bubbles in wet or dry foam is very challenging since foam can have a complicated liquid/gas interface. Each bubble is surrounded by several faces of thin films, which may meet at junctions, where liquid is clustered. These junctions may move, merge and split. In addition, bubbles or liquid drops can merge and split. Thus, the topology of the interface changes often. These behavior causes difficulties for simulations based on the Lagrangian and Eulerian methods. In Lagrangian methods, a thin film may be represented well but topological change is hard to handle. In Eulerian mesh, when the level set representation is used, topological changes are trivially handled, but it is hard to capture thin films. This difficulty was addressed recently by the *regional level set* method proposed in [Zheng et al. 2006], in which each bubble is implicitly associated with a region identifier and a level set function that samples distance to the bubble boundary. The regional level set method allows us to represent a thin film as the boundary between two gas regions.

Bubbles in foam or small bubbles in liquid have smooth surfaces stretched by surface tension. Since the thin film is represented by the regional level set method, the bubbles and foam can be simulated in low resolution grids such as a 64^3 grid. Unfortunately, a fluid simulation on such a low-resolution grid tends to yield slow but steady volume loss [Zheng et al. 2006].

Various chemical or physical reactions may result in volume changes of bubbles. For example, bubbles are inflated in boiling water. In addition, an animator may want to inflate fluid region as a function of time. Animations of these behaviors would require a method to change the volume of bubbles using a prescribed function of time. Therefore, our goal is not only to remove the volume error, i.e., to ensure that the volume of liquid and bubbles is preserved regardless of the simulation time (See Figs. 3 and 4), but also to simulate bubbles that change volume. We achieve these goals by using a new volume control method that applies divergence of the velocity field to fluid regions in order to compensate their volume error. For each of these regions, the divergence is computed carefully by using a model of the volume error in terms of the volume and by constructing nonlinear feedback controllers similar to the proportional and integral feedbacks so that the volume error converge to a small number or zero. We also propose to compute the controller gains using simulation parameters while ensuring fast, stable, and robust volume error correction. We validate the volume change equation and controller by several experiments.

2 Prior Arts

Significant progress has been made since the pioneering work on fluid simulation for computer graphics [Kass and Miller 1990; O'Brien and Hodgins 1995; Foster and Metaxas 1996]. The stability problem of earlier work was addressed in [Stam 1999], in which semi-Lagrangian advection and pressure projection were introduced. This solution became popular for the simulation of incompressible fluids, such as smoke [Fedkiw et al. 2001], and also for free surface flows [Foster and Fedkiw 2001; Enright et al. 2002]. A well-known drawback of the semi-Lagrangian approach [Stam 1999] is the excessive diffusion and dissipation. Several solutions were proposed such as the vorticity confinement method [Fedkiw et al. 2001], vortex particles [Selle et al. 2005], vortex fluid [Park and Kim 2005], and higher-order advection methods [Song et al. 2005; Kim et al. 2005; Selle et al. 2007].

In contrast to the simulation of gaseous phenomena, the simulation of liquids requires the representation of the complicated liquid surface. A representation that has been widely used is the level set method [Osher and Sethian 1988; Osher and Fedkiw 2002; Sethian 1999]. In [Foster and Fedkiw 2001], the level set method was used in fluid simulation to create a realistic liquid simulation. Several researchers used this level set method in [Enright et al. 2002; Losasso et al. 2004; Carlson et al. 2004; Goktekin et al. 2004; Selle et al. 2005; Enright et al. 2005] to simulated liquids without bubbles, considering air as a vacuum. In contrast, liquid and air interactions, such as bubbles, requires the variable density pressure projection method that has been broadly studied in mathematics and fluid mechanics [Sussman et al. 1994; Kang et al. 2000; Haario et al. 2004]. This variable density pressure projection has been used in graphics applications by [Song et al. 2005; Kim et al. 2005], in which splash and bubbles are simulated. The bubble simulation using Lattice-Boltzmann-Method (LBM) is studied in [Nils Thurey 2004; Pohl et al. 2004]. The particle-based methods are found in [Greenwood and House 2004; Muller et al. 2005].

Although the two-phase flow method can simulate bubbles, the practical simulation of foam, in particular, dry foam, is more challenging since the micrometer-thin liquid film is not captured by an Eulerian grid. Consequently most foam simulations have been performed by using Lagrangian methods. For example, the authors of [Kück et al. 2002] approximated foam bubbles as spheres and then studied interactions between them, but the interaction between bubbles and liquid has not been investigated. In contrast, the authors of [Takahashi et al. 2003] used particle to represent foams and simulated interactions with liquids. However, thin liquid film was not simulated. Mihalef et al [Mihalef et al. 2006] simulated boiling water. All bubbles were burst at the surface and they did not attempt to simulate stacking bubbles.

Thin liquid films become stable with surfactants. Such liquid films are governed by molecular interactions rather than by the Navier-Stokes equations. Molecular interactions such as electrostatic, van der Waals, steric, and adsorptional interactions, are the subjects of numerous engineering and scientific studies [Prud'home and Khan 1996; Exerowa and Kruglyakov 1998; Weaire and Hutzler 1999; Stubenrauch and von Klitzing 2003]. Beside these efforts to understand thin film, its simulation is rare. The authors of [Weaire and Hutzler 1999] constructed piecewise curves and patches to simulate a dry foam. The author of [Durian 1997] simulated 2D dry foam using circular bubbles. Because circular bubbles avoid the difficulties related to the thin films, it greatly simplifies the simulation of foam. A drawback of these approximating methods would be the lack of bursting, split, and interactions with liquid. These limitations result in a loss of realism. The authors of [Bazhlekov et al. 2001] simulated the behavior of a foam drop that is made of a few bubbles. They used a Lagrangian mesh to represent the liquid/gas interface. Simulations of various phenomena such as wet foam, formation of dry foam, bubble merging, and bursting were not attempted.

In computer graphics, complex bubble interactions such as merging, splitting, and bursting were simulated in Eulerian grids by using the regional level set method [Zheng et al. 2006]. Since the thin films of bubbles are represented as the boundary between two gas regions, the simulation of thin film can be performed efficiently using a low resolution grid. A remaining challenge is the volume loss of bubbles.

The volume loss property of the level set method has been addressed by various methods. Among them, the easiest way to reduce the volume loss is simply using high order advection methods such as BFECC (Back and Forth Error Compensation and Correction) [Dupont and Liu 2003; Kim et al. 2005], modified MacCormack [Selle et al. 2007], or CIP [Song et al. 2005; Takahashi et al. 2003] methods. These methods reduce the volume loss of the first-order level set methods significantly, but the volume loss may still be substantial. In contrast, the particle level set method (PLS) [Enright et al. 2002] reduces volume loss down to a very small amount. Therefore, the PLS has been broadly used in recent fluid simulations [Carlson et al. 2004; Goktekin et al. 2004; Selle et al. 2005; Enright et al. 2005; Hong and Kim 2005; Wang et al. 2005]. In addition, one can have other benefits such as increased surface detail and visual effects obtained by rendering escaped particles [Guendelman et al. 2005]. Nevertheless, the small volume loss may eventually accumulate to a visible level [Goktekin et al. 2004; Losasso et al. 2006b], especially in a long simulation. In particular, the bubbles and thin film simulated on a coarse resolution grid suffer from noticeable volume loss [Zheng et al. 2006]. Another approach is to combine the level set with the volume of fluid (VOF) method. This combined approach, called Coupled Level Set and Volume Of Fluid (CLSVOF) [Sussman 2003], does not have volume loss as the volume fractions are tracked in each cell. Using this CLSVOF method, the simulation of boiling water was performed in [Mihalef et al. 2006]. As stated in [Sussman 2003], those various Eulerian methods (CLSVOF, PLS, BFECC, MacCormack, CIP) are convergent to the correct solution as the grid is refined. Our volume control method can be used with those methods to correct the relatively large volume error of BFECC, MacCormack, and CIP, to remove the small but accumulating volume error of PLS, or to allow arbitrary volume change while using CLSVOF, PLS, BFECC, and CIP. We note that our volume control does not undermine the overall accuracy of the Navier-Stokes equation. Among those methods, the BFECC method is trivial to implement over first-order advection schemes, and our volume control method can correct the volume loss of BFECC. Therefore, we test our volume control method with BFECC.

Our approach to volume control applies divergence to inflate or deflate gas bubbles or liquid drops so that the desired volume is maintained. Similar ideas have been explored in fluid animation. The scheme called *divergence sourcing* was used in [Losasso et al. 2006a; Feldman et al. 2003]. In [Feldman et al. 2003], particle explosion is achieved by treating particles as divergence sources. In [Losasso et al. 2006a], divergence is applied to model the expansion from solid fuel to gas fuel. Our contribution is to adapt this idea to volume conservation, model the volume change equation, construct the controllers, and provide gain-synthesis methods from simulation parameters so that one can easily compute gains that provide fast and stable volume correction.

3 Fluid Simulation

We use the following Navier-Stokes equation

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla\cdot(\nabla\mathbf{u}) - \frac{1}{\rho}\nabla P + \frac{\mathbf{f}}{\rho},$$

where $\mathbf{u}$ is the velocity of fluid, t is time, ρ is the fluid density, P is the pressure variable, and $\mathbf{f}$ is external forces such as surface tension. We use the operator-splitting steps proposed in [Stam 1999].

The surface tension is implemented by using the ghost fluid method presented in [Hong and Kim 2005]. General topics on fluid simulation are summarized in [Bridson et al. 2006].

3.1 Nonstaggered Octree Grid

Even though the level set method allows simulation of a liquid's surface, the complexity of the surface that can be simulated is limited by the grid resolution. A significant improvement can be obtained by an adaptive grid such as an octree [Losasso et al. 2004; Shi and Yu 2004]. In particular, [Losasso et al. 2004] showed that detailed liquid surfaces can be simulated by using an octree grid.

We simplify the octree-grid representation used in [Losasso et al. 2004], by storing pressure and velocity at the center of octree cell. Since all values are stored at centers of the octree cells, we do not need a parallel data structure for cell corners, where velocities and level set values were stored in [Losasso et al. 2004]. In addition, velocity transfer [Guendelman et al. 2005] to the cell face is not needed either. Therefore, the implementation becomes simpler and more memory-efficient. In our experiments, the 512^3 grid with a flat water surface and 63 bubbles underneath requires 14 million octree leaves. In this case, 2.5GB of memory was used. The 1024^3 grid with five bubbles under a water surface requires 2.2GB of memory. About 65% of the memory was used for the octree. The remaining 35% was used for the symmetric pressure projection matrix and for temporary vectors needed for conjugate gradient iterations.

As proposed in [Losasso et al. 2004], we perturb the sampling points to obtain a symmetric pressure projection matrix. The two phase flow formulation is similar to [Song et al. 2005], except for the differentiation operator across different grid resolutions. Note that all our variables are defined at cell centers. Therefore, the differentiations of these variables are the same as the differentiation of pressure in [Losasso et al. 2004].

3.2 Level Set

We use the level set method to trace the interface between liquid and gas. We use the BFECC (Back and Forth Error Compensation and Correction) for the level set advection and the fast sweeping method [Tsai et al. 2001] for the redistancing.

The BFECC method applied to level set advection tends to induce high-frequency noise on the interface wherever the velocity field is not smooth. Therefore, as suggested in [Dupont and Liu 2007], we add a small amount of diffusion using the following disturbed Courant-Isaacson-Rees (CIR) [Courant et al. 1952] advection

$$\phi^{n+1} = \frac{1}{2}\left[\ \phi^n(-\mathbf{u}\Delta t + \varepsilon\mathbf{e}) + \phi^n(-\mathbf{u}\Delta t - \varepsilon\mathbf{e})\ \right],$$

where we randomly choose $\mathbf{e}$ from the four directions $\{(1,1,1),(1,-1,1),(-1,1,1),(-1,-1,1)\}$ in each time step, and use $\varepsilon = 0.2\Delta x$.

We use the regional level set method [Zheng et al. 2006] to represent multiple fluid regions. When two liquid regions are touching, we merge them into one liquid region. Therefore, a liquid drop merges with other liquid bodies. In contrast, when two gas regions are contacting, we do not merge them so that we can keep the bubble seperate. The interface between two contacting gas regions (bubbles) is treated as a thin liquid film.

When the liquid film is stretched or liquid in a film has evaporated or drained, the liquid film becomes thin. When the film becomes too thin, the thin film ruptures. In regional level set method, one can keep track of each thin film, and when one of these conditions occurs, one can make the film rupture by merging the two gas regions on the two sides of the film. In this way, the bursting and merging of bubbles can be simulated. In our experimental validation, we choose to prevent bubble merging or bursting effects, and focus instead on demonstrating the benefits of our volume control approach for long simulations where bubbles remain separated.

3.3 Interpolation of the Regional Level Set

For the advection of the level set, we use the BFECC method that calls the Courant-Isaacson-Rees (CIR) method three times and combine the results so that the spacial and temporal orders of accuracy are both increased by one. Therefore, the smallest building block of the regional level set advection is the CIR step, which backtracks along the velocity vector and then interpolates the level set value and selects the region number. We implement this interpolation in the following way. First, consider a one-dimensional interpolation problem. In Fig. 2, suppose that we are computing the level set value ϕ_1 and the region number r_1 at the blue point by interpolating two green point values $(\phi_{i,j}, r_{i,j})$ and $(\phi_{i+1,j}, r_{i+1,j})$. The interpolated values ϕ_1 and r_1 are computed as

$$\phi_1 = \begin{cases} (1-m)\phi_{i,j} + m\phi_{i+1,j} & , \text{ if } r_{i,j} = r_{i+1,j} \\ |(1-m)\phi_{i,j} - m\phi_{i+1,j}| & , \text{ if } r_{i,j} \neq r_{i+1,j} \end{cases}$$

$$r_1 = \begin{cases} r_{i,j} & , \text{ if } (1-m)\phi_{i,j} \geq m\phi_{i+1,j} \\ r_{i+1,j} & , \text{ if } (1-m)\phi_{i,j} < m\phi_{i+1,j} \end{cases}$$

Notice that since the region number plays the role of the sign bit of the traditional level set, the level set value ϕ is nonnegative everywhere.

For bilinear interpolation, we first interpolate yellow points to ϕ_2 and r_2 and then interpolate along vertical dotted line to values at the red point as illustrated in Fig. 2. The trilinear interpolations can be easily obtained by extending this 2D example to 3D.

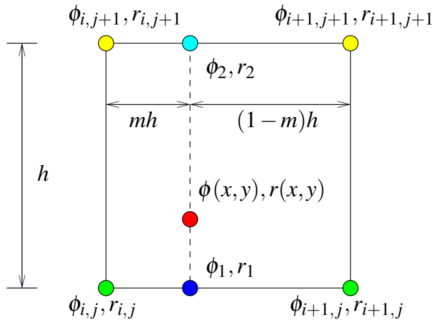

Figure 2: *Interpolation of level set value ϕ and the region number r. Notice that $\phi \geq 0$*

3.4 The Liquid Film and Surface Tension

When two gas regions are touching, there exists a thin liquid film between them. Therefore, we assign the density of the liquid on the grid location next to the interface. This liquid density is used in the variable density pressure projection.

In addition, we apply the surface tension to this thin film by using the ghost fluid method. Since the ghost fluid method requires curvature that is computed by taking the divergence of the normalized level set gradient vectors, i.e., $\nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$, differentiations over the same or different regions are needed. We perform this differentiation in the following way. Suppose that φ is the property that is about to be differentiated. Then, its derivative $\varphi_x = \frac{\partial\varphi}{\partial x}$ at the

ith grid point is computed as

$$\varphi_x = \frac{\varphi_x^+ + \varphi_x^-}{2} \quad , \quad \begin{array}{l} \varphi_x^+ = \left\{ \begin{array}{ll} (\varphi_{i+1} - \varphi_i)/h & , \text{if } r_i = r_{i+1} \\ (-\varphi_{i+1} - \varphi_i)/h & , \text{if } r_i \neq r_{i+1} \end{array} \right. \\ \varphi_x^- = \left\{ \begin{array}{ll} (\varphi_i - \varphi_{i-1})/h & , \text{if } r_i = r_{i-1} \\ (\varphi_i + \varphi_{i-1})/h & , \text{if } r_i \neq r_{i-1} \end{array} \right. \end{array} \quad , \qquad (1)$$

where h is the grid size. Notice that if $r_i = r_{i+1} = r_{i-1}$, then the above is the same as the centered difference. The derivatives along y and z axes can be computed similarly. The gradient of the level set variable ϕ computed from (1) will produce vectors pointing to the interior of each region. Therefore, at the interface, we have normalized gradient vectors $\frac{\nabla\phi}{|\nabla\phi|}$ with opposite directions. However, when we take the divergence of $\frac{\nabla\phi}{|\nabla\phi|}$, we again use (1) that reverse the direction at the neighboring cell that has a different region number. In other words, $\frac{\nabla\phi}{|\nabla\phi|}$ is locally reversed to produce a smooth vector field, and then the divergence of that vector field is computed as the curvature $\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$. The resulting vector $\kappa\frac{\nabla\phi}{|\nabla\phi|}$ computed at a grid point (i, j, k) is the curvature vector of the isosurface $\{(x, y, z)|\phi(x, y, z) = \phi_{i,j,k}\}$, which is smooth across the region boundary.

4 Controlling Fluid Volume

The volume control is used to compensate the volume loss or gain by inflating or deflating regions. The first step is to compute the volume error. Based on this volume error, the second step is to compute the desired divergence of velocity field. The final step is to apply this divergence to the velocity field in a modified pressure projection step. The regions will inflate (positive divergence) or deflate (negative divergence) to compensate the volume error.

The key challenge is to find an explicit formula, called *controller*, that computes the divergence from the volume error. The controller must be derived carefully so that the volume error is quickly corrected without losing stability. The derivation of such controller is complicated since it requires the construction and analysis of a mathematical volume change model. However, the implementation of such a controller is simple since the controller is an explicit equation. Thanks to this simplicity, the volume control is in fact a simple method. Therefore, we present the volume control algorithm without derivations in this section. Detailed derivations are provided in the appendix.

4.1 The Volume Control Method

The first step is to compute the volume of each fluid region; see section 4.2 for detail. Let V_i^n be the volume of the ith region at the nth time step, and let $\tilde{V}_i^n$ be the corresponding desired volume. When we want to preserve the initial volume throughout the simulation, $\tilde{V}_i^n = V_i^0$ for all n. In contrast, when we want to change the volume of the ith region, we can set $\tilde{V}_i^n$ to be the desired value.

After computing volumes, we compute the volume error, for which we propose to use

$$x_i^n = \frac{V_i^n - \tilde{V}_i^n}{\tilde{V}_i^n}.$$

This volume error x_i^n is the normalized difference between the current and desired volume of the ith region at the nth time step. Using this volume error x_i^n as a feedback input, we compute the divergence c_i to compensate the volume loss or gain. The required divergence c_i is computed by a nonlinear controller

$$c_i^n = -k_P \frac{x_i^n}{x_i^n + 1}, \qquad (2)$$

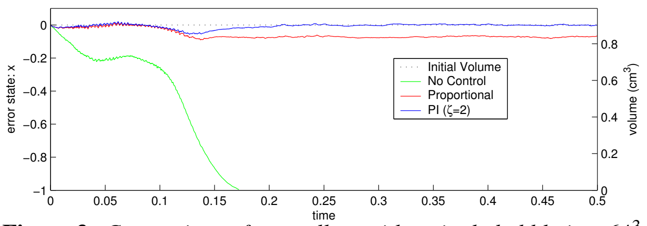

Figure 3: *Comparison of controllers with a single bubble in a 64^3 grid. The proportional controller produces somewhat large error and the PI controller improves it. The high frequency fluctuation come from the error in the volume computation. Notice that the controller is robust against this error.*

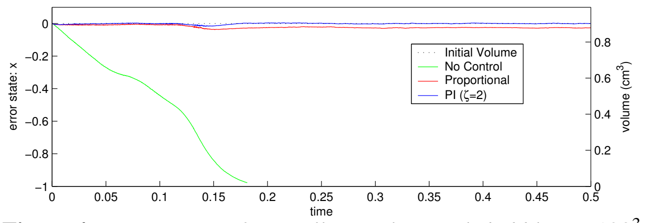

Figure 4: *Comparison of controllers with a single bubble in a 128^3 grid. The proportional controller produces a small error, and the PI controller corrects it.*

where the constant k_P is called controller gain. Note that only if this gain k_P is properly chosen, the controller (2) is able to correct the volume error stably and quickly. The computation of k_P will be discussed later in this section.

If we assume small x_i^n, we have $x_i^n + 1 \approx 1$. Therefore, $c_i^n \approx -k_P x_i^n$, which is a proportional controller. For this reason, we call (2) a *proportional controller* and call k_P *proportional gain*. This proportional controller has a small drift error as shown in Figs. 3 and 4. This drift error can be removed by integrating the volume error over time, and then using it as another feedback input: $c_i^n = \frac{1}{x_i^n+1}\left(-k_P x_i^n - k_I \sum_{m=0}^{n} x_i^m \Delta t\right)$, which is equivalent to

$$\begin{aligned} y_i^n &= y_i^{n-1} + x_i^n \Delta t \\ c_i^n &= \frac{1}{x_i^n + 1}\left(-k_P x_i^n - k_I y_i^n\right). \end{aligned} \qquad (3)$$

Since (3) is approximately a proportional-integral (PI) controller, similarly to the proportional controller case, we call (3) a *PI controller* and call k_I *PI gain*. From Fig. 3 and 4, we can observe that the drift error is eliminated by the PI controller.

We compute the divergence c_i^n for all regions at each time step for $i = 1, 2, ...$, and then apply the computed divergence value c_i^n uniformly to each region to obtain a piecewise constant divergence function c^n at the nth time step.

Once the divergence function c^n is obtained, we modify the pressure projection step so that the projected velocity field has the divergence c^n. The regions that lost volume will have positive divergence values, and hence they will be inflated. In contrast, the regions that gained volume will have negative divergence values, and therefore, they will be deflated. Let $\mathbf{u}^*$ be the velocity computed before the pressure projection is applied, and let c^n be the desired divergence in a region. The modified pressure projection projects $\mathbf{u}^*$ to a velocity $\mathbf{u}^{n+1}$ that has divergence c^n, i.e.,

$$\nabla \cdot \mathbf{u}^{n+1} = c^n \qquad \Rightarrow \qquad \nabla \cdot \frac{\nabla P}{\rho} = \frac{1}{\Delta t}\left(\nabla \cdot \mathbf{u}^* - c^n\right), \qquad (4)$$

The first order discretization of this variable-coefficient Poisson equation is provided in [Song et al. 2005]. Since c^n is simply subtracted from the divergence, the complexity of the pressure projection step is not increased.

If gain coefficients k_P or k_I are not properly chosen, the volume error will fluctuate at large magnitude, or the simulation may become unstable. Therefore, the remaining question is how to compute k_P and k_I. To answer this question, we perform further analysis on the volume change of a region under a constant divergence inside the region to construct the volume change equation. Using this equation, we show that if k_P is computed as

$$k_P = \frac{-\ln 0.1}{n_P \Delta t} = \frac{2.3}{n_P \Delta t} = \frac{2.3}{t_P}, \tag{5}$$

90% of the volume error is corrected in n_P time steps or in time $t_P = n_P \Delta t$. We typically chose $n_P = 25$. In addition, we show that if the integral gain k_I is computed as

$$k_I = \left(\frac{k_P}{2\zeta}\right)^2 \tag{6}$$

the closed loop volume equation is critically damped when $\zeta = 1$ or over damped when $\zeta > 1$. We typically chose $\zeta = 2$ in order to suppress the noise coming from the volume computation. See appendix for derivation details.

4.2 Computation of the Volume of a Region

In order to perform volume control, the volumes of fluid regions must be computed. This volume computation can be performed using various methods. First, suppose that one extracts the isosurface of a region using the marching cubes method [Lorensen and Cline 1987], and then computes the volume bounded by the isosurface. Since the marching cubes method produces piecewise linear surface, the Eucledian distance from the extracted surface to the exact isosurface (or the limit surface) is $O(\Delta x^2)$ for smooth surfaces. Let S be the surface area of the region. The volume error is bounded by $O(\Delta x^2)S = O(\Delta x^2)$. Therefore, the volume computation using marching cubes is in principle second-order accurate for smooth surfaces. Moreover, if a high-order isosurface extraction such as [Schreiner et al. 2006] is used, the volume may be computed with higher accuracy.

However, the isosurface extraction is expensive to perform at each time step. Therefore, we use a method proposed in [Kumar and Lee 2006]. Suppose that we are computing the volume of a region r. We consider the following smoothed Heaviside function $H(\phi)$

$$H(\phi) = \begin{cases} 1 & , \text{ if } \varepsilon < \Phi \\ \frac{1}{2} + \frac{3\Phi}{4\varepsilon} - \frac{\Phi^3}{4\varepsilon^3} & , \text{ if } |\Phi| < \varepsilon \\ 0 & , \text{ if } \Phi < -\varepsilon \end{cases},$$

where Φ is the signed distance function computed from the regional level set function by assigning negative signs to the level set values at cells that does not belong to the region r. Using this smoothed Heaviside function, the volume of region r is computed as $V = \int_V H(\Phi) dV$, which can be integrated by using the 8-point Gaussian quadrature rule. Given that ε is $O(\Delta x)$ and is smaller than the local feature size, this method is expected to be second-order accurate for volume computation, although a rigorous proof is not available. To verify it numerically, we conducted numerical tests to obtain the volume computation error for a sphere on grids of varying resolutions: $20^3, 40^3, 80^3$, and 160^3, and then estimate the order of accuracy by $\log(E_1/E_2)/\log(\Delta x_1/\Delta x_2)$, where $E_{1,2}$ and $\Delta x_{1,2}$ are volume errors and cell sizes of different grids. The convergence rates were 1.98, 1.90, and 2.05, and hence this method is suitable to our need.

4.3 Computation Time

The pressure projection is used to solve a matrix equation in the form $\mathbf{Qx} = \mathbf{y}$, and the volume control adds divergence to the right-hand side $\mathbf{y}$. Although the matrix remains the same, added divergence may cost more iterations in the pressure projection. Our experiments indicate that the slow down in pressure projection is negligible. The most expensive step in the volume control method was the volume computation. The time for volume computation and modified pressure projection increased the total computation time only by about $8 \sim 10\%$.

4.4 Discussions on the Order of Accuracy

An interesting question left is whether the volume control affects the overall order of accuracy or not. To answer this question, first observe that since we use the centered difference, the discretized $\nabla \cdot \mathbf{u}^*$ in (4) has an error of $O(\Delta x^2)$. Therefore, if the divergence c^n is $O(\Delta x^2)$, the order of accuracy of the $\nabla \cdot \mathbf{u}^* - c^n$ is not changed.

Notice that the overall simulation is first-order accurate due to the use of the operator splitting, and in addition, the velocity diffusion and the pressure projection steps are also first-order accurate in both space and time. This implies that $\mathbf{u}^*$ already contains an error of magnitude $O(\Delta x^2, \Delta t^2)$. Therefore, we assume that the volume has the error $V = \tilde{V} + O(\Delta x^2, \Delta t^2)$. Since $x = (V - \tilde{V})/\tilde{V} = O(\Delta x^2, \Delta t^2)$, the divergence is also $c = -k_P x/(x+1) = O(\Delta x^2, \Delta t^2)$ for a constant k_P. Note that this result is valid assuming that the volume computation error is $O(\Delta x^2)$. As discussed in section 4.2, there exist such volume computation methods. Therefore, the overall accuracy of the simulation remains first order.

5 Results

Using the volume control method, we can simulate fluids without volume loss. As shown in Figs. 3, 4, and 5, the volume loss is corrected by using volume control method, and we obtain bubbles that can last for an arbitrarily long period. In Fig. 5, we raise about 400 bubbles to form a foam, where the surface tension coefficient used is $\gamma = 0.07$ N/m. The size of the computational domain is 10cm^3, discretized by a 64^3 grid. The computation takes about 12 seconds per time step at the end when large number of bubbles are stacked. The whole simulation takes about 30 hours on a PC with a 3.4GHz Pentium 4 processor.

As discussed earlier, the volume control method is not only used to correct the volume error, but it can also be used to change the volume of fluid. In the top row of Fig. 6, we demonstrate this by inflating bubbles. We let the desired volume increase linearly over time until all the bubbles become 20 times larger than their initial volumes. As shown in Fig. 6, the volume control produces divergence that inflate bubbles to follow the desired volume. We use a 128^3 equivalent grid, and the surface tension coefficient $\gamma = 0.07$ N/m. The size of the computational domain is 10cm^3. The computation time was about 30 seconds per time step on a Pentium 4 at 2.0 GHz when the bubbles are inflated to large bubbles.

In the bottom row of Fig. 6, we drop liquid droplets on foam obtained during the inflation test. When the droplet is dropped near a vertical film, it moves down along the vertical film. When the droplet is dropped on a nearly horizontal portion of the film, it traverses it, which is the correct behavior, since drops of water pass through soap films without breaking them. The video is available from `http://www-static.cc.gatech.edu/~bmkim/volume_control.avi`.

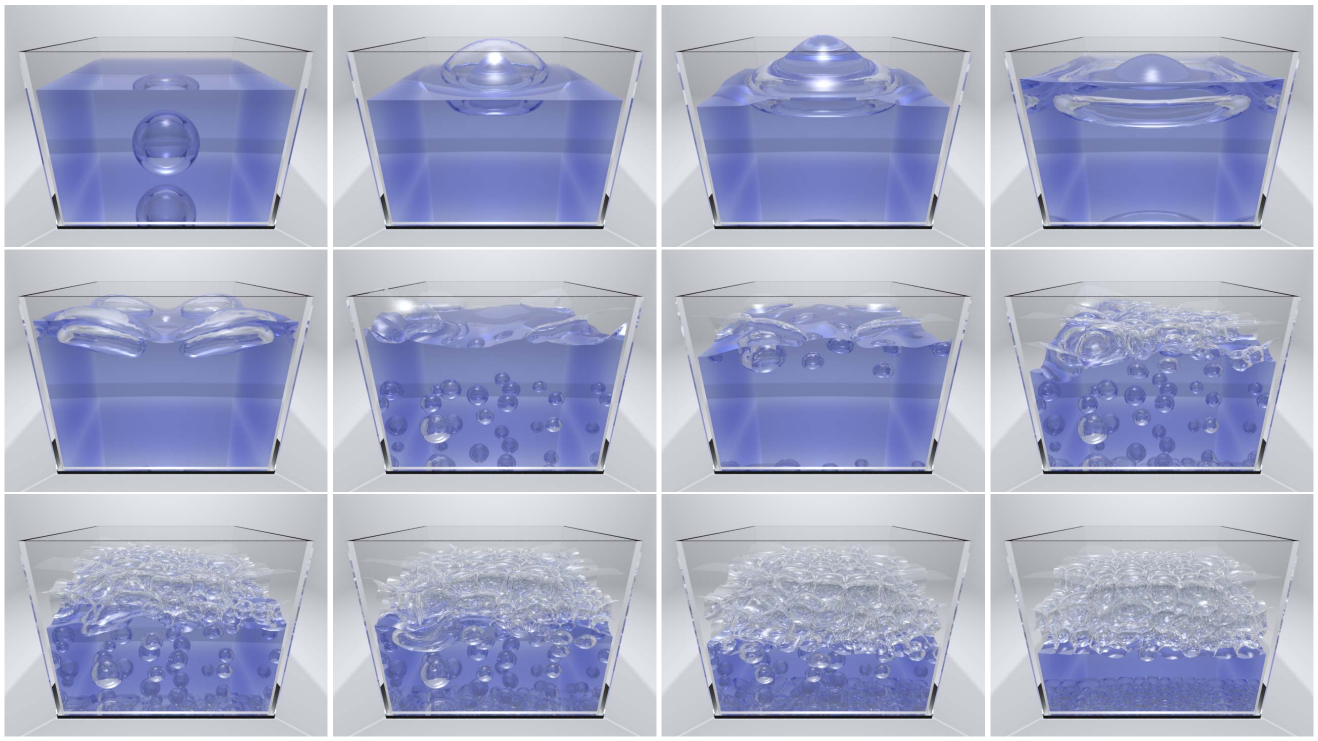

Figure 5: *Simulation of foamy bubbles. A large spherical bubble was raised. After a while, it formed a bubble ring (4th image). 400 bubbles were created in small batches at the bottom of the tank replacing water with air producing foam. The volume of each bubble is carefully maintained throughout the simulation. Since bubbles were created by replacing liquid, the liquid level was lowered. Nevertheless, by applying the volume control, the total volume of liquid and bubbles is preserved regardless of the length of the simulation.*

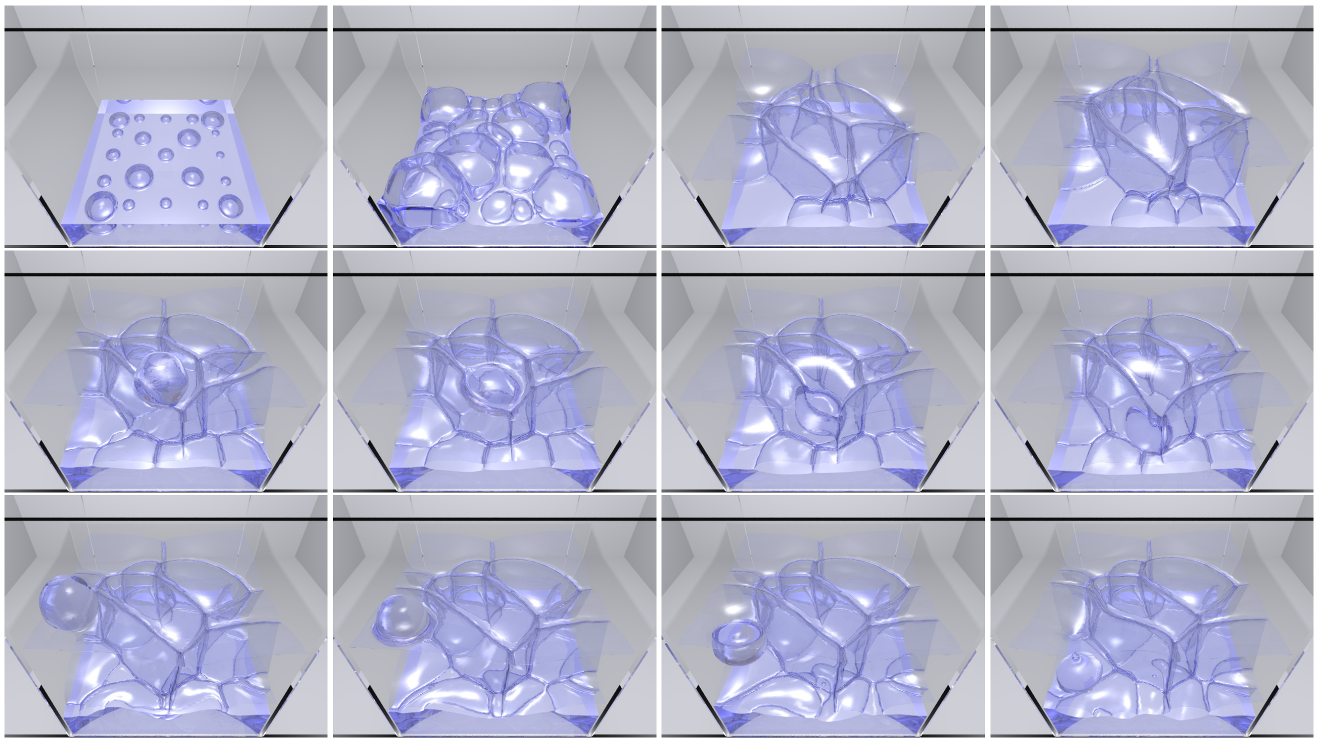

Figure 6: *Bubbles are inflated by setting their desired volumes grow as linear functions of time (the first row). When a water ball is dropped near a vertical film, the drop moves along the film (the second row). When it is dropped on a horizontal film, the drop traverses the film (the third row). The volume of water ball is maintained by the volume control method.*

6 Conclusion

We proposed a method to preserve or control the volume of fluid using the divergence as a control variable. We adopted two nonlinear control strategies, developed a method to compute control gains, and showed its effectiveness on several examples. A liquid or gas region can preserve its volume for an indefinitely long time without adversely affecting the simulation accuracy. The proposed method can be used with any previously proposed interface representation and advection methods to correct volume errors or allow arbitrary volume change.

References

BAZHLEKOV, I. B., VAN DE VOSSE, F. N., AND MEIJER, H. E. 2001. Boundary integral method for 3D simulation of foam dynamics. In *Lecture Notes in Computer Science*, vol. 2179, 401–408.

BRIDSON, R., MULLER-FISCHER, M., GUENDELMAN, E., AND FEDKIW, R. 2006. Fluid simulation. In *SIGGRAPH 2006 course notes, http://www.cs.ubc.ca/ rbridson/fluidsimulation/*.

CARLSON, M., MUCHA, P. J., AND TURK, G. 2004. Rigid fluid: Animating the interplay between rigid bodies and fluid. In *ACM SIGGRAPH*, 377–384.

COURANT, R., ISAACSON, E., AND REES, M. 1952. On the solution of nonlinear hyperbolic differential equations by finite differences. *Comm. Pure Appl. Math. 5*, 243–255.

DUPONT, T. F., AND LIU, Y. 2003. Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *Journal of Computational Physics 190*, 1, 311–324.

DUPONT, T. F., AND LIU, Y. 2007. Back and forth error compensation and correction methods for semi-lagrangian schemes with application to level set interface computations. *Math. Comp. 76*, 647–668.

DURIAN, D. 1997. Bubble-scale model of foam mechanics: Melting, nonlinear behavior, and avalanches. *Physical Review E55*, 2, 1739–1751.

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. In *ACM SIGGRAPH*.

ENRIGHT, D., LOSASSO, F., AND FEDKIW, R. 2005. A fast and accurate semi-lagrangian particle level set method. *Computers and Structures 83*, 479–490.

EXEROWA, D., AND KRUGLYAKOV, P. M. 1998. *Foams and Foam Films*. Elsevier, ISBN 0-444-81922-3.

FEDKIW, R., STAM, J., AND JENSEN, H. 2001. Visual simulation of smoke. In *ACM SIGGRAPH*, 23–30.

FELDMAN, B. E., O'BRIEN, J. F., AND ARIKAN, O. 2003. Animating suspended particle explosions. In *ACM SIGGRAPH*, 708–715.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. In *ACM SIGGRAPH*, 15–22.

FOSTER, N., AND METAXAS, D. 1996. Realistic animation of liquids. *Graphical Models and Image Processing 58*, 5, 471–483.

GOKTEKIN, T. G., BARGTEIL, A. W., AND O'BRIEN, J. F. 2004. A method for animating viscoelastic fluids. In *ACM SIGGRAPH*, 463–468.

GREENWOOD, S., AND HOUSE, D. 2004. Better with bubbles: Enhancing the visual realism of simulated fluid. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 287–296.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. In *ACM SIGGRAPH*.

HAARIO, H., KOROTKAYA, Z., LUUKKA, P., AND SMOLIANSKI, A. 2004. Computational modelling of complex phenomena in bubble dynamics: Vortex shedding and bubble swarms. In *Proceedings of ECCOMAS 2004*.

HONG, J.-M., AND KIM, C.-H. 2005. Discontinuous fluids. In *ACM SIGGRAPH*.

KANG, M., FEDKIW, R. P., AND LIU, X.-D. 2000. A boundary condition capturing method for multiphase incompressible flow. *Journal of Scientific Computing 15*, 3 (Sep).

KASS, M., AND MILLER, G. 1990. Rapis stable fluid dynamics for computer graphics. In *ACM SIGGRAPH*, 49–57.

KIM, B., LIU, Y., LLAMAS, I., AND ROSSIGNAC, J. 2005. FlowFixer: Using BFECC for fluid simulation. In *Eurographics Workshop on Natural Phenomena*.

KÜCK, H. VOGELGSANG, C., AND GREINER, G. 2002. Simulation and rendering of liquid foams. In *Proceedings of Graphics Interface*, 81–88.

KUMAR, A. V., AND LEE, J. 2006. Step function representation of solid models and application to mesh free engineering analysis. *Journal of Mechanical Design 128*, 46–56.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3D surface construction algorithm. In *ACM SIGGRAPH*, 163–169.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating water and smoke with an octree data structure. In *ACM SIGGRAPH*, 457–462.

LOSASSO, F., IRVING, G., GUENDELMAN, E., AND FEDKIW, R. 2006. Melting and burning solids into liquids and gases. *IEEE Transactions on Visualization and Computer Graphics 12*, 343–353.

LOSASSO, F., SHINAR, T., SELLE, A., AND FEDKIW, R. 2006. Multiple interacting liquids. In *ACM SIGGRAPH*, 812–819.

MIHALEF, V., UNLUSU, B., SUSSMAN, M., AND METAXAS, D. 2006. Physically based boiling simulation. In *ACM Siggraph/Eurographics Symposium in Computer Animation*.

MULLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-based fluid-fluid interaction. In *Proceedings of SIGGRAPH'05 Symposium on Computer Animation*, 237–244.

NILS THUREY, U. R. 2004. Free surface lattice-boltzmann fluid simulations with and without level sets. In *Workshop on Vision, Modeling, and Visualization*.

O'BRIEN, J., AND HODGINS, J. 1995. Dynamic simulation of splashing fluids. In *IEEE Computer Animation*, 198–205.

OSHER, S. J., AND FEDKIW, R. P. 2002. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, ISBN 0-387-95482-1.

OSHER, S., AND SETHIAN, J. A. 1988. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics 79*, 12–49.

PARK, S. I., AND KIM, M. J. 2005. Vortex fluid for gaseous phenomena. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

POHL, T., DESERNO, F., THUREY, N., RUDE, U., LAMMERS, P., WELLEIN, G., AND ZEISER, T. 2004. Performance evaluation of parallel large-scale lattice boltzmann applications on three supercomputing architectures. In *Supercomputing, 2004. Proceedings of the ACM/IEEE SC2004 Conference*.

PRUD'HOME, R. K., AND KHAN, S. A. 1996. *Foams: Theory, Measurements, and Applications*. Marcel Dekker, Inc, ISBN 0-8247-9395-1.

SCHREINER, J., SCHEIDEGGER, C., AND SILVA, C. 2006. High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics 12*, 5, 1205–1212.

SELLE, A., RASMUSSEN, N., AND FEDKIW, R. 2005. A vortex particle method for smoke, water and explosions. In *ACM SIGGRAPH*.

SELLE, A., FEDKIW, R., KIM, B., LIU, Y., AND ROSSIGNAC, J. 2007. An unconditionally stable MacCormack method. *J. Sci. Comput. (in review)*.

SETHIAN, J. A. 1999. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, ISBN 0-521-64557-3.

SHI, L., AND YU, Y. 2004. Visual smoke simulation with adaptive octree refinement. In *Computer Graphics and Imaging*.

SHINNERS, S. M. 1978. *Modern Control System Theory and Application*. Addison Wesley.

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

STAM, J. 1999. Stable fluids. In *ACM SIGGRAPH*, 121–128.

STUBENRAUCH, C., AND VON KLITZING, R. 2003. Disjoining pressure in thin liquid foam and emulsion films - new concepts and perspective. *Journal of Physics Condensed Matter 15*, 3-4, 173–181.

SUSSMAN, M., SMEREKA, P., AND OSHER, S. 1994. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics 114*, 1, 146–159.

SUSSMAN, M. 2003. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *Journal of Computational Physics 187*, 110–136.

TAKAHASHI, T., FUJII, H., KUNIMATSU, A., HIWADA, K., SAITO, T., TANAKA, K., AND UEKI, H. 2003. Realistic animation of fluid with splash and foam. In *EUROGRAPHICS*, vol. 22.

TSAI, Y.-H. R., CHENG, L.-T., OSHER, S., AND ZHAO, H.-K. 2001. Fast sweeping algorithms for a class of hamilton-jacobi equations. Tech. Rep. 01-27, University of California, Los Angeles, http://www.math.ucla.edu/applied/cam/index.html.

WANG, H., MUCHA, P. J., AND TURK, G. 2005. Water drops on surfaces. In *ACM SIGGRAPH*.

WEAIRE, D., AND HUTZLER, S. 1999. *The Physics of Foams*. Oxford, ISBN 0-19-851097-7.

ZHENG, W., YONG, Y.-H., AND PAUL, J.-C. 2006. Simulation of bubbles. In *ACM Siggraph/Eurographics Symposium in Computer Animation*.

7 Acknowledgement

This work was supported by the NSF under the ITR Digital clay grant 0121663 and the NSF DMS-0511815.

8 Appendix

8.1 The Volume Change Equation

When the level set is advected by BFECC, a fluid region, such as a gas bubble, gains or loses little volume per time step. It typically takes almost 500 time steps before a bubble loses half of its volume in a 128^3 grid. In other words, the volume change is not a high frequency behavior. This slow dynamic nature implies that a derivative feedback is not necessary and a proportional feedback would suffice. To design an appropriate controller, we must first construct a mathematical model of the volume change. Using this model, we will develop controllers and compute control gains so that the modeled volume change is corrected.

Recall that the volume error is defined as $x_i^n = \frac{V_i^n - \tilde{V}_i^n}{\tilde{V}_i^n}$. Since modeling how the x_i^n changes at each discrete time step is difficult as the volume changes as a result of complex simulation procedure, we use a continuous time model, i.e., we use $V_i = V_i(t), x_i = x_i(t)$ and $c_i = c_i(t)$ instead of V_i^n, x_i^n and c_i^n. The rate of volume change is computed using the divergence theorem as

$$\dot{V}_i = \int_{S_i} \mathbf{u} \cdot \mathbf{n} dS \ = \ \int_{V_i} \nabla \cdot \mathbf{u} dV \ = \ c_i V_i,$$

where S_i is the boundary of the fluid volume V_i, and $\mathbf{n}$ is the unit outer normal vector to the boundary. Therefore, ideally, the volume should not change when $c_i = 0$, but simulation experiments show a volume loss due to the numerical errors in the level set advection and the pressure projection steps. After extensive experiments, we conclude that this volume loss can be modeled by the following equation using an unknown factor $\tilde{b}$

$$\dot{V}_i = c_i V_i + \tilde{b}. \tag{7}$$

When $c_i = 0$, simulation experiments in Figs. 7 and 8 show that the volume graph has large nearby linear segments connected by curved transitions. In each linear segment, the slope is constant, indicating piecewise constant $\tilde{b}$. Rewriting (7) in terms of x_i, we obtain

$$\dot{x}_i \ = \ \frac{\dot{V}_i}{\tilde{V}_i} \ = \ \frac{c_i V_i + \tilde{b}}{\tilde{V}_i} \ = \ c_i(x_i + 1) + b, \tag{8}$$

where $b = \tilde{b}/\tilde{V}$.

The unknown factor b primarily depends on the level set advection method, but it also depends on simulation parameters such as the time step Δt, the cell size Δx, the sizes of bubbles or liquid drops, the surface tension, and others. Figures 7 and 8 show effects of surface tension and the grid size. In addition, b changes during the simulation. For example, b is large when a liquid film is formed. Finally, from Figs. 7 and 8, we can observe that b changes slowly.

8.2 A Proportional Feedback Controller

The volume loss observed in the previous section can be compensated by adding a divergence c_i. We set $\nabla \cdot \mathbf{u} = c_i$ inside each region. Different regions have different values of c_i. We first consider the feedback

$$c_i = -k_p x_i \ / \ (x_i + 1). \tag{9}$$

This is a nonlinear feedback, but since $|x_i| \ll 1$, $c_i \approx -k_p x_i$. Due to the similarity with the proportional feedback $-k_p x_i$, (9) will be called the *pseudo proportional feedback* or *proportional feedback* for simplicity.

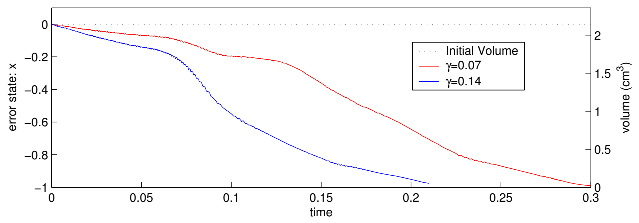

Figure 7: *The volume loss rate b of a rising bubble strongly depends on the surface tension ($c_i = 0$).*

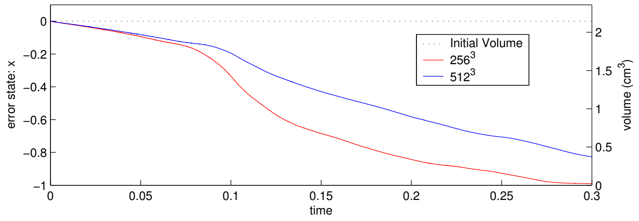

Figure 8: *The volume loss rate b of a rising bubble depends on the mesh resolution ($c_i = 0$).*

Plugging (9) into (8), we obtain a linear equation

$$\dot{x}_i + k_p x_i \ = \ b, \tag{10}$$

which has the following explicit solution

$$x_i(t) = x_i(t_i) e^{-k_p(t-t_i)} + \frac{b}{k_p}\left(1 - e^{-k_p(t-t_i)}\right),$$

where t_i is the time when the i^{th} region is created. If $k_p > 0$, this explicit solution shows that the error $x_i(t)$ does not converge to zero, but

$$x_i(t) \to b \ / \ k_p \quad \text{as} \quad t \to \infty. \tag{11}$$

This steady-state error b/k_p is the drift error we observed with the proportional feedback in the rising bubble experiment in Figs. 3 and 4. This is due to the fact that a rising bubble loses volume until the loss $V_i - \tilde{V}_i$ is large enough for the proportional controller to produce large enough divergence c_i. As a result, the volume of a rising bubble tends to saturate to a slightly smaller volume. However, the amount of shrinkage is unknown since it is a function of many different simulation methods and parameters.

8.2.1 Computing Proportional Gain k_p

The equation (11) shows that the volume error can be reduced down to b/k_p, which may be very small if k_p is sufficiently large. With this observation, a natural approach to compute k_p is first to define an error tolerance ε_v, and then computing the gain as $k_p = |b|/\varepsilon_v$ so that the steady state error is smaller than the tolerance. Since b is unknown, one may estimate b from a number of experiments. However, as shown in the previous section, b is a function of many factors and it is hard to estimate. More importantly, if the error tolerance ε_v is arbitrarily small, the gain $k_p = |b|/\varepsilon_v$ will be arbitrarily large. This will produce an arbitrarily large divergence input $c_i = -k_p x_i$, which can halt the simulation if the time step is not sufficiently small. Therefore, we do not compute k_p from the steady-state error tolerance.

Instead, we use the *rising time* criterion to design the gain k_p. We first assume that a fluid region has a volume error x. The user specifies the number of time steps n_p, during which the volume error x is reduced down to $x/10$. The time $n_p \Delta t$ is called the *rising time*. The use of rising time criterion provides an important advantage. Since

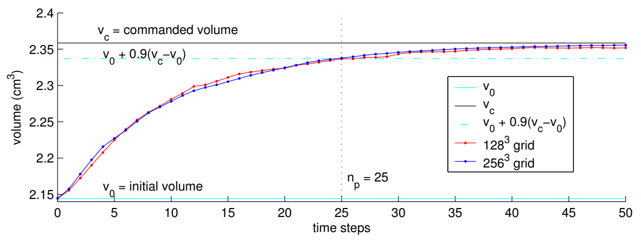

Figure 9: *Step responses of the volume of a bubble show that 90% of the volume error is indeed corrected in about n_P steps.*

we specify the number of time steps n_P, the resulting divergence input will try to fix the volume error in n_P time steps. For example, if we set $n_P = 25$, the volume error will be reduced down to 10% in $25\Delta t$ seconds.

Notice that instabilities may occur if n_P is too small. For example, if $n_P < 1$, the divergence input will be large and the volume of the region may grow or shrink more than necessary in a single time step, making the simulation unstable.

Now, we derive k_P. As will be shown in the following experiments, the steady state volume error is small. Thus, b in (10) can be ignored for the controller construction. Ignoring b, the volume error evolves in the following discrete form

$$x_i^n = \left(e^{-k_P\Delta t}\right)^n x_i^0,$$

where x_i^n is the relative volume error of the i th region in the n th time step. Since n_P is the number of time steps required to reduce the initial error x_i^0 down to 10%, the proportional gain k_P is computed from the condition $\left(e^{-k_P\Delta t}\right)^{n_P} = 0.1$ as (5).

To validate the volume change equation and the proposed controller, we set the volume error $x_i = -0.1$ and then apply the volume control. As shown in Fig. 9, the error x_i decreases towards zero in approximately n_P time steps in various grid resolutions.

8.3 A Proportional-Integral Feedback Controller

As shown in Fig. 3, the proportional control produces a small volume error that does not accumulate. In fact, the proportional control seems sufficient in most cases. For example, Fig. 4 shows that the volume error in 128^3 grid is less than 3%, when the proportional feedback with $n_P = 25$ is applied. However, in some cases, when a low resolution grid is used with large surface tension, the volume error tends to increase when the proportional controller is used, as shown in Fig. 3, where the volume error is about 10%. Although 10% may not be easily noticeable, we develop below an additional control strategy that further reduces this volume error.

In the classical control strategy [Shinners 1978], a natural choice for removing drift error is to use integral feedback, by which the small drift error will be integrated over time and then used as an additional control input. This is an efficient strategy since the small drift error can be accumulated to produce large control input. Consider the following nonlinear feedback with an integral term

$$\begin{aligned} c_i &= \left(-k_P x_i - k_I \int_{-\infty}^{t} x_i dt\right) \;/\; (x_i+1) \\ &= \left(-k_P x_i - k_I \int_{t_i}^{t} x_i dt - k_I \int_{-\infty}^{t_i} x_i dt\right) \;/\; (x_i+1), \end{aligned} \tag{12}$$

where t_i is the time when the i^{th} region, for example a bubble, is created. This is a nonlinear feedback, but since $|x_i| \ll 1$, the feedback is similar to the proportional-integral feedback

$$c_i = -k_P x_i - k_I \int_{-\infty}^{t} x_i dt.$$

Therefore, we call (12) as the proportional-integral (PI) feedback. The term $\int_{-\infty}^{t_i} x_i dt$ is computed by combining the error integrals of previous components that contributed to the i^{th} component. Using this feedback, we obtain a linear equation

$$\dot{x}_i + k_P x_i + k_I \int_{-\infty}^{t} x_i dt \;=\; b. \tag{13}$$

If we choose positive gains, i.e., $k_P > 0$ and $k_I > 0$, the volume error tends to vanish, i.e., $x_i(t) \to 0$ as $t \to \infty$. In addition, by taking the time-derivative of the solution of (13), we can show that $\dot{x}_i(t) \to 0$ as $t \to \infty$. Therefore, from (13), we obtain

$$k_I \int_{-\infty}^{t} x_i dt \;=\; b - \dot{x}_i - k_P x_i \quad \to \quad b \qquad \text{as} \;\; t \to \infty,$$

which shows that $k_I \int_{-\infty}^{t} x_i dt$ is an estimate of the unknown factor b, and that the PI-controller uses this estimate to cancel the b factor.

The next question is the computation of $\int_{-\infty}^{t_i} x_i dt$. When several regions merge or split to create new set of regions, the terms $\int_{-\infty}^{t_i} x_i dt$ in (12) of the new regions can be computed in the following way. Suppose that j_1 th, j_2 th, ..., j_n th regions at the n^{th} time step are merged to form a bigger region that is identified as the i^{th} region at the $(n+1)^{\text{th}}$ time step. To further generalize this, suppose that the volume fractions of $w_1, w_2, ..., w_n$ of each region went to the new i^{th} region. The desired volume of i^{th} region is $\tilde{V}_i = \sum_{k=1}^{n} w_{j_k} \tilde{V}_{j_k}$, and the current volume is $V_i = \sum_{k=1}^{n} w_{j_k} V_{j_k}$. The error integral of the new region $\int_{-\infty}^{t_i} x_i dt$ is computed from the error integrals of the old regions $\int_{-\infty}^{t_{j_k}} x_{j_k} dt, k = 1, 2..., n$ as

$$\int_{-\infty}^{t_i} x_i dt = \int_{-\infty}^{t_i} \frac{V_i - \tilde{V}_i}{\tilde{V}_i} dt = \frac{1}{\tilde{V}_i} \sum_{k=1}^{n} \left(w_{j_k} \tilde{V}_{j_k} \int_{-\infty}^{t_i} x_{j_k} dt \right).$$

8.3.1 Computing Integral Gain k_I

The drift error of proportional control can be removed by adding integral feedback. However, improperly chosen integral gain k_I can cause undesired oscillations in the volume, which indeed occurred in our experiments on stacked bubbles. Therefore, we propose a method to compute the gain k_I in a way that we can specify damping amount so that the unnecessary oscillation is not induced by the PI controller.

Let $y = \int_0^t x_i dt$. Then, from (13), we obtain a second order system

$$\ddot{y} \;+\; k_P \dot{y} \;+\; k_I y = 0,$$

whose natural frequency is $\omega_n = \sqrt{k_I}$ and the damping coefficient is $\zeta = \frac{k_P}{2\sqrt{k_I}}$. Our goal is now choosing good values of ζ that provides enough damping. By the classical control theory, a system that has a good balance between fast error correction and damping would have $\zeta = 0.7$, which contains a small amount of oscillation that settles down quickly. When $\zeta \geq 1$, the system is critically damped or over-damped, and therefore, oscillation in x_i does not exist. Therefore, it would be safe to choose ζ larger than 0.7. In our experiments, $\zeta = 2$ worked well. After ζ is chosen, the integral gain k_I is computed using (6).