On Boundary Condition Capturing for Multiphase Interfaces

Jeong-Mo Hong*, Tamar Shinar†, Myungjoo Kang‡, Ronald Fedkiw§

31 March 2006

Abstract

This paper begins with an overview of the *boundary condition capturing* approach to solving problems with interfaces. Although, the authors' original motivation was to extend the ghost fluid method from compressible to incompressible flow, the elliptic nature of incompressible flow quickly quenched the idea that ghost cells could be defined and used in the usual manner. Instead the boundary conditions had to be implicitly captured by the matrix formulation itself, leading to the novel approach. We first review the work on the variable coefficient Poisson equation, noting that the simplicity of the method allowed for an elegant convergence proof. Simplicity and robustness also allowed for a quick extension to three-dimensional two-phase incompressible flows including the effects of viscosity and surface tension, which is discussed subsequently. The method has enjoyed popularity in both computational physics and computer graphics, and we show some comparisons with the traditional delta function approach for the visual simulation of bubbles. Finally, we discuss extensions to problems where the velocity is discontinuous as well, as is the case for premixed flames, and show an example of multiple interacting liquids that includes all of the aforementioned phenomena.

1 Introduction

Enforcing a variety of boundary conditions or jump conditions at interfaces is important for developing accurate numerical methods in the field of computational physics. For example, Fedkiw et al. [1] developed the ghost fluid

*Computer Science Department, Stanford University
†Institute for Computational and Mathematical Engineering, Stanford University
‡Department of Mathematical Sciences, Seoul National University
§Computer Science Department, Stanford University, email:fedkiw@cs.stanford.edu

method to capture the boundary conditions at a contact discontinuity in the inviscid Euler equations. This method was later extended to treat more general compressible flow discontinuities such as shocks, detonations, and deflagrations in [2]. For a fully conservative implementation, see [3].

Motivated by the ghost fluid method, [4] developed a novel boundary condition capturing approach for solving the variable coefficient Poisson equation in the presence of interfaces where both the variable coefficients and the solution itself may be discontinuous. The method is robust and easy to implement even in three spatial dimensions. Furthermore, the coefficient matrix of the associated linear system is the standard symmetric matrix for the variable coefficient Poisson equation in the absence of interfaces allowing for straightforward application of standard "black box" solvers. A formal convergence proof was given in [5] based on the weak formulation. Both the method's simplicity and its three-dimensional formulation made it an ideal candidate for three-dimensional flows, and [6] extended the method to treat two-phase incompressible flow including the effects of viscosity, surface tension and gravity. While the traditional finite difference techniques for two-phase incompressible flow involved numerical smearing of the discontinuous quantities near the interface, see e.g. [7, 8, 9], [6] treats the interface in a sharp fashion dramatically reducing parasitic currents. Interestingly, [10] has shown that these delta function smearing techniques may lead to $O(1)$ errors in even basic computations if used in the standard fashion (see also [11, 12]). While there have been other sharp interface methods such as the immersed interface method [13], complicated implementations and issues with stability and accuracy have precluded their use in some of the more complex problems. Notably, [14] proposed a new simpler form of the immersed interface method that is quite similar to the boundary condition capturing approach of [4].

The methods proposed in [4] and [6] have also enjoyed popularity outside the computational physics community. For example, [15] independently used these techniques for the visual simulation of incompressible viscous two-phase fluids with realistic small-scale details. Although they used a slightly simplified approach, decoupling the pressure discontinuity from the viscosity discontinuity, they still obtained sharp interface profiles of solution variables and showed that the method is highly preferable to a smeared out delta function approach. Similar techniques were also used to produce realistic three-dimensional simulations of fire in [16]. This work was based on the computational physics research of [17] that extended the methods proposed in [4] and [6] to treat two-phase incompressible flow problems where one phase is being converted into another, e.g. the burning of a premixed flame or the vaporization of liquid water. Most recently, [18] combined all of these techniques into a single simulation framework for multiple interacting liquids.

2 Variable Coefficient Poisson Equation

Based on the ghost fluid method, [4] used a boundary condition capturing approach to develop a new numerical method for the variable coefficient Poisson equation in the presence of interfaces where both the variable coefficients and the solution itself may be discontinuous. The method gives a simple dimension by dimension discretization that can be readily applied in three spatial dimensions. The resulting linear system is the standard symmetric discretization for the Poisson equation, allowing for fast solvers to be used. Notably, the method maintains a sharp profile and does not suffer from numerical smearing at the interface unlike approaches based on a delta function formulation.

Consider a Cartesian computational domain, Ω, with exterior boundary, $\partial\Omega$, and a lower dimensional interface, Γ, that is defined by a level set function, ϕ, and divides the computational domain into disjoint pieces, Ω^- and Ω^+. The variable coefficient Poisson equation is given by

$$\begin{array}{ll} \nabla \cdot (\beta(\vec{x})\nabla u(\vec{x})) = f(\vec{x}), & \vec{x} \in \Omega \\ u(\vec{x}) = g(\vec{x}), & \vec{x} \in \partial\Omega \end{array} \tag{1}$$

where $\vec{x} = (x, y, z)$ are the spatial dimensions, $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$ is the gradient operator, and $\beta(\vec{x})$ is presumed to be continuous on each disjoint subdomain, Ω^- and Ω^+. The jump conditions or internal boundary conditions are specified along the interface Γ as

$$\begin{array}{ll} [u]_\Gamma = a(\vec{x}), & \vec{x} \in \Gamma \\ [\beta u_n]_\Gamma = b(\vec{x}), & \vec{x} \in \Gamma \end{array} \tag{2}$$

where

$$\begin{array}{c} [u]_\Gamma = u^+(\vec{x}) - u^-(\vec{x}) \\ [\beta u_n]_\Gamma = \beta^+(\vec{x})u_n^+(\vec{x}) - \beta^-(\vec{x})u_n^-(\vec{x}) \end{array} \tag{3}$$

specifies the direction of the jump with the "$\pm$" subscripts referring to $\Omega^\pm$. Here $u_n = \nabla u \cdot \vec{N}$ is the normal derivative of u with $\vec{N}$ the local unit normal to the interface. The use of Dirichlet boundary conditions in equation 1 is for exposition only and Neumann boundary conditions of $u_n(\vec{x}) = g(\vec{x})$ for $\vec{x} \in \partial\Omega$ could be used on the outer boundary instead.

2.1 Incorporating Jump Conditions

For simplicity, consider the one-dimensional variable coefficient Poisson equation

$$(\beta u_x)_x = f(x) \tag{4}$$

with fixed Dirichlet boundary conditions on $\partial\Omega$ and a standard second order discretization of

$$\frac{\beta_{i+\frac{1}{2}}\left(\frac{u_{i+1}-u_i}{\triangle x}\right)-\beta_{i-\frac{1}{2}}\left(\frac{u_i-u_{i-1}}{\triangle x}\right)}{\triangle x}=f_i \tag{5}$$

for each unknown u_i. At the fluxes, $\beta_{i\pm\frac{1}{2}}=\beta(x_{i\pm\frac{1}{2}})$ are defined in accordance with the side of the interface the flux is located on as determined by $\phi_{i\pm\frac{1}{2}}$.

If $x_k\in\Omega^-$ and $x_{k+1}\in\Omega^+$, one can see that $\frac{u_{k+1}-u_k}{\triangle x}$ is $O(\frac{1}{\triangle x})$, while all the other terms of the form $\frac{u_{i+1}-u_i}{\triangle x}$ are $O(1)$ and approximate the local derivative. The term $\frac{u_{k+1}-u_k}{\triangle x}$ must be modified in accordance with the jump condition in order to obtain a reasonable approximation of the derivative near the interface. In particular, one should avoid naively mixing terms from different domains. Drawing inspiration from the ghost fluid method, one can define $u^-_{k+1}=u_{k+1}-a_\Gamma$ and $u^+_k=u_k+a_\Gamma$, where

$$a_\Gamma=\frac{a_k|\phi_{k+1}|+a_{k+1}|\phi_k|}{|\phi_k|+|\phi_{k+1}|} \tag{6}$$

gives the jump at the interface. Then in the equation for grid node k, we replace u_{k+1} with u^-_{k+1}, to obtain

$$\frac{\beta_{k+\frac{1}{2}}\left(\frac{(u_{k+1}-a_\Gamma)-u_k}{\triangle x}\right)-\beta_{k-\frac{1}{2}}\left(\frac{u_k-u_{k-1}}{\triangle x}\right)}{\triangle x}=f_k. \tag{7}$$

Similarly, the equation for grid node $k+1$ is modified to be

$$\frac{\beta_{k+\frac{3}{2}}\left(\frac{u_{k+2}-u_{k+1}}{\triangle x}\right)-\beta_{k+\frac{1}{2}}\left(\frac{u_{k+1}-(u_k+a_\Gamma)}{\triangle x}\right)}{\triangle x}=f_{k+1}. \tag{8}$$

Next, we turn our attention to the derivative jump condition where the precise location of the interface plays a larger role, and thus we use

$$\theta=\frac{|\phi_k|}{|\phi_k|+|\phi_{k+1}|} \tag{9}$$

to estimate the subcell interface location. That is, the interface splits the cell into two pieces of size $\theta\triangle x$ on the left and $(1-\theta)\triangle x$ on the right. Denoting the value of u at this subcell interface location by u_I, and interpolating the derivative jump at the interface as

$$b_\Gamma=\frac{b_k|\phi_{k+1}|+b_{k+1}|\phi_k|}{|\phi_k|+|\phi_{k+1}|}, \tag{10}$$

one can discretize the derivative jump condition as

$$\beta^+ \left(\frac{u_{k+1} - u_I}{(1-\theta)\triangle x} \right) - \beta^- \left(\frac{u_I - u_k}{\theta \triangle x} \right) = b_\Gamma \tag{11}$$

and solve for u_I as

$$u_I = \frac{\beta^+ u_{k+1}\theta + \beta^- u_k(1-\theta) - b_\Gamma \theta(1-\theta)\triangle x}{\beta^+\theta + \beta^-(1-\theta)} \tag{12}$$

so that approximations to the derivatives on the left and right sides of the interface can be written as

$$\beta^- \left(\frac{u_I - u_k}{\theta \triangle x} \right) = \hat{\beta} \left(\frac{u_{k+1} - u_k}{\triangle x} \right) - \frac{\hat{\beta} b_\Gamma (1-\theta)}{\beta^+} \tag{13}$$

and

$$\beta^+ \left(\frac{u_{k+1} - u_I}{(1-\theta)\triangle x} \right) = \hat{\beta} \left(\frac{u_{k+1} - u_k}{\triangle x} \right) + \frac{\hat{\beta} b_\Gamma \theta}{\beta^-} \tag{14}$$

where

$$\hat{\beta} = \frac{\beta^+ \beta^-}{\beta^+\theta + \beta^-(1-\theta)} \tag{15}$$

defines an effective β. That is, we have

$$\frac{\hat{\beta}\left(\frac{(u_{k+1} - a_\Gamma) - u_k}{\triangle x} - \frac{b_\Gamma(1-\theta)}{\beta^+} \right) - \beta_{k-\frac{1}{2}} \left(\frac{u_k - u_{k-1}}{\triangle x} \right)}{\triangle x} = f_k \tag{16}$$

and

$$\frac{\beta_{k+\frac{3}{2}} \left(\frac{u_{k+2} - u_{k+1}}{\triangle x} \right) - \hat{\beta}\left(\frac{u_{k+1} - (u_k + a_\Gamma)}{\triangle x} + \frac{b_\Gamma \theta}{\beta^-} \right)}{\triangle x} = f_{k+1} \tag{17}$$

as the equations for the unknowns u_k and u_{k+1} respectively.

Of course, these can be rewritten as

$$\frac{\hat{\beta}\left(\frac{u_{k+1} - u_k}{\triangle x} \right) - \beta_{k-\frac{1}{2}} \left(\frac{u_k - u_{k-1}}{\triangle x} \right)}{\triangle x} = f_k + \frac{\hat{\beta} a_\Gamma}{(\triangle x)^2} + \frac{\hat{\beta} b_\Gamma (1-\theta)}{\beta^+ \triangle x} \tag{18}$$

and

$$\frac{\beta_{k+\frac{3}{2}} \left(\frac{u_{k+2} - u_{k+1}}{\triangle x} \right) - \hat{\beta}\left(\frac{u_{k+1} - u_k}{\triangle x} \right)}{\triangle x} = f_{k+1} - \frac{\hat{\beta} a_\Gamma}{(\triangle x)^2} + \frac{\hat{\beta} b_\Gamma \theta}{\beta^- \triangle x} \tag{19}$$

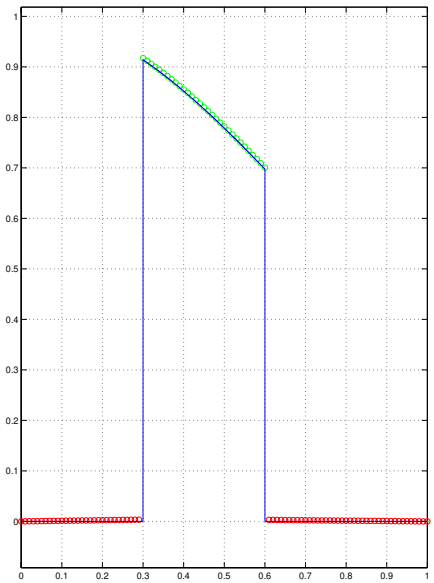


Figure 1: $(\beta u_x)_x = f(x)$, $[u] \neq 0$, $[\beta u_n] \neq 0$

to emphasize that this numerical method yields a symmetric linear system with $\beta_{k+\frac{1}{2}} = \hat{\beta}$.

In figure 1, $\beta = 2$ and $f(x) = (8x^2 - 4)e^{-x^2}$ on the interior region, and $\beta = 1$ and $f(x) = 0$ on the exterior. At $x = .3$, $[u] = -e^{-.09}$ and $[\beta u_n] = -1.2e^{-.09}$, while at $x = .6$, $[u] = -e^{-.36}$ and $[\beta u_n] = 2.4e^{-.36}$. The solution is plotted on top of the exact solution of $u(x) = e^{-x^2}$ on the interior and $u(x) = 0$ on the exterior. Note that the sharp jumps are preserved in both the function and its derivatives.

2.2 Multiple Spatial Dimensions

Consider the two-dimensional Poisson equation

$$(\beta u_x)_x + (\beta u_y)_y = f(\vec{x}) \tag{20}$$

with interface jump conditions, $[u]_\Gamma = a(\vec{x}_\Gamma)$ and $[\beta u_n]_\Gamma = b(\vec{x}_\Gamma)$. The unit normal is $\vec{N} = (n^1, n^2)$ with $\phi \leq 0$ in Ω^- and $\phi > 0$ in Ω^+ implying that the unit normal points from Ω^- into Ω^+.

The normal and tangential derivatives can be defined in terms of u_x, u_y and $\vec{N}$ as

$$u_n = u_x n^1 + u_y n^2 \tag{21}$$

and

$$u_t = u_x n^2 - u_y n^1 \tag{22}$$

respectively. Then

$$u_x = u_n n^1 + u_t n^2 \tag{23}$$

and

$$u_y = u_n n^2 - u_t n^1 \tag{24}$$

follow directly from equations 21 and 22. Multiplying equations 21 and 22 by β and taking the jump across the interface leads to

$$[\beta u_n]_\Gamma = [\beta u_x]_\Gamma n^1 + [\beta u_y]_\Gamma n^2 \tag{25}$$

and

$$[\beta u_t]_\Gamma = [\beta u_x]_\Gamma n^2 - [\beta u_y]_\Gamma n^1 \tag{26}$$

noting that $\vec{N}$ is continuous across the interface. In the same fashion,

$$[\beta u_x]_\Gamma = [\beta u_n]_\Gamma n^1 + [\beta u_t]_\Gamma n^2 \tag{27}$$

and

$$[\beta u_y]_\Gamma = [\beta u_n]_\Gamma n^2 - [\beta u_t]_\Gamma n^1 \tag{28}$$

can be obtained from equations 23 and 24.

Suppose that

$$[\beta u_x]_\Gamma = [\beta u_n]_\Gamma n^1 \tag{29}$$

and

$$[\beta u_y]_\Gamma = [\beta u_n]_\Gamma n^2 \tag{30}$$

are used in place of equations 27 and 28. While equations 29 and 30 are false in general, they still lead to an identity when plugged into equation 25. However, they lead to $[\beta u_t]_\Gamma = 0$ when plugged into equation 26. That is, equations 29 and 30 allow one to correctly capture the jump in the normal derivative while smearing out the jump in the tangential derivative. More importantly, equations 29 and 30 allow the derivative jump condition, $[\beta u_n]_\Gamma = b(\vec{x}_\Gamma)$, to be rewritten as two separate jump conditions, $[\beta u_x]_\Gamma = b(\vec{x}_\Gamma)n^1$ and $[\beta u_y]_\Gamma = b(\vec{x}_\Gamma)n^2$, allowing a dimension by dimension application of the numerical method.

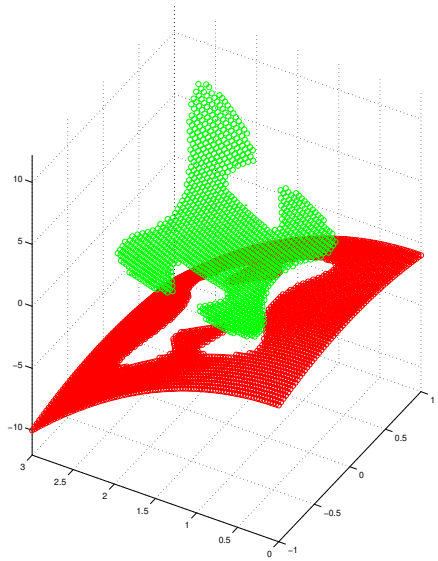


Figure 2: $\nabla \cdot (\beta \nabla u) = f(x,y)$, $[u] \neq 0$, $[\beta u_n] \neq 0$

In two dimensions, the equation is discretized at each grid point (i,j) as

$$\begin{array}{l} \frac{\beta_{i+\frac{1}{2},j}\left(\frac{u_{i+1,j}-u_{i,j}}{\triangle x}\right)-\beta_{i-\frac{1}{2},j}\left(\frac{u_{i,j}-u_{i-1,j}}{\triangle x}\right)}{\triangle x}+ \\ \frac{\beta_{i,j+\frac{1}{2}}\left(\frac{u_{i,j+1}-u_{i,j}}{\triangle y}\right)-\beta_{i,j-\frac{1}{2}}\left(\frac{u_{i,j}-u_{i,j-1}}{\triangle y}\right)}{\triangle y} = f_{i,j}+F^x+F^y \end{array} \tag{31}$$

and included in the linear system of equations. Here $F^x = F^L + F^R$ and $F^y = F^B + F^T$, where F^L, F^R, F^B and F^T incorporate the jump conditions across the left, right, bottom and top arms of the stencil respectively.

Figure 2 shows an interface is defined by $(x(\theta), y(\theta))$ where $x(\theta) = .6\cos(\theta) - .3\cos(3\theta)$, $y(\theta) = 1.5 + .7\sin(\theta) - .07\sin(3\theta) + .2\sin(7\theta)$, $\theta \in [0, 2\pi)$. The exact solution is $u(x,y) = e^x(x^2\sin(y) + y^2)$ on the interior and $u(x,y) = -(x^2+y^2)$ on the exterior. $\beta = 1$ with $f(x,y) = e^x(2 + y^2 + 2\sin(y) + 4x\sin(y))$ on the interior and $\beta = 10$ with $f(x,y) = -40$ on the exterior. The jump conditions are $[u] = -(x^2+y^2) - e^x(x^2\sin(y) + y^2)$ and $[\beta u_n] = (-20x - e^x((x^2+2x)\sin(y) + y^2))n_1 + (-20y - e^x(x^2\cos(y) + 2y))n_2$. Note that the sharp jumps are preserved in both the function and its derivatives even in multiple spatial dimensions.

The three-dimensional scheme follows a similar dimension by dimension framework with $[\beta u_x]_\Gamma = [\beta u_n]_\Gamma n^1$, $[\beta u_y]_\Gamma = [\beta u_n]_\Gamma n^2$, and $[\beta u_z]_\Gamma = [\beta u_n]_\Gamma n^3$ false, but still correctly specifying the normal derivative jump

condition while smearing out the tangential derivatives.

3 Multiphase Incompressible Flow

[6] adapted this boundary condition capturing technique to three-dimensional multiphase incompressible flow calculations including the effects of viscosity, surface tension and gravity. They used a projection method to make an intermediate velocity field divergence free at each time step, thus requiring the solution of a variable coefficient Poisson equation with a discontinuous solution.

The basic equations for viscous incompressible flow are

$$\rho_t + \vec{V} \cdot \nabla \rho = 0 \tag{32}$$

$$\vec{V}_t + \left(\vec{V} \cdot \nabla\right) \vec{V} + \frac{\nabla p}{\rho} = \frac{\left(\nabla \cdot \tau\right)^T}{\rho} + \vec{g} \tag{33}$$

where t is the time, ρ is the density, $\vec{V} =< u, v, w >$ is the velocity, p is the pressure, μ is the viscosity, $\vec{g} =< 0, g, 0 >$ is gravity, and τ is the viscous stress tensor,

$$\tau = \mu \begin{pmatrix} 2u_x & u_y + v_x & u_z + w_x \\ u_y + v_x & 2v_y & v_z + w_y \\ u_z + w_x & v_z + w_y & 2w_z \end{pmatrix} = \mu \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} + \mu \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix}^T . \tag{34}$$

These equations are trivially derived from the viscous compressible Navier-Stokes equations using the divergence free condition, $\nabla \cdot \vec{V} = 0$.

Following the projection method, an intermediate velocity, $\vec{V}^\star$, is calculated via

$$\frac{\vec{V}^\star - \vec{V}^n}{\triangle t} + \left(\vec{V} \cdot \nabla\right) \vec{V} = \frac{\left(\nabla \cdot \tau\right)^T}{\rho} + \vec{g} \tag{35}$$

and then the velocity field at the new time step is obtained through

$$\frac{\vec{V}^{n+1} - \vec{V}^\star}{\triangle t} + \frac{\nabla p}{\rho} = 0. \tag{36}$$

Taking the divergence of equation 36 results in

$$\nabla \cdot \left(\frac{\nabla p}{\rho}\right) = \frac{\nabla \cdot \vec{V}^\star}{\triangle t} \tag{37}$$

after setting $\nabla \cdot \vec{V}^{n+1}$ to zero.

3.1 Jump Conditions

The boundary condition capturing approach, and sharp interface approaches in general, require a set of jump conditions. Although nontrivial for the three-dimensional Navier-Stokes equations including the effects of viscosity, surface tension, and gravity, they were derived in [6]. We repeat that derivation here.

Applying conservation allows one to write the jump conditions for an interface moving with the local fluid velocity in the normal direction as

$$\left[\begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} (pI - \tau)\vec{N}^T \right] = \begin{pmatrix} \sigma\kappa \\ 0 \\ 0 \end{pmatrix} \tag{38}$$

where $\vec{T_1}$ and $\vec{T_2}$ are orthogonal unit tangent vectors, I is the identity matrix, σ is the coefficient of surface tension (a constant), and $\kappa = -\nabla \cdot \vec{N}$ is the local curvature of the interface. Equation 38 states that the net stress on the interface must be zero (since it has no mass). For more details, see [19, 20].

Using the definition of τ in equation 38 leads to

$$\left[\begin{pmatrix} p \\ 0 \\ 0 \end{pmatrix} - \mu \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} \nabla u \cdot \vec{N} \\ \nabla v \cdot \vec{N} \\ \nabla w \cdot \vec{N} \end{pmatrix} - \mu \begin{pmatrix} \nabla u \cdot \vec{N} & \nabla v \cdot \vec{N} & \nabla w \cdot \vec{N} \\ \nabla u \cdot \vec{T_1} & \nabla v \cdot \vec{T_1} & \nabla w \cdot \vec{T_1} \\ \nabla u \cdot \vec{T_2} & \nabla v \cdot \vec{T_2} & \nabla w \cdot \vec{T_2} \end{pmatrix} \cdot \vec{N} \right] = \begin{pmatrix} \sigma\kappa \\ 0 \\ 0 \end{pmatrix} \tag{39}$$

which can be written as three separate jump conditions

$$\left[p - 2\mu \left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N} \right) \cdot \vec{N} \right] = \sigma\kappa \tag{40}$$

$$\left[\mu \left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N} \right) \cdot \vec{T_1} + \mu \left(\nabla u \cdot \vec{T_1}, \nabla v \cdot \vec{T_1}, \nabla w \cdot \vec{T_1} \right) \cdot \vec{N} \right] = 0 \tag{41}$$

$$\left[\mu \left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N} \right) \cdot \vec{T_2} + \mu \left(\nabla u \cdot \vec{T_2}, \nabla v \cdot \vec{T_2}, \nabla w \cdot \vec{T_2} \right) \cdot \vec{N} \right] = 0 \tag{42}$$

Since the flow is viscous, the velocities are continuous

$$[u] = [v] = [w] = 0 \tag{43}$$

as well as their tangential derivatives

$$[\nabla u \cdot \vec{T_1}] = [\nabla v \cdot \vec{T_1}] = [\nabla w \cdot \vec{T_1}] = 0 \tag{44}$$

$$[\nabla u \cdot \vec{T_2}] = [\nabla v \cdot \vec{T_2}] = [\nabla w \cdot \vec{T_2}] = 0 \tag{45}$$

so that the identity

$$\left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{N} + \left(\nabla u \cdot \vec{T_1}, \nabla v \cdot \vec{T_1}, \nabla w \cdot \vec{T_1}\right) \cdot \vec{T_1} + \left(\nabla u \cdot \vec{T_2}, \nabla v \cdot \vec{T_2}, \nabla w \cdot \vec{T_2}\right) \cdot \vec{T_2} = \nabla \cdot \vec{V} = 0 \tag{46}$$

can be used to obtain

$$\left[\left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{N}\right] = 0 \tag{47}$$

emphasizing that the normal derivative of the normal component of the velocity is continuous across the interface allowing equation 40 to be rewritten as

$$[p] - 2[\mu]\left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{N} = \sigma\kappa \tag{48}$$

Next, the family of identities of the form

$$[AB] = \hat{B}[A] + \hat{A}[B] \tag{49}$$

$$\hat{A} = aA_{right} + bA_{left}, \quad \hat{B} = bB_{right} + aB_{left}, \quad a + b = 1 \tag{50}$$

is used along with equations 44 and 45 to rewrite equations 41 and 42 as

$$\left[\left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{T_1}\right] = \frac{-[\mu]}{\hat{\mu}}\hat{\alpha} \tag{51}$$

and

$$\left[\left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{T_2}\right] = \frac{-[\mu]}{\hat{\mu}}\hat{\beta} \tag{52}$$

where

$$\alpha = \left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{T_1} + \left(\nabla u \cdot \vec{T_1}, \nabla v \cdot \vec{T_1}, \nabla w \cdot \vec{T_1}\right) \cdot \vec{N} \tag{53}$$

and

$$\beta = \left(\nabla u \cdot \vec{N}, \nabla v \cdot \vec{N}, \nabla w \cdot \vec{N}\right) \cdot \vec{T_2} + \left(\nabla u \cdot \vec{T_2}, \nabla v \cdot \vec{T_2}, \nabla w \cdot \vec{T_2}\right) \cdot \vec{N} \tag{54}$$

with the "*hat*" superscript defined as outlined above.

Finally, equations 44, 45, 47, 51, and 52 can be compiled to obtain

$$\begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} [\nabla u] \\ [\nabla v] \\ [\nabla w] \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T = \frac{-[\mu]}{\hat{\mu}} \begin{pmatrix} 0 & 0 & 0 \\ \hat{\alpha} & 0 & 0 \\ \hat{\beta} & 0 & 0 \end{pmatrix} \tag{55}$$

or more simply

$$\begin{pmatrix} [u_x] & [u_y] & [u_z] \\ [v_x] & [v_y] & [v_z] \\ [w_x] & [w_y] & [w_z] \end{pmatrix} = \frac{-[\mu]}{\hat{\mu}} \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T \begin{pmatrix} 0 & 0 & 0 \\ \hat{\alpha} & 0 & 0 \\ \hat{\beta} & 0 & 0 \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \tag{56}$$

Alternatively, equations 44, 45, and 47, can be compiled to obtain

$$\begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} [\mu\nabla u] \\ [\mu\nabla v] \\ [\mu\nabla w] \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T = [\mu] \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T +$$
$$[\mu] \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix}^T + \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} [\mu\nabla u] \\ [\mu\nabla v] \\ [\mu\nabla w] \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix}^T \tag{57}$$

or

$$\begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} [\mu\nabla u] \\ [\mu\nabla v] \\ [\mu\nabla w] \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T = [\mu] \begin{pmatrix} \vec{N} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T +$$
$$[\mu] \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix}^T - [\mu] \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix}^T \begin{pmatrix} \vec{N} \\ \vec{0} \\ \vec{0} \end{pmatrix}^T \tag{58}$$

using equations 41 and 42 as well. This can be rewritten as

$$\begin{pmatrix} [\mu u_x] & [\mu u_y] & [\mu u_z] \\ [\mu v_x] & [\mu v_y] & [\mu v_z] \\ [\mu w_x] & [\mu w_y] & [\mu w_z] \end{pmatrix} = [\mu] \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} +$$
$$[\mu] \vec{N}^T \vec{N} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix} \vec{N}^T \vec{N} - [\mu] \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix}^T \begin{pmatrix} \vec{0} \\ \vec{T_1} \\ \vec{T_2} \end{pmatrix} \begin{pmatrix} \nabla u \\ \nabla v \\ \nabla w \end{pmatrix}^T \vec{N}^T \vec{N} \tag{59}$$

noting that the right hand side of this equation only involves derivatives that are continuous across the interface as opposed to equation 56.

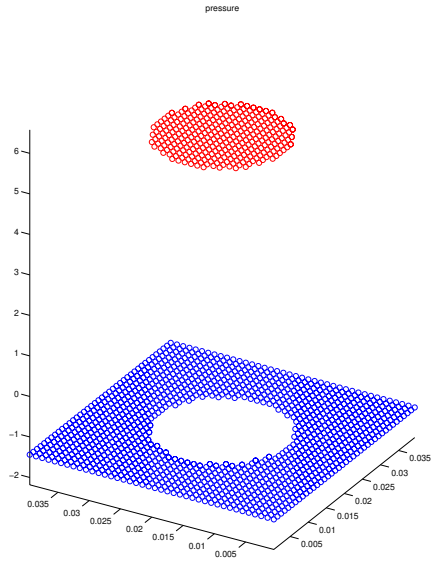


Figure 3: Steady State Air Bubble - Boundary Condition Capturing Method

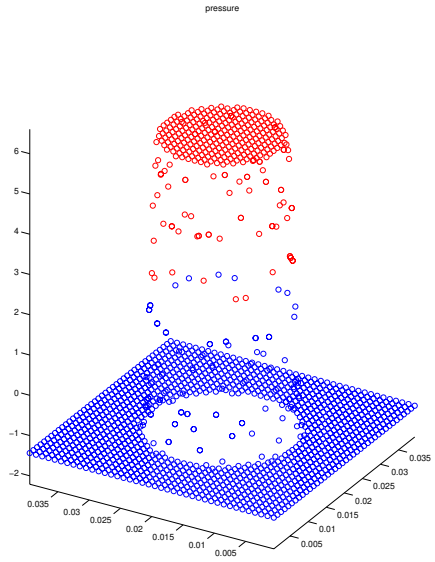


Figure 4: Steady State Air Bubble - Delta Function Method

In viscous flows, the velocity is continuous across the interface implying that the material derivative or Lagrangian acceleration is continuous as well. That is,

$$\left[\frac{Du}{Dt}\right] = \left[\frac{Dv}{Dt}\right] = \left[\frac{Dw}{Dt}\right] = 0 \tag{60}$$

are valid jump conditions allowing one to write

$$\left[\frac{p_x}{\rho}\right] = \left[\frac{(2\mu u_x)_x + (\mu(u_y + v_x))_y + (\mu(u_z + w_x))_z}{\rho}\right] \tag{61}$$

$$\left[\frac{p_y}{\rho}\right] = \left[\frac{(\mu(u_y + v_x))_x + (2\mu v_y)_y + (\mu(v_z + w_y))_z}{\rho}\right] \tag{62}$$

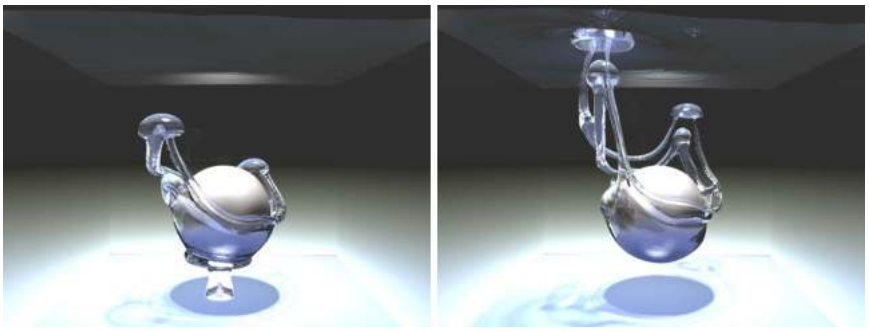
$$\left[\frac{p_z}{\rho}\right] = \left[\frac{(\mu(u_z + w_x))_x + (\mu(v_z + w_y))_y + (2\mu w_z)_z}{\rho}\right] \tag{63}$$

based on equations 33.

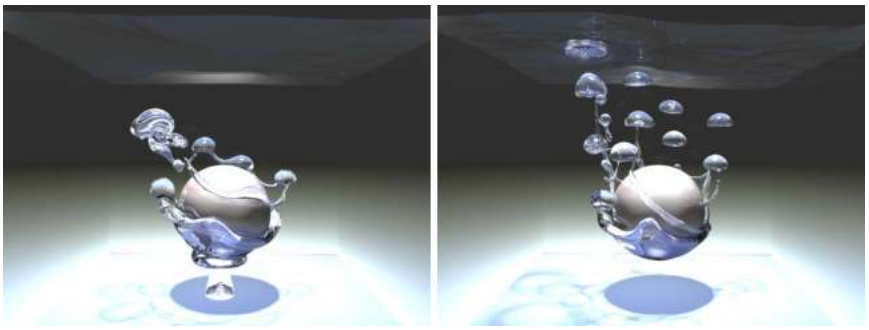
For details on how these jump conditions are all utilized in a sharp interface approach to multiphase incompressible incompressible flow with viscosity and surface tension, we refer the reader to [6]. Notably, they showed that this sharp interface approach can dramatically alleviate parasitic currents because the pressure profile is not smeared out over a characteristic width. In a test example of a stationary air bubble in water (without gravity), they showed that the boundary condition capturing sharp interface approach reduces the parasitic currents by a factor of 1000. Figure 3 shows the sharp pressure profile captured by the boundary condition capturing approach, as opposed to the smeared out delta function approach shown in figure 4.

3.2 Treating Viscosity Implicitly

In [6], the explicit treatment of the viscosity imposed a stringent time step restriction of $\triangle t \sim O(\triangle x^2)$. This poses a severe limitation for high viscosities, and limits the spatial size and time scale of the problems that can be considered. A first step towards an implicit treatment of viscosity for sharp interface methods was proposed in [15]. Assuming μ and ρ are spatially constant within each phase and applying $\nabla \cdot \vec{V} = 0$ allows one to write the viscous term in 33 as $\nabla \cdot (\nu \nabla \vec{V})$ where $\nu = \mu/\rho$. Unfortunately, equation 59 couples the u, v and w terms together meaning that one would require



(a) Delta function



(b) Boundary Condition Capturing

Figure 5: Comparison of the smeared out delta function approach and the boundary condition capturing method for bubble animation.

an implicit method that applies to the entire coupled system of equations, as opposed to the usual component by component approaches. Since [15] was mainly focused on computer graphics applications, they made the simplifying assumption that all the viscous fluxes were balanced setting the right hand side of equation 59 to a three by three zero matrix. This allowed them to fully decouple the viscous terms into three separate scalar equations. The new scheme was applied by removing the viscous terms from equation 35, renaming the result of equation 36 as $\vec{V}^{\star\star}$, and integrating the viscosity implicitly with three decoupled equations for u, v and w of the form

$$u^{\star\star\star} = u^{\star\star} + \triangle t \nabla \cdot (\nu \nabla u^{\star\star\star}) \,. \tag{64}$$

The discontinuous coefficients ν are handled using the method of [4] as outlined in section 2, resulting in an effective viscosity given by equation

15 of

$$\hat{\nu} = \frac{\nu^+ \nu^-}{\nu^+ \theta + \nu^- (1 - \theta)} \tag{65}$$

for stencils that cross the interface. Since this final velocity field cannot be expected to be divergence free, [18] advocated once again using equations 37 and 36 to make the results of the three implicit solves divergence free.

We note that [21] considered a related problem where the u, v and w equations are coupled together in the case of spatially varying viscosity (i.e. fully spatially varying, not just piecewise constant across an interface). They devised a semi-implicit scheme that treated the terms that couple the equations together in an explicit fashion, so that three decoupled scalar equations could be considered in the implicit part of the method. A similar approach could be taken here, adding the jump conditions from equation 59 explicitly so that three decoupled scalar equations could be considered in the implicit part of the method.

Although [15] made other simplifying assumptions as well, such as ignoring the jump in viscosity in equation 48, they carefully considered the jump in pressure due to surface tension effects in that same equation. In fact, they carefully compared the boundary condition capturing approach to the smeared out delta function approach showing dramatic differences between the methods as shown in figure 5.

4 Incompressible Flame Discontinuities

The boundary condition capturing approach was extended further in [17] where a numerical method was developed that allowed for a velocity discontinuity across the interface as well. That paper focused on premixed fuels, where the combustion zone separating the unreacted fuel from the reacted products is assumed to be infinitely thin, allowing for the process to be modeled as a two phase inviscid incompressible flow with discontinuities in velocity and material properties across the interface. Unlike previous methods based on delta function formulations, [17] maintains a sharp velocity profile across the interface allowing the flow to be fully divergence free in each subdomain. In contrast, the delta function approach smears the velocity jump across the interface resulting in a flow with compressible character near the interface. This is especially important since the interface velocity depends on the local velocity of the unreacted fluid, which is difficult to ascertain when the velocity is nonphysically smeared out to be continuous.

For a simple contact discontinuity, the interface velocity is equal to the local fluid velocity, i.e. $\vec{W} = \vec{V}$. Often, only the normal component of the interface velocity is required, i.e. $\vec{W} = D\vec{N}$ where $D = V_N = \vec{V} \cdot \vec{N}$. For

reacting flow, the interface moves at the local velocity of the unreacted fluid plus the flame speed, S, which gives the rate of conversion of the unreacted material into the reacted material. This accounts for the movement of material *across* the interface. If we denote unreacted and reacted material properties with *"u"* and *"r"* subscripts respectively, then the normal component of the interface velocity is given by $D = (V_N)_u + S$. Note that the normal component of the velocity is discontinuous across the interface. The flame speed, S, is typically defined as $S = S_0 + \sigma\kappa$, where S_0 and σ are constants characteristic of the reaction and κ is the curvature.

The equations for conservation of mass and momentum in divergence form are given by

$$\rho_t + \nabla \cdot (\rho\vec{V}) = 0 \tag{66}$$

$$(\rho\vec{V})_t + \nabla \cdot (\rho\vec{V}\vec{V} + pI) = 0 \tag{67}$$

and the divergence free condition is $\nabla \cdot \vec{V} = 0$. The mass and momentum fluxes through an interface surface element (moving in the normal direction with speed D) must be continuous across the interface, implying the standard Rankine-Hugoniot jump conditions across the interface

$$[\rho(V_N - D)] = 0 \tag{68}$$

$$\left[\rho(V_N - D)(\vec{V} - D\vec{N}) + p\vec{N}\right] = 0. \tag{69}$$

If $\vec{T_1}$ and $\vec{T_2}$ are the local unit tangent vectors to the interface, and the mass flux in the moving reference frame is denoted by

$$M = \rho_r \left((V_N)_r - D\right) = \rho_u \left((V_N)_u - D\right) \tag{70}$$

then equation 68 can be rewritten as $[M] = 0$, and equation 69 can be written in terms of normal and tangential components as

$$M\,[V_N] + [p] = 0 \tag{71}$$

$$M\,[V_{T_1}] = M\,[V_{T_2}] = 0. \tag{72}$$

When $D \neq V_N$, then $M \neq 0$ and equation 72 becomes $[V_{T_1}] = [V_{T_2}] = 0$ showing that the tangential components of the velocity must be continuous across the interface. When $D = V_N$, as in the case of a contact discontinuity, $M = 0$, and the tangential velocities are completely uncoupled across the interface. The jump in the normal velocity is derived as

$$0 = [D] = \left[\frac{\rho V_N - \rho(V_N - D)}{\rho}\right] = [V_N] - M\left[\frac{1}{\rho}\right],$$

so that

$$\left[\vec{V}\right] = M\left[\frac{1}{\rho}\right]\vec{N}, \tag{73}$$

which combined with equation 71 implies that

$$[p] = -M^2\left[\frac{1}{\rho}\right]. \tag{74}$$

The velocity jump conditions, 73, are applied to extend the velocity field in each region across the interface. For example, the unreacted velocity field is extended into the reacted region by defining unreacted ghost velocities as

$$u_u^G = u_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right)n_1 \tag{75}$$

$$v_u^G = v_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right)n_2 \tag{76}$$

and

$$w_u^G = w_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right)n_3 \tag{77}$$

where n_1, n_2, n_3 are the components of the interface normal computed at the location of the appropriate velocity component. These ghost values are used in any discretization of the unreacted fluid velocity which crosses the interface. Similarly, reacted ghost velocities are computed in a band on the unreacted side of the interface by adding the jump condition to the local unreacted velocity, and are used in the discretization of the reacted fluid velocity. This avoids combining the discontinuous velocities across the interface. For example, in equation 35, the intermediate velocity field $\vec{V}^\star$ is computed for both the real fluid values and the ghost fluids values, so that the ghost fluids values can be used in the discretization of the right hand side of equation 37, avoiding combination of the intermediate reacted and unreacted velocities across the interface. Also, when solving equation 37 for the pressure, the jump in pressure given by equation 74 is treated using the boundary condition capturing technique outlined in section 2.

Consider two flames both with speed $S = 1$ initially located at $x = -.5$ and $x = .5$. The unreacted material is at rest in the center of the domain. Dirichlet, $p = 0$, boundary conditions are specified at both ends of the domain. Initially, the reacted velocities on the left and right hand sides of the domain were specified as $u = -4$ and $u = 4$ respectively. Figure 6 shows

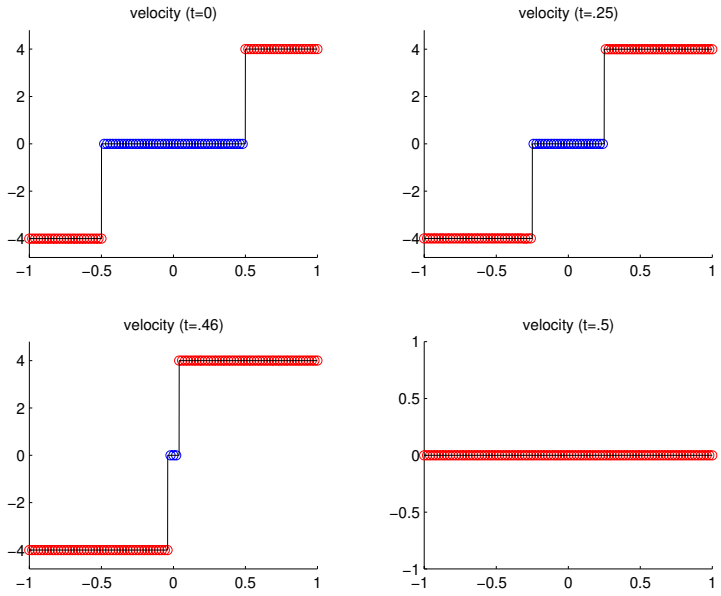
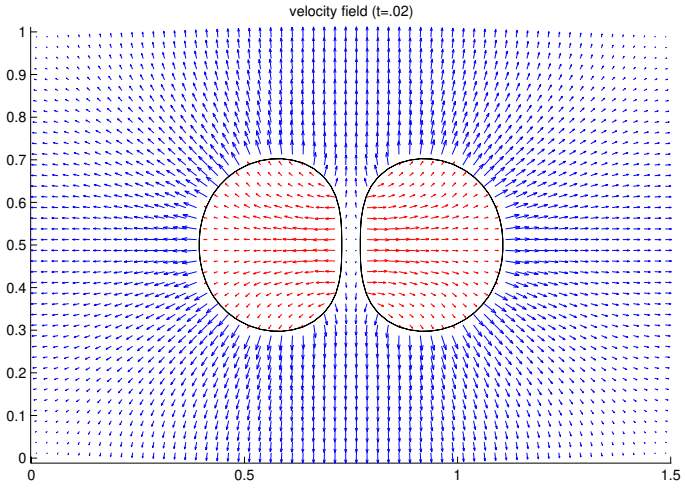


Figure 6: Merging flames.

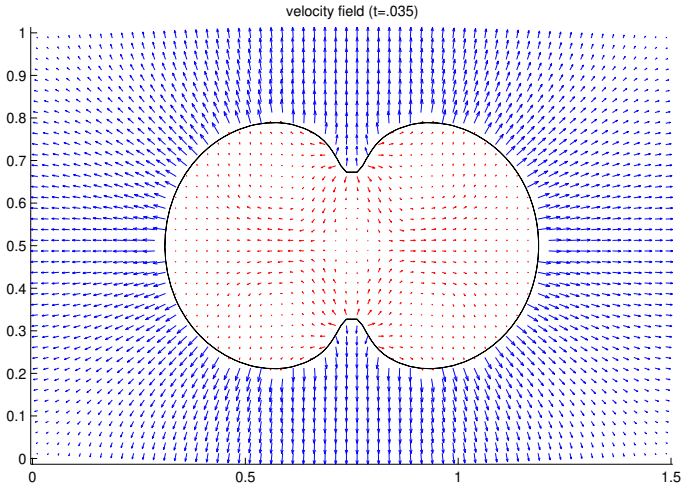
the computed velocity, and illustrates the ability of [17] to treat merging. After merging, the domain contains a single phase incompressible fluid which must have a constant velocity. In the case of compressible flow, a finite speed of propagation rarefaction wave would lower the velocity to the average of the two reacted velocities (zero in this case). For incompressible flow, the "rarefaction wave" moves at infinite speed and the velocity drops to zero in one time step as shown in the figure.

Consider two circular flames with reacted material inside the circles and the unreacted material outside. The flame speed is given by $S = 1 + .01\kappa$, and Dirichlet, $p = 0$, boundary conditions were used on all sides of the domain. Figures 7(a) and 7(b) show the velocity fields at different points in time, before and after the flame fronts merge, respectively. Note that the topological change (merging) requires no special treatment.

[16] exploited the method from [17] to produce visually realistic animation of low speed combustion processes. By modeling the expansion of the fuel as it reacts to form hot gaseous products, they were able to produce high quality animations of visually full flames. Although the level set based premixed flame model suffices for modeling the blue flame core where the reaction is taking place, the blackbody radiation emitted by the hot gaseous products gives the fire its yellowish-orange color. Furthermore,



(a) Velocity field before merging.



(b) Velocity field after merging.

Figure 7: Merging circular flames.

as the temperature cools to the point where the blackbody radiation is no longer visible, smoke and soot are apparent in the flame. Realistic visual depiction of the black body radiation and smoke and soot require that temperature and (smoke) density of each fluid element that has crossed over the reaction zone be tracked with a passively advected reaction coordinate variable.

5 Multiphase Interacting Flows

Recently, [18] presented a level set method for the simulation of multiple (more than two) fluid regions with differing viscosities, densities, and viscoelastic properties. The ghost fluid method and the boundary condition capturing methods of [4] and [6] were used to treat the discontinuous interfaces, and the methods from [17] and [16] were used to incorporate the ability for one material to be converted into another. n distinct fluid regions are represented using n level set functions, resulting in a vector-valued level set function at each point in the domain, $\vec{\phi}(\vec{x}) = (\phi_1(\vec{x}), \ldots, \phi_n(\vec{x}))$. A novel projection method was introduced to decode the vector of level set values, providing a "dictionary" that translates between them and the standard single-valued level set representation.

The multiple level set projection method ensures that the following properties hold at each point in the domain: (P1) If ϕ_j is the smallest element of $\vec{\phi}$, it is the only negative element and its magnitude represents distance to the interface. This property implies that ϕ_i is a signed distance function in region i for all i, and that each point in the domain is assigned to exactly one region. (P2) If P1 holds and ϕ_k is the second smallest element, $\phi_k = -\phi_j$. This implies that the level set for the region that the point is closest to but not inside is a signed distance function as well. These properties are consistent with the standard level set methodology. In particular, a standard level set function ϕ can be regarded as $\vec{\phi} = (\phi_1, \phi_2)$ where $\phi_1 = \phi$ and $\phi_2 = -\phi$. This gives a dictionary that translate between $\vec{\phi}$ and ϕ, by determining ϕ_j and ϕ_k at any point and treating them as ϕ_1 and ϕ_2. The projection algorithm of [18] is simply the following: at each point in the domain, the smallest two elements of $\vec{\phi}$ are determined, and their average is subtracted from each element of $\vec{\phi}$. Subsequently, all geometric information can be computed from the level set which is negative at each point. When level set values are needed between grid points, $\vec{\phi}$ is interpolated at the desired location and the projection method is applied to the resulting vector on the fly.

The particle level set method of [22] was extended to multiple level sets in [18] as well. Each level set has an associated set of particles that are seeded near the boundary of its interior region. After evolving each

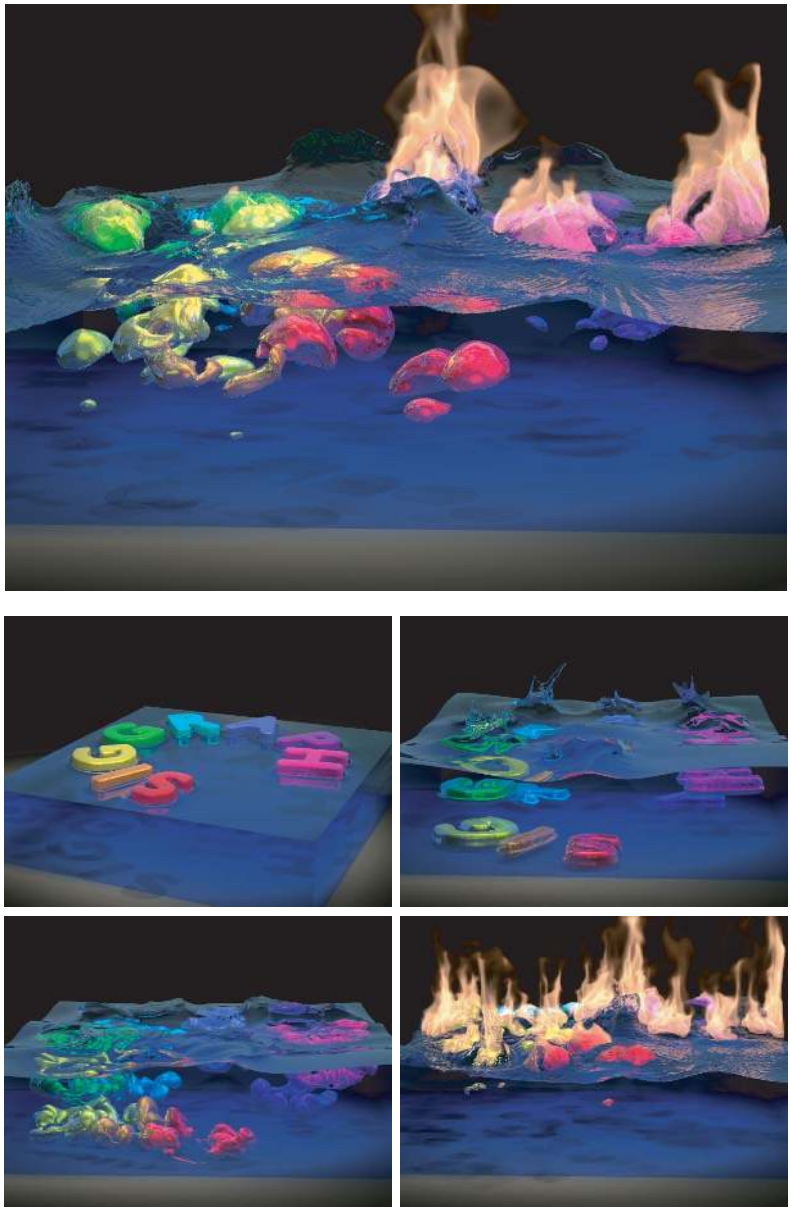


Figure 8: Viscous letters splash into a pool of water, then change into low density inviscid fuel bubbling up and burning when they hit the surface ($350 \times 200 \times 350$ grid, 10 phases).

ϕ_i and all the particles independently in time, each ϕ_i is combined with the particles by rebuilding ϕ_i^- using region i's particles and ϕ_i^+ using the particles of *all* the other regions combined. As in [22], particles are used to correct the level sets after advection and after reinitialization. The projection method is applied after each particle correction. Note that the projection method has the nice property that it preserves signed distance (unlike the method proposed in [23]), so that it can be applied after the reinitialization step without harm.

Figure 8 shows an example with ten separate fluid regions. Initially, the letters are heavier than the water, *highly* viscous, and sink to the bottom almost as if they were rigid bodies. The simulation parameters are then changed, lowering the density, setting viscosity to zero, and adding surface tension. This makes the letters bubble to the surface. To illustrate phase change, the letters are also set to be reactive with air so that they catch on fire when they reach the surface.

6 Conclusion

In this paper, we have shown a variety of applications of the boundary condition capturing method for the variable coefficient Poission equation proposed [4] (motivated in part by the ghost fluid method of [1]) in the field of computational physics as well as computer graphics. Future work is bound to be focused on more accurate methods for a fully implicit viscosity treatment, alleviating the time step restrictions imposed by surface tension forces, and general improvements in the order of accuracy. However, the work to date has proven that sharp interface treatments can be used on fairly complex phenomena for a variety of application areas.

Acknowledgements

Research supported in part by an ONR YIP award and a PECASE award (ONR N00014-01-1-0620), a Packard Foundation Fellowship, a Sloan Research Fellowship, ONR N0014-06-1-0393, ONR N00014-03-1-0071, ONR N00014-02-1-0720, ONR N00014-05-1-0479 (for a SUN computing cluster), ARO DAAD19-03-1-0331, NSF IIS-0326388, NSF ITR-0205671, NSF ITR-0121288, NSF ACI-0323866 and NIH U54-GM072970. J.H. was supported by IT Scholarship Program supervised by IITA (Institute for Information Technology Advancement) & MIC (Ministry of Information and Communication), Republic of Korea.

We would also like to acknowledge our dear departed friend Xu-Dong Liu for many interesting discussions. All this work started when M.K. and R.F. were at UCLA and Xu-Dong was at UCSB, and thus included a lot

of long car rides commuting back and forth, as well as many more (and longer) dinner and lunch conversations. Xu-Dong's impact on computational physics (and computer graphics) will be long felt.

References

[1] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comput. Phys.*, 152:457–492, 1999.

[2] R. Fedkiw, T. Aslam, and S. Xu. The Ghost Fluid Method for Deflagration and Detonation Discontinuities. *J. Comput. Phys.*, 154:393–427, 1999.

[3] D. Nguyen, F. Gibou, and R. Fedkiw. A fully conservative ghost fluid method and stiff detonation waves. In *12th Int. Detonation Symp., San Diego, CA*, 2002.

[4] X.-D. Liu, R. Fedkiw, and M. Kang. A Boundary Condition Capturing Method for Poisson's Equation on Irregular Domains. *J. Comput. Phys.*, 154:151, 2000.

[5] X.-D. Liu and T. C. Sideris. Convergence of the ghost fluid method for elliptic equations with interfaces. *Mathematics of Computation*, 72(244):1731–1746, 2003.

[6] M. Kang, R. Fedkiw, and X.-D. Liu. A boundary condition capturing method for multiphase incompressible flow. *J. Sci. Comput.*, 15:323–360, 2000.

[7] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multifluid flows. *J. Comput. Phys.*, 100:25–37, 1992.

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

[9] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modelling surface tension. *J. Comput. Phys.*, 100:335–353, 1992.

[10] A.-K. Tornberg and B. Engquist. Numerical approximations of singular source terms in differential equations. *J. Comput. Phys.*, 200(2):462–488, 2004.

[11] B. Engquist, A.-K. Tornberg, and R. Tsai. Discretization of dirac delta functions in level set methods. *J. Comput. Phys.*, 207(1):28–51, 2005.

[12] P. Smereka. The numerical approximation of a delta function with application to level set methods. *J. Comput. Phys.*, 211:77–90, 2006.

[13] R. J. Leveque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, 1994.

[14] Z. Li and M.-C. Lai. The immersed interface method for the navier-stokes equations with singular forces. *J. Comput. Phys.*, 171(2):822–842, 2001.

[15] J.-M. Hong and C.-H. Kim. Discontinuous fluids. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 24(3):915–919, 2005.

[16] D. Nguyen, R. Fedkiw, and H. Jensen. Physically based modeling and animation of fire. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 29:721–728, 2002.

[17] D. Nguyen, R. Fedkiw, and M. Kang. A boundary condition capturing method for incompressible flame discontinuities. *J. Comput. Phys.*, 172:71–98, 2001.

[18] F. Losasso, T. Shinar, A. Selle, and R. Fedkiw. Multiple interacting liquids. *ACM Trans. Graph. (SIGGRAPH Proc.), accepted*, 25(3), 2006.

[19] L. D. Landau and E. M. Lifshitz. *Fluid Mechanics, 2nd edition.* Butterworth-Heinemann, Oxford, 1998.

[20] R. Fedkiw and X.-D. Liu. The ghost fluid method for viscous flows. In M. Haferz, editor, *Progress in Numerical Solutions of Partial Differential Equations*, July 1998. Arcachon, France.

[21] N. Rasmussen, D. Enright, D. Nguyen, S. Marino, N. Sumner, W. Geiger, S. Hoon, and R. Fedkiw. Directible photorealistic liquids. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.*, pages 193–202, 2004.

[22] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A hybrid particle level set method for improved interface capturing. *J. Comput. Phys.*, 183:83–116, 2002.

[23] B. Merriman, J. Bence, and S. Osher. Motion of multiple junctions: A level set approach. *J. Comput. Phys.*, 112:334–363, 1994.