Simulating Water and Smoke with an Octree Data Structure

Frank Losasso*
Stanford University
Industrial Light + Magic

Frédéric Gibou†
Stanford University

Ron Fedkiw‡
Stanford University
Industrial Light + Magic

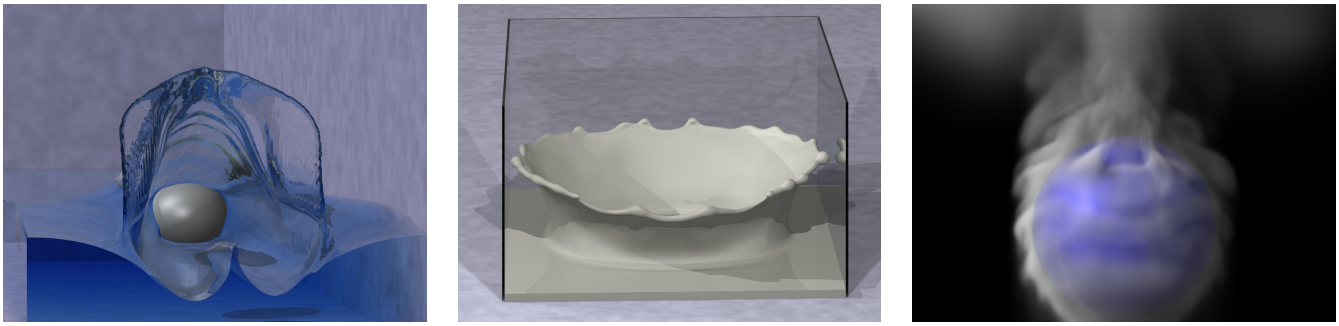

Figure 1: Ellipse traveling through a shallow pool of water (left), formation of a milk crown (center), smoke rising past a sphere (right).

Abstract

We present a method for simulating water and smoke on an *unrestricted* octree data structure exploiting mesh refinement techniques to capture the small scale visual detail. We propose a new technique for discretizing the Poisson equation on this octree grid. The resulting linear system is symmetric positive definite enabling the use of fast solution methods such as preconditioned conjugate gradients, whereas the standard approximation to the Poisson equation on an octree grid results in a non-symmetric linear system which is more computationally challenging to invert. The semi-Lagrangian characteristic tracing technique is used to advect the velocity, smoke density, and even the level set making implementation on an octree straightforward. In the case of smoke, we have multiple refinement criteria including object boundaries, optical depth, and vorticity concentration. In the case of water, we refine near the interface as determined by the zero isocontour of the level set function.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: octree data structure, adaptive mesh refinement, physics-based animation, smoke, water, level set, particles

1 Introduction

Realistic simulations of smoke and water are among the most desired in the special effects industry, since they provide the director with explicit control over the environment enabling the creation of otherwise impossible content. These phenomena contain highly complex motions and rich visual detail, especially when they interact with inanimate objects or the actors themselves. Moreover, a significant portion of the entertainment value and much of the believability relies on an adequate representation and presentation of the small scale visual details such as thin films in water, small rolling vortices in smoke, droplets and sprays, etc. Thus, it is desirable to have both simulation and rendering techniques that can deal with levels of detail.

Recent improvements in simulation techniques have led to impressive simulations of both smoke and water on uniform grids. Empowered by the semi-Lagrangian work of [Stam 1999], [Fedkiw et al. 2001] used vorticity confinement to simulate smoke with visually rich small scale rolling motions. Similarly using both semi-Lagrangian methods and hybridized particle and implicit surface techniques, [Foster and Fedkiw 2001; Enright et al. 2002b] simulated splashing water with both smooth surfaces and thin sheets. While these methods have achieved great success, their application is limited by the computational hardware (i.e. CPU, RAM, disk space) required for the simulations. In an attempt to alleviate this, [Rasmussen et al. 2003] proposed a method that combines interpolation and two-dimensional simulation to obtain highly detailed simulations of large scale smoke-like phenomena. While stunning results were obtained, this method does not faithfully reproduce the three-dimensional Navier-Stokes equations and thus is unable to obtain results for various fully three-dimensional phenomena. Moreover, water was not addressed.

In order to optimize the use of computational resources, we use an adaptive mesh or a level of detail approach where more grid cells are placed in visually interesting regions with rolling smoke or sheeting water. Although adaptive mesh strategies for incompressible flow are quite common, see e.g. [Ham et al. 2002], implementations based on recursive structures, such as the octrees we propose here, are less common. In fact, [Popinet 2003] claims to have the first octree implementation of incompressible flow, although there are certainly similar works such as the nested dyadic grids used for parabolic equations in [Roussel et al. 2003]. We extend the work of [Popinet 2003] in two ways. First, we extend octrees to free surface flows allowing the modeling of a liquid interface. Second, we consider *unrestricted* octrees whereas [Popinet 2003]'s octrees were restricted.

Adaptive meshing strategies lead to nonuniform stencils and thus a nonsymmetric system of linear equations when solving for the pressure, which is needed to enforce the divergence free condition. Although [Popinet 2003] solved this nonsymmetric linear system with a multilevel Poisson solver, [Day et al. 1998] pointed out that these multigrid approaches can be problematic in the presence of

*e-mail: losasso@graphics.stanford.edu
†e-mail: fgibou@math.stanford.edu
‡e-mail: fedkiw@cs.stanford.com

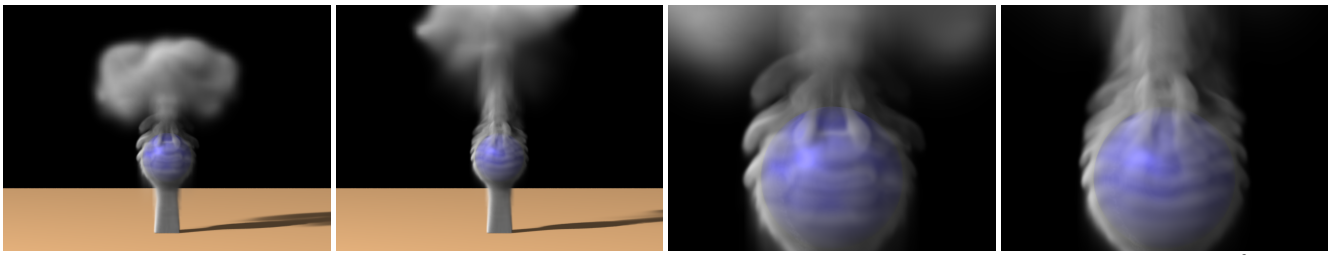

Figure 2: Simulation of smoke past a sphere. The rightmost two figures are close up views. The effective resolution is 1024^3 and the computational time is about 4-5 minutes per frame.

objects with high frequency detail. Moreover, the situation worsens in the presence of interfaces (such as that between water and air), especially since the faithful coarse mesh representation of watertight isosurfaces is a difficult research problem in itself, see e.g. [Lee et al. 2003]. Although multigrid solvers can be efficiently applied if the density is smeared out across the interface as in [Sussman et al. 1999] (resembling a one-phase variable density flow as in [Almgren et al. 1998]), this damps out the surface wave generation that relies on horizontal pressure differences caused by stacking different heights of high-density fluid. That is, damping these high frequency pressure differentials makes multigrid efficient, but also damps the wave motion leading to visually uninteresting overly viscous flows.

More recently, [Sussman 2003] departed from a smeared out density approach and instead solves a free surface problem as in [Enright et al. 2002b]. Moreover, [Sussman 2003] switches from multigrid to a preconditioned conjugate gradient (PCG) method stating that the pressure can be robustly solved for with PCG since the matrix is symmetric. However, the symmetry requirement limits his work to uniform non-adaptive grids. Our new formulation alleviates this restriction by providing a symmetric positive definite discretization of the Poisson equation on an unrestricted octree data structure allowing fast solvers such as PCG to be applied, even in the presence of interfaces.

2 Previous Work

[Kass and Miller 1990] solved a linearized form of the three dimensional Navier-Stokes equations, and [Chen and Lobo 1994] solved the two dimensional Navier-Stokes equations using the pressure to define a height field. The full three dimensional Navier-Stokes equations were solved in [Foster and Metaxas 1996; Foster and Metaxas 1997a; Foster and Metaxas 1997b] for both water and smoke. Large strides in efficiency were made when [Stam 1999] introduced the use of semi-Lagrangian numerical techniques, and [Fedkiw et al. 2001] advocated using vorticity confinement in order to preserve the small scale structure of the flow. [Foster and Fedkiw 2001; Enright et al. 2002b] proposed hybridizing particle and level set methods for water. The incompressible form of the Navier Stokes equations has been used and augmented to model fire [Lamorlette and Foster 2002; Nguyen et al. 2002], clouds [Miyazaki et al. 2002], particle explosions [Feldman et al. 2003], variable viscosity [Carlson et al. 2002], bubbles and surface tension [Hong and Kim 2003], splash and foam [Takahashi et al. 2003], etc. [Treuille et al. 2003] proposed a method for control and used it to make letters out of smoke, and [Stam 2003] solved these equations on surfaces creating beautiful imagery. The compressible version of these equations were used to couple fracture to explosions in [Yngve et al. 2000]. There are also other approaches such as SPH methods [Premoze et al. 2003; Müller et al. 2003].

The representation of implicit surfaces on octree data structures has a long history in the marching cubes community, see the recent papers of [Ju et al. 2002; Ohtake et al. 2003] and the references therein. Moreover, [Frisken et al. 2000; Perry and Frisken 2001] popularized the use of signed distance functions on octree grids. In order to simulate water, we need to solve the partial differential equations that govern the motion of the signed distance function. [Strain 1999b] advocated using quadtrees and semi-Lagrangian methods to solve these equations. Reinitialization for maintaining the signed distance property was addressed in [Strain 1999a], and extrapolation of velocities was considered in [Strain 2000]. One difficulty with semi-Lagrangian methods for solving level set equations is that extreme mass loss (and thus visual artifacts) usually occurs, however [Enright et al. 2004] recently showed that the particles in the particle level set method alleviate this difficulty. A quadtree structure for level set evolution was also proposed in [Sochnikov and Efrima 2003]. However, none of these authors considered level sets in the context of incompressible flows with interfaces such as water.

Starting with the seminal works of [Berger and Oliger 1984; Berger and Colella 1989], adaptive mesh refinement (AMR) typically utilizes uniform overlapping Cartesian grids of various sizes. This is because AMR originally focused on compressible flow with shock waves, and a block structured approach is better able to avoid spurious shock reflections from changing grid levels (since there are less of them). However, in the absence of shocks, a more optimal unrestricted octree approach can be used for incompressible flow.

3 The Octree Data Structure

Figure 3 illustrates our unrestricted octree data structure (see e.g. [Samet 1989]) with a standard MAC grid arrangement [Harlow and Welch 1965], except that all the scalars except the pressure are stored on the *nodes* or corners of the cell. This is convenient since interpolations are more difficult with cell centered data (see e.g. [Strain 1999a]).

Coarsening is performed from the smaller cells to the larger cells, i.e. from the leaves to the root. When coarsening, nodal values are either deleted or unchanged, and the new velocity components at the faces are computed by averaging the old values from that face. Refinement is performed from the larger cells to the smaller cells. The value of a new node on an edge is defined as the average of its two neighbors, and the value of a new node at a face center is defined as the average of the values on the four corners of that face. The velocities on the new faces are defined by first computing the velocities at the nodes, and then averaging back to the face centers. Nodal velocities are computed by averaging the four values from the surrounding cell faces as long as the faces are all the same size. Otherwise, using the coarsest neighboring face as the scale, we compute temporary coarsened velocities on the other faces to be used in the averaging.

For all variables, we constrain T-junction nodes on edges to be linearly interpolated from their neighbors on that edge. Similarly, T-junction nodes on faces are constrained to be the average of the four surrounding corner values. See, e.g. [Westermann et al. 1999] for more details.

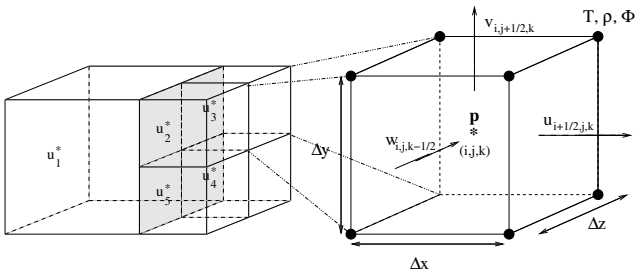

Figure 3: Left: One large cell neighboring four smaller cells. The u_i^* represent the x components of the intermediate velocity $\mathbf{u}^*$ defined at the cell faces. Right: Zoom of one computational cell. The velocity components are defined on the cell faces, while the pressure is defined at the center of the cell. The density, temperature and level set function ϕ are stored at the nodes.

4 Navier Stokes Equations on Octrees

We use the inviscid, incompressible Navier-Stokes equations for the conservation of mass and momentum

$$\begin{aligned} \mathbf{u}_t + \mathbf{u}\cdot\nabla\mathbf{u} &= -\nabla p + \mathbf{f}, & (1)\\ \nabla\cdot\mathbf{u} &= 0, & (2)\end{aligned}$$

where $\mathbf{u} = (u,v,w)$ is the velocity field, $\mathbf{f}$ accounts for the external forces, and the spatially constant density of the mixture has been absorbed into the pressure p. Equation 1 is solved in two steps. First we compute an intermediate velocity $\mathbf{u}^*$ ignoring the pressure term, and then we compute the velocity update via

$$\mathbf{u} = \mathbf{u}^* - \Delta t\nabla p \quad (3)$$

where the pressure is defined as the solution to the Poisson equation,

$$\nabla^2 p = \nabla\cdot\mathbf{u}^*/\triangle t. \quad (4)$$

The external forces are discretized at the cell faces and we postpone the details of their discretization to sections 5 and 6. The convective part of the velocity update is solved using a semi-Lagrangian stable fluids approach as in [Stam 1999]. First we compute nodal velocities, and then we average these values to the cell faces (see section 3). The cell face values are used to trace back characteristics, and trilinear interpolation of nodal values is used to define the new intermediate value of the velocity component on the face in question.

4.1 The Divergence Operator

Equation 4 is solved by first evaluating the right hand side at every cell center in the domain. Then a linear system for the pressure is constructed and inverted. Consider the discretization of equation 4 for a large cell with dimensions $\triangle x$, $\triangle y$ and $\triangle z$ neighboring small cells as depicted in figure 3. Since the discretization is closely related to the second vector form of Green's theorem that relates a volume integral to a surface integral, we first rescale equation 4 by the volume of the large cell to obtain $\mathrm{V}_{\mathrm{cell}}\triangle t\nabla^2 p = \mathrm{V}_{\mathrm{cell}}\nabla\cdot\mathbf{u}^*$. The right hand side now represents the quantity of mass flowing in and out of the large cell within a time step $\triangle t$ in $m^3 s^{-1}$. This can be further rewritten as

$$\mathrm{V}_{\mathrm{cell}}\nabla\cdot(\mathbf{u}^* - \triangle t\nabla p) = 0. \quad (5)$$

This equation implies that the ∇p term is most naturally evaluated at the same location as $\mathbf{u}^*$, namely at the cell faces, and that there is a direct correspondence between the components of ∇p and $\mathbf{u}^*$.

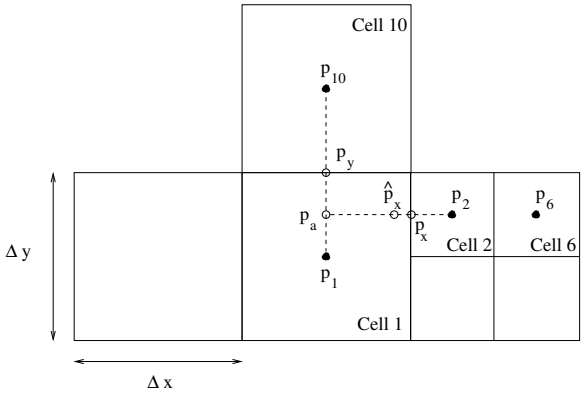

Figure 4: Discretization of the pressure gradient.

Moreover, substituting equation 3 into equation 5 implies $\mathrm{V}_{\mathrm{cell}}\nabla\cdot\mathbf{u} = 0$ or $\nabla\cdot\mathbf{u} = 0$ as desired.

Invoking the second vector form of Green's theorem, one can write

$$\mathrm{V}_{\mathrm{cell}}\nabla\cdot\mathbf{u}^* = \sum_{\mathrm{faces}} (\mathbf{u}^*_{\mathrm{face}}\cdot\mathbf{n})\,\mathrm{A}_{\mathrm{face}},$$

where $\mathbf{n}$ is the *outward* unit normal of the large cell and A_{face} represents the area of a cell face. In the case of figure 3, the discretization of the x component $\partial\mathbf{u}^*/\partial x$ of the divergence reads $\triangle x\triangle y\triangle z\partial\mathbf{u}^*/\partial x = u_2^*\mathrm{A}_2 + u_3^*\mathrm{A}_3 + u_4^*\mathrm{A}_4 + u_5^*\mathrm{A}_5 - u_1^*\mathrm{A}_1$, where the *minus* sign in front of $u_1^*\mathrm{A}_1$ accounts for the fact that the unit normal points to the left. Then $\partial\mathbf{u}^*/\partial x = ((u_2^* + u_3^* + u_4^* + u_5^*)/4 - u_1^*)/\triangle x$. The y and z directions are treated similarly.

Once, the divergence is computed at the cell center, equation 4 is used to construct a linear system of equations for the pressure. Invoking again the second vector form of Green's theorem, one can write

$$\mathrm{V}_{\mathrm{cell}}\nabla\cdot(\triangle t\nabla p) = \sum_{\mathrm{faces}} ((\triangle t\nabla p)_{\mathrm{face}}\cdot\mathbf{n})\,\mathrm{A}_{\mathrm{face}}. \quad (6)$$

Therefore, once the pressure gradient is computed at every face, we can carry out the computation in a manner similar to that of the velocity divergence above. There exist different choices in the discretization of $(\nabla p)_{\mathrm{face}}$, and we seek to discretize the pressure gradient in a fashion that yields a symmetric linear system. Efficient iterative methods such as PCG (see e.g. [Saad 1996]) can be applied to symmetric positive definite matrices offering a significant advantage over methods for nonsymmetric linear systems. Moreover, since data access for the octree is not as convenient as for regular grids, there is a strong benefit in designing a discretization that leads to a symmetric linear system.

4.2 The Pressure Gradient

Consider the configuration in figure 4. In the case where two cells of the same size juxtapose each other, standard central differencing defines the pressure gradient at the face between them, as is the case for $p_y = (p_{10} - p_1)/\triangle y$.

Consider the discretization of the pressure gradient in the x direction at the face between cell 1 and cell 2. A standard approach is to first compute a weighted average value p_a for the pressure, by interpolating between the pressure values p_1 and p_{10}. Then, since standard differencing of $\hat{p}_x = (p_2 - p_a)/(.75\triangle x)$ does not define $\hat{p}_x$ at the cell face but midway between the locations of p_a and p_2, one usually resorts to more complex discretizations. A typical choice is to pass a quadratic interpolant through p_a, p_2 and p_6 and evaluate its derivative at the cell face, see e.g. [Chen et al. 1997]. However, this approach yields a nonsymmetric linear system that is slow to invert. The nonsymmetric nature of the linear system comes from

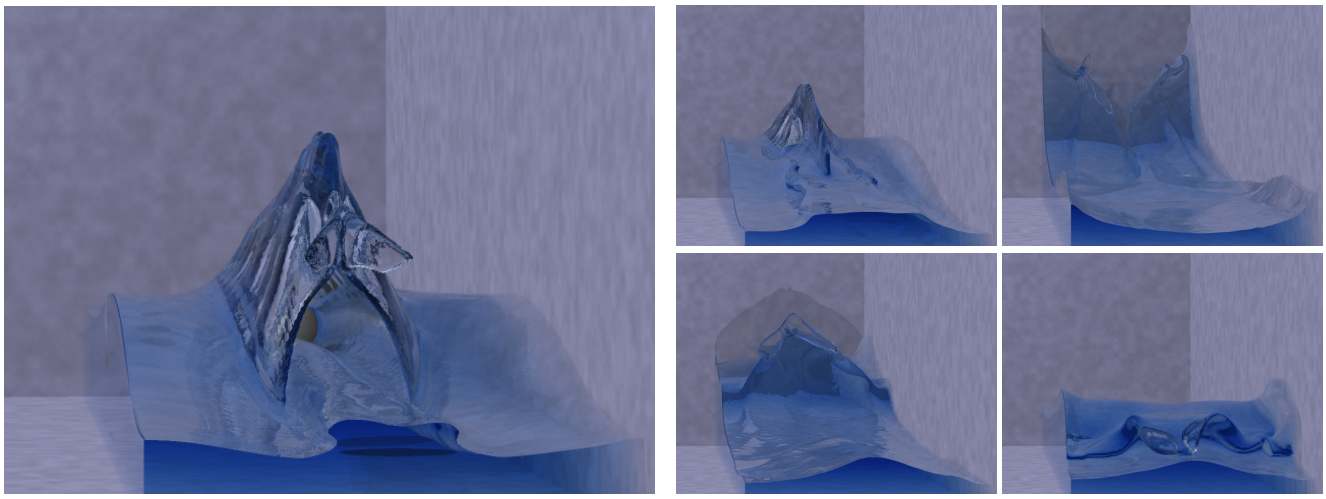

Figure 5: An ellipsoid slips along through shallow water illustrating our method's ability to resolve thin sheets. The effective resolution is 512^3 and the computational time is about 4-5 minutes per frame.

the non-locality of the discretization, i.e. p_a depends on p_{10} and the quadratic interpolation would depend on p_6. Consequently, the equation for cell 1 involves both p_{10} and p_6. It is unlikely that the equation for cell 6 depends on p_1, since cell 6 juxtaposes another cell of the same size, namely cell 2. And even if it did, the coefficients of dependence would not be symmetric.

Our approach is based on the fact that $O(\triangle x)$ perturbations in the pressure location still yield consistent approximations as in [Gibou et al. 2002]. Therefore defining $p_x = (p_2 - p_a)/(.75\triangle x)$ *at* the cell face still yields a convergent approximation, since the location of $\hat{p}_x$ is perturbed by a small amount proportional to a grid cell. Moreover, we can avoid the dependence of p_a on values other than p_1 by simply setting $p_a = p_1$. This corresponds to an $O(\triangle x)$ perturbation of the location of p_1, and therefore still yields a convergent approximation. Thus, our discretization of p_x is simply $p_x = (p_2 - p_1)/(.75\triangle x)$. Moreover, since only p_1 and p_2 are considered, one can define $p_x = (p_2 - p_1)/\triangle$ where $\triangle$ can be defined as the size of the large cell, $\triangle x$, the size of the small cell, $.5\triangle x$, the Euclidean distance between p_1 and p_2, etc. We have carried out numerical tests against known analytic solutions to the Poisson equation demonstrating that all these choices converge. Currently, we are investigating the impact of different $\triangle$ definitions on smoke and water simulations.

In light of equation 6, p_x contributes to both row 1 and row 2 of the matrix representing the linear system of equations, since it is located at the cell face between cell 1 and cell 2. More precisely, the contribution to row 1 occurs through the term

$$\triangle t\, p_x n_1 A_{\text{face}} = \triangle t \frac{p_2 - p_1}{\triangle}(1) A_{\text{face}},$$

since n_1, the x component of the outward normal to cell 1, points to the right (hence $n_1 = 1$). Likewise, the contribution to row 2 occurs through the term

$$\triangle t\, p_x n_1 A_{\text{face}} = \triangle t \frac{p_2 - p_1}{\triangle}(-1) A_{\text{face}},$$

since n_1, the x component of the outward normal to cell 2, points to the left (hence $n_1 = -1$). Therefore, the coefficient for p_2 in row 1 and the coefficient for p_1 for row 2 are identical, namely $\triangle t A_{\text{face}}/\triangle$. The same procedure is applied to all faces, and the discretization of the y and z components of the pressure gradient are carried out in a similar manner. Hence, our discretization yields a symmetric linear system that can be efficiently inverted with a PCG method. The preconditioner we use is based on an incomplete *LU* Cholesky factorization that we modify to ensure that the row sum of *LU* is equal to the row sum of the original matrix (see [Saad 1996]). This yields a significant speed up in the matrix inversion.

The matrix constructed above is negative definite, as is usual when discretizing equation 4. We simply multiply all equations by -1 to make it positive definite. We also note that Dirichlet or Neumann boundary conditions do not disrupt the symmetry. In the case of a Neumann boundary condition, the term $(p_2 - p_1)/\triangle$ disappears from both row 1 and row 2. In the case of a Dirichlet boundary condition, e.g. for p_2, the equation for p_2 drops out of the system and all the terms involving p_2 are moved to the right hand side of the linear system.

4.3 Accuracy

We stress that the dominant errors are due to the first order accurate semi-Lagrangian advection scheme. The velocity averaging is second order accurate and is required in all MAC grid methods in order to define a full velocity vector at a common location for the semi-Lagrangian advection. Dropping the Poisson solver from second to first order accuracy merely puts it on par asymptotically with the semi-Lagrangian scheme. However, we still solve for a fully divergence free velocity field to machine precision just as in a non-adaptive setting. We tested our Poisson solver on many exact solutions and readily obtain several digits of accuracy indicating that the errors from this part of the algorithm are small. See table 1 for a typical result.

5 Smoke

The external forces due to buoyancy and heat convection are modeled as $\mathbf{f}_{\text{buoy}} = -\alpha\rho\mathbf{z} + \beta\,(T - T_{\text{amb}})\,\mathbf{z}$, where $\mathbf{z} = (0,0,1)$, T_{amb} is the ambient temperature and α and β are parameters controlling the influence of the density and the temperature. The density and the temperature are passively advected with the flow velocity and are updated with the semi-Lagrangian method using velocities defined at the nodes (see section 3). Both the density and the temperature are then averaged to the faces in order to evaluate the forcing term.

The vorticity confinement force is calculated as follows. First we define velocities at the centers of cells by using area weighted averaging of face values. Then all the derivatives needed to compute the vorticity, $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, are computed on cell faces using the same method used to compute pressure derivatives. Area weighted averaging is used (again) to define all these derivatives at the cell center, and then we compute the vorticity and its magnitude (at

L^1error	order	L^∞error	order
4.083×10^{-2}	$--$	6.332×10^{-2}	$--$
8.713×10^{-3}	2.22	2.203×10^{-2}	1.523
2.952×10^{-3}	1.56	1.292×10^{-2}	.770
9.980×10^{-4}	1.56	7.745×10^{-3}	.739
4.010×10^{-4}	1.31	4.249×10^{-3}	.866
1.820×10^{-4}	1.14	2.287×10^{-3}	.894

Table 1: Poisson solver accuracy on an unrestricted octree grid.

the cell center). Next, the gradients of the vorticity are computed at the cell faces, and averaged back to the cell center to define $\mathbf{N} = \nabla|\omega|/|\nabla|\omega||$. Finally, the unscaled force can be computed at the cell centers as $\mathbf{N} \times \omega$. Cell face values of this term are obtained by averaging the values from the two cells that contain the face. Then this term is scaled by the diagonal of the face h and a tunable parameter ε.

Inside an object, we set the temperature to the object temperature and the density to zero. For velocity, we clip the component normal to the object so that it is guaranteed to be separating. Furthermore, we apply Neumann boundary conditions to the cell faces that intersect the object when solving for the pressure. This keeps these velocities fixed.

In the case of smoke, we utilize three different refinement criteria. First, we refine near objects since their interactions with smoke will introduce small scales features that enhance believability. Second, we refine near concentrations of high vorticity. Third, we refine in a band of density values (for example $.1 < \rho < .3$). This last criteria prunes out both the low densities that cannot be seen as well as the high densities interior to the smoke which are self-occluded.

6 Water

We use the particle level set method of [Enright et al. 2002a] with $\phi \leq 0$ designating the water and $\phi > 0$ representing the air. When solving for the pressure, one only needs to consider cells in the water. Dirichlet boundary conditions of $p_I = p_{\text{air}} + \sigma\kappa$ are set in the air cells bordering the water, where σ is a surface tension coefficient and $\kappa = \nabla \cdot (\nabla\phi/|\nabla\phi|)$ is the local interface curvature. We note that [Hong and Kim 2003] considered surface tension in the case of bubbles, but not for films. κ is computed by averaging nodal values of ϕ to the cell center, computing derivatives of ϕ on the cell faces, averaging these back to the cell center, using these cell centered values to obtain the normal, computing derivatives of the normal on the cell faces, averaging these values back to the cell center, and finally using these cell centered values to obtain the curvature. The only external force we account for is gravity via $\mathbf{u} \mathrel{+}= \triangle t\mathbf{g}$. The interaction with objects is similar to that of smoke. We apply adaptive refinement to a band about the interface (focusing more heavily on the water side), noting that the signed distance property of ϕ makes this straightforward.

Recently, [Enright et al. 2004] showed that the particle level set method relies on particles for accuracy and the level set for connectivity. Moreover, they showed that one could use a simple semi-Lagrangian method on the level set with no significant accuracy penalty as long as the particles are evolved with at least second order Runge-Kutta. Thus, we update ϕ with the semi-Lagrangian method using velocities defined at the nodes (see section 3). The particles are advected using second order Runge-Kutta and trilinearly interpolated nodal velocities.

We use the fast marching method [Tsitsiklis 1995; Sethian 1996] to maintain the signed distance property of ϕ. First, the signed distance is computed at all the nodes around the interface, and they are marked as *updated*. The nodes adjacent to the *updated* nodes are tagged *trial*. Then we compute potential values of ϕ at all *trial* nodes using only *updated* nodes. The smallest of these is tagged as *updated*, and all its non-updated neighbors are tagged *trial*. This process is repeated to fill in a band of values near the interface. For many grid nodes there are neighboring values of ϕ in all six directions, but at T-junctions there are directions where ϕ is missing some of its neighbors. Since we coarsen as we move away from the interface, these directions will generally not contribute to the potential value of ϕ. Thus, we trivially ignore them.

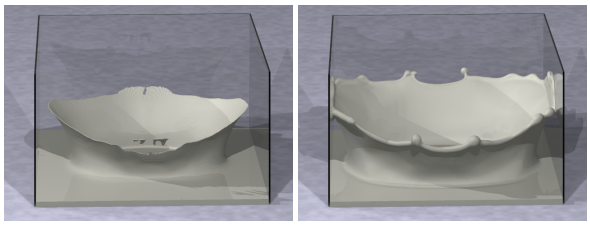

Figure 6: Formation of a milk crown demonstrating the effect of surface tension. We take $\sigma = 0$ (left) and $\sigma = .0005$ (right). The effective resolution is 512^3 and the computational time is about 4-5 minutes per frame.

Velocity extrapolation is carried out by first defining nodal velocities, extrapolating them, and then computing the face velocities. If we perform this algorithm as in [Enright et al. 2002a], we will occasionally encounter grid points that have no neighboring values of velocity and cannot be updated. While this is rare (but not impossible) for uniform grids, T-junctions exacerbate their occurrence on octree grids. When this happens, we skip over these nodes until one of their neighbors is updated (and then we update them in the usual manner).

7 Conclusions

Our new symmetric formulation reduces the pressure solver to approximately 25% of the total simulation time requiring only about 20 iterations to converge to an accuracy of machine precision. This leaves little room for improvement and even a zero cost pressure solver would only make the code 25% faster. On the other hand, nonsymmetric formulations requiring BiCGSTAB or GMRES and nonoptimal preconditioners easily lead to an order of magnitude slowdown, or in the worst case scenario problems with robustly finding a solution at all. The symmetric formulation enables a full octree discretization of the equations that govern both the flow of smoke and water. Moreover, we achieved reasonable computational costs on grids as fine as 1024^3 allowing us to capture fine scale rolling motion in smoke and thin films for water.

8 Acknowledgement

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 N00014-03-1-0071, ONR N00014-02-1-0720, ARO DAAD19-03-1-0331, NSF DMS-0106694, NSF ITR-0121288, NSF IIS-0326388, NSF ACI-0323866 and NSF ITR-0205671. In addition, F. G. was supported in part by an NSF Postdoctoral Fellowship (DMS-0102029).

We'd also like to thank Mike Houston for providing computing resources used for rendering.

References

ALMGREN, A., BELL, J., COLELLA, P., HOWELL, L., AND WELCOME, M. 1998. A conservative adaptive projection method for the variable density incompressible navier-stokes equations. *J. Comput. Phys. 142*, 1–46.

BERGER, M., AND COLELLA, P. 1989. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys. 82*, 64–84.

BERGER, M., AND OLIGER, J. 1984. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys. 53*, 484–512.

CARLSON, M., MUCHA, P., VAN HORN III, R., AND TURK, G. 2002. Melting and flowing. In *ACM SIGGRAPH Symposium on Computer Animation*, 167–174.

CHEN, J., AND LOBO, N. 1994. Toward interactive-rate simulation of fluids with moving obstacles using the navier-stokes equations. *Computer Graphics and Image Processing 57*, 107–116.

CHEN, S., MERRIMAN, B., OSHER, S., AND SMEREKA, P. 1997. A simple level set method for solving stefan problems. 8–29.

DAY, M., COLELLA, P., LIJEWSKI, M., RENDLEMAN, C., AND MARCUS, D. 1998. Embedded boundary algorithms for solving the poisson equation on complex domains. Tech. rep., Lawrence Berkeley National Laboratory (LBNL-41811).

ENRIGHT, D., FEDKIW, R., FERZIGER, J., AND MITCHELL, I. 2002. A hybrid particle level set method for improved interface capturing. *J. Comp. Phys. 183*, 83–116.

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. *ACM Trans. Graph. (SIGGRAPH Proc.) 21*, 3, 736–744.

ENRIGHT, D., LOSASSO, F., AND FEDKIW, R. 2004. A fast and accurate semi-Lagrangian particle level set method. *Computers and Structures, (in press)*.

FEDKIW, R., STAM, J., AND JENSEN, H. 2001. Visual simulation of smoke. In *Proc. of ACM SIGGRAPH 2001*, 15–22.

FELDMAN, B. E., O'BRIEN, J. F., AND ARIKAN, O. 2003. Animating suspended particle explosions. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 3, 708–715.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. In *Proc. of ACM SIGGRAPH 2001*, 23–30.

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

FOSTER, N., AND METAXAS, D. 1997. Controlling fluid animation. In *Computer Graphics International 1997*, 178–188.

FOSTER, N., AND METAXAS, D. 1997. Modeling the motion of a hot, turbulent gas. In *Proc. of SIGGRAPH 97*, 181–188.

FRISKEN, S., PERRY, R., ROCKWOOD, A., AND JONES, T. 2000. Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proc. SIGGRAPH 2000*, 249–254.

GIBOU, F., FEDKIW, R., CHENG, L.-T., AND KANG, M. 2002. A second–order–accurate symmetric discretization of the poisson equation on irregular domains. 205–227.

HAM, F., LIEN, F., AND STRONG, A. 2002. A fully conservative second-order finite difference scheme for incompressible flow on nonuniform grids. *J. Comput. Phys. 117*, 117–133.

HARLOW, F., AND WELCH, J. 1965. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids 8*, 2182–2189.

HONG, J.-M., AND KIM, C.-H. 2003. Animation of bubbles in liquid. *Comp. Graph. Forum (Eurographics Proc.) 22*, 3, 253–262.

JU, T., LOSASSO, F., SCHAEFER, S., AND WARREN, J. 2002. Dual contouring of Hermite data. *ACM Trans. Graph. (SIGGRAPH Proc.) 21*, 3, 339–346.

KASS, M., AND MILLER, G. 1990. Rapid, stable fluid dynamics for computer graphics. In *Computer Graphics (Proc. of SIGGRAPH 90)*, vol. 24, 49–57.

LAMORLETTE, A., AND FOSTER, N. 2002. Structural modeling of natural flames. *ACM Trans. Graph. (SIGGRAPH Proc.) 21*, 3, 729–735.

LEE, H., DESBRUN, M., AND SCHRODER, P. 2003. Progressive encoding of complex isosurfaces. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 3, 471–476.

MIYAZAKI, R., DOBASHI, Y., AND NISHITA, T. 2002. Simulation of cumuliform clouds based on computational fluid dynamics. *Proc. Eurographics 2002 Short Presentation*, 405–410.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *Proc. of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 154–159.

NGUYEN, D., FEDKIW, R., AND JENSEN, H. 2002. Physically based modeling and animation of fire. In *ACM Trans. Graph. (SIGGRAPH Proc.)*, vol. 29, 721–728.

OHTAKE, Y., BELYAEV, A., ALEXA, M., TURK, G., AND SEIDEL, H. 2003. Multi-Level Partition of Unity Implicits. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 3, 463–470.

PERRY, R., AND FRISKEN, S. 2001. Kizamu: a system for sculpting digital characters. In *Proc. SIGGRAPH 2001*, vol. 20, 47–56.

POPINET, S. 2003. Gerris: A tree-based adaptive solver for the incompressible euler equations in complex geometries. *J. Comp. Phys. 190*, 572–600.

PREMOZE, S., TASDIZEN, T., BIGLER, J., LEFOHN, A., AND WHITAKER, R. 2003. Particle–based simulation of fluids. In *Comp. Graph. Forum (Eurographics Proc.)*, vol. 22, 401–410.

RASMUSSEN, N., NGUYEN, D., GEIGER, W., AND FEDKIW, R. 2003. Smoke simulation for large scale phenomena. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 703–707.

ROUSSEL, O., SCHNEIDER, K., TSIGULIN, A., AND BOCKHORN, H. 2003. A conservative fully adaptive multiresolution algorithm for parabolic pdes. *J. Comput. Phys. 188*, 493–523.

SAAD, Y. 1996. *Iterative methods for sparse linear systems*. PWS Publishing. New York, NY.

SAMET, H. 1989. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, New York.

SETHIAN, J. 1996. A fast marching level set method for monotonically advancing fronts. *Proc. Natl. Acad. Sci. 93*, 1591–1595.

SOCHNIKOV, V., AND EFRIMA, S. 2003. Level set calculations of the evolution of boundaries on a dynamically adaptive grid. *Int. J. Num. Methods in Eng. 56*, 1913–1929.

STAM, J. 1999. Stable fluids. In *Proc. of SIGGRAPH 99*, 121–128.

STAM, J. 2003. Flows on surfaces of arbitrary topology. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 724–731.

STRAIN, J. 1999. Fast tree-based redistancing for level set computations. *J. Comput. Phys. 152*, 664–686.

STRAIN, J. 1999. Tree methods for moving interfaces. *J. Comput. Phys. 151*, 616–648.

STRAIN, J. 2000. A fast modular semi-lagrangian method for moving interfaces. *J. Comput. Phys. 161*, 512–536.

SUSSMAN, M., ALMGREN, A., BELL, J., COLELLA, P., HOWELL, L., AND WELCOME, M. 1999. An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys. 148*, 81–124.

SUSSMAN, M. 2003. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comp. Phys. 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. *Comp. Graph. Forum (Eurographics Proc.) 22*, 3, 391–400.

TREUILLE, A., MCNAMARA, A., POPOVIĆ, Z., AND STAM, J. 2003. Keyframe control of smoke simulations. *ACM Trans. Graph. (SIGGRAPH Proc.) 22*, 3, 716–723.

TSITSIKLIS, J. 1995. Efficient algorithms for globally optimal trajectories. *IEEE Trans. on Automatic Control 40*, 1528–1538.

WESTERMANN, R., KOBBELT, L., AND ERTL, T. 1999. Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Vis. Comput. 15*, 2, 100–111.

YNGVE, G., O'BRIEN, J., AND HODGINS, J. 2000. Animating explosions. In *Proc. SIGGRAPH 2000*, vol. 19, 29–36.