Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

Simulating Free Surface Flow with Very Large Time Steps

Michael Lentine† Stanford University Industrial Light + Magic

Matthew Cong† Stanford University

Saket Patkar† Stanford University

Ronald Fedkiw† Stanford University Industrial Light + Magic

Abstract

We provide a novel simulation method for incompressible free surface flows that allows for large time steps on the order of 10-40 times bigger than the typical explicit time step restriction would allow. Although semi-Lagrangian advection allows for this from the standpoint of stability, large time steps typically produce significant visual errors. This was addressed in previous work for smoke simulation using a mass and momentum conserving version of semi-Lagrangian advection, and while its extension to water for momentum conservation for small time steps was addressed, pronounced issues remain when taking large time steps. The main difference between smoke and water is that smoke has a globally defined velocity field whereas water needs to move in a manner uninfluenced by the surrounding air flow, and this poses real issues in determining an appropriate extrapolated velocity field. We alleviate problems with the extrapolated velocity field by not using it when it is incorrect, which we determine via conservative advection of a color function which adds forwardly advected semi-Lagrangian rays to maintain conservation when mass is lost. We note that one might also use a more traditional volume-of-fluid method which is more explicitly focused on the geometry of the interface but can be less visually appealing – it is also unclear how to extend volume-of-fluid methods to have larger time steps. Finally, we prefer the visual smoothness of a particle level set method coupled to a traditional backward tracing semi-Lagrangian advection where possible, only using our forward traced color function solution in areas of the flow where the particle level set method fails due to the extremely large time steps.

1. Introduction

Physically based simulation of water has been one of the most interesting and challenging problems in computer graphics because of the amount of small scale details that can be achieved. Earlier work includes [FM97, Sta99, FF01]. With the increased availability of low cost memory, multi-core machines and software suitable for MPI and threading, grid sizes can be increased achieving even greater detail by reducing the numerical viscosity that damps out the solution on coarser grids. Unfortunately, the computational cost does not scale linearly in the number of grid cells as the time step size must decrease either due to stability restrictions in explicit schemes or accuracy restrictions for implicit schemes. The traditional semi-Lagrangian advection of [Sta99] is unconditionally stable, and thus, does allow for much larger time steps. However, the visual artifacts that are produced with this method at large time steps make this approach undesirable. Although this has been addressed in [LAF11] for smoke simulations, there remain a number of issues for water simulation. For example, because water is treated as a free surface, velocities must be extrapolated across the interface which can create artifacts at large time steps. Figure 2 demonstrates one such case where a ball with constant downward velocity is falling onto a pool of water. With current methods, the bottom of the ball depicted in yellow fails to advect correctly due to inaccurate extrapolated velocities – which could only be accurate if multi-valued. Particle based methods, see e.g. [DC96, MCG03, ZB05, LTKF08] and the references therein, also suffer from accuracy issues and visual artifacts that result from the poor sampling of particles (too sparse or too dense) at these large time steps. There are also additional difficulties in creating surfaces from particle data [YT10, BGB11], and we instead focus our efforts on grid based methods – although addressing very large time steps for particle based methods would also be interesting and useful. Note that front tracking methods [BBB10, WTGT10] also rely on an interface representation via particles and thus would suffer from the same sampling problems as particle based methods but would have

† e-mail: {mlentine|mdcong|patkar|fedkiw}@cs.stanford.edu

DOI: 10.2312/SCA/SCA12/107-116

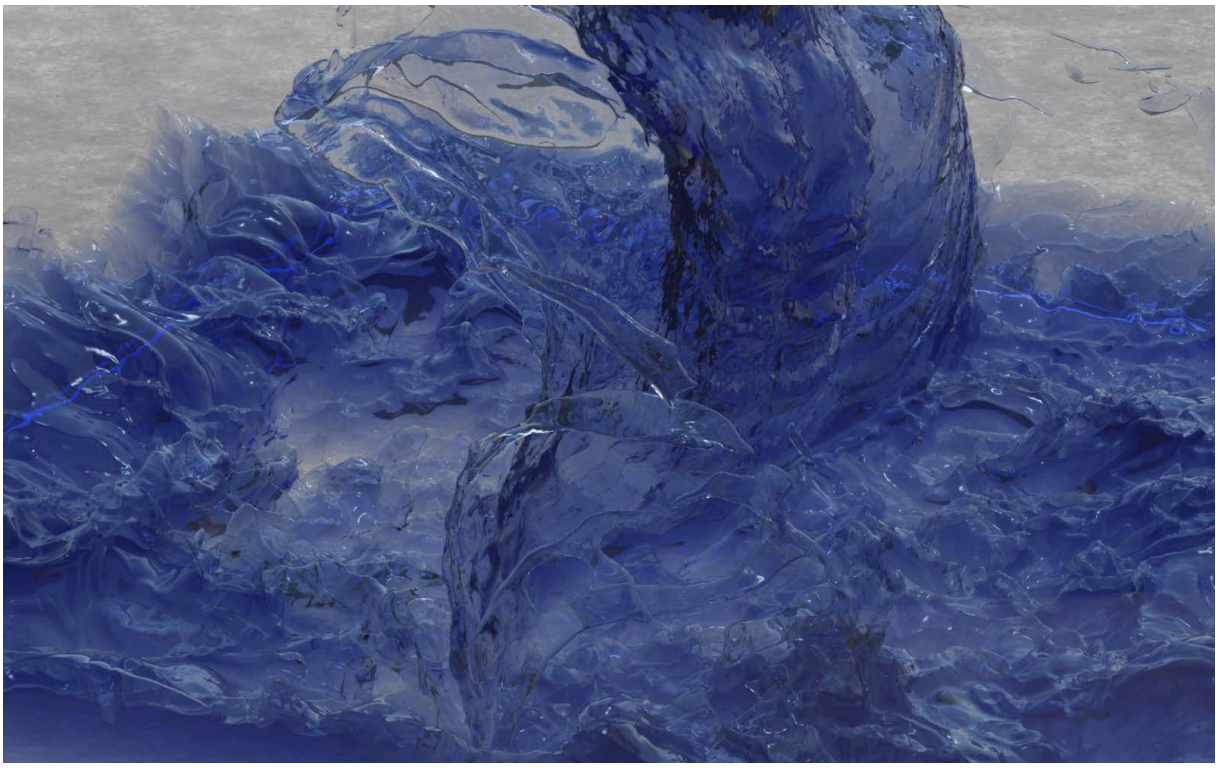
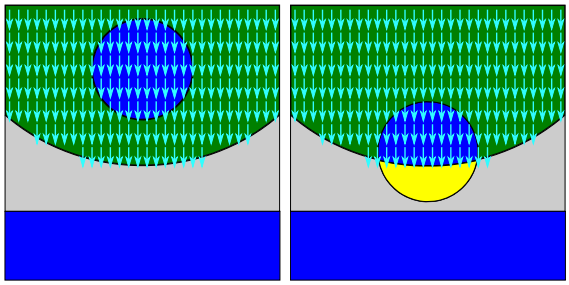


Figure 1: *Water pouring into a box at a resolution of* 512^3. *This example ran with a CFL number ranging from 10-60 and demonstrates the large amount of small scale details that can be achieved by using our method.*

additional difficulties with self intersection of the surface exacerbated by large time steps.

Since [LAF11] increased the visual accuracy of their simulations by conserving both the mass of the smoke and the momentum of the fluid, our main idea will be to enforce a conservation property for the volume of the liquid. In fact, volume conservation is a well-established idea that researchers have explored for some time. They started by advecting a Heaviside or color function, adding various techniques to recompress the interface representation aiming to keep it sharp in spite of the numerical smearing. One early approach was to treat advection by looking at the volume swept by faces (fluxes) and reconstructing the interface using a simple line interface calculation (SLIC) and later a piecewise linear interface calculation (PLIC) resulting in modern day volume-of-fluid (VOF) schemes (see e.g. [PP04] and the references therein). In order to avoid overlapping of the flux swept volumes, these methods must be applied one dimension at a time using dimensional splitting and the time step must be restricted to be one half of that allowed by a standard explicit scheme. Various visual artifacts result from this dimensional splitting, and the interface reconstruction ends up being discontinuous across cell boundaries. [SP00, Sus03] worked to ameliorate these issues by coupling VOF methods to level set methods where a locally constructed and advected level set function is used to compute surface normals alleviating some of the flotsam and jetsam (see also [MMS04, MUM*06]). However, their improvements did not completely eliminate such visual artifacts which would be exacerbated with very large time steps, especially when dimensional splitting is used. Another possible approach involves enforcing volume preservation globally as demonstrated in [KLL*07]. However, this would isotropically expand the blue region in Figure 2 as opposed to adding the yellow region to reconstruct the circle. Therefore, we do not take a VOF type or global volume control approach to the problem but instead use a color function type method that relies on semi-Lagrangian advection and sharpening (see Figure 3). [MMTD07] also proposed an approach to color function advection using a conservative flux based approach which is known to suffer from overshoots and undershoots – the thing VOF methods were created to address. [MMTD07] cited [FM07] as a method for taking larger time steps; however, this dimension by dimension approach suffers from the same limitations discussed in Figure 3.

Simply utilizing a color function type approach would suffer from all the same problems that originally led to the improved VOF methods or coupled level set VOF methods, and we would prefer to use a more visually pleasing particle level set method such as that proposed in [EMF02, EFFM02] (or even a variant such as Marker Level Set [MMS09]). Therefore, we start with the work of [EMF02, EFFM02] addressing various issues with large time steps, noting that without conservation the issue in Figure 2 seems implausible



(a) Extrapolated velocities using closest point extrapolation (b) Result after taking a large time step

Figure 2: *Figure 2a shows the analytic solution for the canonical closest point extrapolation scheme used in free surface flow simulation where the velocity field in the "air" is determined by the closest point in the water surface. This results in a velocity discontinuity along the curved equidistant boundary between the green and grey shaded regions. Everything above this curve has a downward velocity obtained from gravity acceleration of the falling drop whereas everything below this curve has a stationary velocity of 0 obtained from the stationary liquid at the bottom of the figure. For advection, the analytic solution using backwards cast semi-Lagrangian rays gives the result shown in Figure 2b in blue (not yellow) where everything above the curve moves downward and everything below the curve stays stationary. The actual analytic solution is shown by the union of the blue and yellow regions in the figure and we address the loss of mass depicted by the yellow region by instead forward advecting all of that material. One could forward advect the entire drop but that leads to significantly lower accuracy in the blue region where backwards advection works well.*

to address (see Section 3.1). Thus, we hybridize this method with a color function type approach in order to achieve an approximation of the yellow region in Figure 2, and subsequently outline the details for two way coupling of the methods. Our color function approach has many benefits over any dimension by dimension or flux based approach because of the use of the semi-Lagrangian advection from [LGF11] which is especially useful for large time steps (again as shown in Figure 3). The resulting method enables the simulation of free surface flow with time steps over an order of magnitude larger than implied by the CFL condition.

2. Free Surface Flows

Our fluid solver is based on the particle level set method [EMF02] and proceeds by solving the inviscid incompressible Navier-Stokes equations, which are given by

$$\mathbf{u}_t + \mathbf{u}\cdot\nabla\mathbf{u} = -\frac{1}{\rho}\nabla p + \mathbf{f}, \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{2}$$

where $\mathbf{u}$ is the velocity of the fluid, ρ is the density of the fluid, $\mathbf{f}$ is the sum of any external forces (such as gravity)

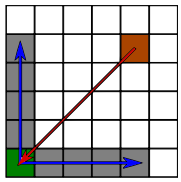


Figure 3: *Assuming a velocity field directed diagonally downward and to the left as depicted by the orange arrow, the correct volume information for the green cell would be obtained from the brown shaded cell. Any method which looks only in orthogonal directions would be limited to ascertain information only from the grey shaded cells depicted in the picture. One could imagine an alternating dimension by dimension exhaustive approach that scans all 25 cells in the neighborhood in order to eventually find the information in the brown cell, but the semi-Lagrangian method is far more efficient.*

scaled by ρ, and p is the fluid pressure. We solve these equations by first calculating an intermediate velocity field $\mathbf{u}^\star$ via

$$\frac{\mathbf{u}^\star - \mathbf{u}^n}{\Delta t} + (\mathbf{u}^n\cdot\nabla)\mathbf{u}^n = \mathbf{f}. \tag{3}$$

This equation is typically solved using the standard backward semi-Lagrangian advection scheme [Sta99] which requires an accurate approximation of the velocities in the region traversed by the fluid during a given time step. In the region not occupied by the liquid, one needs to obtain velocities for use in semi-Lagrangian advection, and this is typically done using any closest-point velocity extrapolation scheme within a band near the surface.

We then subsequently apply the pressure forces via

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^\star}{\Delta t} = -\frac{1}{\rho}\nabla p, \tag{4}$$

where the pressure is calculated by solving the Poisson equation

$$\nabla\cdot\frac{1}{\rho}\nabla\hat{p} = \nabla\cdot\mathbf{u}^\star, \tag{5}$$

where $\hat{p} = p\Delta t$. We also use the pressure modifications of [GFCK02] to achieve second-order accuracy at the surface.

3. Taking Large Time Steps

Generally speaking, there are two main steps in the simulation. The first step is to advect all quantities forward including the fluid velocities, level set, and particles after which the second step is the projection. Both of these steps require modifications for large time steps.

3.1. Problems With Extrapolation

The standard semi-Lagrangian advection requires looking back along characteristic rays. However, for free surface flows, the air region is not modeled and therefore does not have accurate velocities which need to be approximated. One method of obtaining these velocities is to add an air flow such as in [HK05, LSSF06]. The problem with this is that

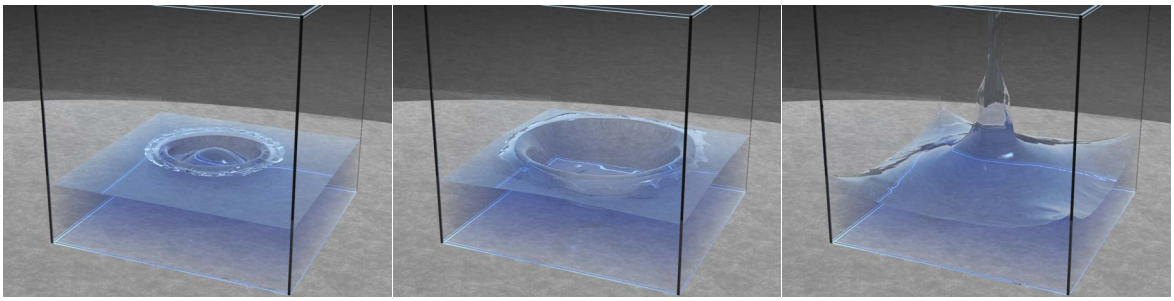


Figure 4: *Sphere of water dropped onto a stationary flat surface of water at a resolution of* 256^3. *This example ran with a CFL number ranging from 10-40. The early part of the simulation when the water first starts to fall was simulated with a smaller CFL number. The later parts of the simulation were ran with a higher CFL number.*

for many graphics simulations, we expect the air to have very little influence on large bodies of water. For example, a falling water drop will deform in a surrounding airflow, and although this can be ameliorated by surface tension, this adds an additional cost and complexity to the simulation.

Instead, as mentioned above, we fill the air region with a pseudo-velocity obtained via the canonical closest point extrapolation scheme which allows the water to move effectively unimpaired by the surrounding air. Typically one extrapolates about three grid cells; however, the extrapolation bandwidth must increase as the size of the time step increases. Previous approaches [CM11, LAF11] address the large time step problem by either performing a global extrapolation or by conserving momentum through advection; however, as seen in Figure 2, even with the analytic solution to the global problem, this method will fail. Other approaches such as [Cho09] which uses a multi-valued velocity field in order to handle collisions such as those shown in Figure 2, will fail in other cases such as when two spheres collide with each other. In this case, the algorithm would either cause the spheres to incorrectly pass through each other without a collision or overlay the spheres with each other resulting in a violation of volume conservation. The only way to properly handle the collision is to construct an extrapolation field that has knowledge of incompressibility.

This was addressed by [Sus03, REN*04] who used divergence free extrapolation to alleviate the issue by ensuring that the extrapolated velocity is discretely mass conserving. However, we observed that their proposed solution deforms a falling spherical drop well before merging takes place. This is because divergence free extrapolation solves a second projection in the narrow band around the outside of the drop with a free surface boundary condition applied on the outside of this narrow band. If this boundary condition is enforced by setting the cells to $p = 0$ without considering the interface geometry, the drop deforms. This deformation of the drop can be resolved by using the second order cut cell method from [GFCK02] on the outer edge of the band (which we point out for the first time in the paper). While this works to improve merging for the small extrapolation bands required when taking small time steps, when it is applied globally or over the large extrapolation bands required when taking large time steps to remove the discontinuity shown in Figure 2, it will produce a velocity field that deforms the drop long before it hits the water surface. Thus, this method, while applicable for improving merging at small time steps, does not alleviate the issues with large time steps.

3.2. Advection

As discussed in the previous section, closest-point extrapolation methods are unable to approximate a velocity field far away from the interface that is usable with large time steps. Thus, we only extrapolate within the standard three grid cell band. Then, we advect the velocity using the conservative method of [LAF11] who show that this method works for globally defined velocity fields and large time steps, as well as small time steps with free surfaces, but do not show that it works for large time steps with free surfaces. However, the method can be used to advect velocities (albeit not a surface) in that case as well without any modifications. Basically, when cells within the three grid cell band look back to advect velocity forward, any velocities that are missed that would have been updated from a larger band are updated in the second step of forward advection which is required in order to conserve the momentum.

Second order Runge-Kutta particle advection can fail at large time steps without an accurate fluid or extrapolated velocity. We advect the particles with forward Euler if a velocity sampled within an RK2 step is outside the extrapolated velocity band allowing us to maintain the higher-order accuracy of the RK2 method in the presence of accurate velocities while improving the robustness of the method when such velocities are absent. Level set advection also has problems at large time steps. While the particle level set method maintains the shape of the air-water interface at small time steps, it does not typically preserve the amount of volume within the liquid region. At small time steps, these changes in volume are small and are unnoticeable in the animation. However, when the particle level set is evolved with a large time step, this volume loss becomes more apparent and detracts significantly from the visual plausibility of the simulation. Furthermore, applying a conservative advection scheme to the level set itself is insufficient to resolve this volume loss

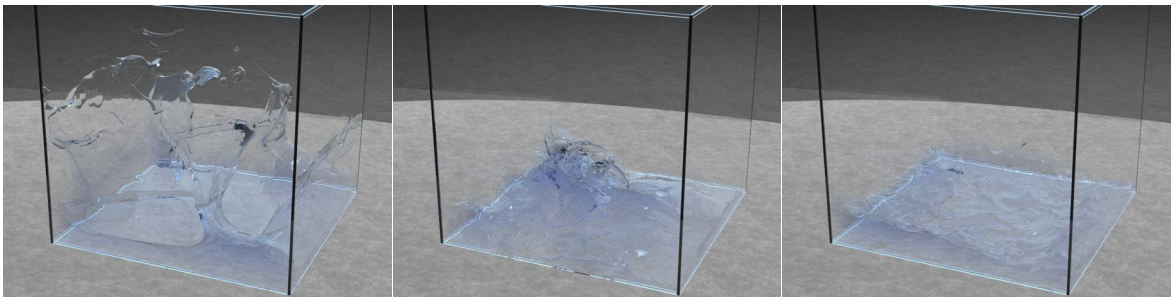


Figure 5: *Dam break water example at a resolution of* 256^3. *This example ran with a CFL number ranging from 10-40. The early part of the simulation when the water first starts to fall under the influence of gravity was simulated with a smaller CFL number. The later parts of the simulation were ran with a higher CFL number.*

because the conservation of signed distance values does not imply conservation of any physical quantity such as volume. We resolve this by advecting a color function using the conservative method of [LAF11] in conjunction with the particle level set surface as discussed in Section 4.

3.3. Projection

We solve the projection step only in liquid regions while enforcing a free surface boundary condition. As a result, small regions of "air" (not in a liquid region) can create large divergences. While this is acceptable with small time steps due to the restrictions on the motion of the fluid, this results in a large amount of volume loss when taking large time steps. One method of solving this problem is to explicitly fill the "air" region and enforce a target divergence [LTKF08] such that the cell is exactly filled in a given time step. However, this method requires the algorithm to detect whether or not the cell will be filled or overfilled for a given velocity field and time step. While simple algorithms, such as thresholding based on the size of the "air" region and the CFL number can be used as an approximate detection scheme, it is difficult to obtain a set of parameters that are suitable for a general fluid flow. We use a threshold that causes the algorithm to only fill small "air" pockets. For larger regions, we instead enforce volume conservation using our conservative color function treatment. This can sometimes lead to cells with $V > 1$, and this is treated along with other advection errors as described in Section 4.

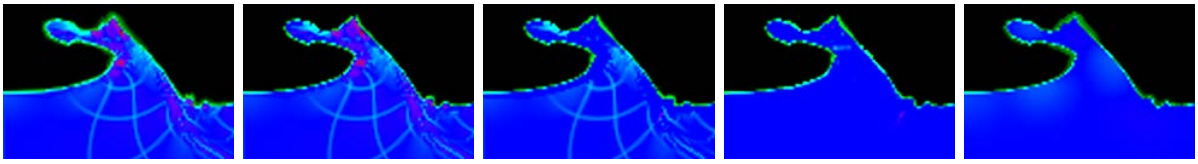
4. Color Function

In addition to the level set function, we also evolve a color function V which represents the fraction of liquid contained within a region (usually a single cell in the case of a uniform grid). Although the color function can be advected using any method, volume is conserved only if the method is conservative. However, numerical smearing, even in a conservative method, will cause inaccuracies in V's spatial distribution. One of the main approaches for increasing accuracy are the VOF type methods. Early on, SLIC approximated the interface in each cell using a plane defined by a normal along one of the Cartesian directions, placing it in a location that accurately represented the fraction of volume in a cell. PLIC later improved on this by allowing for a normal in any direction. Then, advection was carried out by intersecting the material in a cell with the volume swept out by a cell face along its normal during a time step. As mentioned in Section 1, in order to be conservative, these methods have a CFL restriction of 0.5 and produce a number of visual artifacts. [SP00] attempted to improve the VOF method by constructing a level set from the interface which is used to compute the tangent plane normals. The surface reconstructions of this CLSVOF method are typically not as smooth as those from the particle level set method as can seen by the examples of [MMS04, MUM*06], and the resulting visual artifacts will be highly exacerbated at large time steps. While previous methods have achieved limited success for CFL numbers slightly larger than 1 [FM07, CM11], the artifacts will be highly exacerbated for CFL numbers in the range of 10-40 as we take in our examples.

Our method using the color function V is conceptually similar to the VOF method. If $V = 1$ or $V = 0$, the region contains either all liquid or all "air" respectively. If $0 < V < 1$, the region contains both liquid and air. First, to compute the initial data, we compute the color function from a piecewise linear rasterization of the level set. Note that when ϕ is changed for fluid sources, the color function also needs to be modified to maintain consistency. Then we advect the color function forward in time as per

$$V_t + \mathbf{u} \cdot \nabla V = 0 \tag{6}$$

using the conservative method of [LGF11]. Note that we do not use any of the diffusion methods used when advecting smoke density [LAF11]. Our experiments show that diffusion is detrimental in the case of color function advection, since diffusion is based on the heat equation which tends to move the interface faster in regions of higher curvature than lower curvature. This problem is alleviated for momentum advection because diffusion is not applied across the interface, which is well determined before the momentum advection step. The conservative advection method can also result in excess volume accumulating where fluid collides with a rigid body due to the clamping of semi-Lagrangian rays to the surface of the object. This can result in an undesirable



(a) (b) (c) (d) (e)

Figure 6: *Correcting Numerical Dissipation in the Color Function: Figure 6a shows the color function after advection. Due to numerical dissipation, there are regions inside the level set ($\phi \leq 0$) with a color function value $V = 1$ (blue), $V < 1$ (cyan), and $V > 1$ (magenta) along with regions outside the level set ($\phi > 0$) with a color function value $V = 0$ (black) and $V > 0$ (green). Figures 6b, 6c, and 6d demonstrate the results after applying our first, second, and third compression step respectively. We then apply a volume-conserving diffusion algorithm which corrects for errors in compression (usually located in regions of high curvature) to obtain a more accurate color function surface representation shown in Figure 6e.*

viscous fluid appearance which we avoid by reflecting the semi-Lagrangian rays off the surface, and scaling them by a damping coefficient.

After the color function has been conservatively advected forward in time, it will still smear out due to numerical dissipation. We apply a compression procedure diagrammed in Figure 6 to the color function after it has been advected forward in time. First, we compute the .5 isocontour of the color function. Then, we aim to set all cells within the .5 isocontour to have $V = 1$ and all cells outside the .5 isocontour to have $V = 0$. There are two primary discrepancies that need to be addressed. The first problem is that V values outside the isocontour can be more than 0 due to numerical smearing. The second problem is that V values inside the isocontour are not necessarily 1 (can be either more than or less than 1). Although interior cells with $V > 1$ can result from errors in advection, this can also occur from the collapse of air pockets as discussed in Section 3.3. To correct for these errors, we compute differences between the target (0 or 1) and current color function value and then move this difference in the direction of the gradient of V to the interface. For $V > 0$ outside the interface, we move in the direction of the gradient of V until we are at a location $2\Delta x$ interior to the $V = .5$ isocontour. Then, we distribute the difference in color function value to the neighboring cells around this location (excluding cells lying within rigid bodies) in weighted proportions until $V = 1$ in each cell. If all of the difference cannot be distributed at these neighboring cells without overflowing the cells, the remainder of the difference is placed in a location obtained by marching outwards along the gradient of V. For the cells on the interior of the $V = .5$ isocontour with $V > 1$ the difference in color function value $V - 1$ is distributed to the surface in the same way starting from a distance of $2\Delta x$ inside the surface. In the third case, where $V < 1$ for interior cells, we fill V for each cell to 1 using color function values from the surface by distributing a negative value $V - 1$ starting $2\Delta x$ outside the interface and marching inwards in a similar manner in the opposite direction as the first two cases. Both our advection scheme and compression scheme guarantee that values of $V < 0$ cannot occur; however, the compression method can be modified to account for these if a different advection scheme that can result in negative values is used. The compression is done in a Gauss-Seidel fashion using a single iteration (multiple iterations could be used but are not needed since its done at every time step), and thus the order in which the cells are visited matters. One might want to use a method such as Gauss-Jacobi to reduce bias but this will require more iterations, and we have found that Gauss-Seidel is satisfactory for our purposes. This is partially because we couple it with particle level set method for accuracy whenever possible (see Section 5).

5. Coupling

First, we advect both ϕ and V forward in time using semi-Lagrangian advection for ϕ and conservative semi-Lagrangian advection for V. We then correct the values of ϕ using the particles as in the particle level set method, and subsequently reinitialize ϕ. However, we stress that the second particle correction of the reinitialized level set is not yet applied, and the color function is not yet compressed. At this point, we construct a distance function from the color function in almost the same way that [SP00] construct it at the beginning of the time step. That is, we first use a VOF construction of the color function, computing a normal from the gradient of V and the placement of the tangent plane in the cell from the value of V itself, and then initialize values of the signed distance function ϕ^V based on the tangent plane in that cell. Nearby cells are initialized with values from the fast marching method. Unlike [SP00], we compute the normals directly from the gradient of V as opposed to an advected ϕ. This is because an advected ϕ function would be highly inaccurate due to the issue of velocity extrapolation.

While the color function does not represent the interface as accurately as the particle level set method, it is often the only reasonably accurate interface representation available when taking large time steps demonstrated by the yellow region of Figure 2. Thus, our aim is to use the particle level set method wherever possible in order to gain visually pleasing surfaces and only to use the color function to represent the surface where the level set representation has lost large amounts of mass such as shown in Figure 8. Generally speaking if the interfaces agree we use the particle level set representation of the interface. We say

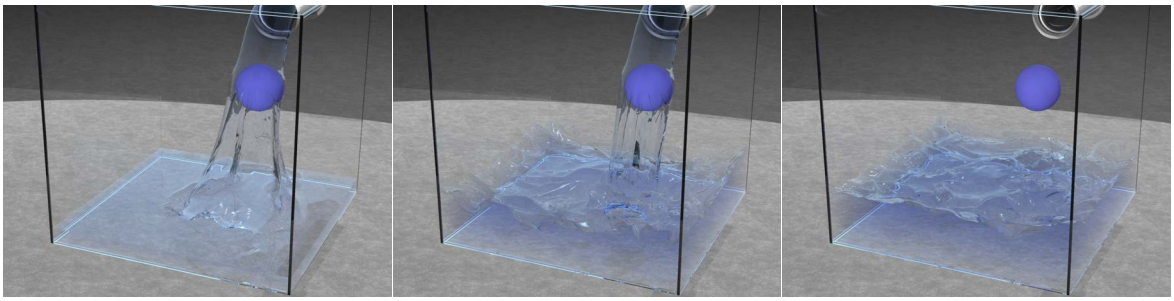
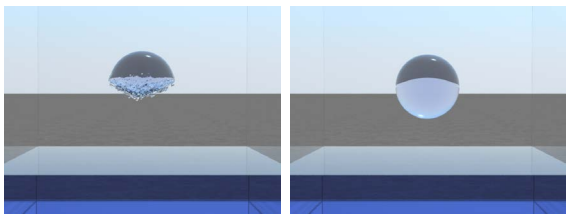


Figure 7: *Water pouring into a box over a rigid sphere at a resolution of* 256^3. *This example ran with a CFL number ranging from 10-40. The earlier section of the simulation from when the source is first activated until the water first hits bottom of the container and the later section after the source was turned off and the water starts to calm were simulated with a smaller CFL number. The more active parts of the simulation which occur between the water from the source hitting the bottom of the container and the source turning off were ran with a higher CFL number.*

that the two interfaces agree if they do not differ by more than $.5\Delta x$ in distance. At every cell (i,j,k) in the computational domain, we compute a blending parameter $\alpha_{i,j,k} = |\phi_{i,j,k} - \phi^V_{i,j,k}|/\Delta x - .5$ and clamp it in the range $[0,1]$. Subsequently, we compute the blended level set function ϕ^B via $\phi^B_{i,j,k} = (1-\alpha_{i,j,k})\phi_{i,j,k} + \alpha_{i,j,k}\phi^V_{i,j,k}$. Note that the minimum pre-clamped value of $\alpha_{i,j,k}$ is $-.5$ and thus when the interface values agree to within half a grid cell, we use the level set version. For $\alpha_{i,j,k} \geq 1$, we use the color function representation, and otherwise we interpolate between them. In order to avoid visual kinks in ϕ^B, α is smoothed locally on the grid using a Gaussian filter before linearly interpolating ϕ and ϕ^V to obtain ϕ^B.

Next, ϕ^B is initialized to a signed distance function. In order to prevent spurious particle corrections to the blended level set representation, a particle is deleted if $\alpha > 0$ at the closest surface cell. This signifies that the closest interface in the cell has changed from the using the ϕ obtained from PLS and is instead using the color function to help define the interface, thus rendering the particle correction incorrect in that cell. Particles are then only reseeded in the areas with $\alpha > 0$ in order to preserve sharp features that cannot be captured by the level set alone. Finally, the reinitialized ϕ^B is corrected via particles in the usual fashion of the particle level set method.



(a) PLS method (b) Our method

Figure 8: *Figure 8a shows the result we get after taking a large time step using the PLS method. Because closest-point extrapolation discussed in 3.1 is unable to provide a good approximation of the velocities in the air region, backward semi-Lagrangian advection fails to accurately advect the level set causing the ball to become clipped at the bottom. Figure 8b shows the same frame using our method where we are able to reconstruct and subsequently maintain the shape of the ball even when taking a very large time step.*

In summary, we advect the level set function ϕ, correct it with particles and reinitialize it, advect the color function V, contour it with a VOF method and produce a signed distance function representation ϕ^V, blend ϕ and ϕ^V to obtain ϕ^B, reinitialize the result, delete particles where the color function representation is being used, reseed particles around the new interface generated by the color function, and correct the combined level set using the particles. We denote the final result once again as ϕ.

At this point, we have the desired level set reconstruction but a smeared out color function. We use the compression method listed in Section 4 to compress the color function using the more accurate normals of ϕ instead of the less accurate gradients of the color function. Moreover, we can accelerate the process of finding the zero isocontour and nearby locations using the fact that ϕ is a signed distance function. We stress that this compression does not use isocontours or geometric information from V in any way but instead strives to compress V into the newly generated ϕ.

While this compression method works well, we have noticed that in regions of high and low curvatures compression tends to produce errors near the interface such as $V > 0$ color function values accumulating on the peaks of the waves. (see Figure 6). Therefore, we propose a diffusion based algorithm, similar to the ones found in [FL04, LAF11] to improve the color function's surface representation. We take the union of the surfaces given by ϕ^V and ϕ and use compression outside this region and diffusion inside. We diffuse the error $e = V - V^\phi$, where V^ϕ is the color function obtained from a piecewise linear rasterization of ϕ, in the color function relative to the level set close to the surface. We have also experimented with solving the Poisson equation with the appropriate target divergences to generate a velocity field inside the union which works as well but is much more expensive than using this sweeping method. After the compression and diffusion steps, the color function yields an improved

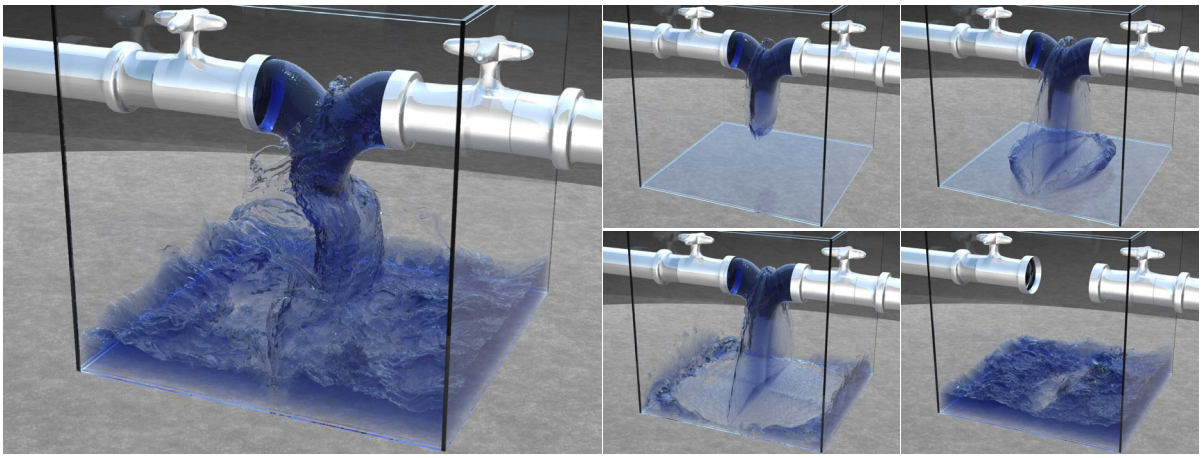


Figure 9: *Water pouring into a box from two opposite-facing sources at a resolution of* 512^3*. This example ran with a CFL number ranging from 10-60. The earlier section of the simulation when the sources are first activated until the water from the sources collides was simulated with a smaller CFL number. The remainder of the simulation was ran with a higher CFL number. Note that we increase the amount of absorption during rendering in order to make the fine scale details more apparent.*

surface representation (as shown in Figure 6) and in order to improve the visual appearance of the final surface, we repeat the blending process using the current V and ϕ.

6. Results

We demonstrated our method on a number of 3D examples as seen in Figures 1, 4, 5, 7, 9, and 11. We first ran a baseline simulation for each example using the standard PLS method at a CFL number of 1 at resolutions of 64^3 and 128^3. However, these simulations were too slow to run in a practical amount of time at the higher resolutions such as 256^3. A

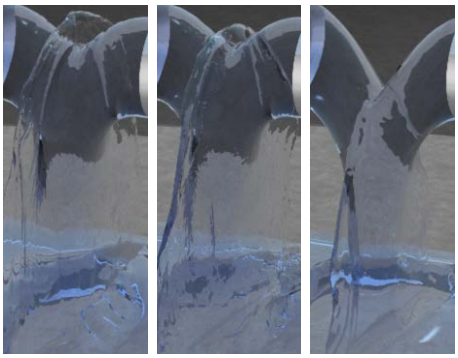


Figure 10: *Comparisons between our method and the particle level set method using similar amounts of computation. The left figure shows the correct answer obtained by running the particle level set algorithm at a resolution of* 256^3 *at CFL 1. The middle figure shows the results of running our algorithm at a resolution of* 256^3 *at CFL 16. This requires a similar amount of computation to running the particle level set method at a resolution of* 128^3 *at CFL 1 which gives the results shown in the right figure. Our algorithm has the same degree of numerical viscosity as the* 256^3 *particle level set simulation but requires the same amount of computation as the* 128^3 *since the resolution is doubled in each dimension and the CFL condition becomes twice as strict.*

CFL number of 1 means that advection does not move information farther than 1 grid cell in any time step. Then, we then ran both the standard PLS method and our method at a high CFL number of 40 at resolutions of 64^3, 128^3 and 256^3. For resolutions of 64^3 and 128^3, this equates to running at the frame rate (meaning one time step per frame) since the maximum CFL number was approximately 5 for 64^3 resolution simulations and approximately 20 for 128^3 resolution simulations. We also ran our method at a resolution of 512^3 in order to demonstrate the large resolution simulations that can be obtained by our method at high CFL numbers. For this simulation, we primarily ran with a CFL number of 60. Figure 10 shows a comparison between our method and the particle level set method using similar amounts of computation. Figure 12 shows a comparison of the volume between the traditional PLS algorithm and our method at a resolution of 128^3. Note the large amount of volume lost without using our method. Compared to the standard particle level set method, the additional steps in our method cause it to execute about twice as slowly for a given time step. However, since our time step is allowed to be 40 times larger, our method approximately achieves a 20 times increase in performance on frames where the CFL number is close to 40 or greater. For frames in specific simulations where the CFL numbers are smaller than 40 (e.g. stationary sphere before it begins falling into water under the influence of gravity), we achieve smaller increases in performance. We emphasize that this is due to the maximum time step size being restricted by the user-specified frame rate for the simulation. Similarly, for the 256^3 resolutions, we ran both the standard PLS method and our method. Our larger examples were ran with a fully parallelized MPI implementation which scales well at relatively low CFL numbers but when running this large of a simulation, our CFL number becomes limited because our current MPI implementation divides the compu-

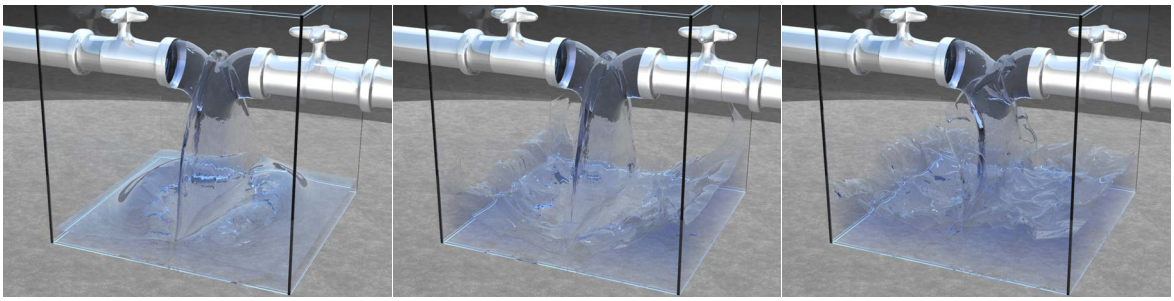


Figure 11: *Water pouring into a box from two opposite-facing sources at a resolution of* 256^3. *This example ran with a CFL number ranging from 10-40. The earlier section of the simulation when the sources are first activated until the water first hits the bottom of the container was simulated with a smaller CFL number. The more active parts of the simulation which occur after the water from the sources hits the bottom of the container were ran with a higher CFL number.*

tational domain into a number of smaller domains and we have not generated the code to cross more than one processor boundary (although this can be done). The limitations on the CFL number of simulations due to the size of the MPI domains bring up the important issue that with a CFL number equal to the size of one of these subdomains, a ghost cell implementation requires 27 times more data for ghost cells than the actual simulation (i.e. each grid being replicated in all Cartesian directions and partially replicated in the diagonal directions) which can quickly deplete memory resources. In the standard ghost cell implementation, it might be advantageous for MPI code (although unnecessary for threaded code) to change the boundary process to have each grid request data from the neighboring grids that contain the needed grid cells, sending that request out to other processors which send the information back. Even so, if every grid cell in the domain is requesting data from another processor, this is equivalent to copying entire grids from one processor to another at every time step, and thus while doubling the CFL number does halve the required number of pressure solves and other steps such as those involved in the particle level set method, it increases the communication cost. Thus, one cannot indefinitely increase the CFL number although we noticed issues in visual accuracy at lower CFL numbers than would lead to concerns with communication bottlenecks. For example, during the first step of a simulation with a horizontally facing source, the velocities are purely horizontal and thus when a large CFL number is taken, the liquid will move a large distance parallel to the ground despite the fact that the fluid should be falling. One alternatively could apply gravity first but that would mean advecting in a divergent flow field which can cause objects to incorrectly advect through solid walls. Instead, these problems are partially resolved by running with a smaller CFL number when such visual artifacts manifest which typically occurs in slow moving fluid flows.

It is important to note that although we can maintain a similar time step size at larger resolutions, the algorithm does scale with the number of grid cells. Therefore, the simulation is a factor of 8 slower when the grid size doubles for a fixed time step (which requires doubling the CFL number of the simulation). However, a number of methods including [LZF10, MST10] improve the scalability when the resolution changes. These methods can be used in conjunction with our method to create even faster simulations.

7. Conclusion

We have presented a novel method method for accurately simulating free surface flow even when taking time steps that are more than an order of magnitude larger than implied by the CFL condition. We accomplish this by conservatively advecting a color function representing the fraction of volume contained in each cell along with the particle level set and using the color function to compute an accurate representation of the surface where the particle level set representation fails to suffice. This allows our method to maintain volume and subsequently achieve visually accurate results. Moreover, we have presented a general framework for combining an interface tracking method with a volume tracking method. Investigating the use of other interface tracking methods and volume tracking methods for use within this framework is a potential subject of future work.

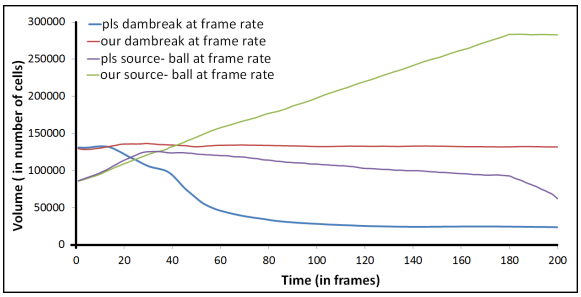


Figure 12: *Volume vs. time for four different simulations. The red and blue lines represent the liquid volume present in the dam break simulation at frame rate. Notice how our method (red) fully conserves volume while the PLS algorithm (blue) loses a tremendous amount of volume. The green and purple lines represent the volume present in a simulation with water from a source flowing over a ball. Notice how the volume increases linearly until the source is turned off when using our method (green) but decreases slowly when using the PLS algorithm (purple).*

Acknowledgements

Research supported in part by ONR N00014-06-1-0505, ONR N00014-09-1-0101, ONR N-00014-11-1-0027, ARL AHPCRC W911NF-07-0027, NSF IIS-1048573, and the Intel Science and Technology Center for Visual Computing. Computing resources were provided in part by ONR N00014-05-1-0479. We would like to thank Christos Kozyrakis for additional computing resources as well as Jacob Leverich for helping us use those resources. We would also like to thank Avi Robinson-Mosher for his early work on coupling PLS and VOF methods.

References

[BBB10] BROCHU T., BATTY C., BRIDSON R.: Matching fluid simulation elements to surface geometry and topology. *ACM Trans. Graph.* (2010). 1

[BGB11] BHATACHARYA H., GAO Y., BARGTEIL A.: A level-set method for skinning animated particle data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2011), SCA '11. 1

[Cho09] CHOPP D.: Another look at velocity extensions in the level set method. *SIAM J. Sci. Comput. 31* (2009), 3255–3273. 4

[CM11] CHENTANEZ N., MÜLLER M.: Real-time Eulerian water simulation using a restricted tall cell grid. In *Proc. of ACM SIGGRAPH 2011* (2011), pp. 82:1–82:10. 4, 5

[DC96] DESBRUN M., CANI M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Comput. Anim. and Sim. '96 (Proc. of EG Wrkshp. on Anim. and Sim.)* (Aug 1996), Boulic R., Hegron G., (Eds.), Springer-Verlag, pp. 61–76. 1

[EFFM02] ENRIGHT D., FEDKIW R., FERZIGER J., MITCHELL I.: A hybrid particle level set method for improved interface capturing. *J. Comput. Phys. 183* (2002), 83–116. 2

[EMF02] ENRIGHT D., MARSCHNER S., FEDKIW R.: Animation and rendering of complex water surfaces. *ACM Trans. Graph. (SIGGRAPH Proc.) 21*, 3 (2002), 736–744. 2, 3

[FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. In *Proc. of ACM SIGGRAPH 2001* (2001), pp. 23–30. 1

[FL04] FATTAL R., LISCHINSKI D.: Target-driven smoke animation. *ACM Trans. Graph. (SIGGRAPH Proc.) 23* (2004), 441–448. 7

[FM97] FOSTER N., METAXAS D.: Controlling fluid animation. In *Comput. Graph. Int.* (1997), pp. 178–188. 1

[FM07] FROLKOVIC P., MIKULA K.: High-resolution flux-based level set method. *SIAM J. Sci. Comput. 29* (2007), 579–597. 2, 5

[GFCK02] GIBOU F., FEDKIW R., CHENG L.-T., KANG M.: A second-order-accurate symmetric discretization of the Poisson equation on irregular domains. *J. Comput. Phys. 176* (2002), 205–227. 3, 4

[HK05] HONG J.-M., KIM C.-H.: Discontinuous fluids. *ACM Trans. Graph. (SIGGRAPH Proc.) 24*, 3 (2005), 915–920. 3

[KLL*07] KIM B., LIU Y., LLAMAS I., JIAO X., ROSSIGNAC J.: Simulation of bubbles in foam with the volume control method. In *Proc. of ACM SIGGRAPH 2007* (2007), pp. 98:1–98:10. 2

[LAF11] LENTINE M., AANJANEYA M., FEDKIW R.: Mass and momentum conservation for fluid simulation. In *SCA '11: Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2011), pp. 91–100. 1, 2, 4, 5, 7

[LGF11] LENTINE M., GRÉTARSSON J., FEDKIW R.: An unconditionally stable fully conservative semi-lagrangian method. *J. Comput. Phys. 230* (2011), 2857–2879. 3, 5

[LSSF06] LOSASSO F., SHINAR T., SELLE A., FEDKIW R.: Multiple interacting liquids. *ACM Trans. Graph. (SIGGRAPH Proc.) 25*, 3 (2006), 812–819. 3

[LTKF08] LOSASSO F., TALTON J., KWATRA N., FEDKIW R.: Two-way coupled sph and particle level set fluid simulation. *IEEE TVCG 14*, 4 (2008), 797–804. 1, 5

[LZF10] LENTINE M., ZHENG W., FEDKIW R.: A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Transactions on Graphics* (July 2010). 9

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *Proc. of the 2003 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* (2003), pp. 154–159. 1

[MMS04] MIHALEF V., METAXAS D., SUSSMAN M.: Animation and control of breaking waves. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* (2004), pp. 315–324. 2, 5

[MMS09] MIHALEF V., METAXAS D. N., SUSSMAN M.: Simulation of two-phase flow with sub-scale droplet and bubble effects. *Comput. Graph. Forum* (2009). 2

[MMTD07] MULLEN P., MCKENZIE A., TONG Y., DESBRUN M.: A variational approach to eulerian geometry processing. *ACM Trans. Graph.* (2007), –1–1. 2

[MST10] MCADAMS A., SIFAKIS E., TERAN J.: A parallel multigrid poisson solver for fluids simulation on large grids. In *SCA/Eurographics Symp. on Comput. Anim.* (2010), pp. 1–10. 9

[MUM*06] MIHALEF V., UNLUSU B., METAXAS D., SUSSMAN M., HUSSAINI M.: Physics based boiling simulation. In *SCA '06: Proc. of the 2006 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* (2006), pp. 317–324. 2, 5

[PP04] PILLIOD J., PUCKETT E.: Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J. Comput. Phys. 199* (2004), 465–502. 2

[REN*04] RASMUSSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* (2004), pp. 193–202. 4

[SP00] SUSSMAN M., PUCKETT E.: A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys. 162* (2000), 301–337. 2, 5, 6

[Sta99] STAM J.: Stable fluids. In *Proc. of SIGGRAPH 99* (1999), pp. 121–128. 1, 3

[Sus03] SUSSMAN M.: A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comput. Phys. 187* (2003), 110–136. 2, 4

[WTGT10] WOJTAN C., THÜREY N., GROSS M. H., TURK G.: Physics-inspired topology changes for thin fluid features. *ACM Trans. Graph.* (2010). 1

[YT10] YU J., TURK G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proc. of the 2010 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* (2010). 1

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph. (SIGGRAPH Proc.) 24*, 3 (2005), 965–972. 1