Ghost SPH for Animating Water

Hagit Schechter*
University of British Columbia

Robert Bridson*
University of British Columbia

Abstract

We propose a new ghost fluid approach for free surface and solid boundary conditions in Smoothed Particle Hydrodynamics (SPH) liquid simulations. Prior methods either suffer from a spurious numerical surface tension artifact or drift away from the mass conservation constraint, and do not capture realistic cohesion of liquid to solids. Our Ghost SPH scheme resolves this with a new particle sampling algorithm to create a narrow layer of ghost particles in the surrounding air and solid, with careful extrapolation and treatment of fluid variables to reflect the boundary conditions. We also provide a new, simpler form of artificial viscosity based on XSPH. Examples demonstrate how the new approach captures real liquid behaviour previously unattainable by SPH with very little extra cost.

CR Categories: Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Animation; Computer Graphics [I.3.5]: Computational Geometry and Object Modeling—Physically based modeling

Keywords: Smoothed Particles Hydrodynamics, SPH, free surface, boundary conditions, volume sampling, liquids, animation

Links: DL PDF

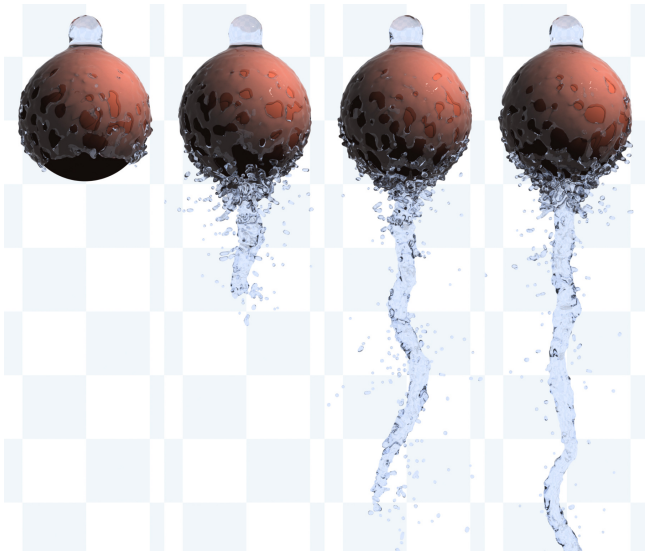


Figure 1: *With our new boundary conditions, water pouring on to a sphere properly coheres to the underside, giving a realistic stream without excessive numerical surface tension.*

1 Introduction

Animating liquids like water via physical simulation remains a beguiling area. Many methods have been developed in graphics to solve various problems, both for realism of the results and for performance. Our focus here is Smoothed Particle Hydrodynamics (SPH), where the fluid is represented by material particles with relatively simple interactions via sums of smooth kernel functions and their gradients: see Monaghan for a detailed overview [2005], which we assume as background knowledge. Most SPH methods enjoy automatic mass conservation (each particle represents a fixed amount of conserved mass, density is estimated by direct summation), accurate tracking of the fluid through space (due to the Lagrangian representation), and a conceptually simple algorithmic kernel (summing weighted kernels over nearby particles, with no systems of equations to solve or tricky geometric discretizations).

However, several problems remain: here we tackle the treatment of free surface and solid boundary conditions. Mass-conserving SPH methods, where density is estimated by a sum rather than evolved as a state variable, suffer serious errors near free surfaces causing unavoidable surface-tension-like artifacts and reduced stability. Prior treatments of arbitrary solid boundaries do not reflect the correct cohesive behaviour visible in many real world scenarios. Our Ghost SPH method resolves both by careful sampling of ghost particles in the air and solid regions near the fluid, and appropriate extrapolation of fluid quantities to the ghost particles. Our new sampling algorithm, an extension of a Poisson-disk method which accurately captures boundaries, is also of interest to other applications.

*e-mail: (hagitsch|rbridson)@cs.ubc.ca

2 Related Work

SPH was independently introduced by Gingold and Monaghan [1977] and Lucy [1977], and has since seen extensive use in physics research. The advantages above have also made it popular in computer graphics for a variety of liquid phenomena.

Monaghan's adaptation [1994] of SPH to free-surface flow serves as a basis for many later works. Müller et al. [2003] used a low stiffness equation of state along with surface tension and viscosity forces for interactive applications. Refined particle sampling near free surfaces for accuracy or efficiency is discussed in several works [Keiser et al. 2006; Adams et al. 2007; Solenthaler and Gross 2011]. Becker and Teschner [2007] reduce compressibility with the stiffer "Tait Equation", and introduce particle initialization with highly damped equations to reach a stable density near the surface, which we directly solve with ghost air particles.

Müller et al. seeded air particles to model bubbles [2003]. Keiser also used a narrow band of air particles to aid in surface tension and small bubbles [2006] — however, whereas this work's mirroring approach may create air sampling with widely varying density, we always seed ghost air pairtlces with a near-rest-state distribution.

Bonet and Kulasegaram [2002] modified the underlying SPH method with corrections to make the scheme numerically consis-

tent, resolving density problems near boundaries as we do, but with significantly heavier calculation.

Müller et al [2005] achieved two-way SPH fluid interaction using "color field"-derived curvature. Solenthaler and Pajarola [2008] introduced a modified density equation for accurate SPH density computation at the interface between the two fluids. However, this is not directly applicable to free surfaces.

Many SPH works represent solids with particles. Repulsion forces from solid particles are often used to avoid fluid penetrating solids [Monaghan 1994]. Two-way interaction between fluid and solid particles is also common [Keiser et al. 2006; Becker et al. 2009]. Colagrossi and Landrini more accurately captured velocity boundary conditions at solids by mirroring nearby fluid particles, but were limited to simple solid geometry [2003]. We handle velocities in a similar manner, but with solid bodies of arbitrary shape. Ihmsen et al. [2010], like us, extrapolated fluid quantities into solid particles to improve solid boundary conditions, avoiding "stacking" and irregular pressure distributions in standing water situations; we further capture accurate cohesion to solids.

Fedkiw and collaborators' "Ghost Fluid Method" [1999], used extensively for Eulerian fluids including coupling to Lagrangian solids [Fedkiw 2002], inspires our velocity and density extrapolation to ghost particles at both solid and free surface boundaries.

Artificial viscosity is critical in SPH for stability and regularity. Many methods use Lucy's equations [1977], e.g. [Monaghan 1994; Becker and Teschner 2007]. Müller et al. suggested another formulation based on second derivatives of a specially designed kernel [Müller et al. 2005; Müller et al. 2003]. Our artificial viscosity approach instead derives from the Monaghan's "XSPH" position update for handling shocks [1989]; we extend this to a velocity update, and find it more intuitive and controllable than prior forms.

Solenthaler and Pajarola's PCISPH [2009] replaces the equation of state, with its stability time step restriction, by an iterative solver for bringing the system to incompressibility. PCISPH allows much larger time steps and reduces the need for parameter tuning. While our examples use an equation of state, we use the same summed-mass density and thus our Ghost SPH method can just as easily be used with PCISPH.

Our seeding algorithm is based on blue noise sampling original developed for rendering. Cook [1986] emphasized the virtues of Poisson-disk distributions for blue noise. Turk [1992] introduced a relaxation method for surface sampling. Dunbar et al. [2006] modified dart throwing to efficiently generate Poisson-disk distributions in 2D; Bridson [2007] simplified this with rejection sampling, extending it to higher dimensions with minimal implementation effort. We extend Bridson's algorithm to tightly sample the boundary of a domain as well as its interior.

3 The Basic Method

There is a wide variety of SPH methods; here we introduce our notation and present a common set of choices for graphics, which we call "Basic SPH". We also call attention to the problems we later will solve with our Ghost-SPH model.

Particle i has position $\mathbf{x}_i$, velocity $\mathbf{v}_i$, and mass m_i (uniform across all particles), with acceleration $\mathbf{a}_i$ computed from force divided by m_i. Kernel function W is radially symmetric with a finite radius of support: our examples use Monaghan's M_4 cubic spline [2005] with radius of support $3h$ for an average particle spacing of h, but this is orthogonal to the contributions of the paper.

Density Evaluation: One critical choice is how to compute density ρ_i at particle i. We advocate the popular direct summation estimate,

$$\rho_i = \sum_j m_j W_{ij}, \tag{1}$$

where $W_{ij} = W(\mathbf{x}_i - \mathbf{x}_j)$. This naturally conserves mass, and forces computed from this density directly correct deficiencies in the particle distribution, generally producing higher quality results.

Equation (1) is problematic near free surfaces. In the interior particles are surrounded by other particles, so an even distribution gives a roughly uniform density. Near a free surface, however, the air part of a particle's neighborhood is empty and the same distribution gives a much lower density estimate: see Figure 3. The equation of state then causes particles to unnaturally cluster in a shell around the liquid, rebalancing density but causing a strong surface-tension-like artifact, stability and accuracy issues due to the distorted distribution, and a deformed initial shape as in Figure 4b and d.

An alternative is to evolve the density as an independent quantity with $\partial\rho/\partial t + \nabla \cdot (\rho \mathbf{v}) = 0$ [Monaghan 1994]: this eliminates the free surface problem, but discretization and integration errors in this equation permit drift from true mass conservation, leading to steadily worse particle distributions particularly around splashes.

Solid Boundaries: We represent solids by sampling with particles. The most common way of preventing liquid particles from penetrating solids is to apply repulsive forces near solid particles: our Basic-SPH examples use the Lennard-Jones approach advanced by Monaghan [1994]. However, while this aims to prevent liquid from entering a solid, it freely allows liquid to separate from a solid — which is often just wrong. In typical scenarios, liquid can only separate from a solid if air can rush in to fill the gap: by default liquid shouldn't be able to leave, leading to the standard no-stick $\mathbf{v} \cdot \hat{\mathbf{n}} = 0$ boundary condition. This provides a visually critical form of cohesion, quite apart from surface tension, which our ghost treatment of solid boundaries enables (see Figures 1 and 11 as well as the accompanying video with real footage).

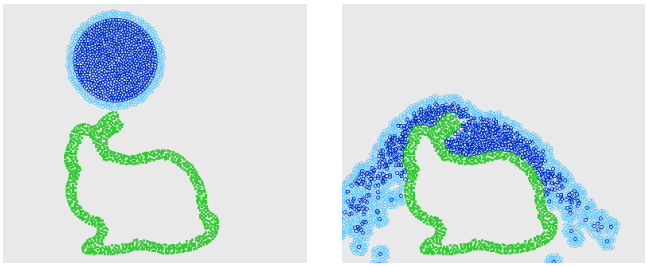
Solid Collisions: Despite treating boundary conditions at the velocity level, truncation errors in time integration may allow liquid particle positions to stray inside solids. Therefore we check them against the solid geometry, represented by level sets for convenience, and if inside project them back to the outside.

Incompressibility: In our examples we computed pressure from density using the Tait equation, $p_i = k\left(\left(\rho_i/\rho_0\right)^7 - 1\right)$ [Monaghan 1994] with $k = 2000$, but other equations or calculations such as PCISPH may serve equally well for the purposes of this paper.

Pressure Force: We used Monaghan's usual pressure gradient, $-m_i \sum_j m_j \left[p_j/\rho_j^2 + p_i/\rho_i^2\right] \nabla W_{ij}$ to compute the pressure force on particle i, but other formulas could be freely used.

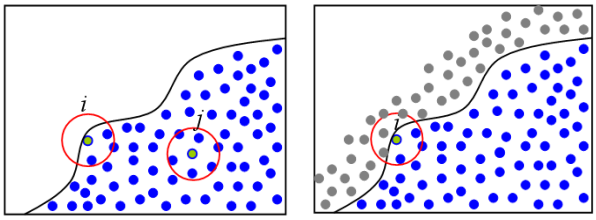
XSPH: Noise in the raw particle velocities can make the simple position update $\mathbf{x}_i^{\text{new}} = \mathbf{x}_i + \delta t \cdot \mathbf{v}_i$ problematic. XSPH [Monaghan 1989] improves matters by blending in surrounding particle velocities with the addition of $\epsilon \sum_j (m_j/\rho_j)\,(\mathbf{v}_j - \mathbf{v}_i)\,W_{ij}$ where ϵ is a user-tuned parameter on the order of 0.5.

Artificial Viscosity: Some form of artificial viscosity is necessary to stabilize the inviscid equations. For Basic SPH we adopt the same model as Becker and Teschner [2007] with $\alpha = 0.0005$.



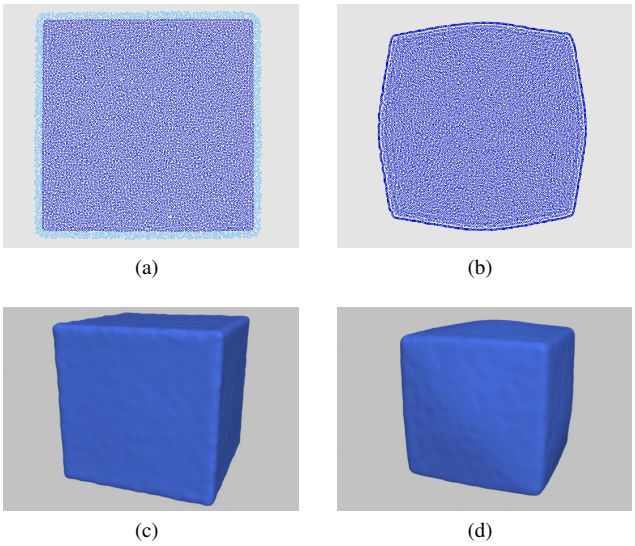
(a) (b)

Figure 2: ***Ghost SPH: 2D Simulation Snapshots.*** *Dark Blue: liquid. Light blue: ghost air. Green: ghost solid.*



(a) (b)

Figure 3: ***Ghost-Air Particles at the Free-Surface.*** *(a) Basic SPH (b) Ghost SPH. Fluid particles are shown in blue, ghost-air particles are shown in gray.*



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

Figure 4: ***Hydrostatic Test.*** *Upper row: 2D square. Lower row: 3D cube. Left: Ghost SPH. Right: Basic SPH. Taken at frame 400.*

4 The Ghost SPH Method

4.1 Algorithm Overview

We solve the particle deficiency at boundaries and eliminate artifacts by (1) dynamically seeding ghost particles in a layer of air around the liquid with a blue noise distribution, (2) extrapolating the right quantities from the liquid to the air and solid ghost particles to enforce the correct boundary conditions, and (3) using the ghost particles appropriately in summations. Figure 2 shows the three classes of particles in a 2D simulation and Algorithm 3 gives full pseudocode for a time step. Mass, density, and velocity are assigned to ghost particles in the spirit of Fedkiw's Ghost Fluid Method: if a quantity should be continuous across a boundary, it is left as is in the ghost; if it is decoupled and may jump discontinuously, the fluid's value is instead extrapolated to the ghost.

4.2 XSPH Artificial Viscosity

Before describing the ghost treatment of boundary conditions, we introduce a simplification of the basic method which eases implementation. The goal of artificial viscosity is just to damp out nonphysical oscillations that plague the undamped method: a full treatment of physical viscosity, with a nonphysical coefficient tuned to stabilize the discretization, is overkill. We propose adopting the simpler XSPH style of damping noise, but at the velocity update level — this works just as well, but is cheaper and easier to tune, and further obviates the need for XSPH in the position update.

Every time step we take the step-advanced velocities $\mathbf{v}_i^* = \mathbf{v}_i + \delta t \cdot \mathbf{a}_i$ and diffuse them with a tunable parameter $\epsilon = 0.05$:

$$\mathbf{v}_i^{\text{new}} = \mathbf{v}_i^* + \epsilon \sum_j \frac{m_j}{\rho_j} \left(\mathbf{v}_j^* - \mathbf{v}_i^*\right) W_{ij}. \tag{2}$$

We can then evolve positions directly with the particle velocity, $\mathbf{x}_i^{\text{new}} = \mathbf{x}_i + \delta t\, \mathbf{v}_i^{\text{new}}$.

4.3 Ghost Particles in the Air

Air particles are generated at the start of simulation within a kernel radius of the liquid, outside of solids: see Section 4.5. The ghost mass of each air particle is set to the mass of a liquid particle, and the ghost velocity of an air particle is set to the velocity of the nearest liquid particle (extrapolating, since in a free surface simulation, there isn't even a defined air mass or velocity). Under the $p = 0$ free surface boundary condition, we set the ghost air density equal to the rest density of the liquid, producing zero pressure.

Ghost air particles contribute to the density summation, and since they fill a thick enough layer around the liquid, this entirely solves the free surface density problem: see Figure 3. As their density is the rest density, they contribute no pressure force on the liquid. We explicitly exclude them from the artificial viscosity, as they would add a slight bias in favour of the outermost particles' velocities.

To make the overhead of resampling negligible, we only resample every 10 time steps or so (twice per frame in our examples), and in interim steps advect the ghost air particles with their velocities. This is frequent enough that no appreciable drift of the ghost particles away from the liquid happens in practice.

Figure 4 demonstrates the results of a zero-gravity hydrostatic test with Basic SPH vs. Ghost SPH. Our method keeps the fluid stable, maintains its original shape and volume, and conserves the initial uniform particle distribution and density. In contrast, in the Basic SPH model pressure pushes the outer particles inwards to reach a density equilibrium, forms a stiff shell of particles at the free-surface, and non-uniformly shrinks the fluid volume.

4.4 Ghost Particles in the Solid

Solid particles, like air particles, have a dual role: correcting the density summation near the boundary and implementing the right boundary condition in lieu of the basic method's Lennard-Jones approach. They too are seeded inside each solid within a kernel-radius

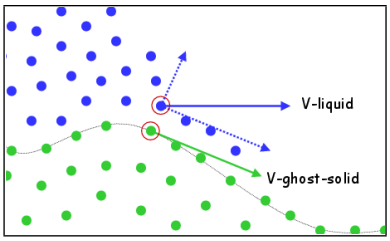


Figure 5: ***Ghost-Solid Velocity*** *in inviscid flow with static solid wall. Blue: liquid particles. Green: ghost-solid particles.*

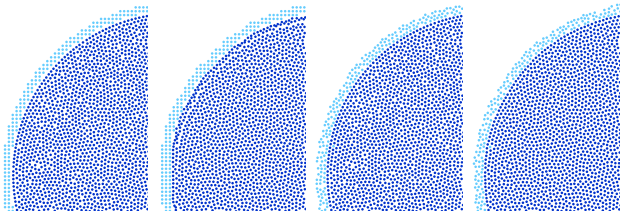


Figure 6: ***Grid sampling the air vs. Poisson disk.*** *(a) Initial grid. (b) 500 frames later, with an anisotropic shell developed. (c) Initial Poisson disk. (d) 500 frames later, essentially unchanged.*

layer of the solid surface, are assigned a ghost mass equal to the liquid particles, and contribute to density summation in the liquid.

The solid boundary condition implies different treatment of ghost density. Pressure should be continuous through the boundary, thus we extrapolate fluid density (which controls pressure via the equation of state) by setting each ghost density to the nearest liquid particle's density. This naturally enforces the no-penetration *and* no-separation boundary condition (up to numerical errors). Note that while this is physically correct, when the boundary layer is under-resolved, as may happen in large-scale simulations, reverting to freely separating boundary conditions by disallowing negative pressures may provide more plausible results [Batty et al. 2007; Chentanez and Müller 2011].

For an inviscid liquid, the associated no-stick condition implies the normal component of liquid and solid velocities match at the boundary, but that tangential components are completely decoupled: the liquid can freely slip past tangentially. We thus construct the ghost velocity at each solid particle by summing the normal component of the solid's true velocity and the tangential component of the nearest liquid particle's velocity:

$$\mathbf{v}^{\text{ghost}} = \mathbf{v}_N^{\text{solid}} + \mathbf{v}_T^{\text{liquid}}. \tag{3}$$

See Figure 5. Ghost solid velocities contribute to XSPH / artificial viscosity of Equation 2, reinforcing the velocity part of the solid boundary condition without introducing unphysical tangential drag.

Of course, for a visibly viscous liquid, the no-slip boundary condition may be more appropriate, where tangential components of solid and liquid velocities also match. In this case, setting the full ghost solid velocity to the real solid velocity, allowing its tangential component to enter the XSPH artificial viscosity term, gives rise to the desired stickiness; the ϵ parameter controls the stickiness and may be adjusted independently for the solid for artistic control.

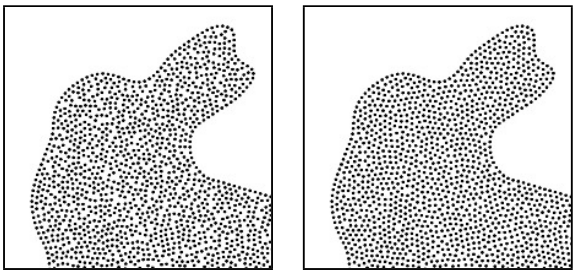


Figure 8: ***Sampling Relaxation Step in 2D.*** *Our sampling algorithm running in a 2D cross section of the Stanford Bunny geometry with zoom in to the head. Left: intermediate result before the last relaxation step. Right: the final result after relaxation.*

4.5 Sampling Algorithm

For seeding particles in the initial liquid, solid layer, and dynamic air layer we need fast isotropic uniform sampling which tightly fits given boundaries. The boundary fit rules out simple periodic patterns: Figure 6 illustrates the artifacts caused by simple grid sampling in the air. We turned to Poisson disk patterns, and specifically Bridson's fast rejection-based approach as the simplest to implement and modify in 3D [2007]. Throughout we take a parameter r proportional to the desired SPH inter-particle spacing and use it as the Poisson disk radius.

There are several components to our sampling, used depending on the context (liquid, solid, or air): sampling on the surface, relaxation on the surface, sampling in the volume, and relaxation in the volume.

For the initial liquid particles, we first sample the surface (represented by a level set for convenience), then improve that initial sampling with surface relaxation, then sample the interior volume, and finish with volume relaxation.

Solids are sampled the same way, but with a distance limit on the volume sampling since we only need a narrow band of particles.

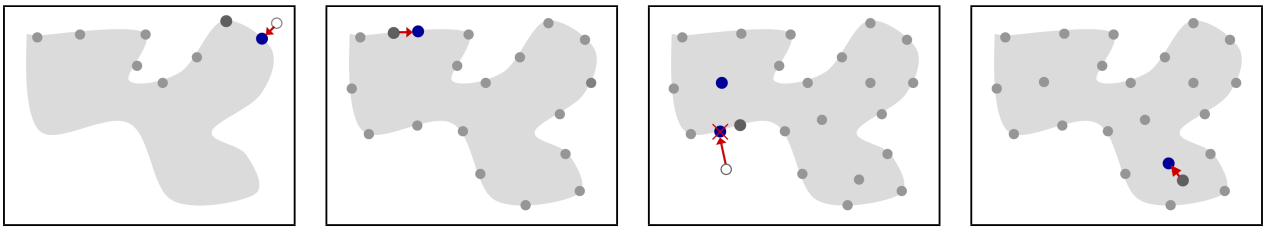
Air sampling, and continuous liquid emission during the simulation, eschew the surface sampling and the relaxation, and instead just sample the required volume starting from existing nearby liquid particles which serve the role of the "surface". As a simple optimization, we do not sample air particles around isolated liquid particles, as their motion is just ballistic anyhow.

We use similar hashed acceleration grids to expedite the the acceptance/rejection search for point samples as we use for SPH summations. For air sampling we also include liquid and solid particles in the rejection test, as we must avoid sampling too close to the liquid or solid, and need not sample beyond one kernel-radius band of the liquid particles.

While reading through the following details of each step, refer to Figure 7 for an illustration of the different components in 2D, and Figure 8 for the interior relaxation in particular. Note that the method applies equally to any dimension, and may be applicable to many problems outside SPH.

4.5.1 Surface Sampling

We assume the surface geometry is given as a signed distance function: this permits fast projection of points to the surface. Pseudocode is provided in Algorithm 1. In the outer loop we search



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

Figure 7: ***Sampling Algorithm Illustration.*** *(a) Sampling the surface. (b) Surface relaxation. (c) Sampling of the interior. (d) Volume relaxation. Blue: newly added or moved particle. Darker gray: particle originating the new particle sample. White with gray line: point sampled to the exterior before projected to the surface.*

for "seed" sample points on the surface, checking every grid cell that intersects the surface (i.e. where the level set changes sign) so we don't miss any components: in a cell we take up to t attempts, projecting random points from the cell to the surface and stopping when one satisfies the Poisson disk criterion, i.e. is at least distance r from existing samples. Once we have a seed sample, we continue sampling from it, taking a step of size $e \cdot r$ from the previous sample along a random tangential direction $\mathbf{d}$, again projecting to the surface and checking the Poisson disk criterion. Parameters $t = 30$ and $e = 1.085$ worked well, but could be further tuned.

Algorithm 1 Surface Sampling

Input: Level set ϕ, radius r, # attempts t, extension e
Output: Sample set S

for all grid cells C where ϕ changes sign **do**
  for t attempts **do**
    Generate random point $\mathbf{p}$ in C
    Project $\mathbf{p}$ to surface of ϕ
    if $\mathbf{p}$ meets the Poisson Disk criterion in S **then**
      $S \leftarrow S \cup \{\mathbf{p}\}$
      Break
  if no point was found in C **then**
    Continue
  while new samples are found **do**
    Generate random tangential direction $\mathbf{d}$ to surface at $\mathbf{p}$
    $\mathbf{q} \leftarrow \mathbf{p} + \mathbf{d} \cdot e \cdot r$
    Project $\mathbf{q}$ to surface of ϕ
    if $\mathbf{q}$ meets the Poisson Disk criterion in S **then**
      $S \leftarrow S \cup \{\mathbf{q}\}$
      $\mathbf{p} \leftarrow \mathbf{q}$

4.5.2 Interior Sampling

In the interior sampling stage we run regular Fast Poisson Disk Sampling [Bridson 2007] but starting with the surface sample points as initial seeds, and using the level set to reject any samples outside the geometry. For solids, we also use the level set to avoid sampling beyond one kernel-radius into the interior. For air or continuous liquid emission, we use the existing liquid particles as the initial "surface" seeds, and for air also avoid sampling too more than a kernel radius away from the liquid. As speed is critical for air and liquid emission during simulation, we reduce the maximum number of random attempts per sample t to 8; for initial liquid shapes and solids we take the usual $t = 30$.

4.5.3 Surface and Volume Relaxation

The goal of the relaxation procedure (Algorithm 2) is to reduce noise in the SPH density by optimizing sample locations. It runs twice during the initial particles seeding process, first for relaxing the surface samples and then for relaxing the volume samples.

We attempt to reposition each sample in turn so that it is a greater distance away from its closest neighbor, again using simple rejection sampling. The code takes $t = 50$ nearby random points (with distance from the sample decreasing through the iteration) and if any are further from other samples than the original position, we take the best. Surface sample candidates are additional projected to the surface of the level set, and interior sample candidates are projected if outside the level set. We sweep through all the samples k times, with $k = 5$ for the surface and $k = 30$ for the volume.

Algorithm 2 Surface/Volume Relaxation

Input: Initial sample set S, level set ϕ, radius r, # sweeps k, and # attempts t
Output: S relaxed sample set

for k sweeps **do**
  for all $\mathbf{p} \in S$ **do**
    Let $B(\mathbf{p}) \subseteq S$ be the samples within $2r$ of $\mathbf{p}$
    $d \leftarrow \min_{\mathbf{q} \in B(\mathbf{p})} \|\mathbf{p} - \mathbf{q}\|$
    $\mathbf{p}^{new} \leftarrow \mathbf{p}$
    for $i = 0 \ldots (t-1)$ **do**
      $\tau \leftarrow \frac{t-i}{t}$
      Generate random vector $\mathbf{f}$ from unit sphere
      $\mathbf{p}^{\text{cand}} \leftarrow \mathbf{p} + r \cdot \tau \cdot \mathbf{f}$
      if $\mathbf{p}^{\text{cand}}$ outside ϕ or came from surface sample **then**
        Project $\mathbf{p}^{\text{cand}}$ to surface of ϕ
      $d' \leftarrow \min_{\mathbf{q} \in B(\mathbf{p})} \|\mathbf{p}^{\text{cand}} - \mathbf{q}\|$
      if $d' > d$ **then**
        $\mathbf{p}^{\text{new}} \leftarrow \mathbf{p}^{\text{cand}}$
        $d \leftarrow d'$
    $\mathbf{p} \leftarrow \mathbf{p}^{\text{new}}$

4.6 SPH Density Distribution

Our stochastic sampling doesn't optimally pack samples together. Therefore we use an empirically determined radius of $r = 0.92\,h$ where h is the desired simulation particle spacing, thereby reaching an SPH density close to the target rest density. At the initial state, we further measure the average density $\bar{\rho}$ and scale the initial particle mass by $\rho_0/\bar{\rho}$ where ρ_0 is the target rest density: the densities then average exactly to ρ_0, and we start very close to a correct physical equilibrium. In contrast, Basic SPH suffers from noticeable acoustic waves in the beginning of the simulation, which requires several hundred damped steps to reach an initial equilibrium.

Algorithm 3 Ghost SPH Simulation Step

Sample new liquid particles, or air particles, if needed this step.
Compute density:
for all liquid particles i **do**
    Compute ρ_i with Equation 1
for all solid particles i **do**
    Find closest liquid particle j
    $\rho_i \leftarrow \rho_j$
Compute pressure:
for all particles i **do**
    Compute pressure p_i using the equation of state
Compute liquid accelerations and velocities:
for all liquid particles i **do**
    Compute the acceleration $\mathbf{a}_i$ from gravity and pressure
    $\mathbf{v}_i^* \leftarrow \mathbf{v}_i + \delta t \cdot \mathbf{a}_i$
Prepare solid boundary conditions:
for all solid particles i **do**
    Find closest liquid particle j
    $\mathbf{v}_i^* \leftarrow \mathbf{v}_i^N + \mathbf{v}_j^T$ as in Equation 3
Apply XSPH artificial viscosity:
for all liquid particles i **do**
    Update $\mathbf{v}_i^{\text{new}}$ using Equation 2
Extrapolate velocity into air:
for all air particles i **do**
    Find closest liquid particle j
    $\mathbf{v}_i^{\text{new}} \leftarrow \mathbf{v}_j$
Update positions:
for all liquid and air particles i **do**
    $\mathbf{x}_i^{\text{new}} \leftarrow \mathbf{x}_i + \delta t \cdot \mathbf{v}_i^{\text{new}}$

4.7 Temporal Coherence

Whenever we resample the air region, there can be a small change in the nearby density summations etc. which in principle could be a problem for temporal coherence. Since we resample twice per frame, this is not an issue — and even if we resampled less frequently we could always smoothly fade out the old air particles while fading in the new. That said, it's worth evaluating how small the change due to resampling is.

Consider a simple constant shear flow where only internal forces (pressure, artificial viscosity) act on the fluid. The exact solution has zero acceleration, which we use as a baseline to measure the general SPH error vs. the change due to air resampling. We ran this example with 6400 liquid particles initially in the unit square centered at $(0, 0)$ with velocity field $\mathbf{v}(x, y) = (0, x)$.

We measure the acceleration $\mathbf{A}_i$ of a particle with the second order finite difference of position, and from that define two metrics,

$$E_i^n = \|\mathbf{A}_i(t^n)\| \tag{4}$$

$$\Delta E_i^n = \|\mathbf{A}_i(t^n) - \mathbf{A}_i(t^{n-1})\|, \tag{5}$$

where t^n is the time at step n. The general SPH error is given by E_i^n, how far the acceleration deviates from the theoretical solution (zero), or in other words the SPH error. We estimate the jump due to resampling by comparing ΔE_i^n on the steps with resampling versus the other steps.

Figure 9 shows both the average and the maximum of the metrics over all particles for the first few dozen steps. Every tenth step, after air resampling, we can see a telltale jump in the ΔE_i value. However, the jump is of the same magnitude as the SPH acceleration error E_i in steps without resampling, which is quite tolerable.

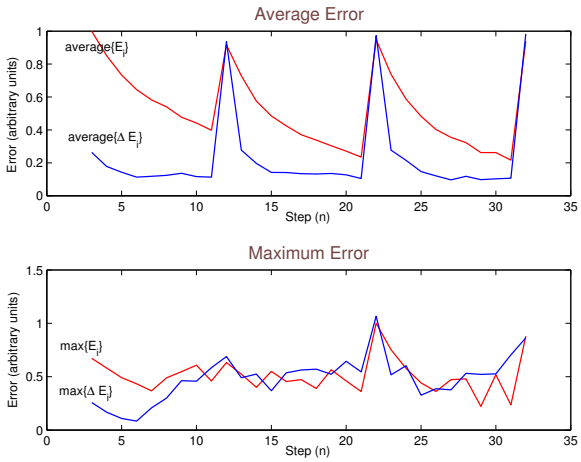


Figure 9: ***SPH Error vs. Resampling Error.*** *Measurements of the general SPH error E_i (red) and the acceleration change ΔE_i (blue) in simulation steps $3 - 32$. Top: average values. Bottom: maximum values. Each graph is normalized by its maximum red plot value. Resampling occurs at every tenth step.*

The accompanying video supports this, demonstrating stable flow in free fall stages and hydrostatic tests for example.

5 Results and Discussion

We ran our simulations on a quad-core Intel i7-2600 (8M Cache, 3.40 GHz) with 8GB memory. Neighbor searches and sampling were accelerated with background grid structures. We used OpenMP to parallelize particle computations.

Figures 10 and 11 show selected frames from our 3D simulation results, referred herein as the "tomato" and the "bunny" examples. Table 1 gives statistics on particle counts, overhead for ghost particles, and run time. The tomato example has a continuous emission of liquid particles, thus we report average liquid particle count and computation time over 16k time steps / 800 frames.

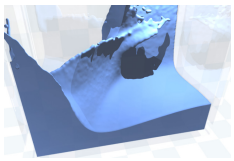
Figure 11 of the tomato example compares Ghost SPH to Basic SPH with the same shared parameters. Basic SPH suffers severe surface tension artifacts at air and solid interfaces, lacks the physical cohesion between liquid and solid, and causes particles to form unnatural clusters instead of spray. Ghost SPH simulation comes much closer to the natural motion of the fluid flow (see the video for real footage), wrapping all the way around the tomato before freely leaving in a stream from the bottom — despite the small number of simulation particles.

The ghost air particles of Ghost SPH added a nontrivial 141% memory overhead in the tomato example, though a much smaller 9% overhead for the bunny. Asymptotically the memory overhead scales with the surface area, not the volume, and thus the overhead is reduced at higher resolutions. Interestingly, Ghost SPH only took 26% more computation time per time step than Basic SPH for the tomato, $1.29s$ versus $1.02s$: we argue the considerable improvement in results at the same resolution is worth this small cost. The computation overhead cause is the air resampling step and the increased number of particles; on average air resampling took 11% of a simulation step computation time, density and particle neighbor data 56%, pressure force 19%, and XSPH for artificial viscosity and boundary conditions took 10% of the time. We also found the improved particle distribution at boundaries meant a much wider range

Statistic	Tomato	Bunny
Liquid particle average count (#)	59k	91k
Ghost-solid particle overhead	102%	9%
Ghost-air particle overhead	136%	40%
Avg. computation time per step	1.92 s	1.29s
Avg. computation time per frame	38.46 s	25.75s

Table 1: ***Simulation statistics for the 3D examples.*** *"Overhead" refers to the ratio of ghost particles to real liquid particles. Each animation frame is subdivided into 20 time steps.*

of stiffness coefficients and artificial viscosities could work, simplifying parameter tuning; in some but not all simulations a much larger stable time step was possible than with Basic SPH, which lead to a net improvement in performance.



A larger 750k particle simulation of a double dam break (frame 250 to the left; also see the accompanying video) averaged $21.4s$ per step, demonstrating the expected linear cost.

6 Conclusions

Together our ghost treatment of solid and free surface boundaries, particle sampling algorithms, and new XSPH artificial viscosity allow significantly higher quality liquid simulations than basic SPH with only a moderate overhead. Our approach should extend easily to related methods such as PCISPH, and the sampling may find use in any particle simulation. Looking to the future, we plan to extend the ghost method to a more accurate treatment of surface tension and to two-way coupling with solids or other fluids.

Acknowledgements

This work was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada and a scholarship from BC Innovation Council and Precarn Incorporated. All bunny-like solids were derived from the scanned bunny model made available by the Stanford Computer Graphics Laboratory.

References

ADAMS, B., PAULY, P., KEISER, R., AND GUIBAS, L. J. 2007. Adaptively sampled particle fluids. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 3.

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3.

BECKER, M., AND TESCHNER, M. 2007. Weakly compressible sph for free surface flows. In *Proc. ACM SIGGRAPH / Eurographics SCA*, 63–72.

BECKER, M., TESSENDORF, H., AND TESCHNER, M. 2009. Direct forcing for Lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics 15*, 3, 493–503.

BONET, J., AND KULASEGARAM, S. 2002. A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *Appl. Math. Comput. 126*, 2-3, 133–155.

BRIDSON, R. 2007. Fast Poisson disk sampling in arbitrary dimensions. In *ACM SIGGRAPH Technical Sketches*.

CHENTANEZ, N., AND MÜLLER, M. 2011. A multigrid fluid pressure solver handling separating solid boundary conditions. In *Proc. Symp. Comp. Anim.*, 83–90.

COLAGROSSI, A., AND LANDRINI, M. 2003. Numerical simulation of interfacial flows by Smoothed Particle Hydrodynamics. *J. Comp. Phys. 191*, 2, 448–475.

COOK, R. L. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graph. 5*, 1.

DUNBAR, D., AND HUMPHREYS, G. 2006. A spatial data structure for fast Poisson-disk sample generation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 25*, 3.

FEDKIW, R., ASLAM, T., MERRIMAN, B., AND OSHER, S. 1999. A non-osillatory Eulerian approach in multimaterial flows (the Ghost Fluid Method). *J. Comput. Phys. 152*, 457–492.

FEDKIW, R. 2002. Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the Ghost Fluid Method. *J. Comp. Phys. 175*, 200–224.

GINGOLD, R. A., AND MONAGHAN, J. J. 1977. Smoothed Particle Hydrodynamics - theory and application to non-spherical stars. *Mon. Not. R. Astron. Soc. 181*, 375–389.

IHMSEN, M., AKINCI, N., GISSLER, M., AND TESCHNER, M. 2010. Boundary handling and adaptive time-stepping for PCISPH. In *Proc. VRIPHYS*, 79–88.

KEISER, R., ADAMS, B., GUIBAS, L. J., DUTRÉ, P., AND PAULY, M. 2006. Multiresolution particle-based fluids. Tech. Rep. 520.

LUCY, L. B. 1977. A numerical approach to the testing of the fission hypothesis. *Astron. J 82*, 1013–1024.

MONAGHAN, J. J. 1989. On the problem of penetration in particle methods. *J. Comput. Phys. 82*, 1–15.

MONAGHAN, J. J. 1994. Simulating free surface flows with SPH. *J. Comput. Phys. 110*, 399–406.

MONAGHAN, J. J. 2005. Smoothed Particle Hydrodynamics. *Reports on Progress in Physics 68*, 8, 1703–1759.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *Proc. ACM SIGGRAPH / Eurographics SCA*, 154–159.

MÜLLER, M., SCHIRM, S., TESCHNER, M., HEIDELBERGER, B., AND GROSS, M. 2004. Interaction of fluids with deformable solids. *J. Comput. Anim. and Virt. Worlds 15*, 3–4, 159–171.

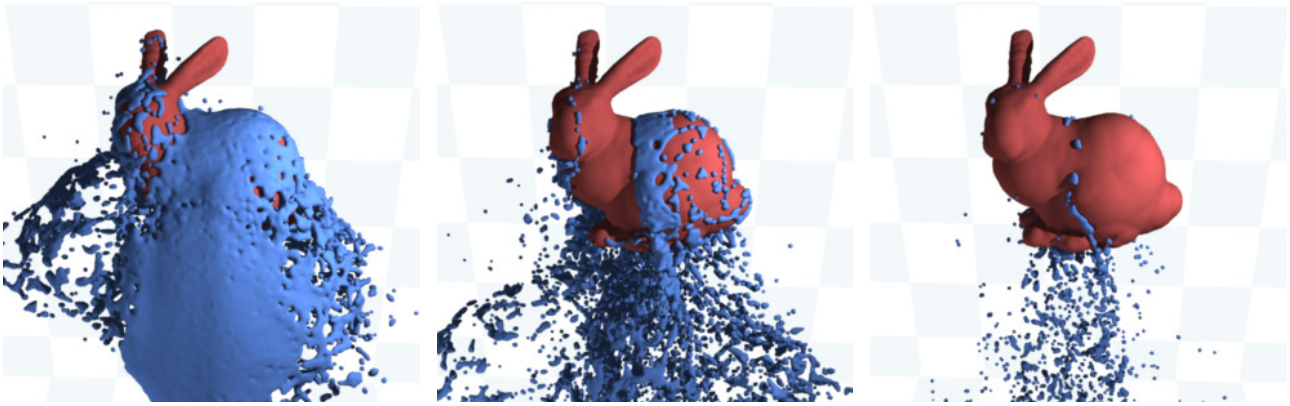
MÜLLER, M., SOLENTHALER, B., AND KEISER, R. 2005. Particle-based fluid-fluid interaction. In *Proc. ACM SIGGRAPH / Eurographics SCA*, 237–244.

SOLENTHALER, B., AND GROSS, M. 2011. Two-scale particle simulation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 30*, 4, 81:1–81:8.

SOLENTHALER, B., AND PAJAROLA, R. 2008. Density contrast SPH interfaces. In *Proc. ACM SIGGRAPH / Eurographics SCA*, 211–218.

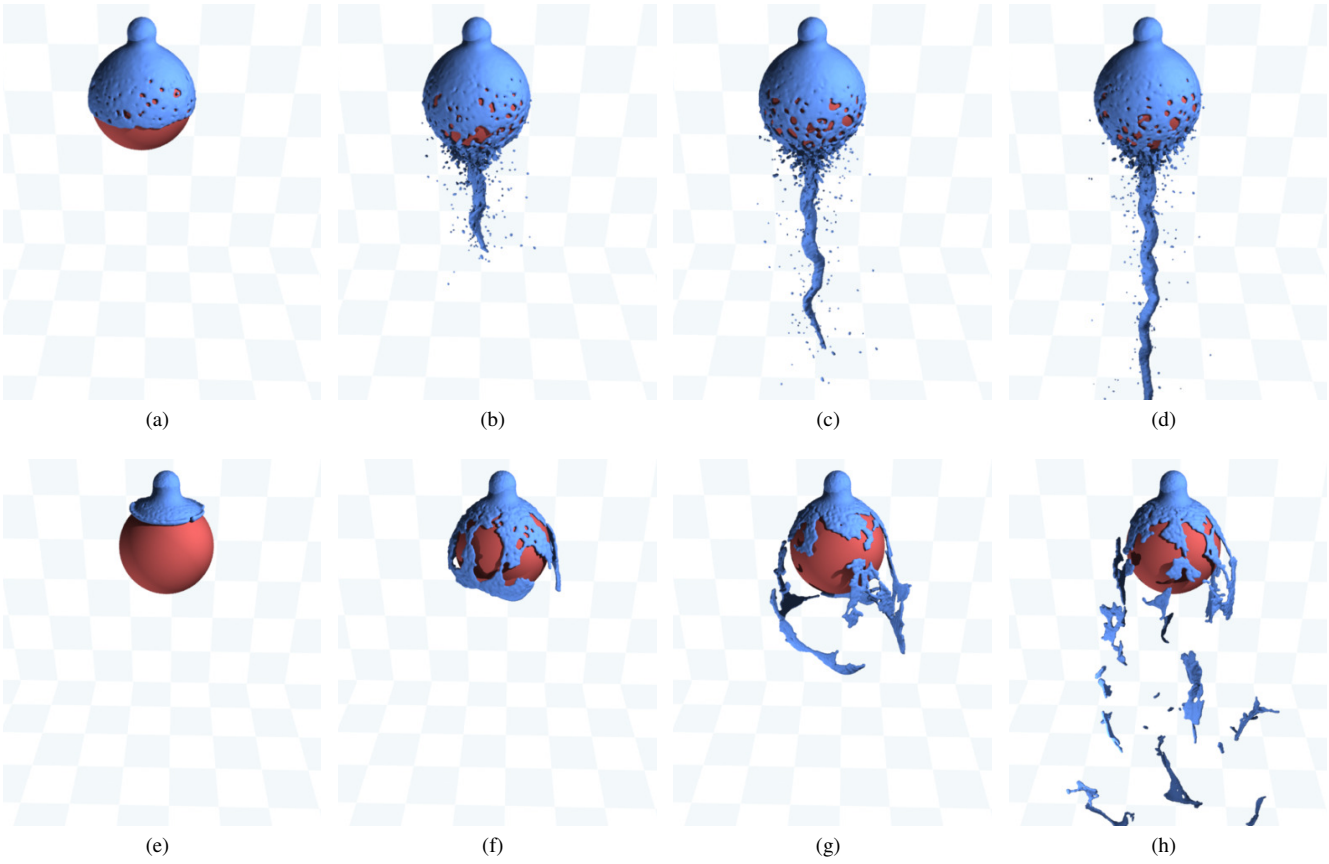
SOLENTHALER, B., AND PAJAROLA, R. 2009. Predictive-corrective incompressible SPH. *ACM Trans. on Graphics (Proc. SIGGRAPH) 28*, 3.

TURK, G. 1992. Re-tiling polygonal surfaces. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 2.



(a) (b) (c)

Figure 10: ***Fluid sphere on solid bunny.***



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

(e) (f) (g) (h)

Figure 11: ***Tomato Under Tap Water. Ghost SPH vs. Basic SPH.*** *Upper row (a)-(d): Ghost SPH. Lower row (e)-(h): Basic SPH. (equivalent frames are presented)*