Matching Fluid Simulation Elements to Surface Geometry and Topology

Tyson Brochu*
University of British Columbia

Christopher Batty*
University of British Columbia

Robert Bridson*
University of British Columbia

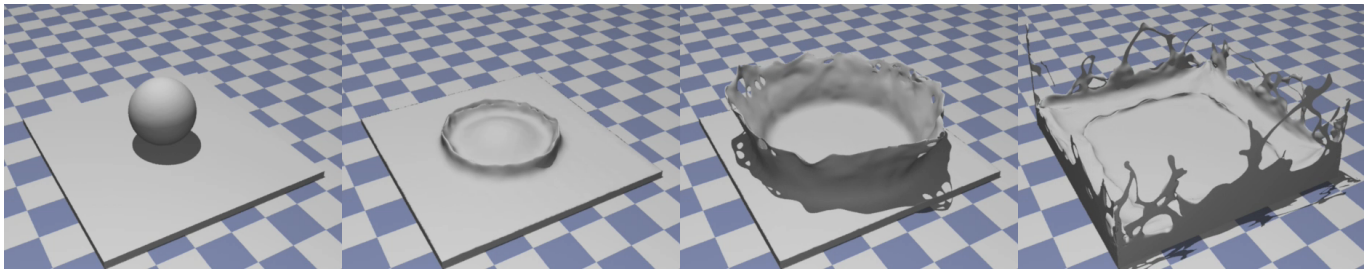

Figure 1: ***Sphere Splash.*** *Coupling an explicit surface tracker to a Voronoi simulation mesh built from pressure points sampled in a geometry-aware fashion lets us capture very fine details in this sphere splash animation that uses only 314K tetrahedra.*

Abstract

We introduce an Eulerian liquid simulation framework based on the Voronoi diagram of a potentially unorganized collection of pressure samples. Constructing the simulation mesh in this way allows us to place samples anywhere in the computational domain; we exploit this by choosing samples that accurately capture the geometry and topology of the liquid surface. When combined with high-resolution explicit surface tracking this allows us to simulate nearly arbitrarily thin features, while eliminating noise and other artifacts that arise when there is a resolution mismatch between the simulation and the surface—and allowing a precise inclusion of surface tension based directly on and at the same resolution as the surface mesh. In addition, we present a simplified Voronoi/Delaunay mesh velocity interpolation scheme, and a direct extension of embedded free surfaces and solid boundaries to Voronoi meshes.

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

Keywords: fluid simulation, liquids, meshes, surface tension

1 Introduction

One of the most visually compelling aspects of liquids is the variety of complex thin sheets and droplets that arise during splashing. However, these remain among the most difficult features to simulate plausibly and accurately with existing techniques. Such detailed behaviour is extremely computationally expensive to resolve because of the tremendous grid resolution required for both the fluid solver and the surface tracking mechanism.

*e-mail: {tbrochu|batty|rbridson}@cs.ubc.ca

Recent advances in explicit surface tracking with triangle meshes [Wojtan et al. 2009; Brochu and Bridson 2009; Müller 2009] have made feasible the geometric representation and manipulation of small features, without the loss of detail exhibited by implicit surface methods. However, when the surface is coupled to a standard Eulerian simulator, the liquid volume must first be resampled onto the simulation mesh or grid to provide geometric information for boundary conditions. As this resampling process typically destroys small details, they are invisible to the fluid solver and cannot be advanced appropriately. This can lead to a variety of visible artifacts including lingering surface noise, liquid behaving as if it were connected when it is not (and vice versa), and thin features simply halting in mid-air because the simulator fails to see them [Bargteil et al. 2006; Kim et al. 2009]. When combined with surface tension forces, noisy sub-mesh details can also severely hamper stability if they are not artificially smoothed out.

We will address these problems by constructing a simulator that "sees" every detail in the explicit liquid surface. We carefully generate pressure sample points near the liquid surface, build a Voronoi simulation mesh from these points and a background lattice, and apply a ghost fluid/finite volume pressure discretization which captures the precise position of the liquid interface. We couple this with a semi-Lagrangian advection scheme and a new approach to surface tension, arriving at a complete liquid simulator.

In summary, our key contribution is coupling an explicit surface tracker to a Voronoi-based liquid simulator with:

- a pressure sample placement strategy that captures the complete liquid surface geometry,
- an accurate surface tension model combining mesh-based curvature estimates and ghost fluid boundary conditions,
- embedded free surface and solid boundary conditions adapted to Voronoi cells, avoiding the need for more onerous conforming tetrahedral mesh generation,
- and a new velocity interpolant over unstructured meshes.

The practical benefits of such a system include:

- improved animation of detailed liquid features, including very thin sheets, tendrils and droplets,
- elimination of noise in explicit surface tracking without non-physical smoothing,
- more detailed and less damped surface tension effects,
- and faster semi-Lagrangian advection on unstructured meshes without increased dissipation.

2 Related Work

2.1 Unstructured Mesh Fluids

Unstructured and semi-structured meshes have a long history in computational fluid dynamics, and have gained traction in computer animation as well. An important reason for their popularity is that careful control of mesh geometry can simplify the discretization or improve accuracy. For example, conforming the simulation mesh to solid walls makes the no-flow boundary condition trivial, and adaptivity can be easily introduced by grading mesh elements as desired. Past work in graphics has extensively explored finite volume methods for tetrahedral meshes [Feldman et al. 2005a; Feldman et al. 2005b; Klingner et al. 2006; Chentanez et al. 2006; Elcott et al. 2007; Wendt et al. 2007; Chentanez et al. 2007], and now many of the features of standard grid-based solvers are supported on tetrahedra, including free surfaces and implicit coupling to dynamic solids. Batty et al. [2010] augmented this approach with embedded boundaries [Enright et al. 2003; Batty et al. 2007], improving free surface accuracy and reducing remeshing complexity. Our method extends these advantages to Voronoi meshes.

In a related approach, Sin et al. [2009] developed a particle method which solves a finite volume pressure projection on the Voronoi diagram of the liquid particles. An advantage of this approach is that the pressure degrees of freedom are directly tied to the number of particles, so there can never be a resolution mismatch between surface geometry and simulator. This idea motivates our work.

Franklin & Lee [2010] subdivide polyhedra into tetrahedra for interpolation similar to our method, but our method is simpler due to use of the Voronoi diagram.

2.2 Surface tracking

Implicit surfaces have long been used to capture liquid geometry in animation; this family of schemes includes level set (LS) methods [Enright et al. 2002a], volume-of-fluid (VOF) [Mihalef et al. 2006; Mullen et al. 2007], and semi-Lagrangian contouring (SLC) [Bargteil et al. 2006]. Implicit approaches naturally yield smooth surfaces and seamlessly handle topological change. However, the resolution of the underlying grid imposes a severe limit on the smallest representable feature, beyond which geometry either vanishes (LS, SLC) or artificially coalesces into grid-scale "flotsam and jetsam" (VOF). Ensuring temporal coherence and avoiding visual artifacts due to the use of regular grids can also be problematic.

The shortcomings of implicit schemes have spurred interest in explicit methods, i.e. "front tracking" [Glimm et al. 1998]. Here the surface is represented explicitly as a triangle mesh, whose vertices are moved with the fluid velocity field. The greatest challenge is handling topological change, due to mesh tangling that may occur during merging and splitting. One solution is to determine problematic regions, switch to an implicit surface to repair the tangles there, then stitch back in a new consistent mesh patch [Du et al. 2006; Wojtan et al. 2009]. Müller [2009] takes a similar grid-based approach to untangling, rebuilding a consistent mesh using marching-cubes-like stencils. Unfortunately these methods still are subject, in complex regions, to a resolution limited by the voxel grid.

Another approach is to work strictly on the triangle mesh itself, using "mesh surgery" for repairs. While this is difficult in general, Brochu & Bridson [2009] recently showed that the problem can be simplified using ideas from cloth animation, enforcing the invariant that the surface remain intersection-free. Topological operations are only allowed when safe, while robust collision processing is used as a last resort to avoid tangles, i.e. the surface is minimally perturbed to avoid problems. We use this method in the presented examples, though note that other front tracking methods could easily be used instead—for example, recent work by Campen & Kobbelt [2010] suggests that the need for collision processing could be obviated with exact Boolean operations.

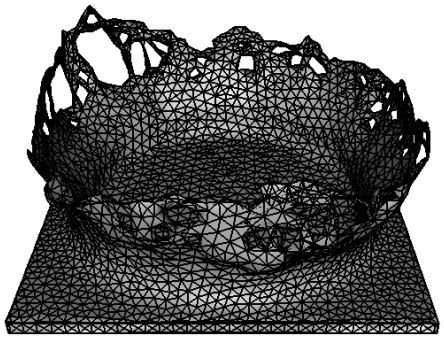

Figure 2: ***Explicit Surface Tracking.*** *Our method exploits the El Topo explicit mesh tracking software to capture thin features.*

2.3 Surface Resolution vs. Simulation Resolution

A prime focus of our work is matching the surface mesh resolution to that of the liquid solver. Most level set-based solvers use one level set sample per pressure grid cell, conservatively avoiding resolution inconsistencies (e.g. [Foster and Fedkiw 2001; Enright et al. 2002b]). Goktekin et al. [2004] experimented with a double-resolution level set, trading better volume conservation for other artifacts. Bargteil et al. [2006] similarly coupled an octree contouring method to a uniform grid fluid solver and explicitly discussed potential artifacts due to resolution mismatch, such as erroneously preserving surface noise and the solver interpreting disconnected fluid regions as connected. Kim et al. [2009] coupled a high resolution particle level set to a low resolution ghost fluid-based liquid solver, but ensured that pressure projection captured all liquid geometry by resampling an inflated level set at the pressure grid resolution—however, this can exacerbate other artifacts, since liquid components behave as if half a cell-width larger than they appear. Kim et al. also introduced extra surface smoothing to prevent retention of small-scale noise.

Mismatched resolutions have been found useful for deformable solids, particularly as surface details are expected to generally persist, unlike in liquids. For example, Wojtan & Turk [2008] used a surface mesh coupled to a lower resolution finite element solver; forcing the simulation mesh to have the same topology, if not resolution, as the embedded surface mesh may improve realism [Teran et al. 2005; Nesme et al. 2009].

2.4 Surface Tension Models

Approaches to surface tension generally fall into two categories: those which apply surface tension as a body force in a region around the interface via smeared delta functions [Brackbill et al. 1992; Hong and Kim 2003; Zheng et al. 2006; Wojtan et al. 2009], and those which apply surface tension *discontinuously* at the interface, typically as a boundary condition in the pressure projection step. The latter is exemplified by the ghost fluid method and related approaches [Enright et al. 2003; Hong and Kim 2005; Hong et al. 2007], and has been shown to provide more realistic results.

Surface tension models can also be compared in terms of how the force itself is approximated. In level set schemes, finite differences

are often used to estimate mean curvature, though this can be quite inaccurate without careful modification (e.g. [Shin 2007]) and cannot capture small details. If a surface mesh is available, a more accurate approach is either to use mesh-based curvature operators (e.g. [Meyer et al. 2002b]), or as proposed recently, to model a physical tension directly in the surface mesh geometry [Perot and Nallapati 2003; Brochu 2006; Wojtan and Turk 2008].

We take the best of each, computing an accurate force from the surface mesh and incorporating it precisely at the surface with the ghost fluid method. We also remedy a shortcoming of existing mesh-based approaches: that surface details below the simulation resolution add energy but cannot be correctly evolved by the solver; without correct feedback from the physics this noise tends to worsen and destroy stability. Wojtan & Turk [2008] handle this with Laplacian smoothing to eliminate small features: note, however, this non-physical operation is dissipative rather than conservative. By instead combining our surface tension model with a geometry-aware sampling, we ensure all relevant details are properly resolved. This yields accurate and comparatively stable surface tension effects without artificial smoothing.

3 Algorithm Outline

We simulate inviscid liquids with semi-Lagrangian advection and an embedded-boundary finite volume pressure projection. We generally follow the tetrahedral scheme of Batty et al. [2010] with modifications to use specially designed Voronoi meshes instead. Like Sin et al. [2009], we place pressure samples on the vertices of a Delaunay tetrahedral mesh, corresponding to the sites of the dual Voronoi diagram (figures 3(a) and 3(b)). Normal components of velocity lie on the faces of the Voronoi cells, so that the velocity sample is parallel to the line segment connecting the pressure samples in the Delaunay mesh. This configuration requires a slightly different velocity reconstruction compared to previous methods, but semi-Lagrangian advection is otherwise straightforward.

For front tracking, we used Brochu & Bridson's *El Topo* code [2009], in particular using its triangle mesh surface to determine the location of pressure samples for our Voronoi simulation mesh.

Purely explicit front tracking algorithms generally use mesh refinement and coarsening to maintain a high quality discretization as the surface deforms. *El Topo* uses a sequence of edge subdivision, collapse and flipping operations, combined with null-space Laplacian smoothing. While these operations change mesh connectivity, they are designed to be geometry-preserving. For example, the smoothing moves vertices only in the null space of the local quadric metric tensor [Garland and Heckbert 1997], as suggested by Jiao [2007]. If the vertex lies on a locally smooth patch it is moved in the plane tangent to the surface, but if on a ridge or corner it is moved only along this line. Therefore, sharp features are preserved, allowing the present paper's algorithm to handle them physically.

The solver runs through the following stages each time step:

1. Advect the explicit surface with *El Topo*.
2. Generate a new simulation mesh as the Voronoi diagram of a lattice with extra samples near the liquid surface (section 5).
3. Advect velocities onto the new mesh with semi-Lagrangian advection (section 6).
4. Add external forces—typically just gravity.
5. Solve for the embedded-boundary pressure projection on the Voronoi mesh, including surface tension forces (section 4).

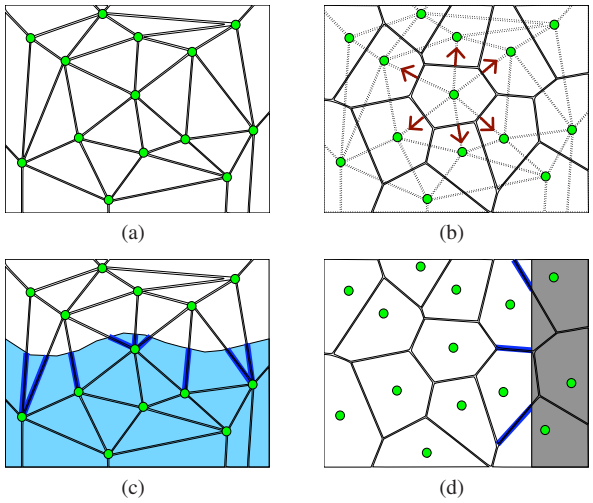

Figure 3: ***Embedded boundaries on Voronoi/Delaunay meshes.*** *Pressure samples are shown as green circles. (a) Delaunay triangulation. (b) Voronoi diagram dual to the Delaunay triangulation (velocity components for the central cell are shown as red arrows). (c) Computation of ghost fluid weights on the edges of the triangulation. (d) Computation of non-solid weights on the faces of the Voronoi diagram. In 2D, Voronoi faces are simply line segments, so solid weights are just fractions of segment lengths. In 3D, Voronoi faces are convex polygons, so determining non-solid weights involves computing polygon areas.*

4 Embedded Boundaries on Voronoi Meshes

We use finite volumes on a Voronoi mesh for the pressure projection step, similar to Sin et al. [2009]. However, rather than applying boundary conditions as they describe, we adapt the embedded boundary methods of Batty et al. [2010] to Voronoi meshes. Conveniently, the duality/orthogonality relationship between Voronoi and Delaunay meshes lets the accuracy benefits of the method carry over. Figure 3 illustrates our mesh configuration, and the computation of the required weights, as discussed below. We solve the resulting symmetric positive definite linear system using incomplete Cholesky-preconditioned conjugate gradients.

To enforce embedded solid boundary conditions, we need to estimate the partial unobstructed area of each element face (figure 3(d)). Batty et al. [2010] used marching triangles cases for computing tetrahedra face fractions from signed distance values on the vertices. However, in the Voronoi setting, the faces are arbitrary convex planar polygons rather than triangles. To handle this, we temporarily place an extra vertex at the face centroid, and use it to triangulate the face. We then use signed distance estimates at the vertices to compute each sub-triangle's partial area, and sum them to determine the partial area for the complete face.

The embedded (ghost fluid) free surface condition uses signed distance estimates at pressure samples to estimate the surface position; these are now located at Voronoi sites rather than tetrahedra circumcenters, but the method is otherwise unchanged (figure 3(c)). A slight improvement can be achieved by casting rays to find the exact position of the surface mesh between pressure samples. In some cases this is much more accurate than the estimate derived from signed distances, but in practice we found it made minimal visual difference. To actually compute the liquid signed distance field on the tetrahedral mesh, we compute exact geometric distance for a narrow band of tetrahedra near the surface, then use a graph-

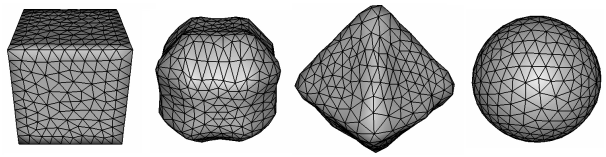

Figure 4: ***Surface Tension.*** *Our accurate surface tension model captures capillary waves even on relatively low resolution meshes. From left to right: A cube in zero gravity begins to collapse due to surface tension, inverts to become an octahedron, and continues to oscillate rapidly before settling down to a sphere.*

based propagation of closest triangle indices to roughly fill in the rest of the mesh. This family of redistancing schemes is described by Bridson [2008], and is easily adapted to tetrahedra.

4.1 Surface Tension

To incorporate surface tension, we follow Enright et al. [2003] in setting the free surface pressure $p_{\text{fs}} = p_{\text{air}} + \gamma\kappa_{\text{fs}}$, where p_{air} is the constant air pressure, γ is the surface tension coefficient and κ_{fs} is the mean curvature of the surface.

Rather than using level set finite differences, we compute curvature directly from the surface mesh to accurately capture high-frequency features. We chose the operator of Meyer et al. [2002b] because it provides high quality estimates using just the one-ring of triangles surrounding each vertex, but others could work too.

Curvature is evaluated at the intersection point between the the triangle mesh surface and the line joining an interior pressure sample to an exterior one. Often this intersection point will coincide with a surface mesh vertex due to our choice of sampling scheme; where it does not, we use simple linear interpolation between the vertices of the surface triangle mesh. This method appears highly accurate, and leads to much less damping than that of Wojtan et al. [2009].

5 Mesh Generation

An advantage of a Voronoi-based discretization is the freedom to explicitly choose pressure sample locations, which is critical for accurate ghost fluid free surface conditions as the signed distance at these samples communicate the surface geometry to the solver. We can visualize the solver's "knowledge" by contouring this level set: figures 5 and 6 illustrate how uniform sampling may fail.

Careful pressure sample placement with respect to the surface helps in three important ways. First, we can inform the solver of all local geometric extrema, allowing the physics to act upon them correctly. This eliminates the accumulation of erroneous surface noise without requiring non-physical smoothing; this is especially vital for surface tension where spurious noise affects the curvature estimates and induces disastrously large yet futile compensating velocities that destabilize the simulation. Second, we can ensure that the solver sees the correct surface *topology* so that the physics responds to merging or splitting only when the surface mesh itself merges or splits. Lastly, grid-scale features often disappear and reappear in regular grid sampling, from the perspective of the solver, as the surface translates through the grid. By specifically placing points inside such small features, we ensure they cannot be missed.

Comparison to Adaptive Lattices: The brute-force approach to these issues is to locally refine using octree grids or graded BCC lattice tetrahedra to capture smaller features. However, this scales poorly since many of the extra samples yield little benefit, while incurring memory and computational overhead. Furthermore, there remains no guarantee that features below the smallest grid cell size will be captured. By choosing sample points to precisely capture the geometry rather than naïvely increasing sample density, we can guarantee sampling of features which would require potentially orders of magnitude more samples with pure adaptive lattices.

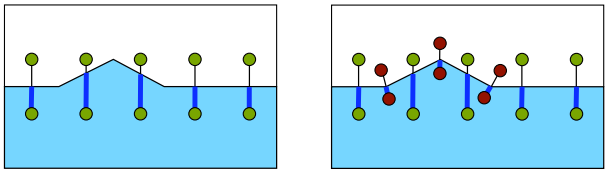

Figure 5: *Left: Even with the ghost fluid method, regular sampling may miss surface details which do not align with the simulation mesh, such as this wave crest. Right: Adaptive samples (shown in red) placed on either side of each mesh vertex ensure that all geometric detail is resolved by the simulation.*

Comparison to Conforming Tetrahedra: While the tetrahedral method of Chentanez et al. [2007] also builds a volumetric mesh that attempts to respect the liquid surface, it matches boundary faces rather than positioning pressure samples. This is considerably more difficult than non-conforming Delaunay tetrahedralization, and generally requires more Steiner points, worse-shaped tetrahedra, and/or the loss of the Delaunay property. Since our method uses embedded boundary conditions, we do not require conforming elements. (Note that this advantage is shared by the method of Batty et al. [2010].) Moreover, the position of pressure samples plays a more important role in free surface conditions than the position of element faces. As accuracy requires that tetrahedral schemes store pressures at circumcenters [Klingner et al. 2006; Batty et al. 2010], and since circumcenters often lie outside their associated tetrahedra, even filling a thin feature with conforming tetrahedra provides no guarantee that its interior will be sampled at all.

5.1 Pressure sample placement strategy

We begin by choosing a characteristic length scale for the simulation, Δx, and configure *El Topo* to try to maintain triangle edge lengths in the range $[\frac{1}{2}\Delta x, \frac{3}{2}\Delta x]$. To resolve all surface details with our volumetric mesh, we need to place pressure samples so that they capture the surface's local geometric extrema, i.e. around surface mesh vertices. In particular, we try to ensure that one edge of the Delaunay triangulation passes through each surface vertex, with one sample inside and one outside. Therefore we take the inward and outward normal at each surface vertex (averaged from the incident surface triangles), and attempt to place a pressure sample a short distance along each. We placed outward samples at $\frac{1}{2}\Delta x$ and inner samples at $\frac{1}{4}\Delta x$, though other ratios would work as well. As a result, surface mesh normal directions will often align exactly with a velocity sample in the simulation mesh; this lends additional accuracy to the vertex's normal motion, and to the incorporation of the normal force due to surface tension calculated at the vertex.

This placement may miss very thin sheets or other fine structures: to robustly sample such features, we check line segments of length Δx from each surface vertex in both offset directions for intersection with the rest of the surface mesh. If we find any triangle closer than Δx, we store the distance d to the closest intersection, and use d in place of Δx in the offset distance calculations above (see figure 7).

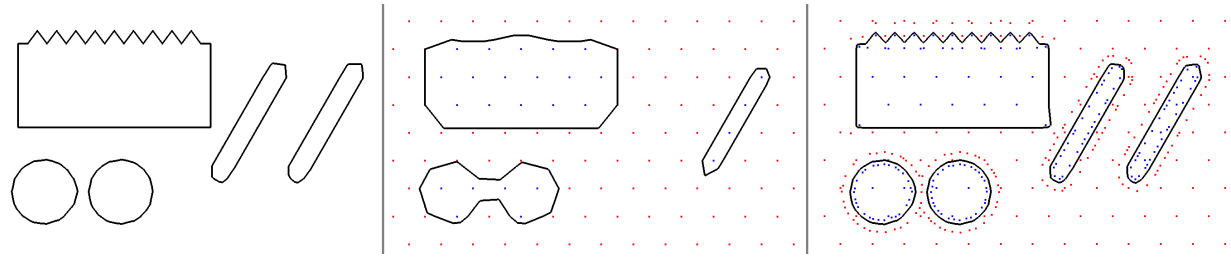

Figure 6: *Left: The input surface geometry. Centre: The resulting surface after resampling onto a regular lattice simulation mesh. Note the spurious topology change, rounding of sharp features, aliasing of high frequency details, and the complete disappearance of one small fluid component due to poor placement relative to the mesh samples. Right: The resampled surface after adding geometry-aware sample points to the simulation mesh; the result is much more consistent with the input. (Mesh sample locations are indicated by points, coloured blue when inside, red when outside.)*

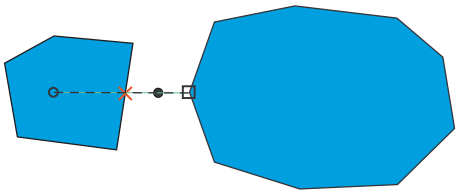

Figure 7: ***Sampling Thin Features.*** *A pressure sample is seeded along the outward normal direction from a surface vertex (black square). The initial proposed pressure location (empty black circle) would land in the wrong component and potentially fail to resolve the intervening air gap. We instead place the final pressure sample (filled black circle) midway between the starting vertex and the first intersection point (red X).*

We further reject new pressure samples which are too close to an existing sample by some epsilon, which would cause a very short edge in the final mesh.

If the distance between the surface vertex and the first intersection is below some threshold (e.g. $\frac{1}{20}\Delta x$) at which we consider the two surfaces to have effectively collided, and the proposed sample is an air sample, we also discard it. This is necessary because the divergence constraint is not enforced on air cells, so they can act as liquid sinks [Losasso et al. 2006] and destroy liquid volume until the geometry finally merges. Unfortunately, merging in this scenario can often take several time steps to resolve because the interpolated velocity in the air gap still averages to zero, thereby preventing surface geometry from actually intersecting and flagging a collision. By not placing a sample point in these very small gaps, our simulator treats the two liquid bodies as merged and prevents volume loss; the geometric merge is usually then processed within a few timesteps. (With regular sampling, merging will depend on where grid points happen to fall with respect to the surface; hence the physics can respond as if merged when the surfaces are still as much as Δx apart, as in figure 9. This generates non-physical air bubbles which linger for many timesteps before they self-collide and are eliminated.)

After placing the surface-adapted pressure samples, we complete the sampling of the domain by adding regularly-spaced points from a BCC lattice with cell size $2\Delta x$, again rejecting samples which fall too near existing samples—of course, a graded octree or any other strategy could also be used to fill the domain. All samples are then run through a Delaunay mesh generator such as *TetGen* [Si 2006]. Figure 8 illustrates in 2D how this sampling approach is able to capture thin features such as splashes. Further experimentation with relative mesh spacing parameters could yield improved results.

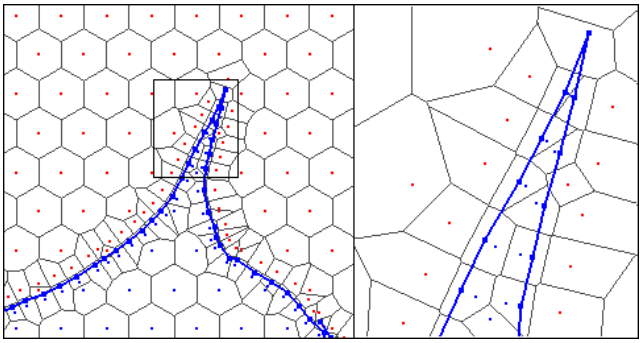

Figure 8: ***Simulating Thin Features.*** *A 2D example of a thin feature simulated with our method. The zoom on the right illustrates the sample placement with respect to surface vertices, and the resulting Voronoi mesh. Notice that even the very sharp tip contains a pressure sample, as indicated by the surrounding Voronoi cell.*

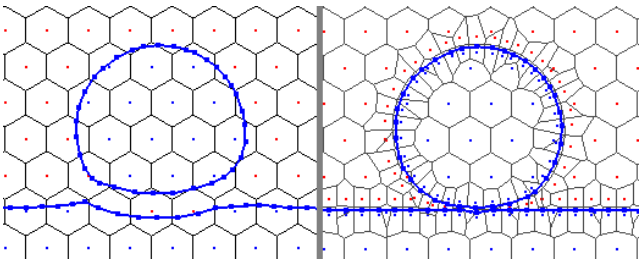

Figure 9: ***Merging.*** *Left: Regular sampling erroneously identifies a topology change, causing a premature reaction in both liquid bodies. Right: Geometry-aware sampling responds correctly.*

6 Interpolation and Advection

Velocity interpolation methods for unstructured meshes typically proceed in two steps [Klingner et al. 2006; Elcott et al. 2007; Batty et al. 2010]. First, a full velocity vector is reconstructed at selected mesh locations using a least-squares fit to the nearby velocity components. Then barycentric or generalized barycentric interpolation between those locations interpolates velocity over the full domain. Given such an interpolant, advection of velocities and geometry is straightforward. We follow this general framework, with two modifications.

In previous work, face normal components on tetrahedra were used to reconstruct velocities at circumcenters (Voronoi vertices). In our configuration, velocity components instead lie along the tetrahedra edges (Voronoi faces) so we perform the least squares fit on this data instead. We could then apply the usual generalized barycentric interpolant over Voronoi cells, but this is expensive [Chentanez et al. 2007] and requires special case handling to avoid degeneracies [Meyer et al. 2002a]. A simple and fast alternative discussed by Klingner et al. and Chentanez et al. is to first interpolate velocities to Voronoi sites (tetrahedra vertices) and apply standard (and fast) barycentric interpolation over each tetrahedron. However, the interpolation onto tetrahedra vertices discards any local extrema at the Voronoi vertices, thereby severely over-smoothing the velocity field in practice, damping out interesting flow behavior.

Rather than discard extrema at Voronoi vertices, we use a slightly refined tetrahedral mesh that *includes* them. We conceptually tetrahedralize the Voronoi cells themselves by placing additional vertices at Voronoi face centroids and Voronoi sites (see figure 10). Velocities for each of these new points need to be computed; while previous work used the generalized barycentric interpolant for this transfer step, we found that simply averaging the velocities of the surrounding ring or cell of Voronoi vertices is quicker and equally effective. For maximum fidelity at the face centroids, we also replace the normal component of the averaged full velocity with the exact normal component already stored at the face. Simple and efficient barycentric interpolations can then be applied on the resulting smaller tetrahedra. Because the sharper, more accurate velocities at the Voronoi vertices are retained and merely augmented with additional data, this is far less dissipative, yielding results that closely match generalized barycentric interpolation (see figure 11).

Lastly, note that reconstructions should only use face velocities which were assigned valid data by the pressure projection, and thus we can only reconstruct reasonable velocities inside the fluid. We therefore extrapolate velocities outwards from the fluid using a breadth-first graph propagation: each unknown point in a layer is set by averaging all adjacent known points from previous layers, repeating until we have a sufficiently large band of velocities surrounding the surface. This simple method, suggested in the context of cloth-fluid coupling by [Guendelman et al. 2005], sufficed for all our animations.

In summary, the steps of our interpolation scheme are:

1. Reconstruct full velocity vectors at Voronoi vertices using least squares.
2. Assign full velocity vectors to Voronoi sites and faces using simple averaging from neighboring vertices.
3. Subdivide the Voronoi cells into sub-tetrahedra using the sites and face centroids (see figure 10).
4. Apply a simple graph-based extrapolation of velocities to fill in velocities near the liquid.

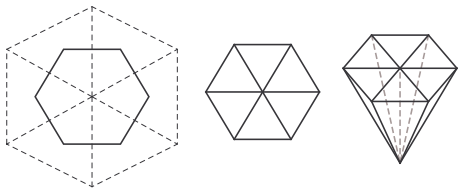

Figure 10: *Rather than interpolating velocity over Voronoi regions directly, we tetrahedralize them and use simple barycentric interpolation. Left: A 2D Voronoi cell with standard dual Delaunay mesh overlaid. Centre: The same cell subdivided into smaller triangles that include the Voronoi vertices. Right: In 3D, each Voronoi face is triangulated using its centroid, and joined to its Voronoi site to build a tetrahedralization.*

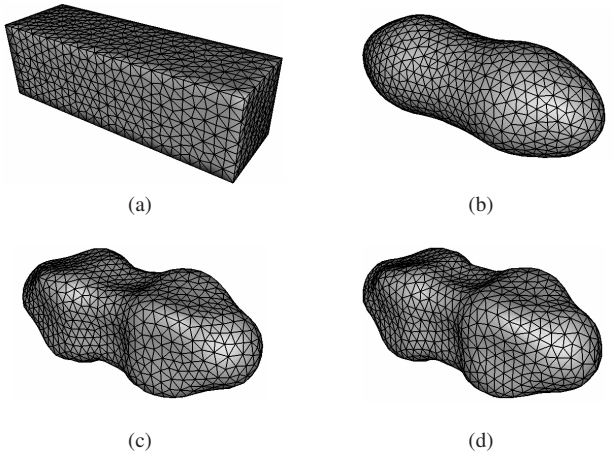

Figure 11: *a) Initial conditions for the collapse of a liquid block due to surface tension in zero gravity. (b) Naïve barycentric interpolation on tetrahedra generates very little detail. (c) Generalized barycentric interpolation over Voronoi cells retains interesting small scale structure. (d) Applying barycentric interpolation over our refined tetrahedra produces qualitatively consistent results.*

5. To interpolate at a point, locate the sub-tetrahedron containing the point and apply basic barycentric interpolation from its four associated data points (i.e. one site, one face centroid, and two Voronoi vertices).

One potential issue, not unique to our method, is that despite enforcing a lower bound on the distance between pressure samples, our unstructured sampling can cause sliver tetrahedra in the unmodified Delaunay tetrahedralization. While we found this posed little problem for the pressure projection, it can cause the least squares velocity reconstructions to be ill-conditioned due to nearly co-planar face normals. This can be readily resolved by requesting that the mesh generator add Steiner points to enforce fairly lax quality bounds; because our embedded pressure projection does not require the mesh generator to match boundaries, this is relatively inexpensive and effective. If mesh quality cannot be improved sufficiently, using additional nearby velocity samples in the reconstruction can ameliorate this at the cost of a smoother result.

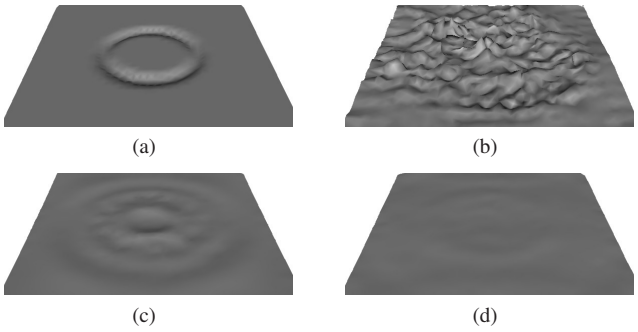

Figure 12: ***Surface noise.*** *(a) A perturbation is introduced into a smooth surface. (b) On a regular tetrahedral mesh, the sub-mesh-resolution noise causes instability. (c, d) With adaptively-placed samples, the surface noise is accurately captured by the fluid solver and initially causes ripples before steadily settling.*

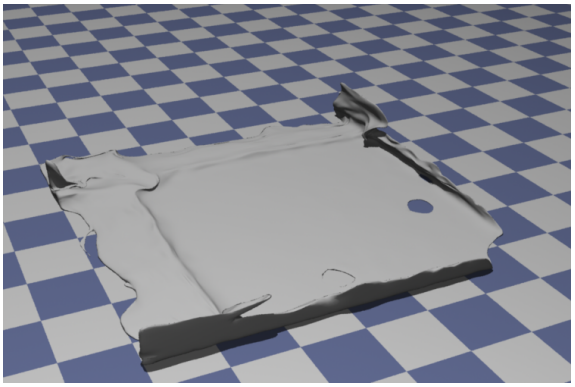

Figure 13: ***Thin Sheet.*** *Seeding pressure samples directly inside the fluid volume allows us to capture almost arbitrarily thin sheets.*

7 Results

7.1 Sampling

The issues that arise from regular, non-geometry-aware pressure sampling are common and consistent across Cartesian grids, octrees, Voronoi meshes, and tetrahedral meshes. We will therefore use Voronoi meshes throughout, and simply compare our geometry-aware sampling against naïve regular sampling.

Surface Noise: As discussed above, regularly-spaced pressure samples can miss fine surface details, resulting in surface noise which is never physically smoothed out. Figure 12 illustrates that our sampling approach successfully resolves and corrects such small surface details. In contrast, regular samples cannot fully capture the initial surface perturbation, so it cannot be rectified. Though the ghost fluid method on regular samples does detect some differences in surface height, this actually exacerbates the problem because noisy sub-mesh details will appear to the simulator as rapid discontinuous changes in surface position over time, inducing noisy responses in the fluid velocity.

Topology Mismatch: Another visible artifact of using mismatched surface and simulation resolutions is topological inconsistencies. For example, a surface with two disjoint volumes of liquid may appear to the solver as one volume, resulting in a premature response. Figure 9 shows a liquid drop impacting a still surface. With regular sampling, the droplet begins to influence the static liquid before the surfaces are actually joined. Because our adaptively-placed samples match the topology of the surface tracker, they easily correct this spurious motion. Figure 1 also features such a topological merge, along with many splitting and tearing operations, with timings listed in table 1.

Thin Features: To illustrate our method's ability to animate thin features, figure 13 shows a scene in which we drop a small sphere of liquid onto the ground. Thin sheets rapidly develop as the fluid spreads out across the floor. With regular pressure samples, sheets of this kind often end up between samples, effectively disappearing from the solver. Our sampling ensures that almost arbitrarily thin sheets of liquid remain visible to the solver, and as such, interesting rippling and splashing motion still occurs.

Our method also resolves thin sheets and small surface details generated by large splashes, as shown in figure 1. To counteract gradual volume drift, we do add a corrective motion-in-the-normal-direction [Brochu 2006; Müller 2009], which further aids in preserving thin sheets. Our video also includes an example of a column of liquid being released into a still pool. Although we are using only first-order semi-Lagrangian advection, the liquid motion remains lively and active throughout. We suspect that because our method retains sharp wave peaks and splashes rather than continually eroding them, their extra kinetic and gravitational potential energy is retained in the simulation, accounting for this reduced dissipation.

Table 1 gives timings for our 3D examples. All figures are averages per frame and all timings are in seconds. These simulations used no more than 320K tetrahedra each, whereas recent tetrahedra-based free surface methods used up to 4 times more tetrahedra to achieve a similar level of detail.

7.2 Surface Tension

Figure 4 illustrates the action of our surface tension model on a low resolution cube in zero gravity. Rather than quickly collapsing into a sphere, a cascade of detailed capillary waves propagate along the surface, causing it to oscillate rapidly. It initially inverts almost completely into an octahedron (the geometric dual of a cube), and continues to oscillate for many subsequent frames. To illustrate the benefits of our sampling approach in the context of surface tension, we launch an identical simulation using the same time steps on a regular mesh. Because this mesh cannot respond and correct high frequency sub-mesh details present in the curvature estimates, the simulation becomes unstable almost immediately. Applying an excessively strict timestep restriction only brings the simulation to a halt as the surface noise introduces increasingly sharp features.

Inspired by an example from the work of Wojtan & Turk [2008], we run another zero gravity simulation on a rectangular block (see figure 11). Because our simulation does not use diffusive Laplacian mesh smoothing and applies accurate mesh-based surface tension forces discontinuously at the interface, we retain substantially greater detail in the resulting capillary wave motion.

7.3 Interpolation

We revisit our surface tension block example to compare different interpolation schemes. As seen in figure 11, our barycentric method is substantially less damped than the naïve barycentric interpolation approach, and matches the more complex generalized barycentric interpolant.

Statistic	Thin sheet	Liquid column	Sphere Splash
# tetrahedra	141,701	197,911	313,587
Velocity reconstruction (s)	3	8	18
Surface tracking (s)	7	37	26
Remeshing (s)	15	39	69
Velocity advection (s)	7	18	15
Redistancing (s)	5	22	42
Pressure solve (s)	0.29	1.8	0.45
Total simulation time (s)	37	127	171

Table 1: *Simulation statistics for 3D examples (all statistics are per-frame values, averaged over all frames).*

8 Discussion and Limitations

Our implementation is not heavily optimized, and we defer various potential performance gains to future work. Obvious optimizations include: reducing the number of tetrahedra through smarter sampling, improving the broad phase algorithm for point-location queries, and streamlining the construction of mesh data structures. More fundamentally, our Voronoi simulator is in many ways dual to a tetrahedral scheme, and for a given mesh the number of velocity samples is identical; we believe that approximately comparable costs are therefore reasonable to expect.

The main contribution of this paper is the coupling of simulation elements to an existing explicit surface tracking method, and not the explicit surface tracking itself. Therefore, not all artifacts due to surface tracking are addressed. For example, *El Topo* delays handling some very difficult collisions for a few timesteps until the topological operations can be safely processed, which occasionally yields visible lingering surface noise. (Reducing the time step size can help by introducing fewer and simpler collisions, and more aggressive simplification can also be enabled by tuning the volume change tolerance that *El Topo* uses to decide whether to accept a given simplification.) Likewise, despite the use of feature-preserving mesh improvement, some popping artifacts due to on-the-fly remeshing are still visible in our animations. We chose *El Topo* because its resolution is not constrained to a regular grid and it is therefore able to showcase very thin features; nevertheless our method could adapt to any of the front tracking methods mentioned in section 2.2.

Surface tension was only used for examples in subsections 7.2 and 7.3. Our goal in many of the other examples was to highlight the ability to track thin sheets, whereas surface tension would break these sheets into droplets. Moreover, explicit surface tension schemes, such as the ghost-fluid-based method used in this paper, suffer from a stringent $O(\Delta x^{\frac{3}{2}})$ time step restriction for stability, which is particularly costly when small scale capillary waves are not erroneously damped out. Pursuing a more efficient, fully implicit surface tension model is a promising future direction.

9 Conclusions and Future Work

We have shown that with careful placement of pressure samples, our Voronoi mesh-based fluid solver makes it possible for explicit surface tracking to achieve its full potential in capturing small scale liquid features. In addition, we adapted embedded boundary pressure projection techniques to Voronoi meshes, introduced a simple improvement to barycentric velocity interpolation for Voronoi/Delaunay meshes, and extended the ghost fluid surface tension model with mesh-based curvature in order to capture complex capillary waves with minimal damping.

Several directions for future work remain. For example, it may be possible to enhance our sampling scheme in various ways, perhaps by exploiting curvature adaptivity, topological information, or measures of vorticity and velocity variation. Likewise, improvements to front tracking would be welcome, such as curvature-driven adaptivity, or greater robustness and efficiency. Lastly, many common extensions to basic inviscid liquid simulation rely on regular grids, and would need to be adapted to accomodate our approach.

10 Acknowledgements

This work was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada. We would like to thank our anonymous reviewers for their helpful comments and suggestions.

References

BARGTEIL, A. W., GOKTEKIN, T. G., O'BRIEN, J. F., AND STRAIN, J. A. 2006. A semi-Lagrangian contouring method for fluid simulation. *ACM Trans. Graph. 25*, 1, 19–38.

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3, 100.

BATTY, C., XENOS, S., AND HOUSTON, B. 2010. Tetrahedral embedded boundary methods for accurate and flexible adaptive fluids. In *Proc. Eurographics 2010*, to appear.

BRACKBILL, J. U., KOTHE, D. B., AND ZEMACH, C. 1992. A continuum method for modeling surface tension. *J. Comput. Phys. 100*, 2, 335–354.

BRIDSON, R. 2008. *Fluid Simulation for Computer Graphics*. A K Peters.

BROCHU, T., AND BRIDSON, R. 2009. Robust topological operations for dynamic explicit surfaces. *SIAM J. Sci. Comput. 31*, 4, 2472–2493.

BROCHU, T. 2006. *Fluid Animation with Explicit Surface Meshes and Boundary-Only Dynamics*. Master's thesis, University of British Columbia.

CAMPEN, M., AND KOBBELT, L. 2010. Exact and robust (self-)intersections for polygonal meshes. *Proc. Eurographics 2010*, to appear.

CHENTANEZ, N., GOKTEKIN, T. G., FELDMAN, B. E., AND O'BRIEN, J. F. 2006. Simultaneous coupling of fluids and deformable bodies. In *Proc. Symp. Comp. Anim. 2006*, 83–89.

CHENTANEZ, N., FELDMAN, B. E., LABELLE, F., O'BRIEN, J. F., AND SHEWCHUK, J. R. 2007. Liquid simulation on

lattice-based tetrahedral meshes. In *Proc. Symp. Comp. Anim. 2007*, 219–228.

DU, J., FIX, B., GLIMM, J., JIA, X., LI, X., LI, Y., AND WU, L. 2006. A simple package for front tracking. *J. Comput. Phys. 213*, 2, 613–628.

ELCOTT, S., TONG, Y., KANSO, E., SCHRÖDER, P., AND DESBRUN, M. 2007. Stable, circulation-preserving, simplicial fluids. *ACM Trans. Graph. 26*, 1, 4.

ENRIGHT, D., FEDKIW, R., FERZIGER, J., AND MITCHELL, I. 2002. A hybrid particle level set method for improved interface capturing. *J. Comp. Phys. 183*, 1, 83–116.

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

ENRIGHT, D., NGUYEN, D., GIBOU, F., AND FEDKIW, R. 2003. Using the particle level set method and a second order accurate pressure boundary condition for free surface flows. In *Proc. 4th ASME-JSME Joint Fluids Engineering Conference*.

FELDMAN, B. E., O'BRIEN, J. F., AND KLINGNER, B. M. 2005. Animating gases with hybrid meshes. *ACM Trans. Graph. (Proc. SIGGRAPH) 24*, 3, 904–909.

FELDMAN, B. E., O'BRIEN, J. F., KLINGNER, B. M., AND GOKTEKIN, T. G. 2005. Fluids in deforming meshes. In *Proc. Symp. Comp. Anim. 2005*, 255–259.

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

FRANKLIN, J. D., AND LEE, J. S. 2010. A high quality interpolation method for colocated polyhedral/polygonal control volume methods. *Computers & Fluids 39*, 6, 1012–1021.

GARLAND, M., AND HECKBERT, P. 1997. Surface simplification using quadric error metrics. In *Proc. SIGGRAPH '97*, 209–216.

GLIMM, J., GROVE, J. W., LI, X. L., SHYUE, K.-M., ZENG, Y., AND ZHANG, Q. 1998. Three-dimensional front tracking. *SIAM J. Sci. Comput. 19*, 3, 703–727.

GOKTEKIN, T. G., BARGTEIL, A. W., AND O'BRIEN, J. F. 2004. A method for animating viscoelastic fluids. *ACM Trans. Graph. (Proc. SIGGRAPH) 23*, 3, 463–468.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. *ACM Trans. Graph. (Proc. SIGGRAPH) 24*, 3, 973–981.

HONG, J.-M., AND KIM, C.-H. 2003. Animation of bubbles in liquid. *Computer Graphics Forum 22*, 3, 253–262.

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

HONG, J.-M., SHINAR, T., KANG, M., AND FEDKIW, R. 2007. On boundary condition capturing for multiphase interfaces. *J. Sci. Comput. 31*, 1-2, 99–125.

JIAO, X. 2007. Face offsetting: A unified approach for explicit moving interfaces. *J. Comput. Phys. 220*, 2, 612–625.

KIM, D., SONG, O.-Y., AND KO, H.-S. 2009. Stretching and wiggling liquids. *Proc. SIGGRAPH Asia 2009*, 120.

KLINGNER, B. M., FELDMAN, B. E., CHENTANEZ, N., AND O'BRIEN, J. F. 2006. Fluid animation with dynamic meshes. *ACM Trans. Graph. (Proc. SIGGRAPH) 25*, 3, 820–825.

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

MEYER, M., BARR, A., LEE, H., AND DESBRUN, M. 2002. Generalized barycentric coordinates on irregular polygons. *J. Graph. Tools 7*, 1, 13–22.

MEYER, M., DESBRUN, M., SCHRODER, P., AND BARR, A. H. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath*.

MIHALEF, V., UNLUSU, B., METAXAS, D., SUSSMAN, M., AND HUSSAINI, M. Y. 2006. Physics based boiling simulation. In *Proc. Symp. Comp. Anim. 2006*, 317–324.

MULLEN, P., MCKENZIE, A., TONG, Y., AND DESBRUN, M. 2007. A variational approach to Eulerian geometry processing. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3, 66.

MÜLLER, M. 2009. Fast and robust tracking of fluid surfaces. In *Proc. Symp. Comp. Anim. 2009*, 237–245.

NESME, M., KRY, P. G., JEŘÁBKOVÁ, L., AND FAURE, F. 2009. Preserving topology and elasticity for embedded deformable models. *ACM Trans. Graph. (Proc. SIGGRAPH) 28*, 3, 52.

PEROT, B., AND NALLAPATI, R. 2003. A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows. *J. Comput. Phys. 184*, 1, 192–214.

SHIN, S. 2007. Computation of the curvature field in numerical simulation of multiphase flow. *J. Comput. Phys. 222*, 2, 872–878.

SI, H. 2006. *TetGen: A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator*.

SIN, F., BARGTEIL, A., AND HODGINS, J. 2009. A point-based method for animating incompressible flow. In *Proc. Symp. Comp. Anim. 2009*, 247–255.

TERAN, J., SIFAKIS, E., BLEMKER, S. S., NG-THOW-HING, V., LAU, C., AND FEDKIW, R. 2005. Creating and simulating skeletal muscle from the visible human data set. *IEEE Trans. Vis. Comp. Graph. 11*, 3, 317–328.

WENDT, J. D., BAXTER, W., OGUZ, I., AND LIN, M. C. 2007. Finite volume flow simulations on arbitrary domains. *Graph. Models 69*, 1, 19–32.

WOJTAN, C., AND TURK, G. 2008. Fast viscoelastic behavior with thin features. *ACM Trans. Graph. (Proc. SIGGRAPH) 27*, 3, 47.

WOJTAN, C., THÜREY, N., GROSS, M., AND TURK, G. 2009. Deforming meshes that split and merge. *ACM Trans. Graph. (Proc. SIGGRAPH) 28*, 3, 76.

ZHENG, W., YONG, J.-H., AND PAUL, J.-C. 2006. Simulation of bubbles. In *Proc. Symp. Comp. Anim. 2006*, 325–333.