Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

A Level-set Method for Skinning Animated Particle Data

Haimasree Bhatacharya
University of Utah

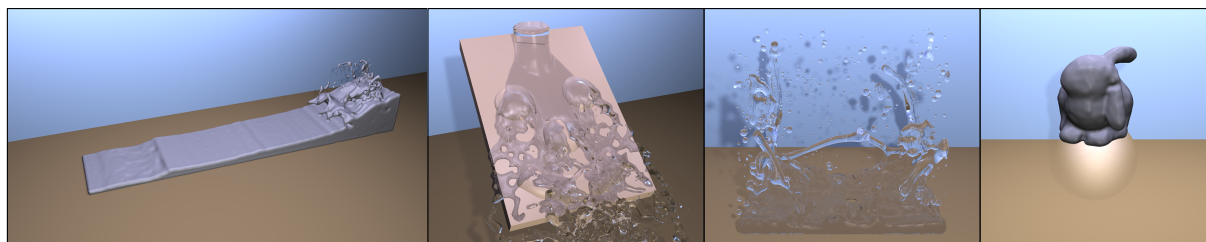Yue Gao
Tsinghua University

Adam Bargteil
University of Utah

Figure 1: *We present a technique for skinning animated particle data and demonstrate it on a variety of particle simulation systems. From left to right: A fluid-implicit particle simulation [ZB05], a simulation using the method of Sin et al. [SBH09], a smoothed particle hydrodynamics simulation [APKG07], and an elasto-plastic smoothed particle hydrodynamics simulation [GBB09].*

Abstract

In this paper, we present a straightforward, easy to implement method for particle skinning—generating surfaces from animated particle data. We cast the problem in terms of constrained optimization and solve the optimization using a level-set approach. The optimization seeks to minimize the thin-plate energy of the surface, while staying between surfaces defined by the union of spheres centered at the particles. Our approach skins each frame independently while preserving the temporal coherence of the underlying particle animation. Thus, it is well-suited for environments where particle skinning is treated as a post-process, with each frame generated in parallel. We demonstrate our method on data generated by a variety of fluid simulation techniques and simple particle systems.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, Surface, Solid, and Object Representations.

Keywords: Particle skinning, level-set methods, particle systems, fluid simulation, surface smoothing, constrained smoothing.

1. Introduction

Particles are a ubiquitous primitive in computer animation. From simple particle systems to high-resolution smoothed particle hydrodynamics simulations, particles have been used to animate a vast range of phenomena and a huge number of special effects. While early particle systems rendered the particles directly into a framebuffer [Ree83, Sim90], more recently it has become common to use the particles to define a volume surrounded by a surface. Consequently, a vast number of techniques and industry tools have been developed to generate surfaces defined by particle animations, an operation we refer to as *particle skinning*.

Inspired by the work of Williams [Wil08], we cast particle skinning as a constrained optimization problem: intuitively, we seek the "smoothest" surface that approximates the geometry implied by the particles. We formalize the intuitive notion of "smoothness" by minimizing the thin-plate energy of the surface. To ensure that the surface captures the geometry implied by the particle set we constrain the surface to lie between two surfaces S_{min} and S_{max}, defined as the constructive solid geometry (CSG) union of spheres of radius
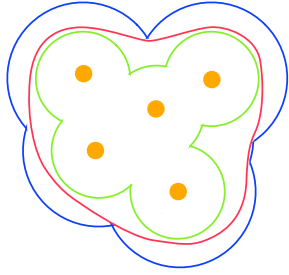
Figure 2: *Our approach seeks a smooth surface (red curve) that is constrained to lie between two surfaces, S_{min} (green curve) and S_{max} (blue curve), defined by the union of spheres centered at the particles (orange points).*

r_{min} and r_{max}, respectively, centered at the particles (see Figure 2).

Our key technical contribution is that, unlike Williams [Wil08], we solve this optimization problem using a level-set approach. We store signed-distance functions that represent the constraint surfaces and the surface being smoothed. Enforcing the constraints is as simple as taking a min and max for each grid-point after each smoothing iteration. Smoothing is accomplished by solving a level-set equation based on the biharmonic operator, ∇^4. Our level-set approach allows us to use the same spatial discretization for every frame, which is key to achieving temporal coherence even though we process every frame independently.

Additionally, our approach is flexible and supports variably sized and anisotropic particles, with anisotropy defined either by particle neighborhoods [YT10] or velocities. In short, our approach produces smooth, temporally coherent surfaces while processing every frame independently. We demonstrate our method on data generated by a variety of fluid simulation techniques and simple particle systems (see Figure 1).

2. Related Work

The pioneering work of Blinn [Bli82] introduced *blobbies*, also known as metaballs, as a primitive for representing shapes. This approach defines a scalar field that is the sum of three-dimensional Guassian kernels defined at a set of points. Surfaces are then taken to be a particular iso-contour of this scalar field. Blobbies tend to smooth regions of the surface between several particles, making them preferable to a union of spheres in many contexts. As the name implies, this approach tends to produce "blobby," or lumpy, surfaces. In practice, smoothing techniques are often applied to these surfaces in an attempt to remove the "blobbiness." One of the earliest attempts to skin animated particle systems [DG98] rasterized blobbies onto a regular grid and applied surface tension forces to smooth the surface.

Recent interest in the graphics community in particle-based fluid simulations has sparked a renewed interest in creating surfaces from particle sets. Müller and colleagues [MCG03] improved upon metaballs by dividing a particle's contribution to the scalar field by the SPH estimate of the density. Zhu and Bridson [ZB05] describe a quite different approach that defines a distance to the surface at every point in space. This distance is computed based on scattered data interpolation of particle radii and a weighted average of the distance to neighboring particles. While this approach produces very good results early in the animation, without a way to update particle radii, the results deteriorate over time. Shen and Shah [SS07] use a similar approach and note temporal discontinuities, which they address by blending adjacent frames. Adams and colleagues [APKG07] improved upon the results of Zhu and Bridson [ZB05] by recomputing the particle-to-surface distances every timestep. Unfortunately, the changing distances make surface generation a sequential process, ill-suited to treatment as a parallel post-process. Moreover, both of these approaches sometimes produce spurious surfaces in concavities. Solenthaler and colleagues [SSP07] address this problem by preventing the generation of surface in regions where the average neighbor position changes quickly. Museth and colleagues [MCZ07] describe a very high-resolution particle surfacing pipeline that incorporates a variety of post-processing techniques including temporal and spatial anti-aliasing.

Perhaps, the first to cast the problem of generating surfaces from particle data as a constrained optimization problem was Williams [Wil08]. We embrace this formulation, but diverge in how the optimization is performed. Whereas they tessellate an initial surface using a variation of marching cubes [LC87] and perform mesh-based smoothing to optimize the surface, we perform the optimization using a level-set approach. Mesh-based smoothing methods are highly sensitive to mesh topology and Williams' *marching tiles* mesh extraction method does not guarantee that meshes adjacent in time have the same topology. Consequently, some temporal incoherence is expected. By adopting a level-set framework, we can ensure that the level-set mesh is identical for every frame allowing us to preserve the temporal coherence in the particle data. We note that Williams [Wil08] suggested a level-set approach as future work and that Sin and colleagues [SBH09] briefly mention using such a variation (actually, an early version of our code). This paper details the approach and provides a variety of examples and comparisons with alternatives.

Very recently, Yu and Turk [YT10] demonstrated very impressive results for the particle skinning problem. Their approach used a single pass of Laplacian smoothing of particle positions, followed by defining a metaballs-like surface with anisotropic smoothing kernels. The anisotropic kernels conform far better to the surface defined by the particles than isotropic kernels. Theirs is the first approach we know of that achieves appealing, smooth surfaces for each frame indepen-

dently without introducing any temporal artifacts. This paper only addresses the surface smoothing aspect of the problem. Instead of performing a single pass of Laplacian smoothing we minimize the thin-plate energy using a level-set method. This approach results in smoother surfaces than those of Yu and Turk [YT10] (see Figure 8). Moreover, our approach can also make use of anisotropic kernels (see Figure 10).

Our work is also related to the rich body of research on surface smoothing, or fairing. Graphics researchers have largely focused on smoothing surfaces with explicit, mesh-based representations (see e.g., [WW94, Tau95, Kob96, DMSB99, SKS01, BS05, WMKG07, WBH*07, HP07]). However, to avoid flickering artifacts from changing mesh structure, we prefer a level-set approach. Level-set methods are well known in computer graphics and vision and have been used for a wide variety of problems from virtual sculpting [BC02] to surface reconstruction [ZOF01] to liquid surface tracking [FF01]. A principal use of level-set methods is for surface smoothing. Early work examined motion by mean curvature, which is closely related to Laplacian smoothing. Chopp and Sethian [CS99] (see also [TWBO03]) examined motion by intrinsic Laplacian of curvature and describe the numerical difficulties they encounter. As such motion minimizes the bending energy independent of the quality of the parameterization, we initially took this approach. However, we found that the linear biharmonic operator gave as good results and was numerically much more stable. Premoze and colleagues [PTB*03] also seek a smooth surface that approximates particle data and take a level-set approach. However, their approach initializes the optimization for every frame with the surface from the previous frame. Surface generation then becomes a sequential process, ill-suited to parallel processing. For a treatment of level-set methods in general see the texts by Sethian [Set99] and Osher and Fedkiw [OF03].

Generating surfaces from animated particles can be cast as a surface tracking problem and in this way is related to the vast literature on surface tracking for the simulation of liquids (e.g. [EMF02,HK05,BGOS06,WT08,MÖ9,WTGT09]). Most closely related to our approach are the particle level-set methods [EMF02, HK05]. Like our approach these methods combine tracked particles with level-set methods. However, these approaches allow bidirectional feedback between the particle and level-set representations and are sequential in nature. In contrast, in our approach the particles initialize the level-set and provide constraints, but are static during the smoothing iterations.

3. Method

Our method comprises three steps. First, a regular grid is initialized with signed-distance fields representing S_{min}, S_{max}, and S_0, the initial guess of the smooth surface. Second, a constrained optimization smoothes the surface. Third, an explicit representation of the surface is extracted and rendered.

3.1. Level-set Initialization

We begin by initializing three scalar fields: ϕ_{min}, ϕ_{max}, and ϕ_0. The initialization of ϕ_{min} and ϕ_{max} is straightforward and involves a distance calculation to the nearest particle and subtraction of r_{min} and r_{max}, respectively. More formally,

$$\phi_{min}(i,j,k) = \min_p \|\mathbf{T}(\mathbf{x}(i,j,k) - \mathbf{x}(p))\| - r_{min}$$

$$\phi_{max}(i,j,k) = \min_p \|\mathbf{T}(\mathbf{x}(i,j,k) - \mathbf{x}(p))\| - r_{max}, \qquad (1)$$

where (i,j,k) refers to a grid point, $\mathbf{x}(i,j,k)$ and $\mathbf{x}(p)$ are the world-space positions of the grid point and the particle, p. $\mathbf{T}$ is an optional transformation to allow for anisotropy. We perform this initialization by rasterizing the particles onto the level-set gridpoints. Choosing the initial surface too near either constraint surface can increase the number of smoothing iterations required. While many values work well, we generally initialize ϕ_0 as,

$$\phi_0(i,j,k) = 0.5\left(\phi_{min}(i,j,k) + \phi_{max}(i,j,k)\right). \qquad (2)$$

After initialization, a fast sweeping method [Zha04] is run to ensure that ϕ_{min}, ϕ_{max}, and ϕ_0 are signed-distance functions. In all our examples, we apply a small number of iterations (15) of Laplacian smoothing ($\phi_t = \nabla^2\phi\|\nabla\phi\|$) to ϕ_0. Like Williams [Wil08], we have found that this preprocessing reduces the number of constrained optimization iterations required.

As is typical in level-set methods, our approach requires that we maintain ϕ only in a narrow band around the surface. In our implementation, ϕ is updated at all points within three grid cells of the zero levelset. We store all values on six regular grids (ϕ, ϕ_{min}, ϕ_{max} and three grids for intermediate computations) of doubles, leading to a memory cost of 48 bytes per grid-point. Thus, grids on the order of 200^3 typically fit in less than 0.5 GB, while 400^3 requires more than 3 GB. However, we note that the streaming nature of the smoothing computations leads to near-optimal cache performance. If very high resolution is desired the dynamic tubular grid (DT-Grid) structure [NM06] could be used.

All our levelset grids can be thought of as being sub-grids of an infinite background grid with grid-spacing h and containing the origin $(0,0,0)$. The fact that every frame uses the same background grid is the key to maintaining the temporal coherence of the particle data.

3.2. Constrained Optimization

We seek to minimize thin-plate energy,

$$E_{thinplate}(\phi) =$$
$$\frac{1}{2}\int_\Omega \left(\phi_{xx}^2 + \phi_{yy}^2 + \phi_{zz}^2 + 2\phi_{xy}^2 + 2\phi_{yz}^2 + 2\phi_{zx}^2\right) dxdydz. \quad (3)$$

The thin-plate energy is not an intrinsic property of the surface and penalizes the parameterization of the surface as well

as the shape of the surface. However, if ϕ is nearly a signed-distance field, $E_{thinplate}$, like the various intrinsic bending energies, is at a minimum for a sphere (see Figure 7). The variational derivative of $E_{thinplate}$ is the biharmonic (or bi-Laplacian) operator leading to the level-set equation,

$$\phi_t = -\nabla^4\phi\|\nabla\phi\| =$$
$$- \left(\phi_{xxxx} + \phi_{yyyy} + \phi_{zzzz} + 2\phi_{xxyy} + 2\phi_{yyzz} + 2\phi_{zzxx} \right)\|\nabla\phi\|, \tag{4}$$

which we integrate through fictitious time by

$$\phi^{t+\Delta t} = \phi^t - \Delta t \nabla^4\phi\|\nabla\phi\|. \tag{5}$$

The various high-order derivatives of ϕ in Equation (4) can be straightforwardly discretized using two rounds of second-order centered finite differences. We also use a second-order centered difference for the $\|\nabla\phi\|$ term. Constraints are enforced after every smoothing iteration by taking

$$\phi(i,j,k) = \min(\phi_{min}(i,j,k), \max(\phi_{max}(i,j,k), \phi(i,j,k))). \tag{6}$$

Periodically, a fast sweeping method ensures that ϕ is approximately a signed-distance function, preventing $\|\nabla\phi\|$ from becoming near zero everywhere.

To avoid applying different degrees of smoothing to different frames, which may lead to temporal incoherence, we do not iterate until convergence. Instead, we apply a fixed number of smoothing passes to each frame. This approach does imply that the number of smoothing passes is determined by the frame that requires the most, but this restriction has not been problematic in practice. In fact, as described in Section 3.4 we set the default number of smoothing passes sufficiently large to handle all of our examples.

Finally, we note that we do not have a proof that this simple optimization algorithm necessarily converges to the minimum thin-plate energy. However, in practice we have always gotten good results.

3.3. Surface Extraction

The final step in our approach is to extract the surface—the zero-set of ϕ_{final}. Our implementation applies marching tetrahedra [Blo94] using trilinear interpolation on the level-set grid. Note that the extracted surface is only used for rendering and, while it does introduce typical marching cubes artifacts, does not introduce any temporal incoherence.

3.4. Parameters

At first it may seem that there are many parameters to our approach: r_{min}, r_{max}, the number of smoothing passes, frequency of fast sweeping, timestep, and grid spacing. While hand-tuning these parameters can produce slightly improved results and reduce running times, good default values can be obtained by requiring the user to specify only the grid-spacing, h, which is related to the inter-particle spacing and

the desired level of detail in the resulting surfaces. The grid spacing can be quickly found by viewing ϕ_0 and ensuring that there is the desired amount of overlap between particles. Given h, we set

$$r_{min} = \frac{\sqrt{3}h}{2} \tag{7}$$

Equation (7) guarantees that every particle rasterizes to at least one grid point and prevents particles from disappearing in animation. The ratio between r_{min} and r_{max} provides a tradeoff between surface smoothness and faithfulness to the underlying particles (see Figure 3). Larger ratios generate smoother surfaces, while smaller ratios more faithfully represent the particles. We have found that a ratio of 4 works well for many examples and this value was used for all of the examples in Figure 1. We default to 500 passes of bi-harmonic smoothing with fast sweeping performed every 50 passes. The timestep defaults to

$$\Delta t = 0.01h^4 \tag{8}$$

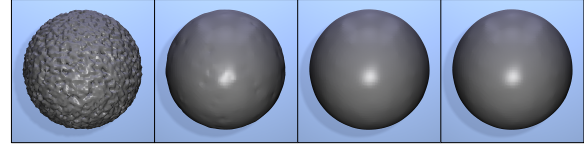though this relation is not perfect and we anticipate that some examples will require a smaller timestep.

Figure 3: *The effect of changing the ratio, r_{min}/r_{max}, from left to right, 1, 2, 4, and 8.*

4. Results and Discussion

We have tested our approach with a variety of particle based simulation systems (see Figure 1) as well as several common analytical test examples. Timing results for the examples in Figure 1 are in Table 1. These examples make clear that our method generates temporally coherent, smooth surfaces and is able to preserve much of the richness of the underlying particle motion.

example	particles	init	smooth	extract
left	217K	90	135	3
center-left	38K	8.0	70	2.5
center-right	52K	15	62	9
right	40K	13	54	2

Table 1: *Timing results (in seconds) for the examples in Figure 1 recorded on a single-core of an Intel Harpertown Xeon processor at 2.4 GHz with 16 GB of memory.*

Figure 5 shows the result of the "Enright" test. In this example we have tested our method with different resolutions
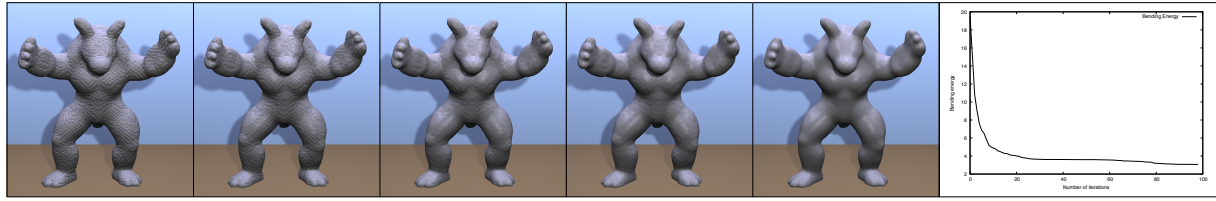
Figure 4: *The surface after (from left to right) 0, 5, 20, 30, and 100 smoothing iterations. Far right: bending energy as a function of smoothing iterations. This graph indicates that the thin-plate energy is a good proxy for bending energy.*
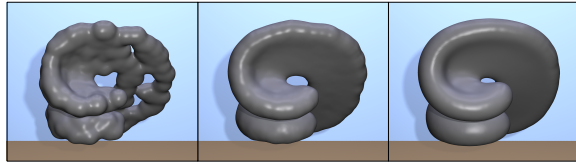

Figure 5: *The "Enright" test. The leftmost image is a low resolution example, the middle image is an intermediate resolution, and the rightmost example is a very dense particle set. With very coarse particles sets, the underlying discretization becomes apparent, but as the videos demonstrate, our technique maintains temporal coherence.*

to show that even for very coarse particle sets and grids, temporal coherence is maintained. The supplementary material also shows the Zalesak notched sphere example, which also does not flicker. We note that in these tests the particles are passively advected through the analytic flow field and, consequently, we would not expect to see the same sort of distortions commonly found in other surface tracking methods.
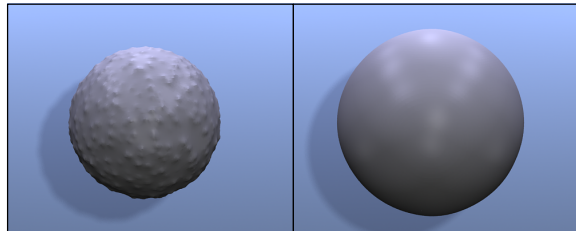

Figure 6: *Laplacian smoothing (left) shrinks the sphere until it starts interacting with the constraints, while our biharmonic smoothing (right) converges to a sphere.*

In Figure 6 we also show the advantage of biharmonic smoothing over the simpler Laplacian smoothing that is commonly employed. At first Laplacian smoothing quickly smoothes the surface, but additional iterations shrink the surface such that it starts interacting with the constraints. Volume correction techniques [DMSB99] could be employed to address the shrinkage, but these techniques lead to other artifacts (e.g. volume moving from one region of the fluid to an-

other). Alternatively, a limited number of smoothing passes could be applied, however it is difficult to know, a priori, how many passes to apply. Biharmonic smoothing, on the other hand, converges to a sphere without shrinking the surface and applying additional smoothing passes produces no ill effects (see Figure 4).
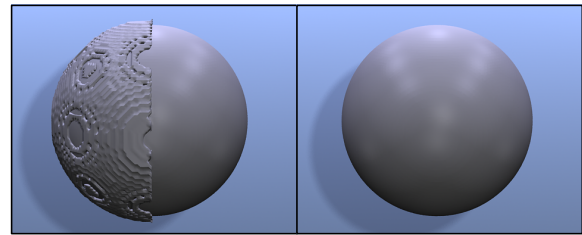

Figure 7: *With a bad parameterization (left), smoothing fails. With a signed distance function (right), biharmonic smoothing converges nicely.*

We also demonstrate one of the primary limitations of our approach in Figure 7. For the example on the left, we created a scalar field that is a signed distance function for $x > 0$, but is a function that rapidly increases away from the surface for $x < 0$. While smoothing by the intrinsic Laplacian of curvature [CS99] could theoretically handle this case because such motion is independent of the parameterization, the bad parameterization interferes with our linear operator and causes the smoothing to fail. Fortunately, this limitation can be addressed by occasionally applying a fast sweeping method to re-establish the signed-distance property. Another limitation, not so easily addressed, is that our method has difficulty generating very thin surfaces, even at high grid-resolutions. While for a mesh-based approach, the topology of the surface is fixed before smoothing begins, in a level-set approach the topology is free to change throughout the smoothing process. Thus, with our approach if r_{min} is set small enough that spheres centered at the particles do not overlap, the surface can break apart. With a mesh-based approach if pieces of surface are initially connected, they may get arbitrarily thin without breaking apart.

We also provide comparisons with previous approaches in the accompanying video. In particular, we compare with the

method of Adams and colleagues [APKG07] in the falling armadillo example; and with the mesh-based method of Williams [Wil08] in the example with the smiley face board. In both these cases it is clear that our approach maintains better temporal coherence. In Figure 8, we show a comparison with the method of Yu and Turk [YT10]. Our surface is quite smooth at modest particle counts, while theirs retains some lumpy features even with many more particles. An additional comparison with their double dam break is included in the supplementary material.

In Figure 9, we demonstrate our method's ability to handle variably sized particles. In this case r_{min} and r_{max} vary per particle and are a function of both the particle "radius" given by the simulator and the level-set grid-spacing. In Figure 10, we demonstrate our method's ability to handle anisotropic particles. The anisotropy can be determined either using a particle's local neighborhood as in Yu and Turk [YT10] or using velocity stretching where $\mathbf{T}$ in Equation (1) is given by

$$\mathbf{T} = \begin{pmatrix} \mathbf{v} & \mathbf{t}_0 & \mathbf{t}_1 \end{pmatrix} \begin{pmatrix} \frac{1}{(1+s)^2} & 0 & 0 \\ 0 & 1+s & 0 \\ 0 & 0 & 1+s \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \mathbf{t}_0 \\ \mathbf{t}_1 \end{pmatrix}, \quad (9)$$

where $\mathbf{v}$ is the normalized particle velocity, $\mathbf{t}_0$ and $\mathbf{t}_1$ are orthogonal to $\mathbf{v}$ and to each other, and s is the particle's (scaled) speed. This transformation has the effect of reducing distances in the velocity direction (and increasing distances in directions tangent to the velocity), thereby stretching particles in the direction of movement. In this way, our contribution is complementary to the anisotropic kernels advocated by Yu and Turk [YT10].

As future work, we would like to incorporate boundaries into our approach. Ideally, the surface should "snap" to solid boundaries. Currently, the surface can glide on top of or penetrate boundaries. We also believe it should be possible to
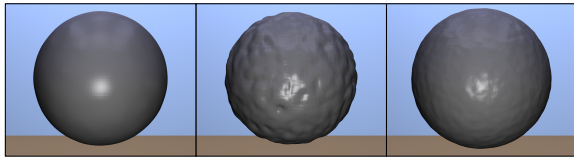

Figure 9: *A comparison of Adams et al [APKG07] approach (left) with ours (right) on an example with variably sized particles.*

derive boundary conditions for the level-set method that address the "reverse meniscus" that occurs along solid-fluid-air boundaries. The fact that level-set methods offer simple computations of CSG operations should help in these endeavors. It would also be interesting to investigate skinning multiple interacting fluids at once. Adding motion blur to our framework would be another interesting, and particularly useful direction. Motion blur is the motivation behind our velocity-based anisotropy, but the effect is somewhat different from traditional motion blur. It would also be interesting to enhance our approach to favor thin sheets, perhaps by borrowing ideas from Hamano and colleagues [HTNM06] or Kim and colleagues [KSK09]. While we believe our running times are fast enough for practical use, we would like to improve the speed of our implementation and explore parallel and GPU computation.

We have presented a method for skinning particle animations that processes each frame independently *and* maintains the temporal coherence of the particle data. Our method is fast and easy to implement, can handle variable-sized particles and anisotropy, and generates appealing, smooth surfaces. We fully expect our method to find its way into many production toolboxes where it will complement the vast array of particle skinning tools currently available.

Figure 8: *A comparison with the method of Yu and Turk [YT10]. A sphere is sampled (uniformly at random) with particles and given as input. Our approach (left) nearly recreates the sphere with 50,000 particles. The method of Yu and Turk [YT10], produces a fairly lumpy surface at 50,000 particles (center), but does much better with 1,000,000 particles (right), though some lumpiness is still present. These results imply that our approach may be especially well-suited for low particle counts.*
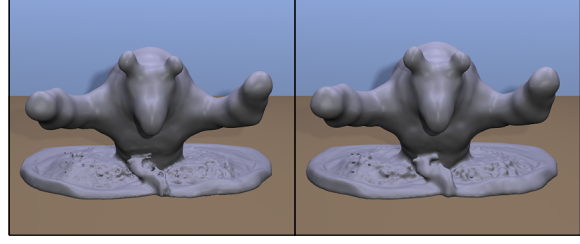
References

[APKG07]   ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. *ACM Trans. Graph. 26*, 3 (2007), 48.
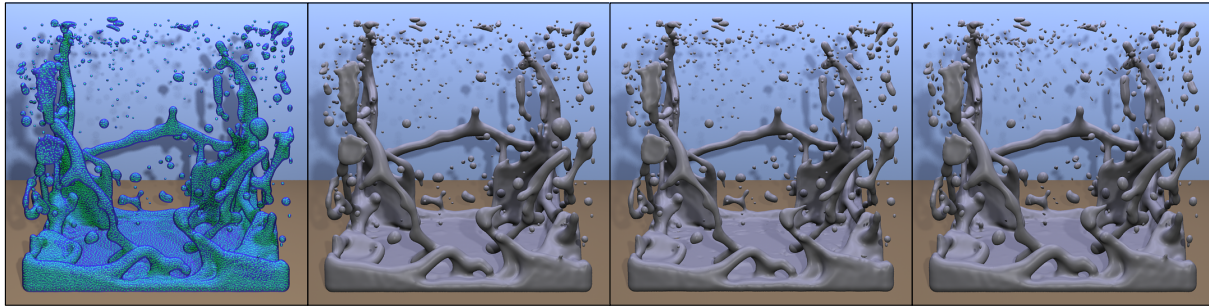
Figure 10: *From left: a rendering of the particles inside the generated surface, isotropic constraints, neighbor-based anisotropy and velocity-based anisotropy*

[BC02] BÆRENTZEN J. A., CHRISTENSEN N. J.: Interactive modelling of shapes using the level-set method. *International Journal of Shape Modelling 8*, 2 (2002), 79–97.

[BGOS06] BARGTEIL A. W., GOKTEKIN T. G., O'BRIEN J. F., STRAIN J. A.: A semi-lagrangian contouring method for fluid simulation. *ACM Trans. Graph. 25*, 1 (2006).

[Bli82] BLINN J. F.: A generalization of algebraic surface drawing. *ACM Trans. Graph. 1*, 3 (1982), 235–256.

[Blo94] BLOOMENTHAL J.: An implicit surface polygonizer. In *Graphics Gems IV*. Academic Press Professional, Inc., 1994, pp. 324–349.

[BS05] BOBENKO A. I., SCHRÖDER P.: Discrete willmore flow. In *Symposium on Geometry Processing* (2005), pp. 101–110.

[CS99] CHOPP D. L., SETHIAN J. A.: Motion by intrinsic laplacian of curvature. *Interfaces and Free Boundaries 1* (1999), 1–18.

[DG98] DESBRUN M., GASCUEL M.-P.: Active implicit surface for animation. In *Graphics Interface* (1998), pp. 143–150.

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *The Proceedings of ACM SIGGRAPH* (1999), pp. 317–324.

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

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

[GBB09] GERSZEWSKI D., BHATTACHARYA H., BARGTEIL A. W.: A point-based method for animating elastoplastic solids. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aug 2009).

[HK05] HIEBER S. E., KOUMOUTSAKOS P.: A lagrangian particle level set method. *J. Comput. Phys. 210*, 1 (2005), 342–367.

[HP07] HILDEBRANDT K., POLTHIER K.: Constraint-based fairing of surface meshes. In *The Proceedings of the Eurographics symposium on Geometry processing* (Aire-la-Ville, Switzerland, Switzerland, 2007), Eurographics Association, pp. 203–212.

[HTNM06] HAMANO R., TSUMURA N., NAKAGUCHI T., MIYAKE Y.: Rigid core particle: stabilization technique in large time step fluid simulation. In *ACM SIGGRAPH 2006 Research posters* (New York, NY, USA, 2006), SIGGRAPH '06, ACM.

[Kob96] KOBBELT L.: Discrete fairing. In *The Proceedings of the IMA Conference on the Mathematics of Surfaces* (1996), pp. 101–131.

[KSK09] KIM D., SONG O.-y., KO H.-S.: Stretching and wiggling liquids. *ACM Trans. Graph. 28* (December 2009), 120:1–120:7.

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3d surface construction algorithm. In *The Proceedings of ACM SIGGRAPH* (1987), pp. 163–169.

[MÖ9] MÜLLER M.: Fast and robust tracking of fluid surfaces. In *The Proceedings of the Symposium on Computer Animation* (New York, NY, USA, 2009), ACM, pp. 237–245.

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *The Proceedings of the Symposium on Computer Animation* (2003), pp. 154–159.

[MCZ07] MUSETH K., CLIVE M., ZAFAR N. B.: Blobtacular: surfacing particle system in "pirates of the caribbean 3". In *SIGGRAPH '07: ACM SIGGRAPH 2007 sketches* (New York, NY, USA, 2007), ACM, p. 20.

[NM06] NIELSEN M. B., MUSETH K.: Dynamic tubular grid: An efficient data structure and algorithms for high resolution level sets. *J. Sci. Comput. 26*, 3 (2006), 261–299.

[OF03] OSHER S., FEDKIW R.: *The Level Set Method and Dynamic Implicit Surfaces*. Springer-Verlag, New York, 2003.

[PTB*03] PREMOŽE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R.: Particle-based simulation of fluids. *Computer Graphics Forum 22*, 3 (2003), 401–410.

[Ree83] REEVES W. T.: Particle systems—a technique for modeling a class of fuzzy objects. *ACM Trans. Graph. 2*, 2 (1983), 91–108.

[SBH09] SIN F., BARGTEIL A. W., HODGINS J. K.: A point-based method for animating incompressible flow. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aug 2009).

[Set99] SETHIAN J. A.: *Level Set Methods and Fast Marching Methods*, 2nd ed. Cambridge Monograph on Applied and Computational Mathematics. Cambridge University Press, Cambridge, U.K., 1999.

[Sim90] SIMS K.: Particle animation and rendering using data parallel computation. In *The Proceedings of ACM SIGGRAPH* (New York, NY, USA, 1990), ACM, pp. 405–413.

[SKS01] SCHNEIDER R., KOBBELT L., SEIDEL H.-P.: Improved bi-laplacian mesh fairing. In *Mathematical Methods for Curves and Surfaces*. Vanderbilt University, 2001, pp. 445–454.

[SS07] SHEN C., SHAH A.: Extracting and parametrizing tem-

porally coherent surfaces from particles. In *The Proccedings of ACM SIGGRAPH (sketches)* (2007), p. 66.

[SSP07]   SOLENTHALER B., SCHLÄFLI J., PAJAROLA R.: A unified particle model for fluid-solid interactions. *Journal of Visualization and Computer Animation 18*, 1 (2007), 69–82.

[Tau95]   TAUBIN G.: A signal processing approach to fair surface design. In *The Proceedings of ACM SIGGRAPH* (New York, NY, USA, 1995), ACM, pp. 351–358.

[TWBO03]   TASDIZEN T., WHITAKER R., BURCHARD P., OSHER S.: Geometric surface processing via normal maps. *ACM Trans. Graph. 22*, 4 (2003), 1012–1033.

[WBH*07]   WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete quadratic curvature energies. *Comput. Aided Geom. Des. 24*, 8-9 (2007), 499–518.

[Wil08]   WILLIAMS B.: *Fluid Surface Reconstruction from Particles*. Master's thesis, University of British Columbia, 2008.

[WMKG07]   WARDETZKY M., MATHUR S., KÄLBERER F., GRINSPUN E.: Discrete laplace operators: no free lunch. In *The Proceedings of Eurographics symposium on Geometry processing* (2007), pp. 33–37.

[WT08]   WOJTAN C., TURK G.: Fast viscoelastic behavior with thin features. *ACM Trans. Graph. 27*, 3 (2008), 1–8.

[WTGT09]   WOJTAN C., THÜREY N., GROSS M., TURK G.: Deforming meshes that split and merge. *ACM Trans. Graph. 28*, 3 (2009), 1–10.

[WW94]   WELCH W., WITKIN A.: Free-form shape design using triangulated surfaces. In *The Proceedings of ACM SIGGRAPH* (New York, NY, USA, 1994), ACM, pp. 247–256.

[YT10]   YU J., TURK G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2010), SCA '10, Eurographics Association, pp. 217–225.

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

[Zha04]   ZHAO H.: A fast sweeping method for eikonal equations. *Mathematics of Computation 74* (2004), 603–627.

[ZOF01]   ZHAO H., OSHER S., FEDKIW R.: Fast surface reconstruction using the level set method. In *IEEE Workshop on Variational and Level Set Methods* (2001), pp. 194–202.