A Robust Volume Conserving Method for Character-Water Interaction

Minjae Lee
Stanford University
mjlgg@stanford.edu

David Hyde
Stanford University
dab@stanford.edu

Kevin Li
Stanford University
kevli@cs.stanford.edu

Ronald Fedkiw
Stanford University
Industrial Light + Magic
rfedkiw@stanford.edu

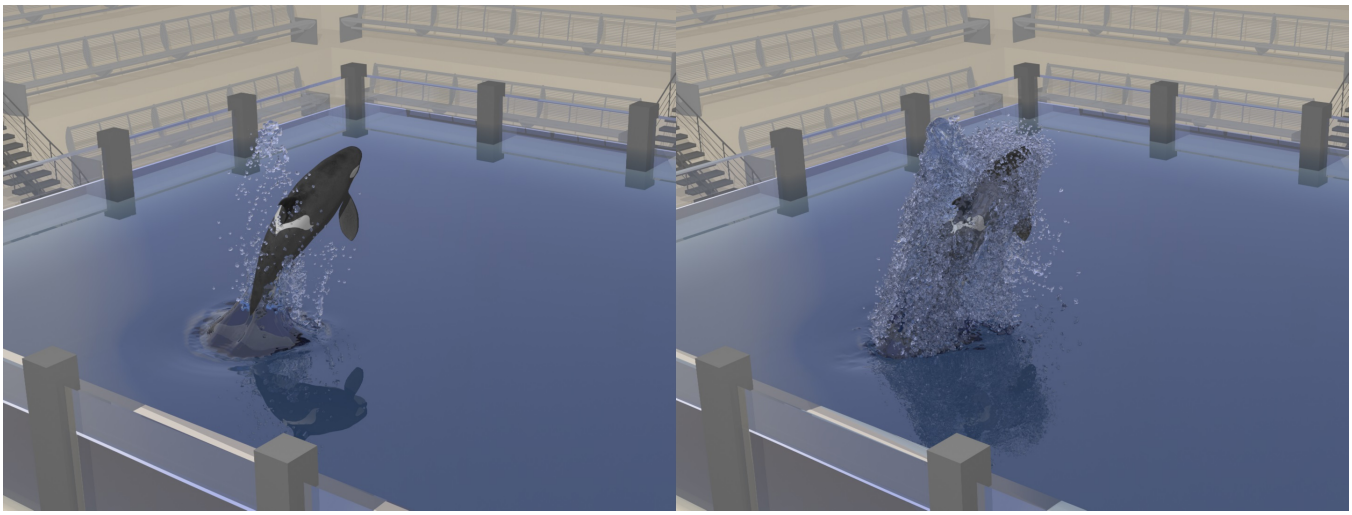

Figure 1: (Left) Whale breaching with the PLS method using automatically generated removed particles for spray. Very little of the water volume follows the whale's motion because of volume loss on the relatively coarse Eulerian background grid. (Right) Using the same Eulerian grid, our ALE based VOF method on the KDSM produces much more visually interesting sheeting and spray effects.

ABSTRACT

We propose a novel volume conserving framework for character-water interaction, using a novel volume-of-fluid solver on a skinned tetrahedral mesh, enabling the high degree of the spatial adaptivity in order to capture thin films and hair-water interactions. For efficiency, the bulk of the fluid volume is simulated with a standard Eulerian solver which is two way coupled to our skinned arbitrary Lagrangian-Eulerian mesh using a fast, robust, and straightforward to implement partitioned approach. This allows for a specialized and efficient treatment of the volume-of-fluid solver, since it is only required in a subset of the domain. The combination of conservation of fluid volume and a kinematically deforming skinned mesh allows us to robustly implement interesting effects such as adhesion, and anisotropic porosity. We illustrate the efficacy of our method by simulating various water effects with solid objects and animated characters.

SCA '19, July 26–28, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6677-9/19/07…$15.00
https://doi.org/10.1145/3309486.3340244

CCS CONCEPTS

• **Computing methodologies** → **Animation**; **Physical simulation**;

KEYWORDS

adaptive water simulation, volume conservation, sheeting, sprays

ACM Reference Format:
Minjae Lee, David Hyde, Kevin Li, and Ronald Fedkiw. 2019. A Robust Volume Conserving Method for Character-Water Interaction. In *SCA '19:The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '19), July 26–28, 2019, Los Angeles, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3309486.3340244

1 INTRODUCTION

Character-water interaction is a widespread phenomenon in the visual effects industry, and there have been many efforts to push for higher quality water interaction with animated characters such as King Kong in *Kong: Skull Island (2017)*, Hank the octopus in *Finding Dory (2016)*, and various characters in *Moana (2016)*.

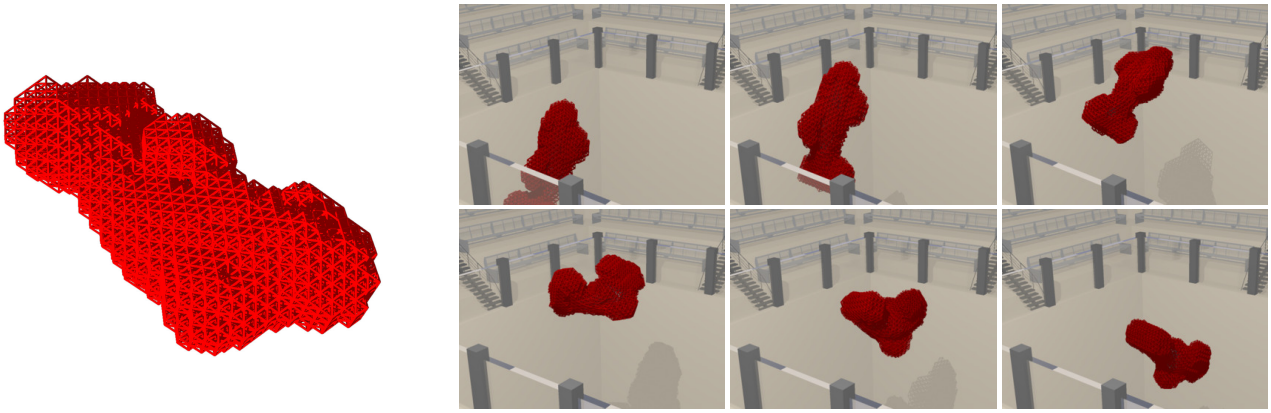

Figure 2: (Left) A KDSM mesh around a whale in a normalized pose (also known as T-pose or rest pose). (Right) A sample animation showing the KDSM skinned to follow an animation of a whale breaching.

Arguably, the most obvious approach to obtaining more detailed features anywhere in the domain is to place more degrees of freedom in the region of interest. A number of adaptive methods have been developed such as Adaptive Mesh Refinement (AMR) [Sussman et al. 1999], octree data structures [Losasso et al. 2004], [Losasso et al. 2006], [Aanjaneya et al. 2017], lattice based tetrahedral methods [Chentanez et al. 2007], [Batty et al. 2010], [Ando et al. 2013], and Chimera grids [English et al. 2013a], [English et al. 2013b]. While these methods greatly improve water simulation detail through adaptivity, various authors have noted numerous drawbacks including the need to remesh very often, difficulties in implementation, performance bottlenecks induced by high communication costs, and issues related to domain decomposition due to a large number of small patches. These issues are exacerbated when the adaptivity is required near boundaries with animated characters, since the character motion can rapidly change the region in space where the adaptivity is required. A more natural approach would be to use an adaptive mesh that moves with the character such as the recently proposed kinematically deforming skinned mesh (KDSM) of [Lee et al. 2018]. This allows one to prebake the adaptivity so that on-the-fly refinement is not required during the simulation. This makes the method straightforward to implement and robust in its handling of delicate phenomena.

Even with additional degrees of freedom near the animated character, the highly dynamic water motion and thin films are notoriously difficult to simulate due in large part to both volume loss and difficulties with imposing proper boundary conditions between the water and the character. We address volume conservation by proposing a novel volume-of-fluid (VOF) method implemented on the KDSM. Although our proposed VOF method is novel, it is similar in spirit to other VOF methods such as [Mihalef et al. 2004], [Mihalef et al. 2006] in that no fluid volume is lost, especially as compared to typical Eulerian methods. VOF method is a well known technique as demonstrated in [Hirt and Nichols 1981], [Brackbill et al. 1992], [Rider and Kothe 1998], and [Sussman and Puckett 2000]. There have been some recent interesting works on boundary conditions between solids and fluids such as [Zhang et al. 2016], [Zarifi and Batty 2017] using an Eulerian fluid grid (see [Akinci et al. 2013a] for SPH); however, it is more natural to specify these types of boundary conditions when the fluid grid is moving along with the solid in its Lagrangian frame, even if it is deforming a bit in that frame as is the case with KDSM. With this treatment, much of the fluid moves along with the mesh being driven by the character animation (which is also driving the mesh) meaning that less fluid volume flows from one computational cell to another. This is the typical arbitrary Lagrangian-Eulerian (ALE) approach, see for example [Feldman et al. 2005a], [Feldman et al. 2005b], [Klingner et al. 2006]. Notably, our method significantly differs from existing ALE implementations in that our ALE mesh is prebaked based on kinematically prescribed motion and has topology that remains consistent throughout the entire animation sequence. This separation of the remeshing step resolves a key problem of ALE based methods which can lack robustness due to the numerical instabilities caused by ill-formed elements–this can now be addressed during a preprocessing step.

In order to increase the overall efficiency and efficacy of our approach, we only utilize the ALE based VOF method on the KDSM near the animated character while using a standard Eulerian based Cartesian grid solver in the rest of the domain, in our case the particle level set (PLS) method [Enright et al. 2002a], [Enright et al. 2002b], including spray particles [Losasso et al. 2008]. It is important to note that the PLS method is a hybrid method combining particle and grid representations, and early work was presented in [Foster and Metaxas 1996] where they implemented a precursor to PIC/FLIP while removing unneeded interior particles far from the surface to boost performance and using level set to reconstruct smooth surface. Recently, [Ferstl et al. 2016] proposed further improvements of PLS and FLIP hybrid method. Thus, PLS and PIC/FLIP share important commonalities, and our VOF method can improve PLS or PIC/FLIP or any other method combining particle and grid representations.

Note that our spray particles carry mass and momentum as in [Losasso et al. 2008] for visual quality when spray particles interacts with water surface instead of using massless particles as in vanilla PLS. Importantly, the conservative nature of our VOF solver allows for relaxation of the numerical approach especially since the VOF solver is only required in a small subset of the domain near the character while a standard Eulerian solver is used elsewhere. Thus, we devise a straightforward partitioned (as compared to monolithic) approach to the coupling of the fluid flow equations

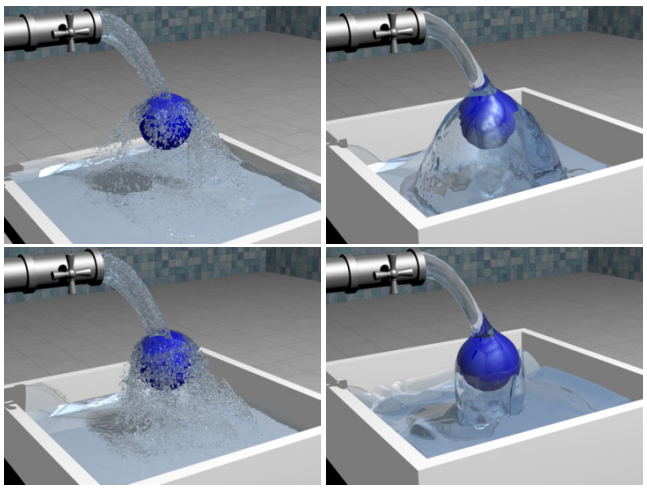

Figure 3: (Top Left) A standard PLS simulation on a relatively coarse Eulerian grid. (Top Right) Our ALE based VOF method on the KDSM achieves better water sheeting and volume conservation using the same Eulerian grid. (Bottom Row) Applying adhesion forces to both simulations produces the desired clinging to the ball with our method but has almost no effect on the PLS simulation.

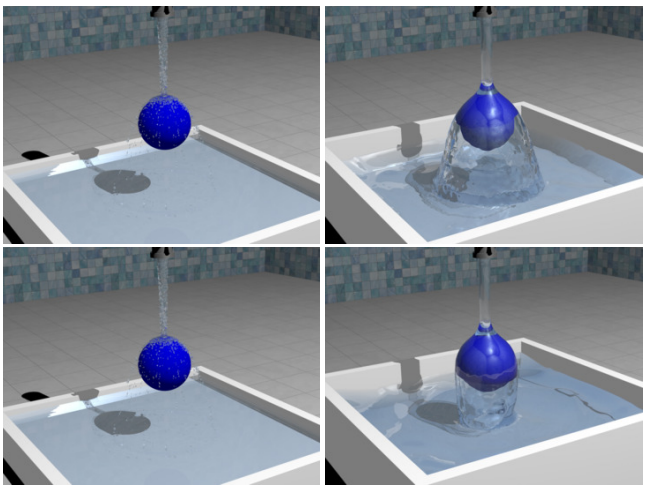

Figure 4: Same as Figure 3, but using an even smaller water stream accentuating the benefits of our approach especially when considering volume conservation.

between the Eulerian Cartesian grid and the ALE based VOF solver on the KDSM; see Section 4. Importantly, this combination of a standard Eulerian solver on the bulk of the domain with an ALE based KDSM mesh near the character allows the proposed VOF scheme to be incredibly simple, as discussed in Section 3, which is quite notable given the typical high level of complexity one usually confronts with VOF methods.

The first contribution of this paper is our strategy of prebaking the dense ALE mesh which occupies the space near the object or creature of interest, taking advantage of the adaptivity to capture detailed water phenomena based on the intuition that most interesting water effects are focused near the creature. We achieve a robust simulation method by separating the nontrivial mesh processing operations from the simulation stage and incorporating them into a preprocessing stage, where we precompute various auxiliary data in order to improve the performance of the simulation. Our second contribution is our novel VOF method, which conserves volume within the ALE mesh, whereas the PLS method in the background alone does not. Our approach of conserving volume near the object or creature of interest allows us to implement various adhesion and porosity effects robustly and with mechanisms for artistic control. The third contribution of our method is the straightforward partitioned approach for coupling the coarse background Eulerian grid and our fine ALE mesh, which greatly streamlines the development process.

2 KDSM

Following [Lee et al. 2018], we generate a KDSM from a tetrahedral BCC (body centered cubic) lattice as in [Molino et al. 2003] using a thickened level set of the triangulated surface skin mesh of a creature or an object in a normalized pose (see Figure 2 left). Then, given an animation sequence of the creature's triangulated surface skin mesh, the KDSM nodes inside the creature are morphed to follow the animation as per [Ali-Hamadi et al. 2013; Cong et al. 2015] capturing the kinematic deformation of the creature's skin and its volumetric interior. We connect the KDSM nodes that are exterior to the skin mesh of the creature to one another and to the internal nodes via a constitutive model (mass spring), so that the KDSM nodes that are external to the creature also follow the animation (see Figure 2). In addition, zero length spring attachments are connected between the creature's skin mesh and corresponding barycentric locations in the KDSM in order to obtain more accurate deformations of the KDSM near the creature's skin.

[Lee et al. 2018] embedded hair particles in the KDSM so that they would follow the skinned animation sequence (see Figure **??**). Each hair's base particle is embedded on the surface of the character's triangle mesh skin and the rest of hair particles are embedded in the tetrahedral mesh using hard bindings as in [Sifakis et al. 2007]. They also showed how a duplicated KDSM which is following the original constrained with zero length springs could be used to produce more dynamic hair behavior. Since the original KDSM sequence contains a significant portion of the desired motion, further iterations for dynamic hair behavior becomes much more efficient. In addition, they showed the benefits of the KDSM when simulating individual hairs as well as how to use the KDSM to implement a blendshape system including the effects of clumping, sagging, and matting. They simulated individual hairs using [Selle et al. 2008] and soft bindings (see [Sifakis et al. 2007]) connecting hair particles and their corresponding desired positions in the KDSM via zero length springs. They proposed a shape-preserving tetrahedral column to maintain the original style of the hair also connected to hair particles via zero length springs. The blendshape hair component is implemented as a collection of barycentric coordinates for each hair particle, and can be interpolated in time to implement a change in hairstyle over time (e.g. bear getting wet in the water). Then length preservation and interpenetration is run as a post-process, shortening the each hair segment to its rest length

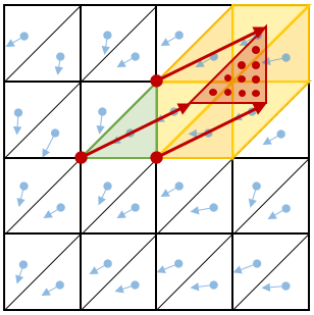

Figure 5: The red backtraced triangle uses each of its 10 quadrature point samples to transport water from the old mesh to the new one. Here, we render both the old and the new mesh in the same location assuming that the KDSM is not moving for simplicity of depiction. Each red dot point sample would attempt to remove 1/10 of the area of the red triangle from the specific yellow triangle that it is interior to.

(see [Müller et al. 2012; Sánchez-Banderas et al. 2015]) and applying pushout method from [Bridson et al. 2003]. Interestingly, they demonstrated how the air volume enclosed by the KDSM could be utilized to add adhesion and drag effects, porosity for hair, etc.

A major shortcoming of this prior work in regards to water simulation is that the KDSM is merely used to provide information such as forces that augment the treatment of the Eulerian grid. Thus, all the typical drawbacks of volume loss, etc., are not only still present but potentially worsened by these additional forces. See, for example, Figure 3 and 4.

3 KDSM FLUIDS

Our novel ALE based VOF method on the KDSM produces compelling results even though it has none of the complexities associated with typical VOF methods–even the volume conservation step is quite simple. We stress that the ability to use such a simple method is due in large part to the partitioned coupling discussed in Section 4, the adaptivity of the KDSM, exact volume conservation, and the Lagrangian nature of the KDSM as it follows the animated creature. Our VOF method fully conserves volume within the KDSM, while the rest of the domain simulated using the PLS method does not.

3.1 Precomputation

Starting from the original KDSM animation, we precompute auxiliary information such as adaptivity by subdividing the KDSM until a desired resolution is reached (for our examples, we subdivided KDSM multiple times until the size of tetrahedron is 4 to 8 times smaller than the background grid cell size). Prebaking subdivisions to obtain per-frame ALE meshes with consistent topology greatly increases the robustness and minimizes computation time as compared to typical ALE remeshing. We subdivide using [Molino et al. 2003], but only utilized the subdivision operation part without any adaptivity features as we wanted an evenly subdivided ALE mesh. Although we opt to precompute our KDSM, one could subdivide on-the-fly if desired. Note that we use a relatively coarse KDSM to run a mass spring simulation and then subdivide the resulting

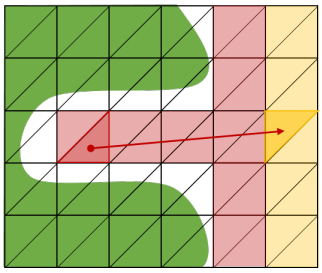

Figure 6: The triangles cut by the green solid surface region would be assigned rank 0, their node neighbors shown in red would be rank 1, and their node neighbors shown in yellow would be rank 2. The arrow shows how a rank 1 triangle needs to look non-locally in order to find a rank 2 triangle that it can deposit excess water into.

KDSM for performance reasons. One can always use a dense KDSM to obtain better boundaries near the creature's skin, but we found the current scheme sufficient for our examples. We additionally precompute a number of useful quantities such as per node velocities and a level set volume for every frame of the animation. Every cut cell tetrahedron, those containing a part of the creature's surface, is assigned a surface normal and object velocity. Those surface normals and object velocities are extrapolated to every tetrahedron of the KDSM exterior to the creature using the level set information.

Per tetrahedron solid occupancy is precomputed in the cut cells using point samples and the quadrature formula of [Zhang et al. 2009] testing how many point samples are inside of the creature using the level set representation. Instead of using the exact volume, we simply use the fraction of point samples inside and outside the creature to compute an approximate volume. This added simplicity is equivalent to a slight sub-tetrahedral perturbation of the solid surface. Obviously, this could be done more accurately, but we found that this simple method worked quite well, and simplicity and efficiency is highly desirable when one might want to refine near the surface of the creature on-the-fly.

Our proposed volume conservation method is motivated by the shock propagation for rigid bodies from [Guendelman et al. 2003]. As such, we precompute the rank of each tetrahedron as its topological distance from the creature, similar in spirit to the contact graph from [Guendelman et al. 2003]. Tetrahedra fully inside the creature are assigned a rank of -1, and partially filled tetrahedra are assigned rank 0. Then, all tetrahedra with unassigned ranks that are node neighbors of rank 0 tetrahedra are assigned rank 1. Rank 2, rank 3, etc. are assigned similarly.

3.2 Advection

We store vector valued velocities as per tetrahedron values, and multiplying by the mass of water in a tetrahedron yields momentum. Our ALE based VOF method updates the velocity and the amount of water in each tetrahedron on a new mesh given values on an old mesh.

First, for each tetrahedron, we trace its nodes backwards in time using its vector valued velocity multiplied by Δt. As shown in Figure 5, this rigidly translates the tetrahedron. Alternatively, one

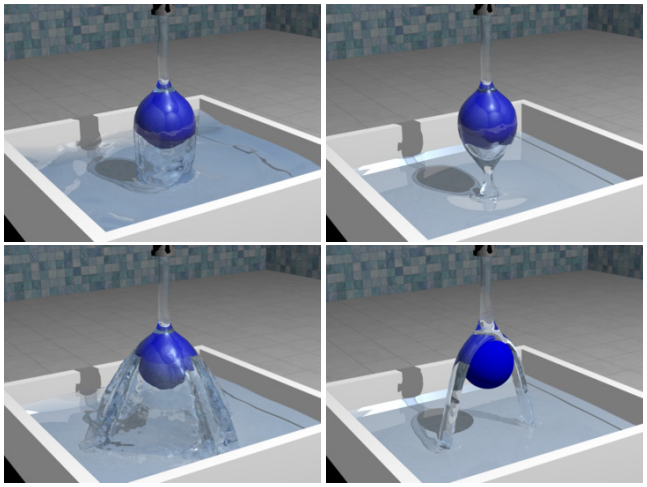

Figure 7: Our ALE based VOF method provides robust adhesion control to produce various effects. All four of these examples are obtained by merely varying adhesion effects.

could instead compute per node velocities, but we have found this unnecessary. If a backtraced node collides with a solid surface, it is clamped to that location. Thus, the backtraced tetrahedron does not deform unless it hits a solid surface. The resulting backtraced tetrahedron (shown in red in Figure 5) is used to collect water from the old mesh in order to deposit it on the new mesh. Instead of performing the usual complex geometric intersections between backtraced tetrahedra and the old mesh, we take a simpler approach using a number of quadrature formula point samples (again from [Zhang et al. 2009]). Each point sample (shown as a red dot in Figure 5) attempts to transport a certain amount of water from the old mesh tetrahedron it lies within (shown in yellow in Figure 5) to the original tetrahedron on the new mesh (shown in green in Figure 5). This potential amount of transported water is calculated as the volume of the backtraced tetrahedron divided by its number of point samples. Both water volume and associated momentum are transported. If a point sample falls outside the KDSM, we compute the amount and momentum of water to transport using interpolation from the background Eulerian grid. Note that this water is not removed from the background grid, since the background grid is treated in an Eulerian fashion and updated properly via the coupling proposed in Section 4.

The aforementioned advection might attempt to transport more water out of a tetrahedron on the old mesh than that tetrahedron contains. This could occur because of size differences between tetrahedra or because multiple point samples from separate tetrahedra of the new mesh request water from the same tetrahedron on the old mesh. Thus, in a second step, we visit every tetrahedron on the old mesh and scale down the amount of water each point sample removes in order to match the total volume of water in this old mesh tetrahedron as in [Lentine et al. 2011].

In a third step, we identify any tetrahedra on the old mesh that may have excess water, which was not transported to the new mesh, and forward advect this water to the new mesh similar to [Lentine et al. 2011], [Lentine et al. 2012], [Klingner et al. 2006]. However, our method can be much simpler because we track the exact volume of water with our VOF method, and therefore do not mind overfilling tetrahedra because they are drained to their appropriate volume during the volume conservation step (see Section 3.3). For each tetrahedron on the old mesh that requires forward advection, we use that tetrahedron's velocity to advect its nodes forward in time (in the opposite direction of Figure 5) and use its point samples to locate which tetrahedra in the new mesh will receive its water. Similar to backward advection, we clamp nodes that collide with solids, use the number of point samples and the original tetrahedron volume to decide on the fraction of water deposited in each target tetrahedron, and utilize special treatment for any point sample that leaves the KDSM and lands on the background Eulerian grid. In particular, we find particle creation to be quite useful for transporting water off of the KDSM (see Section 4.1).

3.3 Volume Preservation

Our volume conservation method is used to enforce incompressibility on the new mesh using our precomputed rank. The method consists of three parts: smear, pushout, and velocity correction. Pushout transports excess water and associated momentum outward from the creature's skin mesh using rank motivated by [Guendelman et al. 2003]. However, this ignores the fact that the fluid can rotate and move laterally, so we first apply what we refer to as a smearing step to account for this behavior. Both the smear and the pushout step transport the volume and associated momentum together. Finally, velocity correction is used to apply boundary conditions on the water from the creature.

We refer to tetrahedra as oversaturated when they contain more water than their volume should allow. The smearing step loops over oversaturated tetrahedra, except those on the boundary of the KDSM, and distributes the excess fluid equally to face neighbors. The boundary tetrahedra are taken care of in the pushout phase.

For pushout, we iterate over oversaturated tetrahedra in order from the lowest rank to the highest starting with tetrahedra of rank 0 which intersect the creature's skin mesh. For each oversaturated tetrahedron, we distribute as much excess fluid as possible equally to its face neighbors that are not yet fully saturated. If there is still excess fluid after every neighbor is saturated, it is distributed equally to all face neighbors with strictly higher ranks. Note that it is possible to have a tetrahedron without any valid neighbors to distribute water to, since the creature skin mesh can have narrow space between solid surfaces as illustrated in Figure 6. To handle this case, we preprocess the neighbor information using breadth first search to look for non-local neighbors with higher ranks to properly transport excess water to. Although not trivial, this can be done in the preprocessing step after determining rank. For boundary tetrahedra, we transfer excess fluid to the background Eulerian grid for particles as discussed in Section 4 and 4.1, respectively.

Finally, velocity correction is used to apply boundary conditions on the fluid from the creature. We assign a Boolean flag per tetrahedron indicating whether its fluid velocity needs to be corrected, and initialize all flags to false. Then, we iterate over all cut cell tetrahedra with water and set flags to true. Subsequently, we loop over the tetrahedra in the same order as in the pushout phase, clamping the normal component of the fluid velocity to be the precomputed

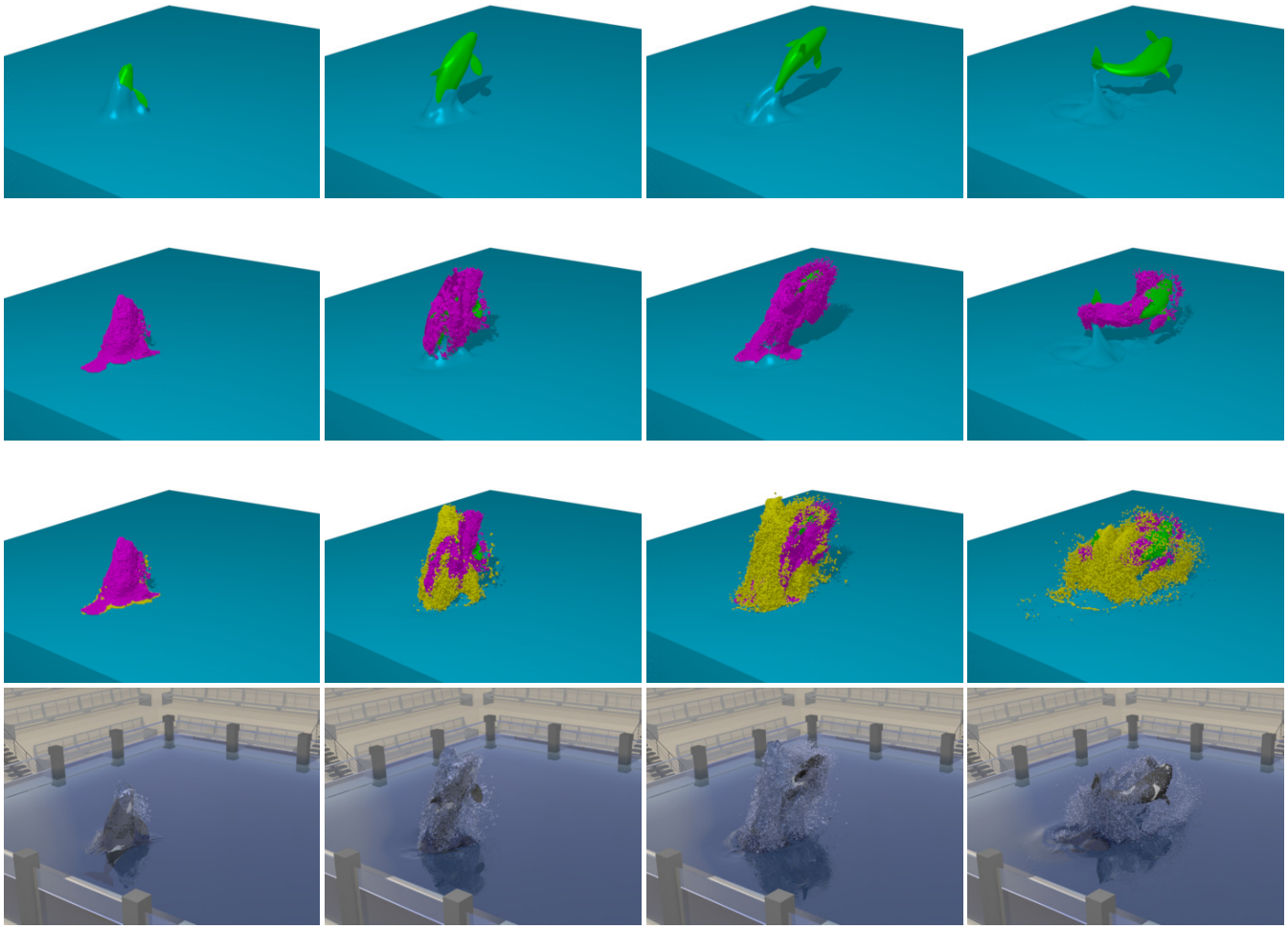

Figure 8: A whale breaches out of the water. (Top Row) Visualization of the Eulerian water. (Second Row) VOF tetrahedra water is rendered in pink. (Third Row) Particles are shown in yellow. (Fourth Row) Final rendering.

object normal velocity when the flag is set to true and the normal component of velocity is smaller than the precomputed object normal velocity. Higher rank tetrahedra have their Boolean flag set to be true when they have a lower rank face neighbor which is fully saturated that required clamping. This limits the enforcement of boundary conditions to those tetrahedra exposed to the creature surface by a column of water. Although this removes circulation on the KDSM, the circulation is restored from the background Eulerian grid during coupling as discussed in Section 4.

This approach is not a standard projection scheme, but still enforces a divergence free condition thereby enforcing incompressibility. One other notable approach that is not an advection-projection scheme is [Zehnder et al. 2018]. To elaborate, fluid simulated using the Navier-Stokes equations is assumed to obey the conservation of mass equation (i.e. no fluid is created or destroyed):

$$\partial\rho/\partial t + \nabla\cdot(\rho\mathbf{u}) = 0 \tag{1}$$

$$\partial\rho/\partial t + \rho\nabla\cdot\mathbf{u} + \mathbf{u}\cdot\nabla\rho = 0 \tag{2}$$

Here, $\mathbf{u}$ is the fluid velocity, t is time, and ρ is the fluid density. By the product rule, (1) equivalent to (2). Using this equation, it can be seen that setting $\nabla\cdot\mathbf{u} = 0$ is equivalent to setting $\partial\rho/\partial t + \mathbf{u}\cdot\nabla\rho = 0$. Either condition implies the other–they are equivalent from the conservation of mass. Setting $\rho = m/V$ and using the product rule:

$$(1/V)(\partial m/\partial t) - (m/V^2)(\partial V/\partial t) + \mathbf{u}\cdot((1/V)\nabla m - (m/V^2)\nabla V) = 0$$

$$(1/V)(\partial m/\partial t + \mathbf{u}\cdot\nabla m) - (m/V^2)(\partial V/\partial t + \mathbf{u}\cdot\nabla V) = 0$$

The first term $(1/V)(\partial m/\partial t + \mathbf{u}\cdot\nabla m)$ must equal zero since mass is conserved along streamlines. This means $\partial V/\partial t + \mathbf{u}\cdot\nabla V$ must also equal zero, and taking this with the divergence free condition $\nabla\cdot\mathbf{u} = 0$ yields:

$$\partial V/\partial t + \nabla\cdot(\mathbf{u}V) = 0$$

Thus, conserving volume is equivalent to enforcing a divergence-free velocity field.

With regard to maintaining the incompressibility of a simulated fluid, we note that a standard projection scheme such as the classic method introduced by [Chorin 1968] is just one proposed algorithm for this task. In fact, Chorin advocated at least two distinct schemes [Chorin 1967], though the advection-projection procedure became most popular. Chorin-style projection claims that one can advect a fluid state ad-hoc off of the manifold of all incompressible fluid fields and then correct the ensuing error by projecting back onto that manifold. However, these projection-style schemes are known to be brittle (e.g. they require small time steps and do not even always converge under temporal refinement), and they are well-known to be unable to capture important physics of fluids (such as viscosity, due to its parabolic nature). Thus, solving a pressure Poisson equation to enforce a divergence-free velocity field is not the ultimate, and certainly not the only, numerical scheme to simulate incompressible flow. Our technique, which indeed differs from a pressure projection, is able to successfully and robustly conserve volume, which as shown above implies the standard incompressibility condition. Hence our volume conservation method maintains a faithful relationship with the underlying fluid principles and equations. Moreover, at a high level, we remark that even the Navier-Stokes equations quickly fail to be a physically accurate model when considering real-world flow problems i.e. turbulence; however, such approximations are heavily relied upon in computer graphics due to their computational amenability and their ability to produce visually plausible results.

3.4 Adhesion

We allow an artist to paint adhesion coefficients α and force directions $\vec{d}$ on the triangulated surface mesh of the creature, and then we rasterize this information to the KDSM setting adhesion quantities in rank 0 tetrahedra. We propagate adhesion quantities to face neighbors (or non-local neighbors) of strictly higher rank tetrahedra by averaging the adhesion quantities from lower rank neighbors for which adhesion had already been specified. We apply an adhesion force $\alpha\vec{d}$ when a tetrahedron is within a prescribed distance ϕ_a from the creature's surface with linear falloff, i.e. $\alpha(\phi_a - \phi)/\phi_a\vec{d}$ where ϕ is the distance from the creature's surface (similar to [Zhu et al. 2014]).

Figure 7 illustrates some of the many interesting visual effects obtainable by varying adhesion parameters. Notably, it is the robust volume conservation of our VOF method and the adaptivity of the KDSM that allows for such interesting effects. We attempted similar simulations using a standard Eulerian method and mostly achieved disturbing volume loss. The top row shows how increasing adhesion (from left to right) makes the water to stick to the ball and flow around to the bottom surface before separating. The bottom left image was created with vectors $\vec{d}$ pointing outwards from the ball at various locations to produce thickened streams. In contrast, the bottom right figure shows how the vectors $\vec{d}$ can be used to direct water away from parts of the ball's surface, drying it out.

4 PARTITIONED COUPLING

We utilize three different representations for water: besides the VOF representation on the KDSM, we also use both free particles and velocities on the background Eulerian Cartesian grid as is typical for the standard PLS method (see e.g. [Enright et al. 2002b]). See Figure 8. Our partitioned coupling method consists of four major steps. In the first step, each of our three representations (VOF tetrahedra, particles, and Eulerian Cartesian grid) are advected forward in time. The method of Section 3.2 is used for the VOF tetrahedra, while the standard PLS method is used to advect the Eulerian grid velocities and to move the particles. In a second step, momentum is transferred between the three representations in order to maximize the visual efficacy of the results. Then, external forces are independently added to each representation, before projecting the velocity into a divergence free state acceptable to all three representations. The steps are summarized below:

Algorithm 1 Pseudocode

```
function Partitioned Coupling
    Advection()
    Momentum Transfer()
    Add External Forces()
    Volume Conservation()
function Advection (each stage is independent)
    VOF advection (Section 3.2)
    Particle advection
    Eulerian advection
function Momentum Transfer
    Transfer momentum from VOF to Eulerian (optional)
    Transfer momentum from Eulerian to VOF
    VOF particle reincorporation
function Add External Forces (each stage is independent)
    VOF external forces
    Particle external forces
    Eulerian external forces
function Volume Conservation
    VOF volume conservation (Section 3.3)
    Eulerian Projection()
    Transfer momentum from Eulerian to VOF
function Eulerian Projection
    Eulerian particle reincorporation
    Projection
```

Both the particles and the VOF tetrahedra carry accurate Lagrangian momentum information, as compared to the typically more smeared out velocities obtained using semi-Lagrangian advection (see e.g. [Stam 1999]) on the Eulerian grid. Note that we allow VOF tetrahedra to overlap with the Eulerian water. Thus, we allow for the option to first transfer some momentum from the VOF tetrahedra to the Eulerian grid. Typically, this increases the turbulence near the boundaries of a moving creature. This is accomplished by iterating over tetrahedra with water and averaging their momentum with the values on the background Eulerian grid using an artist controllable multiplier. The result is used to overwrite the value on the Eulerian grid.

Next, the velocities of the Eulerian grid are used to overwrite the momentum value of any tetrahedron which has all four of its nodes inside the water surface representation of the Eulerian grid. The tetrahedron's volume is also set to be fully saturated with

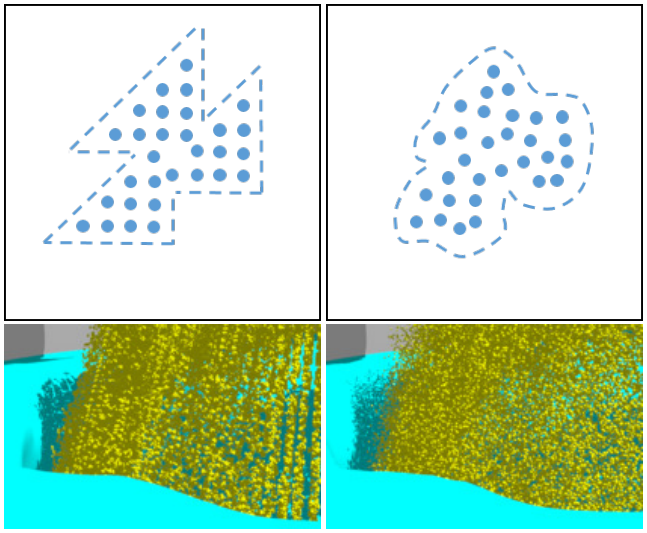

Figure 9: Top row compares the particle positions obtained with uniform versus jittered sampling emphasizing how well our eyes capture structured information (even when we do not want them to). Bottom row compares the two approaches for an actual simulation.

water. This overwrite operation does not use averaging since the background Eulerian grid has a full-fledged pressure solver that tracks velocities more accurately preserving various effects such as the circulation (discussed in Section 3.3). Importantly, cut cell tetrahedra are not overwritten allowing them to more accurately track volume and momentum close to the boundary of the creature. Higher ranks of tetrahedra could also be allowed to preserve their information if desired, although we did not experiment with this option.

Finally, any particle that lies within a VOF tetrahedron that contains water is deleted, and its volume and momentum are added to that tetrahedron. This allows particles to freely move through the region of space occupied by the KDSM only being reincorporated into the VOF representation when they impact water regions as defined by the VOF tetrahedra.

The volume conservation step starts out with the method proposed in Section 3.3, i.e. smear, pushout, and velocity correction, in order to create an adequate velocity for the VOF tetrahedra on the KDSM. Then, particles are reincorporated into the background Eulerian grid as Eulerian water when appropriate applying a local momentum force, altering the level set, and adding an expansion force similar to [Losasso et al. 2008]. Following the standard PLS projection scheme, the results of the pressure solve are subsequently added to the Cartesian grid velocity in order to obtain a divergence free field. As a final step, the divergence free Eulerian grid velocities are used to overwrite the momentum in any tetrahedron that has all four of its nodes interior to the Eulerian grid water representation.

4.1 Particle Generation

The automatic generation of particles in visually compelling locations by hybrid particle level set methods has been one of their strengths even predating the PLS method, see [Foster and Fedkiw 2001]. Thus, we devise a method similar in spirit for our ALE based VOF method on the KDSM. As discussed in Section 3.2, advection might dictate that water moves off of the KDSM. This occurs when part of a forward advected tetrahedron lies outside of the KDSM. This is detected by checking whether or not the point samples of the tetrahedron lie outside of the KDSM. Each point sample had already been assigned a certain amount of water to transport, so we use that water's volume and momentum to create a particle with appropriate radius and velocity. Note that we use a standard volume equation for a sphere, $V = 4/3\pi r^3$, to get radius. Since a straightforward approach leads to noticeable aliasing, we jitter the particle locations by a small amount–we used a fraction ranging from .1 to 1 multiplied by maximum edge length of a tetrahedron for our jitter magnitudes (see Figure 9). As discussed in Section 3.3, tetrahedra on the exterior boundary of the KDSM may contain excess water that needs to be transported off of the KDSM. In this scenario, there is no natural advection direction. Thus, we move the particles across the exterior face of the tetrahedron while also applying appropriate jittering. Note that when water leaves the KDSM, it always goes through particle phase before rejoining the level set.

4.2 Rendering

As is the case for many of the state-of-the-art Lagrangian methods, rendering smooth surfaces is quite difficult. Many authors have proposed various strategies, such as applying smoothing kernel on implicit surfaces as in [Blinn 1982], [Zhu and Bridson 2005], [Adams et al. 2007], [Solenthaler et al. 2007], [Museth et al. 2007], [Williams 2008], [Sin et al. 2009], [Onderik et al. 2011], [Bhatacharya et al. 2011], [Müller and Chentanez 2011], explicitly tracking fluid surfaces as in [Brochu and Bridson 2009], [Brochu et al. 2010], and polygonalizing fluid surfaces as in [Akinci et al. 2012a], [Akinci et al. 2012b], [Akinci et al. 2013b], [Wu et al. 2016]. Since most of

Figure 10: Our anisotropic porosity model is implemented to influence the VOF method on the KDSM accounting for both limited volume fraction and drag/adhesion yielding visually compelling results.

Figure 11: (Left) Whale breaching with the PLS method on high resolution grid. The whale pulls very little water along with it. (Middle) FLIP method, which also produces similar amount of sprays. (Right) Our method pulls more water into the air with the whale, producing interesting effects.

this research has been focused on rendering particles as opposed to triangles, we do not use a marching tetrahedra approach as in [Doi and Koide 1991], [Müller and Wehle 1997] (the resulting surface via marching suffered from excessive bumps and sharpness, which was not suitable for graphics applications). Instead, we convert the VOF tetrahedra water into particles and render them along with the other particles. This is done by creating point samples per tetrahedron based on the quadrature formula (without jittering). Then, we attract the particles that are near the level set representing the water surface towards that level set in order to flatten out bumps created by the cut cell tetrahedra near the boundary of the level set. Finally, we use [Yu and Turk 2010] (see also a slight variant of this method [Yu and Turk 2013]) to create an implicit surface from the particle data, and merge this implicit surface with the Eulerian grid level set to obtain the final water surface that we render. The resulting level set still has some bumps due to the limitations of the anisotropic kernel, so we additionally smooth the normals during rendering. We stress the fact that we only render the final merged level set on a high resolution grid, and do not directly render the particles.

5 HAIR-WATER INTERACTION

We embed hair particles in the KDSM and treat the hair using the KDSM as in [Lee et al. 2018] (we also refer the interested reader to [Rungjiratananon et al. 2012], [Lin 2014], [Fei et al. 2017], [Fei et al. 2018] for more discussion on hair-water interaction). Our hair-water approach is volumetric in nature, rasterizing multitude of hair representation into KDSM as opposed to [Fei et al. 2017], where they focus on a reduced model for individual hair strands. As a result, our method handles hair-water interaction with 540k hairs as opposed to 5k and 30k as given in [Rungjiratananon et al. 2012] and [Fei et al. 2017], respectively. For each tetrahedron containing hair we precompute the volume fraction occupied by the hair and reduce the water that this tetrahedron may contain at saturation by this amount. This gives a very accurate representation of the porosity. We also compute the average direction of the hair strands in each tetrahedron, so that we may treat the porosity anisotropically. Essentially, more drag is applied orthogonal to the average direction of the hair strands. See Figures 10, 12, 13.

6 RESULTS AND DISCUSSION

3.06GHz CPU (12 cores) and 96GB RAM was used. KDSM generation for the whale and bear examples took 10 min / frame, and each frame is temporally independent so we ran them in parallel. The ball examples (Figures 4 and 7) took 1 min and 2 sec / frame to run with a 100x100x100 Eulerian grid, 5.6 million KDSM elements, and .9 million KDSM particles. The bear pour example (Figure 10) took 7 min and 3 sec / frame with a 200x200x400 grid, 8.2 million KDSM elements, and 1.4 million KDSM particles. The bear walk example (Figure 12) took 4.5 min / frame with a 100x200x200 grid, 8.2 million KDSM elements, and 1.4 million KDSM particles. The whale example took 20 min / frame with a 200x300x200 grid, 8.5 million KDSM elements, and 1.4 million KDSM particles whereas the PLS method-only example took 29 min / frame using a 350x525x350 grid. If we run the PLS method for the whale example at an even higher resolution, we can eventually achieve higher quality results by carrying more water volume with the whale, but this would require significant time investment. Figure 11 qualitatively compares PLS, FLIP, and our method, and illustrates the benefit of our method regarding visual quality. We used Neumann boundary condition for solid boundaries for PLS method. For all our examples, we generated 5 to 35 samples per tetrahedron based on the quadrature formula.

There are fundamental limitations of ALE based methods especially regarding meshing problems, so we implement a couple of simple remedies below to fix the occasional degeneracy in order to run all of our examples robustly. Note that the animated creature can move in a way that inverts its elements or prevents volume preservation of the surrounding space unless the artist is very careful–most of issues appear near joints and are worsened by linear blend skinning. We only need to iterate a couple of times in the preprocessing stage to resolve most of these issues, and we disable any remaining degenerate elements (inverted or collapsed) so that they cannot participate in the VOF solver. Thus, whenever a sample point falls in degenerate elements, particles will be formed instead of the located element receiving water. While one could better prevent element inversion by using FEM, in practice our simple mass spring model was sufficient. Rarely when we cannot properly advect or enforce incompressibility during the simulation because a VOF tetrahedron is completely surrounded by the solid due to an extreme creature deformation, we simply keep the water in that tetrahedron in order to exactly conserve volume until the issue is resolved as the surrounding solid opens up.

Our method fully conserves volume in the KDSM, although floating point drift causes small volume error throughout the simulation. We measured the volume error per frame for ball example in top right of Figure 4 for 400 frames. The average volume error per frame was 0.00089%, and the maximum error was 0.00189%.

Figure 12: A bear walks out of a pool onto land, still carrying and dripping a large amount of water from its fur. Our anisotropic porosity model accounts for the correct volume fraction of water in the fur and uses adhesion to pull that water out of the pool with the bear, subsequently slowing dripping the water out of the fur. This example emphasizes the efficacy of the adaptivity of the KDSM as well as the ability to preserve volume and avoid disappearing water with our VOF method.

As future work, one could implement a different solver such as [Jiang et al. 2015], [Ihmsen et al. 2012], or [Chentanez et al. 2014] to simulate fluid in the background grid or in the KDSM, and the adaptivity of our method will improve the accuracy of chosen method. In order to generalize our method to a pure FLIP/PIC/APIC variant, the data transfer function would need to be rewritten in order to refer to the KDSM when the particle is inside of the KDSM, and the interpolation scheme would need to be modified to use barycentric weights for tetrahedron. Then, [Ando et al. 2013] could be used to handle non-advection steps. Thus, FLIP/PIC/APIC variant can benefit from the dense KDSM mesh instead of using the coarse background Eulerian grid when the method transfers data from particles to the KDSM. We emphasize the technical insight that the coarse background grid captures a low frequency fluid surface whereas around the creature with high frequency boundaries we use the dense KDSM mesh to capture high frequency fluid motion. We chose the PLS method because it generates a very smooth surface, which is suitable for background motion, whereas our ALE based VOF method is more geared towards capturing detailed fluid motion by preserving volume to compensate for the PLS method's limitation. Additionally, one can subdivide on-the-fly if adaptive remeshing based on the fluid motion is desired.

7 CONCLUSIONS

We proposed a new fluid simulation framework for character-water and hair-water interaction using our novel volume conserving VOF method based on an adaptive tetrahedral mesh from the KDSM, which moves with the creature. We prebake the adaptivity of the ALE mesh, separating the nontrivial remeshing issue from the simulation phase and improving the robustness of our ALE based VOF method; we further preprocess auxiliary data wherever possible in order to make the simulation efficient and streamlined. A coarse background Eulerian grid and our fine ALE mesh are two way coupled using a partitioned approach which is fast, efficient, and straightforward to implement. We use our volume conserving VOF method only on the KDSM near the creature while using a standard PLS method on the background Eulerian grid. We robustly implement interesting effects such as adhesion and anisotropic porosity. We demonstrated how the coarse background Eulerian grid captures the bulk behavior of the water, while our VOF method captures detailed water effects near the creature and the particles capture the spray—all of which make important contributions to the final result.

Figure 13: A close-up of the bear example, showing water sticking to fur and splashes generated from our VOF method.

ACKNOWLEDGMENTS

The authors would like to thank Ed Quigley for writing and Winnie Lin for narrating the video. Research supported in part by ONR N00014-13-1-0346, ONR N00014-17-1-2174, ARL AHPCRC W911NF-07-0027, and NSF CNS1409847. M.L. was supported by Samsung Scholarship and MPC-VCC Summer Scholarship. Computing resources were provided in part by ONR N00014-05-1-0479.

REFERENCES

M. Aanjaneya, M. Gao, H. Liu, C. Batty, and E. Sifakis. 2017. Power diagrams and sparse paged grids for high resolution adaptive liquids. *ACM TOG* 36 (2017).

B. Adams, M. Pauly, R. Keiser, and L. J. Guibas. 2007. Adaptively sampled particle fluids. *ACM Trans. Graph. (SIGGRAPH Proc.)* 26, 3 (2007).

G. Akinci, N. Akinci, M. Ihmsen, and M. Teschner. 2012a. An Efficient Surface Reconstruction Pipeline for Particle-Based Fluids. In *Proceedings of Virtual Reality Interactions and Physical Simulations*.

G. Akinci, N. Akinci, E. Oswald, and M. Teschner. 2013b. Adaptive Surface Reconstruction for SPH Using 3-level Uniform Grids. *WSCG 2013* (2013).

G. Akinci, M. Ihmsen, N. Akinci, and M. Teschner. 2012b. Parallel Surface Reconstruction for Particle-Based Fluids. *Computer Graphics Forum* (2012).

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013a. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Trans. Graph.* 32, 6, Article 182 (Nov. 2013), 8 pages. https://doi.org/10.1145/2508363.2508395

Dicko Ali-Hamadi, Tiantian Liu, Benjamin Gilles, Ladislav Kavan, François Faure, Olivier Palombi, and Marie-Paule Cani. 2013. Anatomy Transfer. In *ACM SIGGRAPH Asia 2013 papers (SIGGRAPH ASIA '13)*. 188:1–188:8.

R. Ando, N. Thürey, and C. Wojtan. 2013. Highly Adaptive Liquid Simulations on Tetrahedral Meshes. *ACM Trans. Graph. (Proc. SIGGRAPH 2013)* (July 2013).

C. Batty, S. Xenos, and B. Houston. 2010. Tetrahedral embedded boundary methods for accurate and flexible adaptive fluids. In *Comput. Graph. Forum (Eurographics Proc.)*, Vol. 29. 695–704.

H. Bhatacharya, Y. Gao, and A. Bargteil. 2011. A level-set method for skinning animated particle data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '11)*.

J. Blinn. 1982. A generalization of algebraic surface drawing. *ACM Trans. Graph. (Proc. SIGGRAPH 1982)* (1982).

J. U. Brackbill, D. B. Kothe, and C. Zemach. 1992. A Continuum method for modelling surface tension. *J. Comput. Phys.* 100 (1992), 335–353.

R. Bridson, S. Marino, and R. Fedkiw. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of the 2003 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 28–36.

T. Brochu, C. Batty, and R. Bridson. 2010. Matching fluid simulation elements to surface geometry and topology. *ACM Trans. Graph. (SIGGRAPH Proc.)* (2010), 47:1–47:9.

T. Brochu and R. Bridson. 2009. Robust topological operations for dynamic explicit surfaces. *SIAM Journal on Scientific Computing* 31, 4 (2009), 2472–2493.

N. Chentanez, B. E. Feldman, F. Labelle, J. F. O'Brien, and J. R. Shewchuk. 2007. Liquid Simulation on Lattice-Based Tetrahedral Meshes. In *ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 219–228.

Nuttapong Chentanez, Matthias Müller, and Tae-Yong Kim. 2014. Coupling 3D Eulerian, Heightfield and Particle Methods for Interactive Simulation of Large Scale Liquid Phenomena. (2014), 1–10. http://dl.acm.org/citation.cfm?id=2849517.2849519

A.J. Chorin. 1967. A Numerical Method for Solving Incompressible Viscous Flow Problems. *J. Comput. Phys.* 2, 1 (1967), 12–26.

A.J. Chorin. 1968. Numerical solution of the Navier-Stokes Equations. *Math. Comput.* 22, 1 (1968), 745–762.

M. Cong, M. Bao, J. L. E, K. S. Bhat, and R. Fedkiw. 2015. Fully Automatic Generation of Anatomical Face Simulation Models. In *Proc. of the 14th ACM SIGGRAPH / Eurographics Symp. on Comput. Anim.* 175–183.

A. Doi and A. Koide. 1991. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE Trans. Inf. & Syst.* E74-D (1991), 214–224.

R.E. English, L. Qiu, Y. Yu, and R. Fedkiw. 2013a. An adaptive discretization of incompressible flow using a multitude of moving Cartesian grids. *J. Comput. Phys.* 254, 0 (2013), 107 – 154. https://doi.org/10.1016/j.jcp.2013.07.032

R.E. English, L. Qiu, Y. Yu, and R. Fedkiw. 2013b. Chimera Grids for Water Simulation. In *SCA '13: Proceedings of the 2013 ACM SIGGRAPH/Eurographics symposium on Computer animation*.

D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. 2002a. A Hybrid Particle Level Set Method for Improved Interface Capturing. *J. Comput. Phys.* 183 (2002), 83–116.

D. Enright, S. Marschner, and R. Fedkiw. 2002b. Animation and Rendering of Complex Water Surfaces. *ACM Trans. Graph. (SIGGRAPH Proc.)* 21, 3 (2002), 736–744.

Y. Fei, H. T. Maia, C. Batty, C. Zheng, and E. Grinspun. 2017. A Multi-Scale Model for Simulating Liquid-Hair Interactions. *ACM Trans. Graph.* 36, 4 (2017).

Yun (Raymond) Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2018. A Multi-scale Model for Simulating Liquid-fabric Interactions. *ACM Trans. Graph.* 37, 4, Article 51 (Aug. 2018), 16 pages. https://doi.org/10.1145/3197517.3201392

B. Feldman, J. O'Brien, and B. Klingner. 2005a. Animating Gases with Hybrid Meshes. *ACM Trans. Graph. (SIGGRAPH Proc.)* 24, 3 (2005), 904–909.

B. Feldman, J. O'Brien, B. Klingner, and T. Goktekin. 2005b. Fluids in Deforming Meshes. In *Proc. of the ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 255–259.

Florian Ferstl, Ryoichi Ando, Chris Wojtan, Rüdiger Westermann, and Nils Thuerey. 2016. Narrow Band FLIP for Liquid Simulations. *Comput. Graph. Forum* 35, 2 (May 2016), 225–232. https://doi.org/10.1111/cgf.12825

N. Foster and R. Fedkiw. 2001. Practical Animation of Liquids. In *Proc. of ACM SIGGRAPH 2001*. 23–30.

N. Foster and D. Metaxas. 1996. Realistic animation of liquids. *Graph. Models and Image Processing* 58 (1996), 471–483.

E. Guendelman, R. Bridson, and R. Fedkiw. 2003. Nonconvex rigid bodies with stacking. *ACM TOG* 22, 3 (2003), 871–878.

C. Hirt and B. Nichols. 1981. Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries. *J. Comput. Phys.* 39 (1981), 201–225.

Markus Ihmsen, Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2012. Unified spray, foam and air bubbles for particle-based fluids. *Vis. Comput.* 28, 6-8 (2012), 669–677.

C. Jiang, C. Schroeder, and J. Teran. 2015. An Affine Particle-In-Cell Method. *ACM Trans. Graph. (Proc. SIGGRAPH 2015)* (2015).

B. M. Klingner, B. E. Feldman, N. Chentanez, and J. F. O'Brien. 2006. Fluid animation with dynamic meshes. *ACM Trans. Graph. (SIGGRAPH Proc.)* 25, 3 (2006), 820–825.

M. Lee, D. Hyde, M. Bao, and R. Fedkiw. 2018. A Skinned Tetrahedral Mesh for Hair Animation and Hair-Water Interaction. *IEEE TVCG* (2018).

M. Lentine, M. Aanjaneya, and R. Fedkiw. 2011. Mass and momentum conservation for fluid simulation. In *SCA '11: Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 91–100.

M. Lentine, M. Cong, S. Patkar, and R. Fedkiw. 2012. Simulating Free Surface Flow with Very Large Time Steps. In *ACM SIGGRAPH/Eurographics Symp. on Comput. Anim. 2012*. 107–116.

W. Lin. 2014. Coupling hair with smoothed particle hydrodynamics fluids. *Proc. of VRIPHYS* (2014).

F. Losasso, R. Fedkiw, and S. Osher. 2006. Spatially Adaptive Techniques for Level Set Methods and Incompressible Flow. *Computers and Fluids* 35 (2006), 995–1010.

F. Losasso, F. Gibou, and R. Fedkiw. 2004. Simulating Water and Smoke with an Octree Data Structure. *ACM Trans. Graph. (SIGGRAPH Proc.)* 23 (2004), 457–462.

F. Losasso, J. O. Talton, K. Nipun, and R. Fedkiw. 2008. Two-way Coupled SPH and Particle Level Set Fluid Simulation. *IEEE TVCG* 14 (July 2008), 797–804.

V. Mihalef, D. Metaxas, and M. Sussman. 2004. Animation and control of breaking waves. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 315–324.

V. Mihalef, B. Unlusu, D. Metaxas, M. Sussman, and M. Hussaini. 2006. Physics Based Boiling Simulation. In *SCA '06: Proc. of the 2006 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 317–324.

N. Molino, R. Bridson, J. Teran, and R. Fedkiw. 2003. A Crystalline, Red Green Strategy for Meshing Highly Deformable Objects with Tetrahedra. In *12th Int. Mesh. Roundtable*. 103–114.

H. Müller and M. Wehle. 1997. Visualization of Implicit Surfaces Using Adaptive Tetrahedrizations. *Scientific Visualization*.

M. Müller and N. Chentanez. 2011. Solid Simulation with Oriented Particles. *ACM TOG* 30, 4, Article 92 (2011), 10 pages.

Matthias Müller, Tae-Yong Kim, and Nuttapong Chentanez. 2012. Fast Simulation of Inextensible Hair and Fur. *VRIPHYS* 12 (2012), 39–44.

K. Museth, M. Clive, and N. B. Zafar. 2007. Blobtacular: Surfacing particle systems in "Pirates Of The Caribbean 3". *SIGGRAPH Sketches* (2007).

J. Onderik, M. Chladek, and R. Durikovic. 2011. SPH with small scale details and improved surface reconstruction. *SCCG* (2011).

W. J. Rider and D. B. Kothe. 1998. Reconstructing Volume Tracking. *J. Comput. Phys.* 141 (1998), 112–152.

W. Rungjiratananon, Y. Kanamori, and T. Nishita. 2012. Wetting effects in hair simulation. In *Comput. Graph. Forum*, Vol. 31. Wiley Online Library, 1993–2002.

R Sánchez-Banderas, H Barreiro, I García-Fernández, and M Pérez. 2015. Real-time inextensible hair with volume and shape. In *Congreso Español de Informática Gráfica, CEIG'15*.

A. Selle, M. Lentine, and R. Fedkiw. 2008. A Mass Spring Model for Hair Simulation. *ACM Trans. Graph. (SIGGRAPH Proc.)* 27, 3 (Aug. 2008), 64.1–64.11.

E. Sifakis, T. Shinar, G. Irving, and R. Fedkiw. 2007. Hybrid Simulation of Deformable Solids. In *Proc. of ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 81–90.

F. Sin, A. W. Bargteil, and J. K. Hodgins. 2009. A point-based method for animating incompressible flow. In *Proc. of the 2009 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 247–255.

B. Solenthaler, J. Schlüfli, and R. Pajarola. 2007. A Unified Particle Model for Fluid-Solid Interactions. *Computer Animation and Virtual Worlds* (2007).

J. Stam. 1999. Stable Fluids. In *Proc. of SIGGRAPH 99*. 121–128.

M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome. 1999. An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys.*

148 (1999), 81–124.
M. Sussman and E. Puckett. 2000. A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys.* 162 (2000), 301–337.
B. W. Williams. 2008. *Fluid Surface Reconstruction from Particles.* Ph.D. Dissertation. The University of British Columbia.
W. Wu, H. Li, T. Su, H. Liu, and Z. Lv. 2016. GPU-accelerated SPH fluids surface reconstruction using two-level spatial uniform grids. 33 (07 2016).
J. Yu and G. Turk. 2010. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proc. of the 2010 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.* 217–225.
J. Yu and G. Turk. 2013. Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Trans. Graph.* 32, 1, Article 5 (Feb. 2013), 12 pages. https://doi.org/10.1145/2421636.2421641
O. Zarifi and C. Batty. 2017. A positive-definite cut-cell method for strong two-way coupling between fluids and deformable bodies. In *Proceedings of the 2016 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '17).*
Jonas Zehnder, Rahul Narain, and Bernhard Thomaszewski. 2018. An Advection-reflection Solver for Detail-preserving Fluid Simulation. *ACM Trans. Graph.* 37, 4, Article 85 (July 2018), 8 pages. https://doi.org/10.1145/3197517.3201324
L. Zhang, T. Cui, and H. Liu. 2009. A Set of Symmetric Quadrature Rules on Triangle and Tetrahedra. *Journal of Computational Mathematics.*
X. Zhang, M. Li, and R. Bridson. 2016. Resolving Fluid Boundary Layers with Particle Strength Exchange and Weak Adaptivity. *ACM Trans. Graph.* 35, 4 (2016).
B. Zhu, E. Quigley, M. Cong, J. Solomon, and R. Fedkiw. 2014. Codimensional Surface Tension Flow on Simplicial Complexes. *ACM Trans. Graph. (SIGGRAPH Proc.)* 33, 4 (2014), 111:1–111:11.
Y. Zhu and R. Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph. (SIGGRAPH Proc.)* 24, 3 (2005), 965–972.