ACM SIGGRAPH / Eurographics Symposium on Computer Animation 2024
M. Skouras and H. Wang
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 8

Reconstruction of implicit surfaces from fluid particles using convolutional neural networks

C. Zhao[1], T. Shinar[1], and C. Schroeder[†1]

[1]University of California Riverside

Abstract

In this paper, we present a novel network-based approach for reconstructing signed distance functions from fluid particles. The method uses a weighting kernel to transfer particles to a regular grid, which forms the input to a convolutional neural network. We propose a regression-based regularization to reduce surface noise without penalizing high-curvature features. The reconstruction exhibits improved spatial surface smoothness and temporal coherence compared with existing state of the art surface reconstruction methods. The method is insensitive to particle sampling density and robustly handles thin features, isolated particles, and sharp edges.

CCS Concepts

• ***Computing methodologies*** → ***Physical simulation; Point-based models;***

1. Introduction

Fluid simulation techniques that utilize particles as the primary material representation are ubiquitous in computer graphics. These methods include the Lagrangian smoothed particle hydrodynamics (SPH) method [MCG03; KBST22], as well as the hybrid Lagrangian-Eulerian particle-in-cell (PIC) method [Har64], fluid-implicit-particle (FLIP) method [BR86; ZB05], and the material point method (MPM) [SZS95; JSS*15].

Though meshless particle-based fluid simulation methods have become increasingly popular due their strengths, including flexibility in handling topological changes and resolution of thin features, they do not impose a well-defined surface, which is required for tasks such as rendering and computation of normals and curvature. Reconstruction of the fluid surface from the particle data remains a challenge [YT13; SLW*23]. Specific challenges include generating smooth surfaces without spurious bumps while retaining fine features, accurately reconstructing geometric properties such as normals and curvature, used in the computation of surface tension, and conserving volume. These issues are exacerbated by the irregular distribution of particles that results from fluid deformation.

While previous approaches to fluid surface reconstruction in the physics-based animation community have relied on classical geometric processing, we take inspiration from the recent success of data-driven approaches to surface reconstruction from noisy point cloud data. We introduce a novel network-based approach for reconstructing signed distance functions (SDFs) from fluid particles. First, a weighting kernel is used to transfer particle spatial information onto a regular grid, which serves as input for a convolutional neural network. To enhance surface quality, we propose a regression-based regularization method that reduces surface noise without penalizing high-curvature features. This polynomial surface regularization is efficient, relying on a precomputed matrix and avoiding the need for derivative computation as in other regularization approaches.

Our reconstruction demonstrates excellent spatial surface smoothness and temporal coherence compared to state-of-the-art methods. Notably, our approach exhibits insensitivity to particle sampling density and robustly handles thin features, isolated particles, and sharp edges. We validate our network on various test cases encompassing both fluids and solids, showcasing its ability to maintain temporal coherence, produce smooth surfaces, and represent visually plausible fluid features.

2. Related Work

There have been several approaches to surface reconstruction from particle data. Here, we briefly review previous methods used for particle-based fluid animation. We also review recent data-driven approaches to the related problem of surface reconstruction from noisy surface point cloud data.

Traditional scalar field reconstruction. Many approaches construct an implicit scalar field representation of a surface from particle data, which can be rendered with ray casting or from which a mesh can be constructed using Marching Cubes [LC98]. Motivated by the problem of rendering molecular models, [Bli82] developed a

† Corresponding author: craigs@cs.ucr.edu

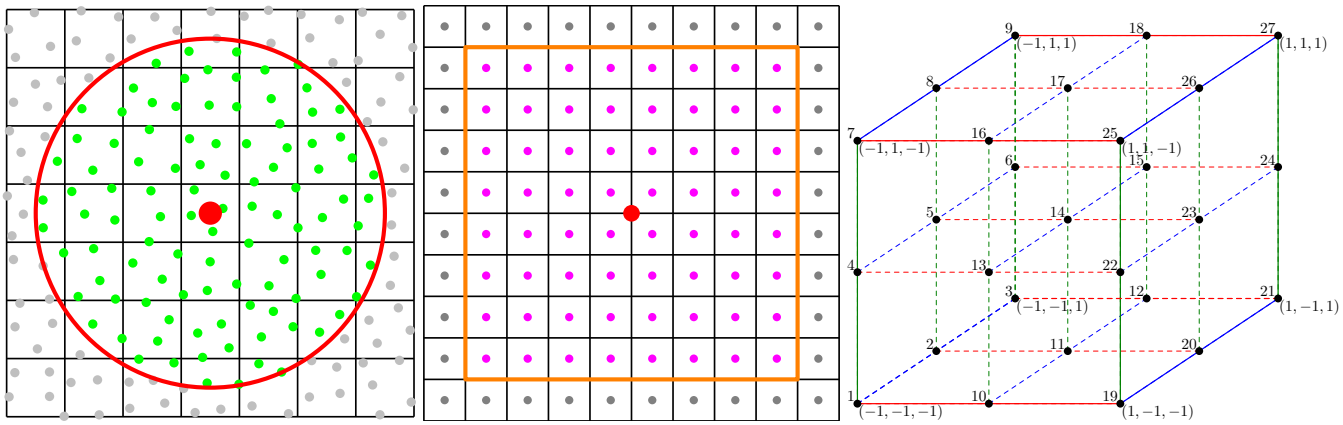


Figure 1: ***Left:*** *Grid values m_c (●) are computed from fluid particles (●) within a $3\Delta x$ radius (red circle).* ***Middle:*** *Each node-centered vector (●) is computed from the cell-centered features m_c (●) within a 8^3 box (orange).* ***Right:*** *Numbering of ϕ_i within Φ. The cluster of 27 ϕ values is used for polynomial regularization.*

method for rendering blobby implicit surfaces defined by a collection of particles. Applied to fluid simulations, this approach yields an undesirable bumpy appearance at flat surfaces. [DC98] applied a tunable amount of surface tension to reduce the bumpiness of the surface, though this also undesirably smooths finer flow features. [MCG03] used the continuous SPH color function to determine surface particles and normals for use in a particle splatting approach [ZPVBG01] and to directly extract an isosurface using marching cubes [LC98; NY06], with both approaches exhibiting bumpiness at flat surfaces. [CGB16] also used the SPH color function but accounted for topological neighborhood information to reduce unwanted blending between unmerged surfaces. [PTB*03] advected a signed distance function with forces computed from the particle data to maintain consistency with the primary SPH particle representation, leveraging the smoothing inherent to the level set advection. [SS07] reconstructed an implicit surface from sparse particle data by defining special reference particles and using backtracing to improve temporal coherence.

[ZB05] partially mitigated the surface bumpiness by using local weighted averages of the particle data in reconstructing a signed distance function and applying a small amount of unidirectional surface smoothing to avoid destruction of fine features. [SSP07] improved on this approach by detecting and correcting errors in concave regions. [APKG07] introduced a novel SPH method with adaptively sampled particles, and extended the surface reconstruction algorithm of [ZB05] by tracking particle-to-surface distance to alleviate temporal incoherence under particle resampling.

A widely used approach was developed in [YT13], which replaces the spherical kernels at the heart of previous implicit surface reconstruction approaches with elliptical kernels whose anisotropy depends on the local particle distribution, resulting in improved surface smoothness. Recently, researchers developing the Pahi water simulation pipeline [SLW*23] proposed a modification to the anisotropic transformation of [YT13], which instead uses the dimensionally consistent square roots of the singular values and further reduces undesirable surface artifacts. In this work, we compare against this state-of-the-art method developed for Pahi [SLW*23].

Explicit mesh reconstruction. Other methods reconstruct an explicit surface mesh directly from the particle data, such as [Wil08] which devised a novel constrained optimization approach resulting in good surface properties ([BGB11] also formulated an implicit surface reconstruction method inspired by this approach). [YWTY12] tracks an explicit triangle mesh representing the liquid surface of an SPH simulator, advecting mesh vertices with nearby particle velocities and projecting the mesh surface onto the implicit surface defined by [YT13] to maintain consistency. Surface bumpiness has also been addressed by post-processing the bumpy meshes with variants of Laplacian smoothing, such as in splashsurf [LBJB23] which computes weights based on flow features to avoid overly damping desired details. [Aki14] also proposed post-processing of the surface mesh using decimation and subdivision to improved surface quality.

Surface reconstruction speed. There has also been interest in improving the speed of the surface reconstruction process. [AIAT12] presented a generic method to parallelize Marching Cubes-based isocontouring of scalar fields in a narrow band near the surface and demonstrates their method on [SSP07]. [WLS*17] further improved efficiency with a two-level spatial uniform grid structure. Recently, [QP22] presented a simple but fast approach for fluid surface reconstruction based on filtering a scalar field defined as the number of particles in each cell.

Neural implicit surfaces. Surface reconstruction from surface point cloud data, such as acquired by 3D scanning or LiDAR, has received a great deal of attention in the vision and graphics community. In contrast to particles from fluid simulation, which sample the interior, the point cloud represents a noisy and incomplete sampling of the object surface. [BTS*17] gives an overview of classical methods, including the popular screened Poisson surface reconstruction [KH13]. Recently, researchers have developed a wide variety of data-driven approaches to address challenges in surface reconstruction, object representation, shape generation, and object completion [GWH*20; FGC*23]. DeepSDF [PFS*19] was one of the first works to represent a signed distance field with a neural network whose input is a query point and a global latent vector rep-

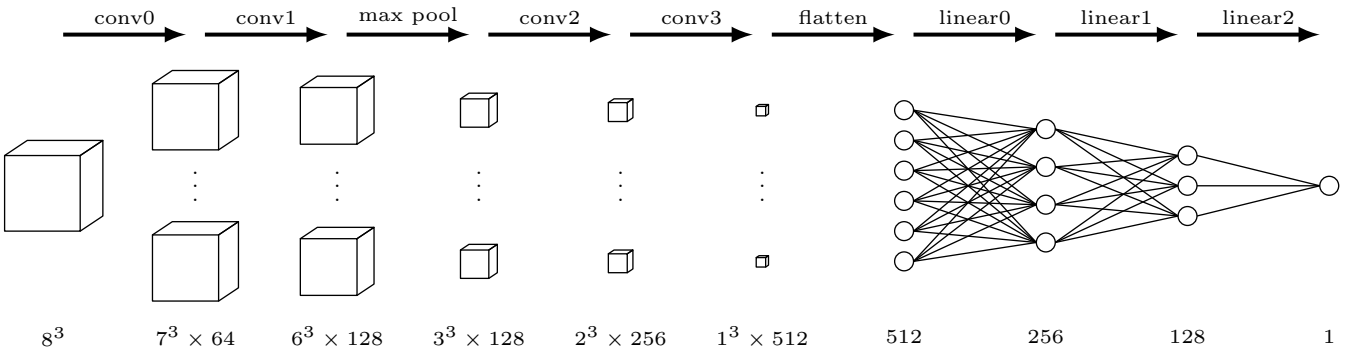


Figure 2: *Network structure for $\mathcal{NN}$. conv0, ..., conv3 are convolutional layers with $2 \times 2 \times 2$ kernels and stride 1. The max pool layer uses a $2 \times 2 \times 2$ kernel and stride 2. ReLU activations are applied after conv0, max pool, conv2, and conv3. The layers linear0, ..., linear2 are fully connected layers with Tanh activations after linear0 and linear1.*

resenting the point cloud. While resulting in a highly compressed representation of a class of shapes, it does not generalize well to other classes of shapes. Subsequent works improved generalizability, such as [CLI*20] which uses a grid of local latent vectors. To improve generalizability, [EGO*20] proposed inferring absolute distance from local patches and using a sparse sampling of the global point cloud to infer the sign. Similarly, we use only local particle information to infer the SDF at a query point and do not require any global information to infer sign since fluid particles are always inside the volume.

Direct learning on point clouds. Several methods have explored learning features directly on point clouds [QSMG17; QYSG17] and point cloud convolutions [LBS*18; HTY18; AML18; TQD*19; UPTK19; BPM20], and these architectures have been used in surface reconstruction from point cloud data [GFK*18; EGO*20; AL20; BM22]. Deep marching cubes [LDG18] developed an end-to-end trainable model that predicts explicit surface meshes from point cloud data. Inspired by neural marching cubes [CZ21], which used a neural approach to generate high quality smooth surfaces from discrete SDFs, neural dual contouring [CTFZ22] applied a deep learning approach to dual contouring to generate surface meshes directly from various inputs including point cloud data.

Grid-based convolutional neural networks. Though we experimented with point encodings and point cloud convolutions, we found improved results by constructing features from particle data on a regular grid. Many works have converted point cloud data to a grid representation to facilitate processing with convolutional neural networks (CNNs), which exhibit translational invariance and by learning local spatial filters, can capture local spatial relationships and patterns inherent in the data. Architectures utilizing regular grids as in VoxNet [MS15] and adaptive octree grids as in OctNet [ROUG17] have been demonstrated to be effective for tasks such as object classification and segmentation, orientation estimation, and point cloud labeling. CNNs have been used in surface reconstruction from point cloud data to generate multiscale features [CAPM20; CPM*20], while [UK21] proposes aggregating point cloud data into an adaptive grid, such as an octree, and then processing with multiscale convolutional kernels. [YFM*22] also makes use of octrees to improve the scalability of the surface reconstruction process. [LGL*24] uses an attention mechanism to enhance the construction of grid data from the point cloud.

Regularization. To avoid overfitting and improve the smoothness and accuracy of the results, several methods have incorporated regularization terms that impose desirable surface properties or properties of the unsigned or signed distance function. RegSDF [ZYL*22] improves the robustness of surface reconstruction to noisy or incomplete data using Hessian regularization and minimal surface regularization. [MHLZ20] formulated a pulling loss that pulls query locations to their closest surface points using the predicted signed distance and gradient. This was used in PCPNet [MLZH22] and in [CLH23] which also incorporated total variation, surface, and gradient loss terms. Several methods have made use of the Eikonal equation satisfied by the signed distance field for regularization. [GYH*20] proposed an unsupervised approach with their loss driving the implicit function to vanish on the point cloud and the function gradient to have unit norm. [YPN*22] also employed a regularizing Eikonal term in their loss. Recently, [JWZ23] devised a novel self-loop loss for regularization.

3. Approach

Our general strategy is to construct a network that is able to predict the signed distance of a query point given the particles in the local neighborhood. We can then use this to populate a level set on a regular grid. After experimentation, we found that it was most effective to first transfer the particle data onto a regular grid and use the data on the regular grid as features for the network. Although this requires the use of a separate preprocessing step, this allows us to utilize convolutional neural networks in our network. The regular grid conveniently encodes spatial information about the location and density of particles. This also avoids complications relating to neighborhood size and particle coverage since the network only sees normalized grid data. An outline of our approach is shown in Figure 1. The network is trained to predict a narrow-band signed distance function with the surface as the zero level set. The training data consists of analytical implicit surfaces sampled with interior particles, and the loss function includes a polynomial regularization term to enhance the smoothness of the resulting surface.

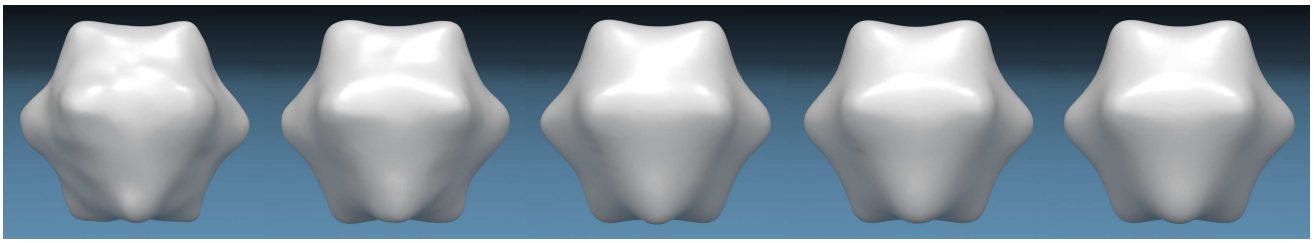


Figure 3: *Reconstruction of a blobby sphere sampled with 1, 2, 4, 8, and 16 particles per cell. The reconstruction is noisy when particles are sparse, since insufficient data coverage is available to infer a smooth surface. At higher sampling densities, a smooth surface is reconstructed, and this reconstruction is insensitive to particle sampling density.*

3.1. Feature construction

The first step of our algorithm is constructing features m_c on the cell centers c of a uniform grid with grid spacing Δx. We do this using a kernel-based transfer of particle location information to the grid, normalized by the local particle density

$$m_c = \sum_{p \in N_c} \frac{1}{\rho_p} W(\|\mathbf{x}_c - \mathbf{x}_p\|, R) \quad \rho_p = \sum_{q \in N_p} W(\|\mathbf{x}_p - \mathbf{x}_q\|, R),$$

where $\mathbf{x}_c$ is the location of grid cell c, N_c is the set of all particles p within radius R of $\mathbf{x}_c$, $\mathbf{x}_p$ is the position of the particle p, W is the smoothing kernel, and ρ_p is the density of particle p where N_p is the set of particles within radius R, including the particle p itself. We use the `Poly6` kernel with $R = 3\Delta x$ as our kernel support radius.

$$W(r,R) = \frac{315}{64\pi R^9}(R^2 - r^2)^3. \tag{1}$$

The choice of grid Δx is discussed in Section 3.5.

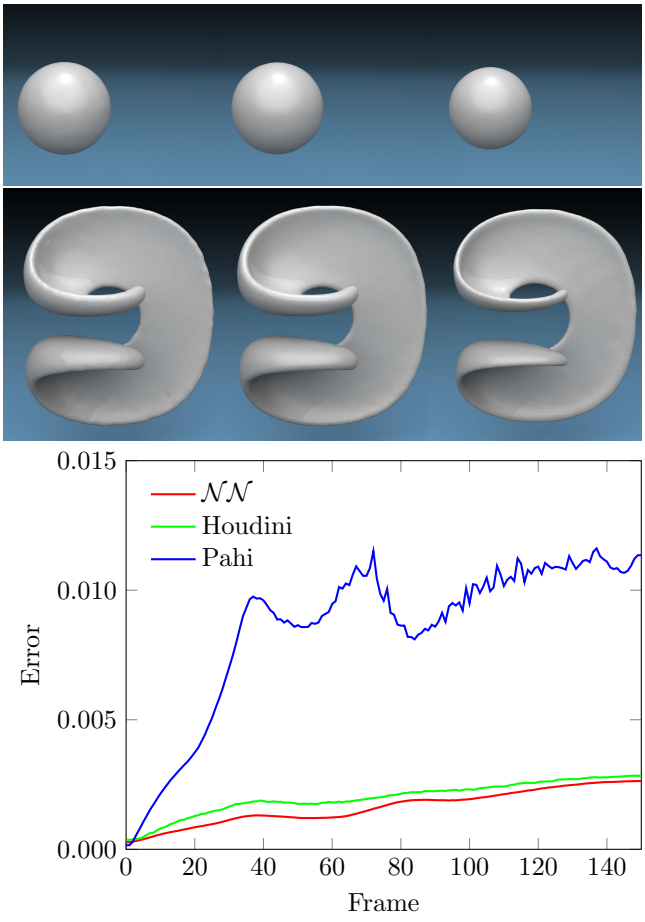


Figure 4: *Quantitative analysis on 3d vortex deforming sphere, comparing Houdini (left), ours (middle), and Pahi (right).* ***Top:*** *Initial reconstructed sphere at frame 0.* ***Middle:*** *Deformed shape at frame 150.* ***Bottom:*** *Error over all frames, measured at the surface particle locations.*

3.2. Network architecture

Next, we construct a convolutional neural network $\phi_i \leftarrow \mathcal{NN}(\mathbf{M}_i)$ that predicts the signed distance value ϕ_i for the grid node i from the features $\mathbf{M}_i$ surrounding it, as illustrated in Figure 1. The feature vector $\mathbf{M}_i$ consists of the 8^3 cell-centered features m_c (blue) surrounding the grid node i (red). The structure of our network is illustrated in Figure 2. It consists of two convolutional layers, a max pooling layer, two more convolutional layers, and three fully connected layers.

3.3. Loss

During training, we construct feature blocks of size 10^3. Since a 8^3 block is required to compute one value of ϕ_i, each 10^3 block gives us 3^3 values of ϕ_i. We assemble these into a vector Φ of size 27 in the order shown in the right of Figure 1. Let Φ^t represent the corresponding ground truth Φ values. Our loss function takes the form

$$L = \|\Phi - \Phi^t\|_2^2 + \lambda L_p(\Phi), \tag{2}$$

where the first loss term is the sum of squares error of the predicted signed distance values and $L_p(\Phi)$ is a regularization term that we describe in the next section.

3.3.1. Polynomial regularization

Using signed distance errors as the only loss term leads to bumpy surfaces. There are many possible ways to regularize ϕ. Penalizing $\|\nabla\phi\|$ would be problematic, since this quantity should not be zero even for flat surfaces. Instead we could add a term like $(\|\nabla\phi\| - 1)^2$, which would enforce the Eikonal equation and prevent wild variations in the derivatives of ϕ [GYH*20; YPN*22].

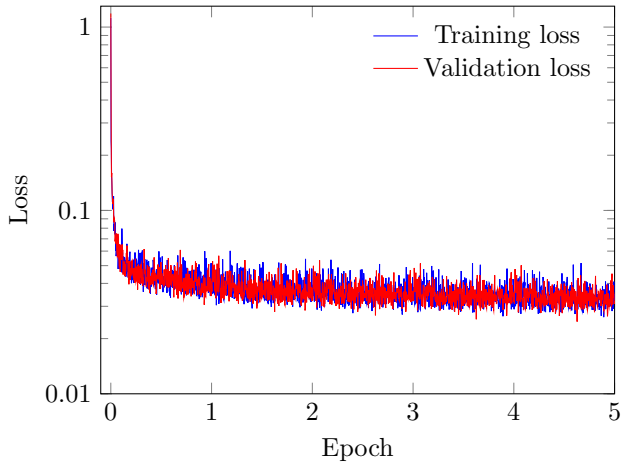


Figure 5: *Training and validation loss versus epoch for our model. The training loss converges within five epochs without signs of overfitting.*

Alternatively, one might consider a term that penalizes the Hessian or curvature of ϕ [ZYL*22], but this is also undesirable. Fluid surfaces contain small features like droplets that *should* have significant curvature, and we want our network to be able to accurately learn these features. We also note that derivatives in ϕ would require derivatives of our network's features with respect to a uniform shift to all of the input particles, which would increase the size of the training data by a factor of four. We found this to be impractical. Instead, we consider a strategy for encouraging smoothness without using derivatives.

An alternative to derivatives is to compute ϕ at several nearby nodes. Although we could use these to approximate derivatives using finite differences, we pursue an alternative strategy. We want to penalize noise, that is, irregularity, not curvature. To do this, we fit a quadratic polynomial to the 3^3 neighborhood using least squares and construct a loss from the fitting error. This fitting can be done very efficiently. Let (x_i, y_i, z_i) with $x_i, y_i, z_i \in \{-1, 0, 1\}$ be the locations of the 27 entries in Φ. Given the Vandermonde matrix

$$A = \begin{pmatrix} x_1^2 & x_1y_1 & x_1z_1 & y_1^2 & y_1z_1 & z_1^2 & x_1 & y_1 & z_1 & 1 \\ x_2^2 & x_2y_2 & x_2z_2 & y_2^2 & y_2z_2 & z_2^2 & x_2 & y_2 & z_2 & 1 \\ \vdots & & & & & & & & & \\ x_{27}^2 & x_{27}y_{27} & x_{27}z_{27} & y_{27}^2 & y_{27}z_{27} & z_{27}^2 & x_{27} & y_{27} & z_{27} & 1 \end{pmatrix},$$

we wish to choose coefficients C for our polynomial so that $\|AC - \Phi\|_2^2$ is minimized. These coefficients are given by $C = (A^TA)^{-1}A^T\Phi$. Then, the error is $L_p(\Phi) = \|A(A^TA)^{-1}A^T\Phi - \Phi\|_2^2$. Defining the projection operator $K = I - A(A^TA)^{-1}A^T$ the polynomial fit error is simply

$$L_p(\Phi) = \|K\Phi\|_2^2 = \Phi^T K \Phi. \tag{3}$$

Note that A is constant, so that K is a fixed 27×27 matrix. We provide the full (x_i, y_i, z_i), A, and K in the accompanying technical document. We note that a quadratic fit here is ideal, since a linear polynomial would penalize curvature and a cubic polynomial would be ineffective since it has almost as many coefficients (20) as degrees of freedom (27).

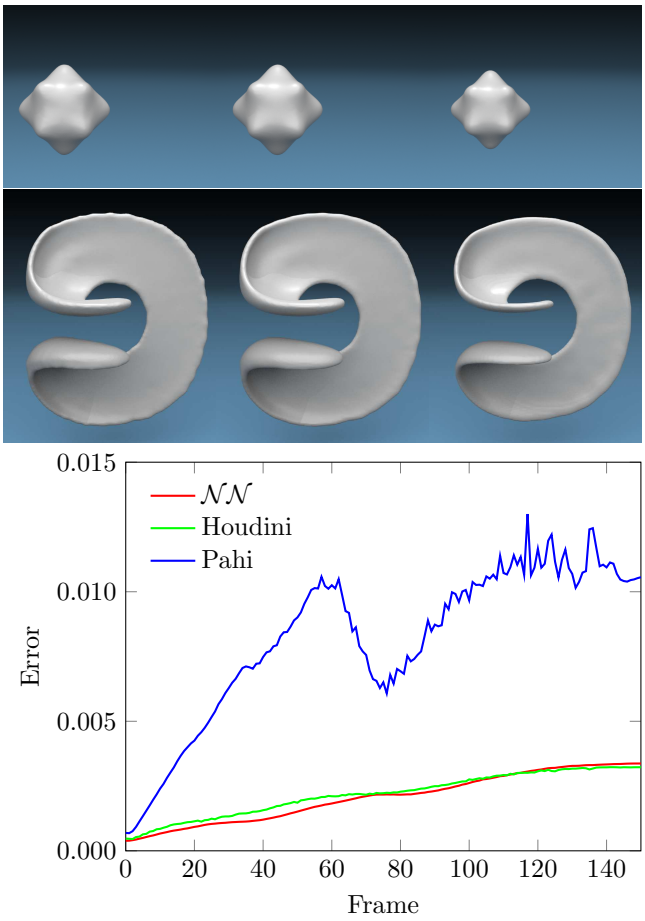


Figure 6: *Quantitative analysis on 3D vortex deforming bumpy sphere comparing Houdini (left), ours (middle), and Pahi (right).*

3.4. Training data and training process

We train our network using only analytic shapes. For volumetric shapes, we use four basic shapes: sphere, cone, torus, and cylinder. We generate these shapes using random position, orientation, and shape parameters. For each shape, we seed particles using three sampling strategies: inside the object, outside the object (but within a slightly enlarged bounding box), or within a small delta from the boundary (to mimic thin sheets). We also add isolated particles to our training set, which are basically single points whose analytic signed distance value we choose to define as a small sphere with radius of Δx so that it can be resolved on the grid and be visually pleasing. Particle seeding is performed using Poisson disk sampling [Bri07]. We seed objects at three different particle separation distances to help the network generalize across particle sampling density. Since particles sampled in this way have distinctive statistical properties that fluid particles will not, we jitter the particles after sampling while making sure all particles remain inside. We also leave a small gap ($0.25\Delta x$) between the outermost particles and the

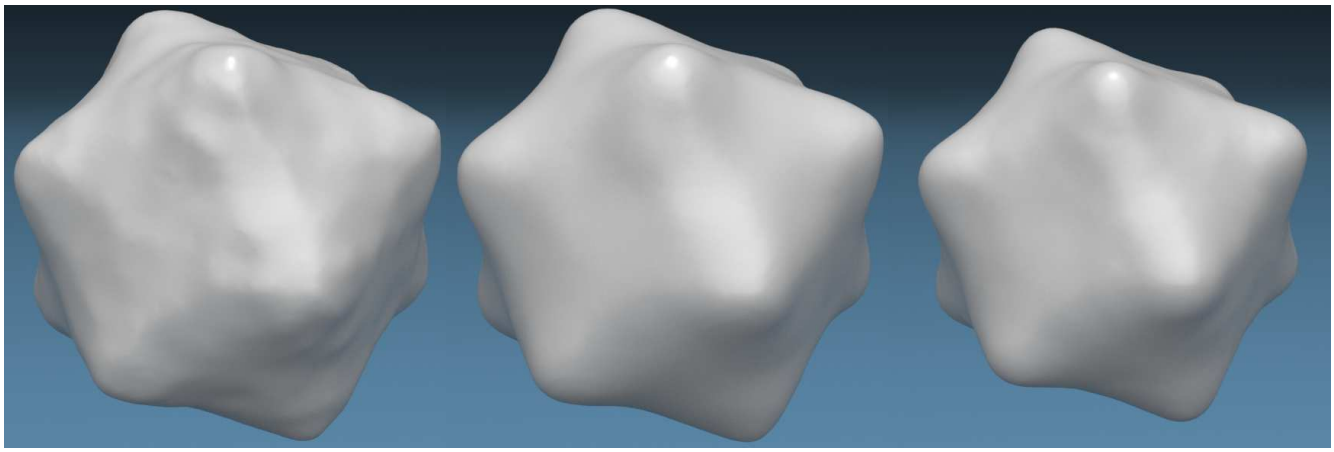


Figure 7: *Translating and rotating particles using Houdini (left), our Pahi's method (right), and ours (middle). Observe that Houdini's reconstruction is noisy, and Pahi's reconstruction artificially shrinks the surface.*

analytical zero level set surface to avoid teaching the network to generate extremely thin shapes that cannot be resolved by grid. The training set consists of pairs $(\mathbf{M}_i, \Phi_i)$ of feature blocks and associated analytic signed distance values.

We use a data set composed of 1.3M training data samples and 80K validation data samples uniformly split into the four basic shapes and the three sampling strategies. Among them, 2/3 of the regular and inverted shapes are sampled with 1ppc and 1/3 with 2ppc, while 2/3 of the thin sheets are sampled with 0.5ppc and 1/3 with 1ppc. We adopted the Adam optimizer with the initial learning rate set as 0.0002. The weight on the polynomial term in the loss function is set as 1. The model was trained for 5 epochs with a batch size of 256. The training time for the model was 40 minutes on an NVIDIA GeForce RTX 4090. The loss plot is given in Figure 5.

During the training process, for each shape, we first find grid nodes j whose ϕ values range from $[-2\Delta x, 2\Delta x]$. For each j, we compute the input features for a neighboring group of 27 nodes i centered at j. We store this 10^3 block of features and the corresponding 3^3 values of ϕ_i for training. A set of 256 blocks corresponds to a batch during training and allows us to compute the ϕ fitting error term and the polynomial regularization term for the loss function in Equation (2).

3.5. Higher resolution level sets

The reconstruction algorithm presented has two critical length scales: the characteristic separation distance h between particles and the grid cell size Δx used to compute features m_c. The relationship between these can be expressed in terms of the average number of particles per cell (ppc) of the reconstruction grid. To encourage our network to be insensitive to this parameter, we included samples in the training set with 1, 2, and 4 particles per cell. As can be seen in Figure 3, the reconstruction is relatively bumpy at 1 ppc. Since the particles do not sample the surface but are rather randomly distributed in the interior, accurate surface determination is only possible through particle positioning statistics. At 1 ppc, there are not enough particles to reconstruct a smooth surface. The reconstruction is much smoother at 2 ppc and has visually converged by 4 ppc, after which the reconstruction is insensitive to particle density. This is important since particle density often varies considerably within a fluid simulation.

The need to retain a sufficient number of particles per cell limits how small Δx can be, which in turn limits the geometric length scale that can be represented in the reconstructed level set.

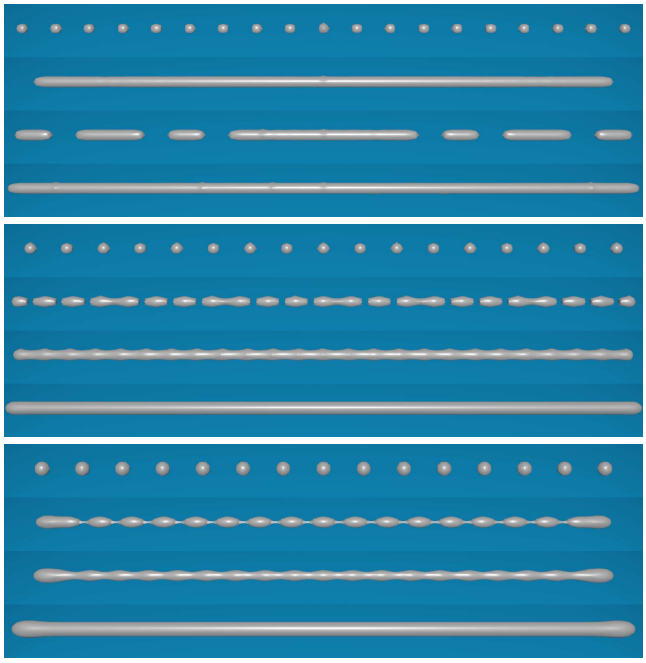
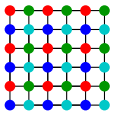


Figure 8: *A line of particles moves closer together to observe the transition from a string of isolated particles to a solid line using Houdini (top), ours (middle), and Pahi (bottom). Houdini's merging is not monotonic, and the final shape has artifacts. Pahi's final surface has bulges at the ends.*

Figure 9: *Rendering of rings falling into a pile, with surfaces reconstructed using Houdini (left), ours (middle), and Pahi (right). The Houdini reconstruction is bumpy. The Pahi reconstruction exhibits a bulge along the sharp edge. Our method is smooth and without edge artifacts.*

Therefore, to achieve a higher resolution level set reconstruction, rather than reduce Δx, we do multiple, staggered reconstructions. In particular, consider a double fine grid with resolution $\Delta x/2$. We can label its nodes as $(2i+a, 2j+b, 2k+c)$ for $a,b,c \in [0,1]$. Each of the eight combinations of a,b,c gives a separate grid of resolution Δx. We reconstruct level set values on each of these eight grids independently, so that we can maintain the cell size Δx while achieving a double sampling of the reconstructed level set. This is illustrated analogously for two dimensions in the inset figure. For example, the red nodes comprise a grid of the original resolution Δx, while the union of all the nodes comprise a grid of resolution $\Delta x/2$. In contrast to using a double fine grid directly, this approach maintains the cell size for computing features and hence the number of particles per cell, resulting in a smoother surface. We perform this resolution doubling on all of our reconstructions.



3.6. Pruning nodes

To make the surface reconstruction process efficient, we avoid doing network-based inference on grid nodes far from the surface. Instead, these nodes are assigned a fixed negative or positive ϕ value. For a given grid node, we count the number of neighboring cells that contain particles. If the number is greater than an empirically predetermined threshold (2000 of 4096 double-fine cells within the feature radius), the node is labeled as inside and assigned the fixed negative ϕ value. If no neighboring grid cells contain particles, the node is labeled as outside and assigned the fixed positive ϕ value. Using this strategy, the majority of the grid nodes are pruned and only a relatively small number of ϕ_i values will be inferred by the network. Counting the number of neighboring cells with particles can be done efficiently using dynamic programming.

4. Results

Throughout this section we compare our surface reconstruction method with the Pahi method [SLW*23], based on [YT13], and with the surface reconstruction tool available in Houdini [Sid]. The domain size of the double dam break example in Figure 15 is $1m \times 1m \times 1.5m$ and the sphere drop example in Figure 16 is $1m \times 1m \times 3m$. The other examples have a domain size of $1m \times 1m \times 1m$. We perform resolution doubling on a grid of 100 cells per meter for our method while we apply other methods on a grid of 200 cells per meter for fair comparison.

Reconstruction error. To give a quantitative analysis of the accuracy of the three methods, we perform the Enright test [EFFM02]. Rather than advecting a level set as in the original test, we seed the initial sphere with particles, advect the particles through the velocity field using third order Runge-Kutta, and reconstruct the surface from the particles at each step. To track the actual interface, we seed particles on the sphere's surface at the beginning of the test and advect them through the velocity field. We then evaluate each of the reconstructed surfaces $\phi(\mathbf{x})$ at the locations of the advected surface particles $\mathbf{x}_i$. We compute the error as $E = \frac{1}{N}\sum_i \frac{|\phi(\mathbf{x}_i)|}{\|\nabla\phi(\mathbf{x}_i)\|}$, where the gradient is computed with finite differences. Note that this choice of error does not assume that the reconstruction should be a signed distance field, and instead captures the error in ϕ relative to the change in ϕ over a single grid cell. Since surface construction algorithms tend to construct the surface at some user-chosen offset, we move the surface particles to the isocontour that minimizes the error E in the initial state to account for this. The results are shown in Figure 4. We repeat the same test but with a bumpy sphere as the initial state (which is not in our training data); the results of this test are shown in Figure 6. Note that the calibration of the surface particles causes the reconstruction errors at the initial state to be similar. While the errors in Houdini's and our reconstructions are similar, the errors in Pahi's reconstruction grow rapidly. Note also that Pahi tends to shrink the surface quite significantly. We observe that our reconstruction is generally smoother than the others, which is especially noticeable at the edges.

Rigid body motion. To demonstrate the temporal coherence of our approach, we reconstruct a translating and rotating bumpy sphere. We observe Houdini's reconstruction to be bumpier (as seen in Figure 7) with moderate temporal fluctuations (as seen in the supplementary video). Our reconstruction is smooth in both time and space.

Merging line of particles. We take a line of equally spaced particles and slowly move them closer together to observe the transition from isolated particles to a unified structure, as shown in Figure 8. In Houdini's reconstruction, the particles merge, separate, and then merge again as particles move closer. The final reconstructed surface is uniform in diameter but has occasional bumps along it.

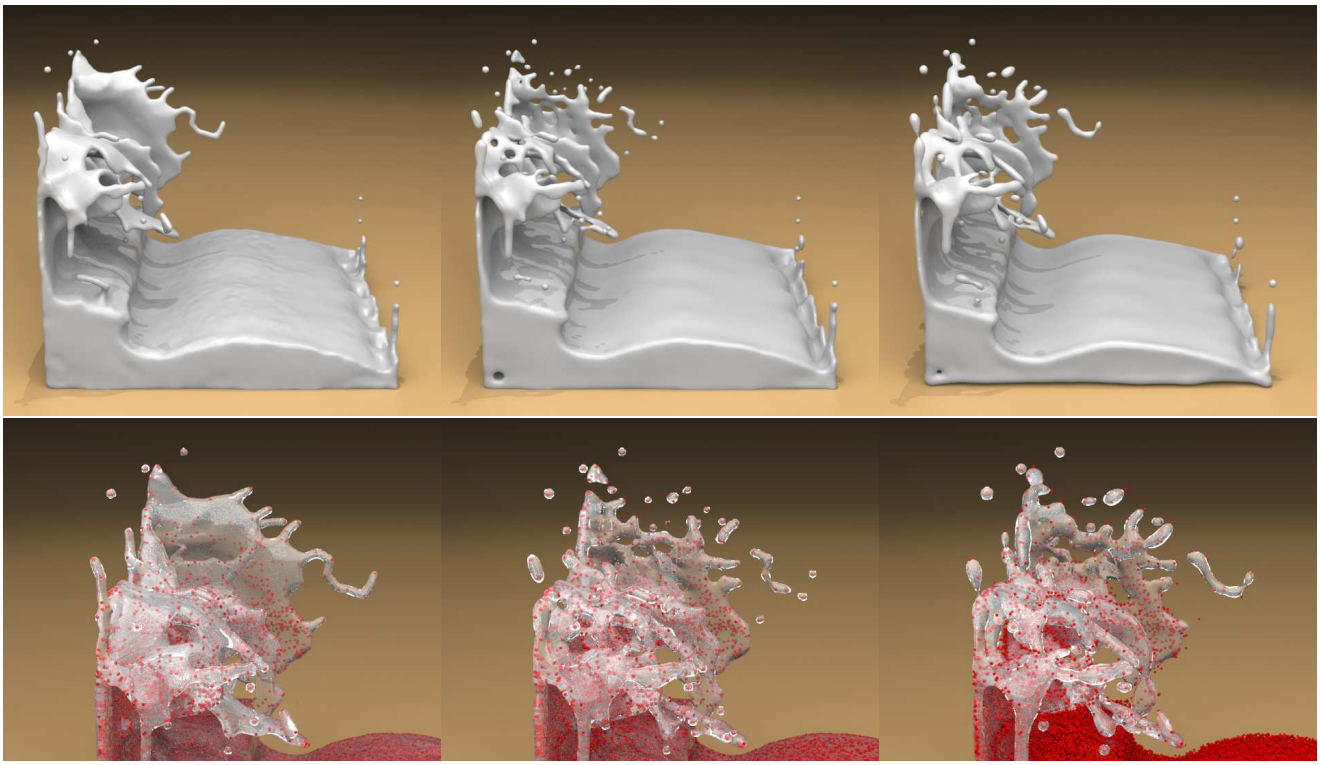


Figure 10: *Dam break simulation, comparing Houdini (left), ours (middle), and Pahi (right). Bottom row figures are zoomed in version with particles shown.*

Pahi's reconstruction shows a regular and clean merge, though the final shape has bulges at the ends. As in many of the tests, the thickness of Pahi's reconstruction tends to vary quite significantly from the others. Our method's merging behavior is monotonic, and the final shape is smooth with a uniform diameter.

Merging plane of particles. We take a plane of randomly spaced particles and slowly move them closer together to observe the transition from isolated particles to a plane, as shown in Figure 11. Houdini's reconstruction creates random holes in the plane as the particles merge which loses time coherence. Pahi's reconstruction shows a regular and clean merge, though it gives a blobby plane in the end. Our method's merging behavior exhibits better time coherence and a smooth final plane.

Ring drop. Figure 9 depicts the surface reconstruction of a particle-based solid simulation of three deformable rings dropped into a pile. The rings have sharp edges, which allows us to compare the behavior of the reconstruction methods on sharp features. We observe Houdini's reconstruction to be rather bumpy. Pahi's reconstruction is smooth, but it creates a noticeable bulge along the sharp edges. Our reconstruction is smooth and does not introduce artifacts at sharp edges. **Dam break.** We reconstruct the fluid surface from a dam break test as shown in Figure 10. We observe that Houdini's fluid surface is not as smooth as the others. We also observe Houdini's tendency to create filaments from isolated particles. Pahi's reconstruction is similar to our own on this example. As can be observed in the supplementary video, along with Houdini, our method better reconstructs the sharp edges of the box than Pahi.

Bunny. Figure 12 depicts an example where chocolate is poured over a bunny. We observe that Houdini tends to fill in large gaps between particles, and its reconstruction of the fluid surface on the

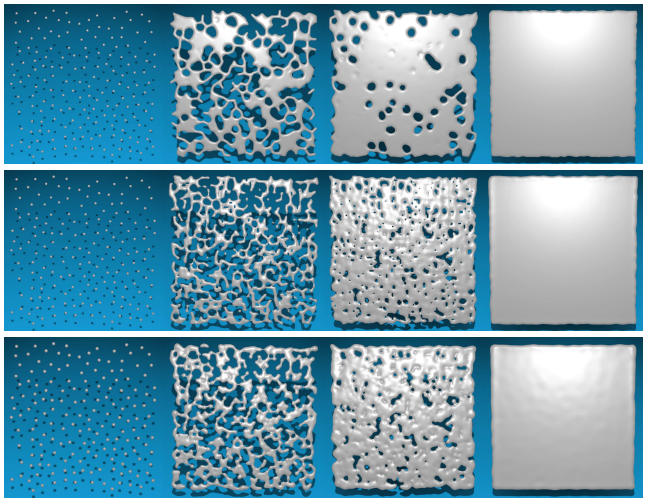


Figure 11: *A plane of particles moves closer together to observe the transition from randomly sampled isolated particles to a solid plane using Houdini (top), ours (middle), and Pahi (bottom). Houdini's merging is not time consistent and Pahi's final plane is bumpy.*

Figure 12: *Chocolate is poured over a bunny. The surfaces are reconstructed with Houdini (left), ours (middle), and Pahi (right).*

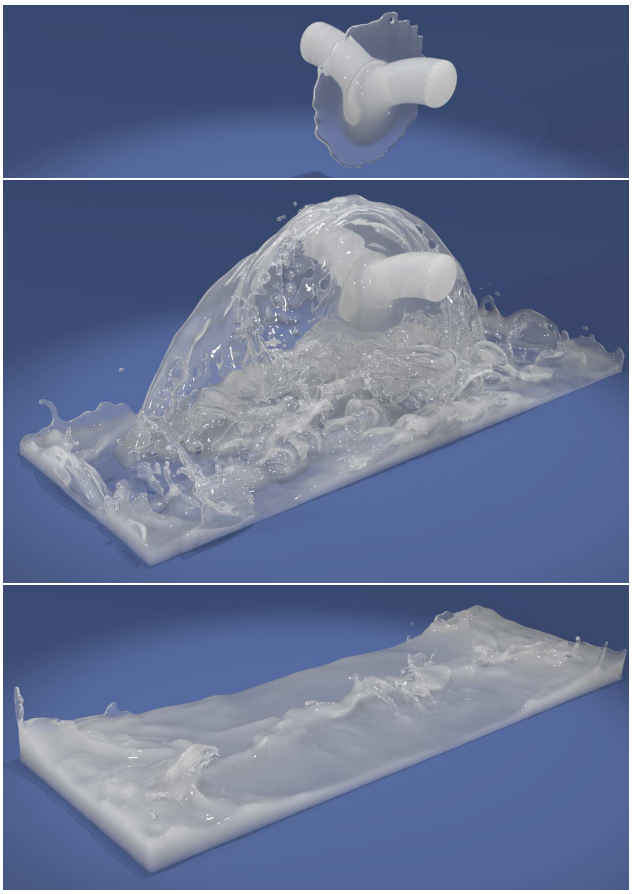


Figure 13: *Two fluid sources collide, producing dynamic water features. The surface is reconstructed using our method.*

floor is bumpier. Pahi's reconstruction is better, but it smooths out relatively large features on the floor. Our reconstruction produces a smooth fluid surface without losing larger scale surface features.

Pour on box. In this test, water is poured onto a box, causing it to flatten out into a sheet (see the accompanying video). Houdini tends to merge the chaotic splashes along the wall into a solid surface (See Figure 14). We also observe that Pahi's reconstruction of the source column is much thinner than it should be.

Ablation study In this test, we reconstruct the surface of a blobby sphere using neural networks with different model setups, as shown in Figure 17 (best viewed under magnification). From left to right are models with (1) feature size 8 and kernel radius 3 (ours), (2) same as (1) except without polynomial loss, (3) feature size 8 and kernel radius 2, (4) feature size 4 and kernel radius 3, (5) feature size 6 and kernel radius 3 (6) feature size 8 and kernel radius 4. The reconstruction looks most smooth in (1) and (6); we choose (1) since a larger radius tends to incorrectly close holes in fluid surfaces.

Fluid simulations. We finish with three dynamic fluid simulations to show the quality of our reconstruction in different scenarios. Figure 15 shows a double dam break, where two columns of fluid collapse into a pool. Figure 13 shows two fluid sources colliding with each other. Figure 16 is a classical sphere drop. Our surface reconstruction is able to consistently produce smooth thin sheets and splashes. In calm water, our method can produce almost glassy smooth fluid surfaces. Our method reconstructs the sharp edges and corners of the initial fluid shape without undesirable bulging edge artifacts and with only mild smoothing of corners.

Timing information. Timing information for each of the simulation reconstructions is shown in Figure 18. We run our method on the CPU, with only network evaluation computed on the GPU. Houdini is overall fastest, Pahi's reconstruction is slowest, and ours is in between. For Pahi and our method, the Linux `time` command was used for timing, which includes all steps used to generate the signed distance field from particles. For Houdini, we used the time stamps of the dumped signed distance files to estimate timings.

5. Conclusion and Limitations

In this paper, we presented a network-based scheme for reconstructing fluid surfaces from particles. The reconstruction exhibits improved spatial surface smoothness and temporal coherence compared with existing state of the art surface reconstruction methods. The method is insensitive to particle sampling density and robustly handles thin features, isolated particles, and sharp edges. Although artifacts are significantly less than existing methods, the reconstruction does still have artifacts. Some of these artifacts, such as the surface smoothness at low particle sampling density, are related to

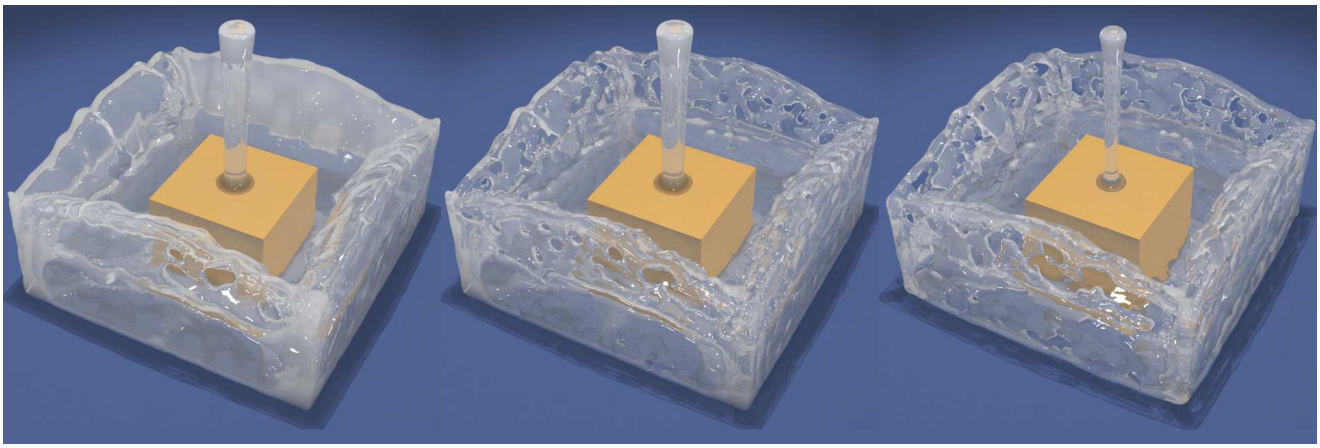


Figure 14: *Water is poured on a box, showing thin sheets and dynamic splashes. The surfaces are reconstructed with Houdini (left), ours (middle), and Pahi (right). Houdini merges the splashes at the boundary, and Pahi thins the spout.*

the statistical nature of the reconstruction problem. The proposed reconstruction is independent per time step; improved temporal coherence may be possible by using information from particles at adjacent time steps. Although the reconstruction is time competitive with Pahi, it is slower than Houdini's reconstruction. Our implementation of the feature computation is single threaded for simplicity; we exploit parallelism by performing reconstructions in multiple frames in parallel. Threading or GPU implementation may be used to accelerate the surface reconstruction. In this work, the reconstructed surface has been used for the sole purpose of rendering. An interesting avenue for future work is to devise a network-based surface reconstruction that yields good accuracy in surface tension computations, where errors in curvature estimation are a significant source of spurious currents. For example, our approach could be combined with recent machine learning algorithms for estimating curvature from level set values [LCG22; LCG23].

We have not observed the model to fail in practice, in the sense that it has always given us a reconstructed surface that is reasonable. As is generally the case with network-based approaches, we cannot prove that the model will not do unexpected things in unusual circumstances. We have tested the model on anticipated failure cases such as failing to put a surface around isolated or small groups of particles, but we have not observed it to fail such tests.

6. Acknowledgements

This work was supported by National Science Foundation award NSF-2006570 and University of California award M23PL6076.

References

[AIAT12] AKINCI, G., IHMSEN, M., AKINCI, N., and TESCHNER, M. "Parallel Surface Reconstruction for Particle-Based Fluids". *Computer Graphics Forum* 31.6 (2012), 1797–1809. ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2012.02096.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2012.02096.x 2.

[AL20] ATZMON, MATAN and LIPMAN, YARON. "Sal: Sign agnostic learning of shapes from raw data". *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, 2565–2574 3.

[AML18] ATZMON, MATAN, MARON, HAGGAI, and LIPMAN, YARON. "Point Convolutional Neural Networks by Extension Operators". 2018 3.

[APKG07] ADAMS, BART, PAULY, MARK, KEISER, RICHARD, and GUIBAS, LEONIDAS J. "Adaptively sampled particle fluids". *ACM SIGGRAPH 2007 papers*. 2007, 48–es 2.

[Aki14] AKINCI, GIZEM. "Efficient surface reconstruction for SPH fluids". *PhD diss., Dissertation, Universität Freiburg* (2014) 2.

[BGB11] BHATACHARYA, HAIMASREE, GAO, YUE, and BARGTEIL, ADAM. "A level-set method for skinning animated particle data". *Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on computer animation*. 2011, 17–24 2.

[BM22] BOULCH, ALEXANDRE and MARLET, RENAUD. "Poco: Point Convolution for Surface Reconstruction". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, 6302–6314 3.

[BPM20] BOULCH, ALEXANDRE, PUY, GILLES, and MARLET, RENAUD. "FKAConv: Feature-kernel Alignment for Point Cloud Convolution". *Proceedings of the Asian Conference on Computer Vision*. 2020 3.

[BR86] BRACKBILL, J. and RUPPEL, H. "FLIP: A method for adaptively zoned, Particle-In-Cell calculations of fluid flows in two dimensions". *J Comp Phys* 65 (1986), 314–343 1.

[BTS*17] BERGER, MATTHEW, TAGLIASACCHI, ANDREA, SEVERSKY, LEE M., et al. "A Survey of Surface Reconstruction from Point Clouds". *Computer Graphics Forum* 36.1 (2017), 301–329. ISSN: 1467-8659. DOI: 10.1111/cgf.12802. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12802 2.

[Bli82] BLINN, JAMES F. "A generalization of algebraic surface drawing". *ACM transactions on graphics (TOG)* 1.3 (1982), 235–256 1.

[Bri07] BRIDSON, R. "Fast Poisson disk sampling in arbitrary dimensions." *SIGGRAPH sketches*. 2007, 22 5.

[CAPM20] CHIBANE, JULIAN, ALLDIECK, THIEMO, and PONS-MOLL, GERARD. "Implicit functions in feature space for 3d shape reconstruction and completion". *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, 6970–6981 3.

[CGB16] CANEZIN, FLORIAN, GUENNEBAUD, GAËL, and BARTHE, LOÏC. "Topology-aware neighborhoods for point-based simulation and reconstruction". *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*. 2016 2.

[CLH23] CHEN, CHAO, LIU, YU-SHEN, and HAN, ZHIZHONG. "Gridpull: Towards scalability in learning implicit representations from 3d point clouds". *Proceedings of the ieee/cvf international conference on computer vision*. 2023, 18322–18334 3.

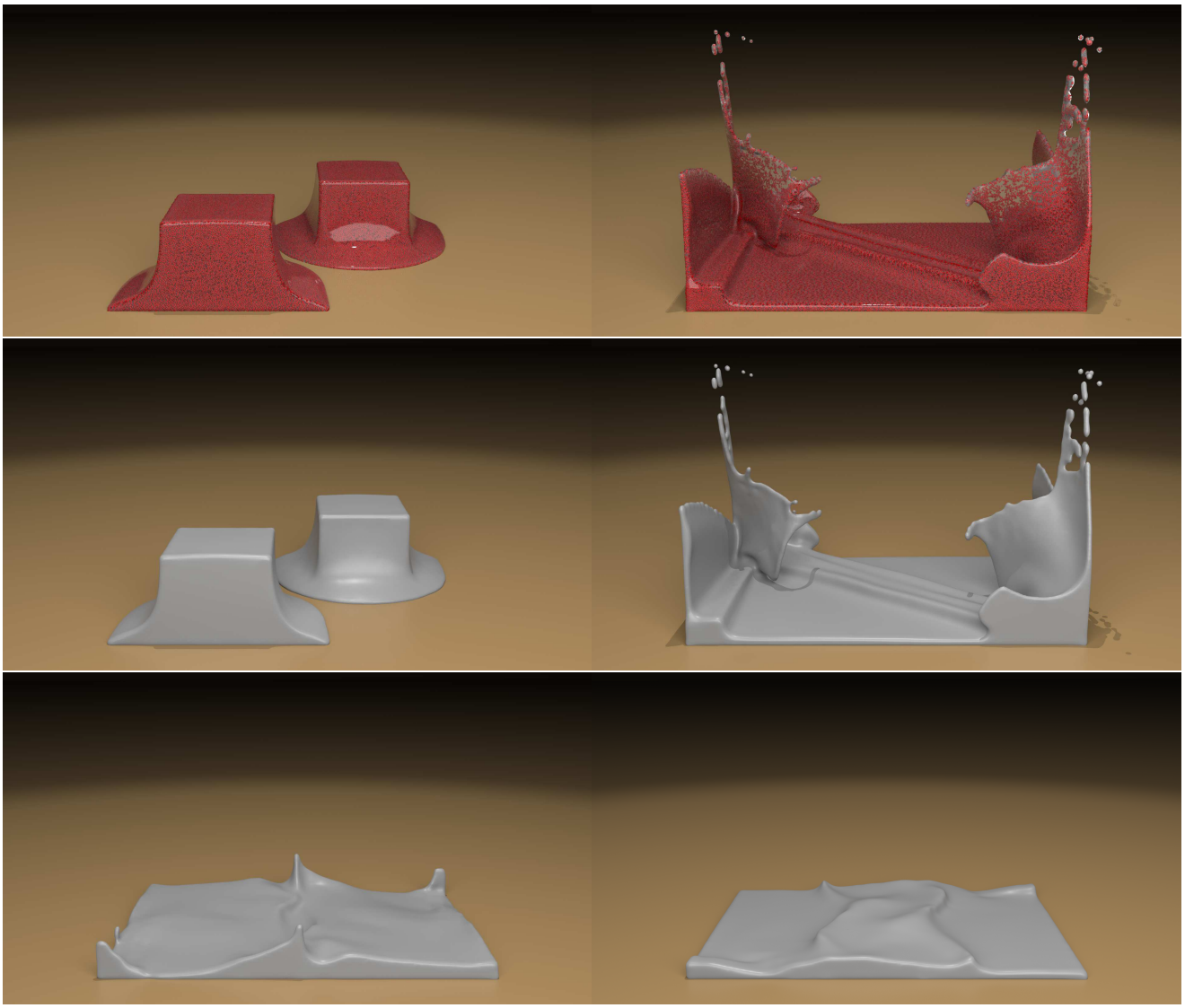


Figure 15: *Two blocks of fluid collapse and splash into a pool. The surface is reconstructed using our method and results in a smooth, temporally coherent render surface. The first row shows the particles for the second row frames.*

[CLI*20] CHABRA, ROHAN, LENSSEN, JAN E., ILG, EDDY, et al. "Deep Local Shapes: Learning Local Sdf Priors for Detailed 3d Reconstruction". *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*. Springer, 2020, 608–625 3.

[CPM*20] CHIBANE, JULIAN, PONS-MOLL, GERARD, et al. "Neural unsigned distance fields for implicit function learning". *Advances in Neural Information Processing Systems* 33 (2020), 21638–21652 3.

[CTFZ22] CHEN, ZHIQIN, TAGLIASACCHI, ANDREA, FUNKHOUSER, THOMAS, and ZHANG, HAO. "Neural dual contouring". *ACM Transactions on Graphics (TOG)* 41.4 (2022), 1–13 3.

[CZ21] CHEN, ZHIQIN and ZHANG, HAO. "Neural marching cubes". *ACM Transactions on Graphics (TOG)* 40.6 (2021), 1–15 3.

[DC98] DESBRUN, MATHIEU and CANI, MARIE-PAULE. "Active implicit surface for animation". *Graphics Interface*. 1998, 143–150 2.

[EFFM02] ENRIGHT, DOUGLAS, FEDKIW, RONALD, FERZIGER, JOEL, and MITCHELL, IAN. "A hybrid particle level set method for improved interface capturing". *Journal of Computational physics* 183.1 (2002), 83–116 7.

[EGO*20] ERLER, PHILIPP, GUERRERO, PAUL, OHRHALLINGER, STEFAN, et al. "Points2surf Learning Implicit Surfaces from Point Clouds". *European Conference on Computer Vision*. Springer, 2020, 108–124 3.

[FGC*23] FARSHIAN, ANIS, GÖTZ, MARKUS, CAVALLARO, GABRIELE, et al. "Deep-Learning-Based 3-D Surface Reconstruction—A Survey". *Proceedings of the IEEE* (2023) 2.

[GFK*18] GROUEIX, THIBAULT, FISHER, MATTHEW, KIM, VLADIMIR G, et al. "A papier-mâché approach to learning 3d surface generation". *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, 216–224 3.

[GWH*20] GUO, YULAN, WANG, HANYUN, HU, QINGYONG, et al. "Deep learning for 3d point clouds: A survey". *IEEE transactions on pattern analysis and machine intelligence* 43.12 (2020), 4338–4364 2.

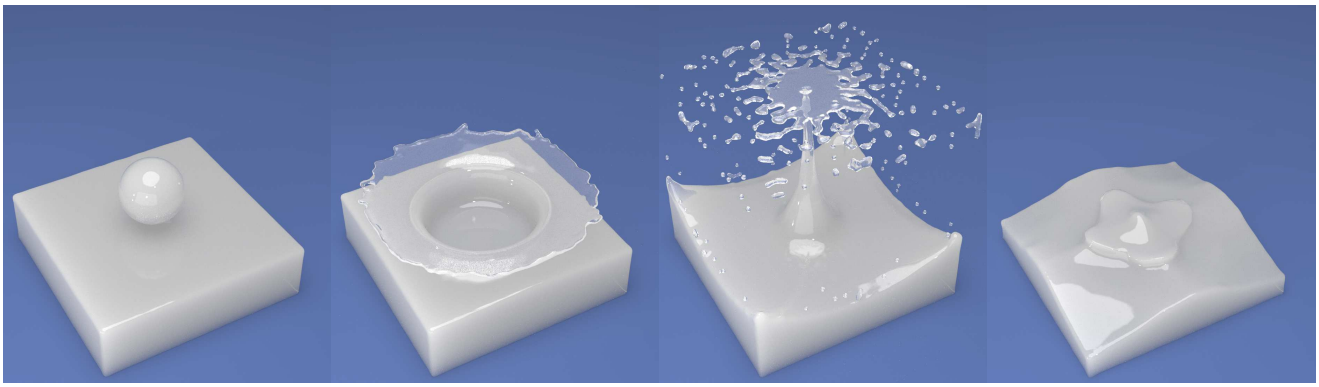


Figure 16: *A sphere drops into a standing pool. Our reconstruction produces a smooth, glassy surface when the fluid is calm.*

[GYH*20] GROPP, AMOS, YARIV, LIOR, HAIM, NIV, et al. "Implicit geometric regularization for learning shapes". *arXiv preprint arXiv:2002.10099* (2020) 3, 4.

[HTY18] HUA, BINH-SON, TRAN, MINH-KHOI, and YEUNG, SAI-KIT. "Pointwise Convolutional Neural Networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 984–993 3.

[Har64] HARLOW, F. "The particle-in-cell method for numerical solution of problems in fluid dynamics". *Meth Comp Phys* 3 (1964), 319–343 1.

[JSS*15] JIANG, C., SCHROEDER, C., SELLE, A., et al. "The Affine Particle-In-Cell Method". *ACM Trans Graph* 34.4 (2015), 51:1–51:10 1.

[JWZ23] JIN, CHUAN, WU, TIERU, and ZHOU, JUNSHENG. "Multi-grid representation with field regularization for self-supervised surface reconstruction from point clouds". *Computers & Graphics* 114 (2023), 379–386 3.

[KBST22] KOSCHIER, DAN, BENDER, JAN, SOLENTHALER, BARBARA, and TESCHNER, MATTHIAS. "A survey on SPH methods in computer graphics". *Computer graphics forum*. Vol. 41. 2. Wiley Online Library. 2022, 737–760 1.

[KH13] KAZHDAN, MICHAEL and HOPPE, HUGUES. "Screened poisson surface reconstruction". *ACM Transactions on Graphics (ToG)* 32.3 (2013), 1–13 2.

[LBJB23] LÖSCHNER, FABIAN, BÖTTCHER, TIMNA, JESKE, STEFAN RHYS, and BENDER, JAN. "Weighted Laplacian Smoothing for Surface Reconstruction of Particle-based Fluids". (2023). DOI: 10.2312/vmv.20231245 2.

[LBS*18] LI, YANGYAN, BU, RUI, SUN, MINGCHAO, et al. "Pointcnn: Convolution on x-Transformed Points". *Advances in neural information processing systems* 31 (2018) 3.

[LC98] LORENSEN, WILLIAM E and CLINE, HARVEY E. "Marching cubes: A high resolution 3D surface construction algorithm". *Seminal graphics: pioneering efforts that shaped the field*. 1998, 347–353 1, 2.

[LCG22] LARIOS-CÁRDENAS, LUIS ÁNGEL and GIBOU, FRÉDÉRIC. "Error-correcting neural networks for two-dimensional curvature computation in the level-set method". *Journal of Scientific Computing* 93.1 (2022), 6 10.

[LCG23] LARIOS-CÁRDENAS, LUIS ÁNGEL and GIBOU, FRÉDÉRIC. "Machine learning algorithms for three-dimensional mean-curvature computation in the level-set method". *Journal of Computational Physics* 478 (2023), 111995 10.

[LDG18] LIAO, YIYI, DONNE, SIMON, and GEIGER, ANDREAS. "Deep marching cubes: Learning explicit surface representations". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 2916–2925 3.

[LGL*24] LI, SHENGTAO, GAO, GE, LIU, YUDONG, et al. "GridFormer: Point-Grid Transformer for Surface Reconstruction". *arXiv preprint arXiv:2401.02292* (2024) 3.

[MCG03] MÜLLER, MATTHIAS, CHARYPAR, DAVID, and GROSS, MARKUS. "Particle-based fluid simulation for interactive applications". *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Citeseer. 2003, 154–159 1, 2.

[MHLZ20] MA, BAORUI, HAN, ZHIZHONG, LIU, YU-SHEN, and ZWICKER, MATTHIAS. "Neural-Pull: Learning Signed Distance Functions from Point Clouds by Learning to Pull Space onto Surfaces". 2020 3.

[MLZH22] MA, BAORUI, LIU, YU-SHEN, ZWICKER, MATTHIAS, and HAN, ZHIZHONG. "Surface Reconstruction from Point Clouds by Learning Predictive Context Priors". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, 6326–6337 3.

[MS15] MATURANA, DANIEL and SCHERER, SEBASTIAN. "Voxnet: A 3d convolutional neural network for real-time object recognition". *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2015, 922–928 3.

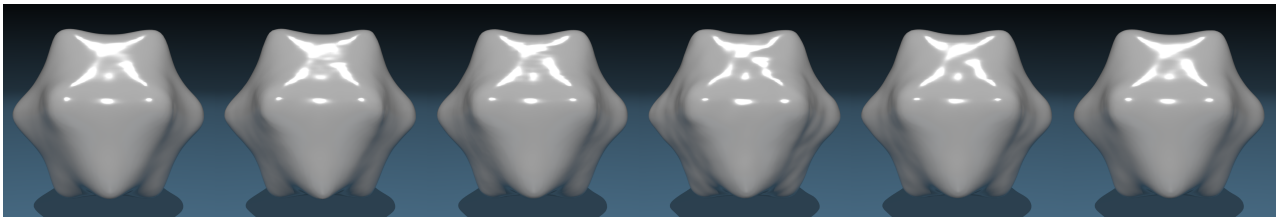


Figure 17: *Reconstruction of a blobby sphere sampled with different model setups. From left to right are models with (1) feature size 8 and kernel radius 3, (2) same as (1) except without polynomial loss, (3) feature size 8 and kernel radius 2, (4) feature size 4 and kernel radius 3, (5) feature size 6 and kernel radius 3 (6) feature size 8 and kernel radius 4.*

[NY06] NEWMAN, TIMOTHY S and YI, HONG. "A survey of the marching cubes algorithm". *Computers & Graphics* 30.5 (2006), 854–879 2.

[PFS*19] PARK, JEONG JOON, FLORENCE, PETER, STRAUB, JULIAN, et al. "Deepsdf: Learning Continuous Signed Distance Functions for Shape Representation". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, 165–174 2.

[PTB*03] PREMŽOE, SIMON, TASDIZEN, TOLGA, BIGLER, JAMES, et al. "Particle-based simulation of fluids". *Computer Graphics Forum*. Vol. 22. 3. Wiley Online Library. 2003, 401–410 2.

[QP22] QUISPE, FILOMEN INCAHUANACO and PAIVA, AFONSO. "Counting Particles: A Simple and Fast Surface Reconstruction Method for Particle-Based Fluids". *2022 35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. Vol. 1. IEEE, 2022, 145–149 2.

[QSMG17] QI, CHARLES R., SU, HAO, MO, KAICHUN, and GUIBAS, LEONIDAS J. "Pointnet: Deep Learning on Point Sets for 3d Classification and Segmentation". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 652–660 3.

[QYSG17] QI, CHARLES RUIZHONGTAI, YI, LI, SU, HAO, and GUIBAS, LEONIDAS J. "Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space". *Advances in neural information processing systems* 30 (2017) 3.

[ROUG17] RIEGLER, GERNOT, OSMAN ULUSOY, ALI, and GEIGER, ANDREAS. "Octnet: Learning deep 3d representations at high resolutions". *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, 3577–3586 3.

[SLW*23] STOMAKHIN, ALEXEY, LESSER, STEVE, WRETBORN, JOEL, et al. "Pahi: A Unified Water Pipeline and Toolset". *Proceedings of the Digital Production Symposium*. 2023, 1–13 1, 2, 7.

[SS07] SHEN, CHEN and SHAH, APURVA. "Extracting and parametrizing temporally coherent surfaces from particles." *SIGGRAPH Sketches* 66.10.1145 (2007), 1278780–1278860 2.

[SSP07] SOLENTHALER, BARBARA, SCHLÄFLI, JÜRG, and PAJAROLA, RENATO. "A Unified Particle Model for Fluid–Solid Interactions". *Computer Animation and Virtual Worlds* 18.1 (2007), 69–82. ISSN: 1546-427X. DOI: 10.1002/cav.162. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.162 2.

[SZS95] SULSKY, D., ZHOU, S., and SCHREYER, H. "Application of a particle-in-cell method to solid mechanics". *Comp Phys Comm* 87.1 (1995), 236–252 1.

[Sid] SIDEFX. *Houdini*. Computer software. SideFX. URL: https://www.sidefx.com/products/houdini/ 7.

[TQD*19] THOMAS, HUGUES, QI, CHARLES R., DESCHAUD, JEAN-EMMANUEL, et al. "Kpconv: Flexible and Deformable Convolution for Point Clouds". *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, 6411–6420 3.

[UK21] UMMENHOFER, BENJAMIN and KOLTUN, VLADLEN. "Adaptive Surface Reconstruction with Multiscale Convolutional Kernels". *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, 5651–5660 3.

[UPTK19] UMMENHOFER, BENJAMIN, PRANTL, LUKAS, THUEREY, NILS, and KOLTUN, VLADLEN. "Lagrangian fluid simulation with continuous convolutions". *International Conference on Learning Representations*. 2019 3.

[WLS*17] WU, WEI, LI, HONGPING, SU, TIANYUN, et al. "GPU-accelerated SPH Fluids Surface Reconstruction Using Two-Level Spatial Uniform Grids". *The Visual Computer* 33.11 (Nov. 1, 2017), 1429–1442. ISSN: 1432-2315. DOI: 10.1007/s00371-016-1289-x. URL: https://doi.org/10.1007/s00371-016-1289-x 2.

[Wil08] WILLIAMS, BRENT WARREN. "Fluid surface reconstruction from particles". PhD thesis. University of British Columbia, 2008 2.

[YFM*22] YUAN, GANZHANGQIN, FU, QIANCHENG, MI, ZHENXING, et al. "Ssrnet: Scalable 3d surface reconstruction network". *IEEE Transactions on Visualization and Computer Graphics* (2022) 3.

Fig	Ours	Pahi	Houdini
10	13.5	33.4	11.1
15	9.2	32.7	8.1
13	32.2	80.2	16.3
14	10.5	2.7	7.4
16	13.1	119.3	13.3
12	3.0	0.6	4.4
9	3.4212	1.7796	3.1667
geomean	9.2	11.7	8.0

Figure 18: *Reconstruction times (in minutes) for each of the fluid simulation reconstructions along with the geometric mean ("geomean").*

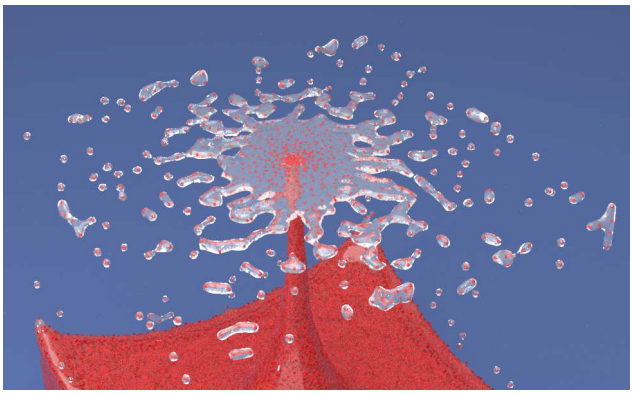


Figure 19: *Detail of splash showing particles along with the reconstructed surface.*

[YPN*22] YU, ZEHAO, PENG, SONGYOU, NIEMEYER, MICHAEL, et al. "Monosdf: Exploring Monocular Geometric Cues for Neural Implicit Surface Reconstruction". *Advances in neural information processing systems* 35 (2022), 25018–25032 3, 4.

[YT13] YU, JIHUN and TURK, GREG. "Reconstructing Surfaces of Particle-Based Fluids Using Anisotropic Kernels". *ACM Transactions on Graphics (TOG)* 32.1 (2013), 1–12 1, 2, 7.

[YWTY12] YU, JIHUN, WOJTAN, CHRIS, TURK, GREG, and YAP, CHEE. "Explicit mesh surfaces for particle based fluids". *Computer graphics forum*. Vol. 31. 2pt4. Wiley Online Library. 2012, 815–824 2.

[ZB05] ZHU, Y. and BRIDSON, R. "Animating sand as a fluid". *ACM Trans Graph* 24.3 (2005), 965–972 1, 2.

[ZPVBG01] ZWICKER, MATTHIAS, PFISTER, HANSPETER, VAN BAAR, JEROEN, and GROSS, MARKUS. "Surface splatting". *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 2001, 371–378 2.

[ZYL*22] ZHANG, JINGYANG, YAO, YAO, LI, SHIWEI, et al. "Critical regularizations for neural surface reconstruction in the wild". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, 6270–6279 3, 5.