GARM-LS: A Gradient-Augmented Reference-Map Method for Level-Set Fluid Simulation

XINGQIAO LI*, School of IST & National Key Lab. of AGI, Peking University, China
XINGYU NI*, School of CS & National Key Lab. of AGI, Peking University, China
BO ZHU, Georgia Institute of Technology & Dartmouth College, United States of America
BIN WANG†, Beijing Institute for General Artificial Intelligence, China
BAOQUAN CHEN†, School of IST & National Key Lab. of AGI, Peking University, China

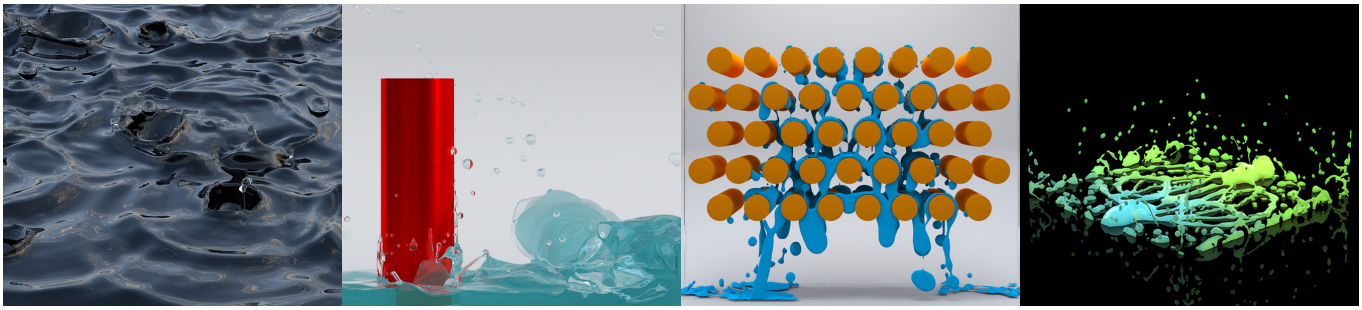

Fig. 1. Our new GARM-LS framework allows for highly detailed fluid simulation with surface tension on a relatively coarse grid. (Far Left) Rain drops falling into the pond. (Middle Left) Flow from a breaking dam. (Middle Right) A water ball falling through a Galton board. (Far Right) Splashes from a collision of two droplets.

This paper presents a novel level-set method that combines gradient augmentation and reference mapping to enable high-fidelity interface tracking and surface tension flow simulation, preserving small-scale volumes and interface features comparable to the grid size. At the center of our approach is a novel reference-map algorithm to concurrently convect level-set values and gradients, both of which are crucial for reconstructing a dynamic surface exhibiting small-scale volumes. In addition, we develop a full pipeline for the new level-set scheme by incorporating a novel extrapolation algorithm and an enhanced reinitialization procedure into our reference-map method. We test our algorithm by simulating complex surface tension flow phenomena such as raindrop collision, merging, and splashing. We also showcase the efficacy of our approach by performing validation tests and comparing it to a broad range of existing level-set algorithms.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; • **Applied computing** → *Physics*.

*joint first authors
†corresponding authors

Authors' addresses: Xingqiao Li, lixingqiao@pku.edu.cn, School of IST & National Key Lab. of AGI, Peking University, China; Xingyu Ni, nixy@pku.edu.cn, School of CS & National Key Lab. of AGI, Peking University, China; Bo Zhu, bo.zhu@gatech.edu, Georgia Institute of Technology & Dartmouth College, United States of America; Bin Wang, binwangbuaa@gmail.com, Beijing Institute for General Artificial Intelligence, China; Baoquan Chen, baoquan@pku.edu.cn, School of IST & National Key Lab. of AGI, Peking University, China.

0730-0301/2023/12-ART192 $15.00
https://doi.org/10.1145/3618377

Additional Key Words and Phrases: level-set methods, reference maps, gradient augmentation

ACM Reference Format:
Xingqiao Li, Xingyu Ni, Bo Zhu, Bin Wang, and Baoquan Chen. 2023. GARM-LS: A Gradient-Augmented Reference-Map Method for Level-Set Fluid Simulation. *ACM Trans. Graph.* 42, 6, Article 192 (December 2023), 20 pages. https://doi.org/10.1145/3618377

1 INTRODUCTION

Level-set methods play a fundamental role in simulating free-surface flow. By representing the fluid volume with a signed distance function, level-set methods can simultaneously capture the fluid's topological transition and interfacial dynamics on a background grid. However, traditional level-set methods suffer from two inherent weaknesses in representing and preserving small features. When a fluid volume gets small, especially when the feature size is comparable to the grid size, a traditional level-set function cannot accurately characterize its local geometry due to the linear assumption of the signed distance within a cell. On the other hand, there is no mechanistic guarantee that a level set can preserve its volume, either locally or globally, when advecting in a divergence-free velocity field. Small volumes, such as droplets and thin sheets, smear out quickly due to this inherent volume loss during advection.

The fluid simulation community has made progress in solving these two problems on different fronts. For one, high-resolution level-set surface tracking is combined with a low-resolution fluid solver and filters (e.g., see [Bojsen-Hansen and Wojtan 2013; Goldade et al. 2016; Kim et al. 2009]) to simulate detailed interface dynamics at an affordable cost. For another, a level set is enhanced by high-order techniques such as additional gradient fields [Nave et al. 2010] or

local polynomials [Saye 2014; Song et al. 2005] to track and preserve its volume in a variational way. Moreover, a level set is coupled with or even locally replaced by other geometric representations to characterize small-scale features around its interface. Examples of these representations include particles [Enright et al. 2002a, 2005; Losasso et al. 2004], meshes [Wojtan et al. 2011, 2009], and volume of fluid (VOF) [Chentanez and Müller 2014; Sussman and Puckett 2000]. Despite these inspiring progresses that has been made in computer graphics and computational physics, to the best of our knowledge, the existing level-set methods still have difficulties in robustly tracking and preserving very small fluid volumes for a long period, such as small fluid droplets, thin sheets, and filaments. This limitation consequently hinders the simulation of many visually appealing interfacial phenomena observed in our daily life (e.g., raindrops splashing on a puddle).

We propose a new high-order level-set method, named GARM-LS (Gradient-Augmented Reference Map for Level Set), to better address the aforementioned challenges (Fig. 1). Specifically, we intend to simulate interfacial fluid whose dynamics is governed by small volumes and strong surface tension. Our level-set method can robustly track abundant small-scale fluid volumes degenerated to the size of a single grid cell and solve their interfacial physics. Our method is purely Eulerian, without leveraging any additional Lagrangian structures to augment material tracking or interface representation. We showcase the capability of the method by simulating surface tension flow that exhibits rich interfacial details on a comparatively small-scale grid resolution. In essence, our level-set method combines two emerging numerical techniques in computational physics and computer graphics — *gradient augmentation* and *reference maps* — to achieve fourth-order accurate and robust interface tracking for simulating complex interfacial processes. However, simply integrating these two cutting-edge techniques into a fluid solver is insufficient to obtain satisfactory simulation results. To address this issue, we carefully redesigned every key stage of a conventional level-set pipeline, including *advection*, *interpolation*, *extrapolation*, and *reinitialization*, to devise a novel, full-stack interface tracking framework, ultimately enable highly accurate numerical simulations of strong surface-tension flow with rich interfacial details.

The overall technical contributions of our method include

- A unified approach that couples the level-set method with flow-map and gradient-augmentation techniques,
- A novel reference map extrapolation algorithm for plugging reference maps into narrow-band fluid surfaces, and
- A practical and robust algorithm to reinitialize gradient-augmented level sets.

2 RELATED WORK

Level-set methods. Since the pioneering work of Foster and Fedkiw [2001], much literature in the graphics community has paid attention to free-surface fluid simulation, which puts forward high requirements of surface tracking. The level-set methods [Osher and Sethian 1988] dominate the existing studies [Enright et al. 2002b; Hong et al. 2007; Kim et al. 2013; Losasso et al. 2006] from an Eulerian perspective. Compared to Lagrangian surface-tracking methods, such as those used in PIC/FLIP [Zhu and Bridson 2005], MPM [Tampubolon et al. 2017], and PBD [Macklin and Müller 2013], a level-set method shows its smoothness and consistency, both in tracking and mesh reconstruction of surfaces, which is required by simulating interfacial phenomena [Ni et al. 2020; Zheng et al. 2015] driven by surface tension. However, unlike Lagrangian tracking, level-set methods may suffer from volume loss. For this reason, there are schemes proposed to reduce the volume dissipation of the level-set method. Among these, the attempts to couple it with volume-of-fluid (VOF) methods [Chentanez and Müller 2014; Mullen et al. 2007; Sussman and Puckett 2000] have gained great effect, though numerical diffusion (resulting from first-order accuracy) and instability remain as limitations. Combining high-resolution level-set surface tracking with a lower-resolution grid-based fluid solver [Bojsen-Hansen and Wojtan 2013; Goldade et al. 2016; Kim et al. 2009] emerged as an attractive alternative to obtain visually appealing flow details. These approaches typically relied on employing additional filters to align the interface tracker and the background solver, and they have no internal mechanism to ensure high-order convergence. Another category relies on introducing auxiliary Lagrangian data structures to correct the interface, including the Lagrangian level-set [Hieber and Koumoutsakos 2005], Lagrangian VOF [Karnakov et al. 2020], and many hybrid particle-grid methods [Chen et al. 2021; Hyde et al. 2020; Saye 2014; Wang et al. 2020]. A typical example is the particle level set method (PLS) [Enright et al. 2002a, 2005; Losasso et al. 2004], which tracks abundant particle samples on both sides of the interface and uses these particles to indicate sides and correct the local level-set value. These particles, especially the escaped ones, can be further used to model visually appealing spray dynamics for large-scale water simulations [Losasso et al. 2008]. These hybrid representations often require the reconstruction or modification of the surface with respect to the particles, making it difficult to preserve a smooth surface, especially when the number of particles is limited. Last but not least, high-order interpolation and advection schemes, including CIP [Song et al. 2005], BFECC [Kim et al. 2005], modified MacCormack [Selle et al. 2008], and gradient augmentation [Nave et al. 2010], are developed both in the fields of computational physics and computer graphics, which internally improve the tracking ability of a level-set function.

Reference-map methods. Originated from the work of Wiggert and Wylie [1976], reference-map methods (also known as methods of characteristic mapping) have been consistently receiving attention. Semi-Lagrangian advection, which can be regarded as utilizing characteristic mapping during one time step, first became popular in atmospheric science [Staniforth and Côté 1991] and has been widely used in graphics for fluid simulation since Stam [1999]. Beginning with Hachisuka [2005] and Tessendorf and Pelfrey [2011], in recent years, many studies have been devoted to long-term mapping in flows [Nabizadeh et al. 2022; Qu et al. 2019; Sato et al. 2017]. Its efficacy in tracking and preserving flow features, particularly vortical structures, and solving solid-fluid interaction has been demonstrated in the recent literature. However, when it comes to free-surface fluid, reference-map methods have not fully realized their potential in facilitating simulations with complex interfacial physics. In the field of computational physics, Theillard et al. [2019;

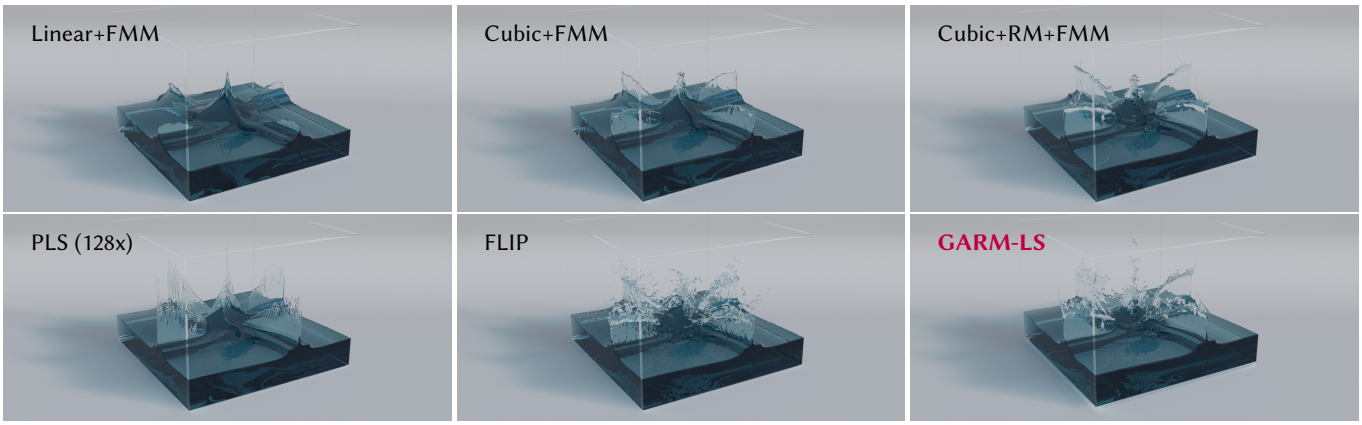

Fig. 2. Comparisons of a 4-Way Dam Break simulation among different interface-tracking methods at frame 50. Our algorithm can make splashes without aid of particles, while the fluid thin sheet is well preserved.

2021] were inspired by the coupled level-set and reference-map method (CLSRM) in the context of solid mechanics [Pons et al. 2006]. They only transport the reference map and use it to reconstruct the level set for simulating fluids, in which volume preservation is demonstrated in relatively simple scenes. Mercier et al. [2020] further combined gradient augmentation with characteristic mapping to achieve a high-order advection scheme for level sets, but the omission of extrapolation and reinitialization steps made it unavailable for long-period simulations when dramatic topological changes are present. In the community of computer graphics, Narita and Ando [2022] leveraged characteristic mapping for free-surface fluid, employing multiple localized tiled characteristic maps to mitigate volume loss introduced by frequent global reinitialization. They demonstrated their method retains small-scale splashes and waves that are quickly smeared out in a traditional level-set method. Our method differs from that of Narita and Ando [2022] in terms of its gradient-augmentation on the reference map, reinitialization, and extrapolation schemes, establishing a set of high-order algorithms to address the accuracy issue of characteristic mapping.

3 BACKGROUND

We will briefly review the background of level-set and reference-map methods in solving free-surface incompressible flow. Table 1 lists physical quantities involved in the following sections.

Free-Surface Flow. We model free-surface flow by solving incompressible Navier–Stokes equations:

$$\begin{cases} \rho\left(\dfrac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\boldsymbol{\nabla}\boldsymbol{u}\right) = -\boldsymbol{\nabla}p + \mu\boldsymbol{\nabla}^2\boldsymbol{u} + \rho\boldsymbol{g} - \delta_{\partial\Omega}\sigma\kappa\hat{\boldsymbol{n}}, & (1) \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} = 0. & (2) \end{cases}$$

Here, $\boldsymbol{u}$ is velocity; p is pressure; ρ, μ, $\boldsymbol{g}$, and σ denote density, dynamic viscosity, gravity, and surface tension, respectively. A generalized Dirac delta function $\delta_{\partial\Omega}(\boldsymbol{x})$ is introduced so that surface tension exists on the free surface $\partial\Omega$ only.

Level Sets. We use level sets [Osher and Sethian 1988] to track the fluid surface. The level set function φ is defined in the entire domain that satisfies

$$\varphi(\boldsymbol{x},t)\begin{cases} <0, & \boldsymbol{x}\in\Omega, \\ =0, & \boldsymbol{x}\in\partial\Omega, \\ >0, & \boldsymbol{x}\notin\Omega\cup\partial\Omega, \end{cases} \quad (3)$$

where Ω stands for the space occupied by the fluid and $\partial\Omega$ denotes its boundary. Since φ equals to 0 if and only if $\boldsymbol{x}$ is on $\partial\Omega$, we state that the liquid surface is implicitly captured by the 0-level set. For any point on the surface, $\hat{\boldsymbol{n}} = \boldsymbol{\nabla}\varphi/|\boldsymbol{\nabla}\varphi|$ coincides with the unit (outward) normal vector of the surface,with the mean curvature calculated by $\kappa = \boldsymbol{\nabla}\cdot\hat{\boldsymbol{n}} = \boldsymbol{\nabla}\cdot(\boldsymbol{\nabla}\varphi/|\boldsymbol{\nabla}\varphi|)$. The reinitialization step converts φ into a signed distance field (SDF) by solving an eikonal equation $\partial\varphi/\partial\tau + \mathrm{sgn}(\varphi)(|\boldsymbol{\nabla}\varphi| - 1) = 0$ over pseudo time τ until a steady state is reached [Sussman et al. 1994]. A conventional and robust way for solving this equation is the fast-marching method (FMM) [Sethian 1996], inspired by Dijkstra's algorithm.

Table 1. Notations used through the following sections.

Notation	Meaning
ρ	Density of fluid
p	Pressure of fluid
$\boldsymbol{u}$	Velocity of fluid
μ	Dynamic viscosity of fluid
σ	Surface tension coefficient of fluid
Ω	Fluid region
$\partial\Omega$	Fluid surface
κ	Mean curvature of fluid surface
$\hat{\boldsymbol{n}}$	Normal of fluid surface
φ	Level-set function
ξ	Reference-map function
$\mathcal{B}$	Back-trace mapping function
$\mathcal{H}$	Hermite interpolation
$\mathcal{L}$	Bi-linear or tri-linear interpolation
Ω_a	Advection region
Ω_e	Extended region

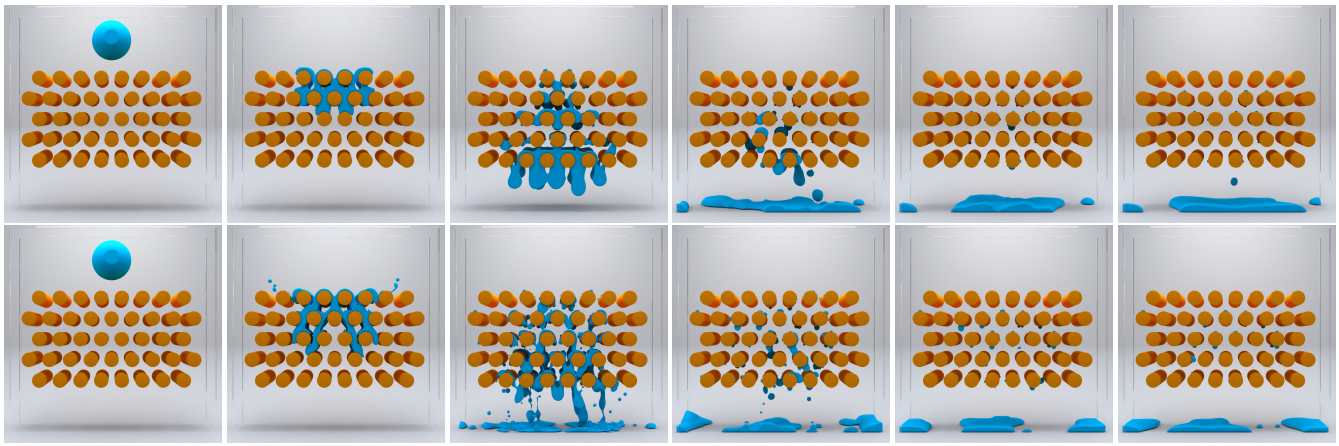

Fig. 3. Water ball falling through a Galton board. The first row is simulated with high surface tension while the second row is simulated with low surface tension. Frame numbers from left to right: 0, 35, 100, 200, 300, 400.

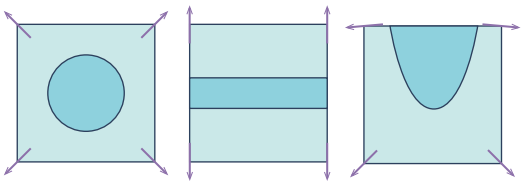

Fig. 4. A gradient-augmented level-set function, with arrows indicating gradients here, is capable of carrying sub-cell features, in particular small droplets and thin films.

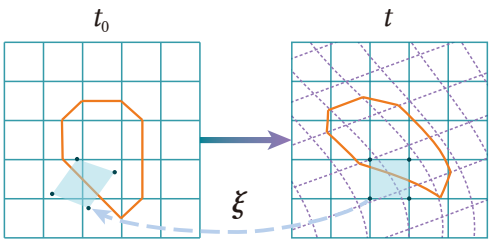

Fig. 5. As time flows, the regular grid is deformed as well as the liquid surface. We can track the deformed surface by combining the original level-set surface with the reference map.

Gradient-Augmented Level Sets (GALS). A gradient-augmented level-set method (e.g., see the work of Nave et al. [2010]) maintains the level set's gradient $\nabla\varphi$ as an auxiliary field and uses it to enhance the level-set function's local expressiveness with high-order interpolation. As shown in Fig. 4, with the help of a fourth-order Hermite interpolation scheme (§A), sub-cell features such as small droplets and thin films can be preserved during their advection. To reinitialize a gradient-augmented level set, quasi-Newton methods [Anumolu and Trujillo 2013; Chopp 2001; Saye 2014] can be used to update φ and $\nabla\varphi$ around the interface by iteratively solving for the nearest point on $\partial\Omega$.

Reference Maps. As shown in Fig. 5, a reference map $\boldsymbol{\xi}(\boldsymbol{x},t) : \Omega^t \rightarrow \Omega^{t_0}$ $(t \geq t_0)$ is defined as a vector field indicating the initial (or reference) location of the fluid element currently occupying position $\boldsymbol{x}$, with t_0 and t denoting the initial and current times. For conciseness, we rewrite $\boldsymbol{\xi}(\boldsymbol{x},t)$ and $\varphi(\boldsymbol{x},t_0)$ as $\boldsymbol{\xi}(\boldsymbol{x})$ and $\varphi^*(\boldsymbol{x})$ in short. A reference map can carry out the advection of any material points (i.e., $\mathrm{D}(\cdot)/\mathrm{D}t = 0$) by establishing mappings between locations in different frames based on the flow field.

Level Sets on Reference Maps. The reference map is advected instead of the level set. This reference map is then used to reconstruct the new level-set field at time t based on mapping values from the level-set field at time 0 by $\varphi(\boldsymbol{x},t) = \varphi(\boldsymbol{\xi}(\boldsymbol{x},t),t_0)$. It is noteworthy that a conventional SDF-based level set only satisfies $\mathrm{D}\varphi/\mathrm{D}t = 0$ on the interface ($\varphi = 0$), which means the reconstructed level-set value may deviate from an SDF. Practically, we accept slight deviation and obtain level-set values by $\varphi(\boldsymbol{x}) = \varphi^*(\boldsymbol{\xi}(\boldsymbol{x}))$ with confidence in this situation. However, for the purposes of smoothness and continuity, if φ is deformed excessively, φ^* needs to be reinitialized by the redistanced φ. Besides, the calculation of $\varphi^*(\boldsymbol{\xi}(\boldsymbol{x}))$ relies on successively interpolating $\boldsymbol{\xi}$ and φ^* on a background discretization (e.g., a Cartesian grid).

4 THE GARM-LS METHOD

4.1 Gradient Augmentation on Reference Maps

We incorporate gradient augmentation and a reference map into a unified pipeline to facilitate a level-set method. Distinguished from previous literature, which focused on addressing the feature representation (with gradient augmentation, e.g., see the work of Nave et al. [2010]) or volume preservation (with reference maps, e.g., see the work of Theillard [2021]) separately. Our proposed approach combines these two techniques. Our central idea is to (1) accurately convect the reference-map gradients and (2) leverage the reference-map gradients to reconstruct the level-set gradients and its function values. Next, we provide a brief motivational analysis to justify this design decision followed by a detailed description of the two steps.

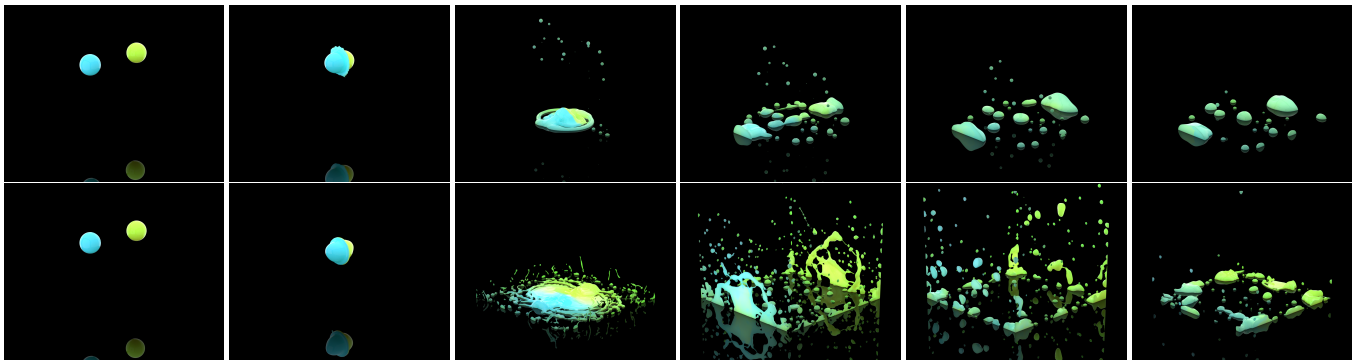

Fig. 6. Two droplets collide in opposite directions. The two simulations with identical initial configuration, but different surface tension coefficients are demonstrated in each individual rows. Specifically, the surface tension of the top line is twice that of the bottom line. Frame numbers from left to right: 0, 18 (40 for the bottom), 100, 150, 200, 300.

4.1.1 Motivation. A natural and reasonable way to combine level sets and reference maps is to store the initial level set on the background grid and use its values to reconstruct the level sets in future frames based on an evolving reference map $\boldsymbol{\xi}$. In this way we can combine a conventional interpolation scheme and the reference map technique to produce a modified advection scheme [Bellotti and Theillard 2019; Narita and Ando 2022]. To improve its accuracy using high-order information, we can use a gradient-augmented level set for the initial frame, by storing additional $\varphi^*(\boldsymbol{\xi})$, and conduct Hermite interpolation for the future level set reconstruction. However, we want to emphasize that simply augmenting $\varphi^*(\boldsymbol{\xi})$ with its gradients, without maintaining a high-order reference map at the same time, is insufficient to obtain a high-order advected level set. This can be seen by performing a simple numerical error analysis on the mapping function: We assume that $\varphi^*(\boldsymbol{\xi})$ and $\boldsymbol{\xi}(\boldsymbol{x})$ are respectively approximated by $\tilde{\varphi}^*(\boldsymbol{\xi})$ and $\tilde{\boldsymbol{\xi}}(\boldsymbol{x})$. Then we can obtain the approximation of $\varphi(\boldsymbol{x})$ as a composite function:

$$\begin{aligned}\tilde{\varphi}(\boldsymbol{x}) = \tilde{\varphi}^*\left(\tilde{\boldsymbol{\xi}}(\boldsymbol{x})\right) &= \varphi^*\big(\boldsymbol{\xi}(\boldsymbol{x}) + \overbrace{O(\Delta x^p)}^{\text{error}}\big) + \overbrace{O(\Delta x^q)}^{\text{error}} \\ &= \varphi(\boldsymbol{x}) + O\left(\Delta x^{\min(p,q)}\right), \end{aligned} \tag{4}$$

with Δx denoting the grid spacing, which is derived from $\varphi(\boldsymbol{x}) = \varphi^*(\boldsymbol{\xi}(\boldsymbol{x}))$. It suggests that the accumulated errors stem from both the reference-map and level-set interpolation. A more rigorous proof can be found in the work of Mercier et al. [2020]. Because our gradient augmentation for the initial level set is fourth-order accurate, we want to match this order in the reference map. To this end, we choose to advect the reference map gradient $\boldsymbol{\nabla\xi}$ (the derivative of the mapping from the current frame to the first frame w.r.t. the coordinates in the current frame) using a fourth-order accurate advection scheme.

4.1.2 Updating the reference map and its gradients. Similar to the GALS method [Nave et al. 2010], we write the advection equations for $\boldsymbol{\xi}$ and $\boldsymbol{\nabla\xi}$ as follows:

$$\begin{cases} \dfrac{\mathrm{D}\boldsymbol{\xi}}{\mathrm{D}t} = \boldsymbol{0}, & (5) \\ \dfrac{\mathrm{D}\boldsymbol{\nabla\xi}}{\mathrm{D}t} = -\boldsymbol{\nabla u}\cdot\boldsymbol{\nabla\xi}. & (6) \end{cases}$$

Instead of solving Eq. (6) by explicitly updating the velocity gradients and accumulating their stretching effects in the time integration, we choose to solve them from a reference-mapping perspective. As stated as the *superconsistency* condition in [Nave et al. 2010], Eqs. (5) and (6) can be rewritten as a reference-mapping problem from frame $t - \Delta t$ to frame t:

$$\begin{cases} \boldsymbol{\xi}(\boldsymbol{x}_i, t) = \boldsymbol{\xi}(\mathcal{B}(\boldsymbol{x}_i, \boldsymbol{u}, \Delta t), t - \Delta t), & (7) \\ \boldsymbol{\nabla\xi}(\boldsymbol{x}_i, t) = \boldsymbol{\nabla}\mathcal{B}(\boldsymbol{x}_i, \boldsymbol{u}, \Delta t)\cdot\boldsymbol{\nabla\xi}(\mathcal{B}(\boldsymbol{x}_i, \boldsymbol{u}, \Delta t), t - \Delta t), & (8) \end{cases}$$

where the process $\mathcal{B}(\boldsymbol{x}, \boldsymbol{u}, \Delta t)$ returns the back-traced point of $\boldsymbol{x}$ on a flow distribution $\boldsymbol{u}$. In our numerical practice, we adopt a third-order total variation diminishing Runge–Kutta (TVD-RK3) scheme to back-trace each grid point from frame t to frame $t - \Delta t$ (Alg. 2). For a better comprehension of the superconsistency, a first-order scheme is provided here. See Alg. 1. We point out that $\mathcal{B}$ plays as the role of a reference map.

ALGORITHM 1: First-Order Superconsistent Semi-Lagrangian Advection of the Gradient-Augmented Reference Map

Input: the reference map $\boldsymbol{\xi}$ and its gradient $\boldsymbol{\nabla\xi}$, the velocity field $\boldsymbol{u}$, and the time step Δt.
Output: the advected fields $\boldsymbol{\xi}'$ and $\boldsymbol{\nabla\xi}'$.
Acquire $\boldsymbol{\nabla u}$ by finite differences of $\boldsymbol{u}$;
for *every cell center* $\boldsymbol{x}_i$ **do**
  $\boldsymbol{x}^{(0)} \leftarrow \boldsymbol{x}_i$;
  $\boldsymbol{u}^{(0)}, \boldsymbol{\nabla u}^{(0)} \leftarrow \mathcal{L}(\boldsymbol{u}, \boldsymbol{x}^{(0)}), \mathcal{L}(\boldsymbol{\nabla u}, \boldsymbol{x}^{(0)})$;
  /* one step back-tracking */
  $\boldsymbol{x}^{(1)} \leftarrow \boldsymbol{x}^{(0)} + \Delta t\, \boldsymbol{u}^{(0)}$;
  $\boldsymbol{\nabla x}^{(1)} \leftarrow \boldsymbol{I} + \Delta t\, \boldsymbol{\nabla u}^{(0)}$;
  /* Hermite interpolation */
  $\boldsymbol{\xi}_i', \boldsymbol{\nabla\xi}_i' \leftarrow \mathcal{H}(\boldsymbol{\xi}, \boldsymbol{\nabla\xi}, \boldsymbol{x}^{(1)})$;
  $\boldsymbol{\nabla\xi}_i' \leftarrow \boldsymbol{\nabla x}^{(1)} \cdot \boldsymbol{\nabla\xi}_i'$;

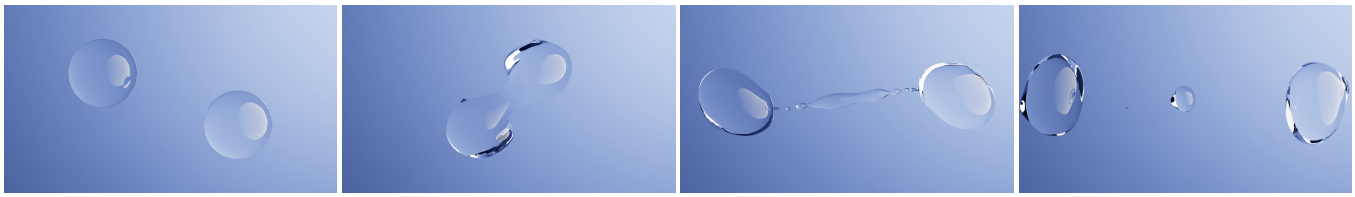

Fig. 7. Two droplets collide in diagonal directions. Frame numbers from left to right: 0, 125, 369, 450.

As a side note, we want to bring to the readers' attention of the geometric connection between Eqs. (7), (8) and the characteristic mapping scheme of impulse in the work of Cortez [1995] (Proposition 1 and Proposition 3) and Nabizadeh et al. [2022] (Eqs. (2) and (3)): the gradient of a reference map is mathematically identical to an impulse field advected in an incompressible flow field.

ALGORITHM 2: High-Order Superconsistent Semi-Lagrangian Advection of the Gradient-Augmented Reference Map

Input: the reference map $\boldsymbol{\xi}$ and its gradient $\nabla\boldsymbol{\xi}$, the velocity field $\boldsymbol{u}$, and the time step Δt.
Output: the advected fields $\boldsymbol{\xi}'$ and $\nabla\boldsymbol{\xi}'$.
Acquire $\nabla\boldsymbol{u}$ by finite differences of $\boldsymbol{u}$;
for *every cell center* $\boldsymbol{x}_i$ **do**
  $\boldsymbol{x}^{(0)} \leftarrow \boldsymbol{x}_i$;
  $\boldsymbol{u}^{(0)}, \nabla\boldsymbol{u}^{(0)} \leftarrow \mathcal{L}(\boldsymbol{u}, \boldsymbol{x}^{(0)}), \mathcal{L}(\nabla\boldsymbol{u}, \boldsymbol{x}^{(0)})$;
  /* TVD-RK3 back tracking */
  $\boldsymbol{x}^{(1)} \leftarrow \boldsymbol{x}^{(0)} + \Delta t\, \boldsymbol{u}^{(0)}$;
  $\nabla\boldsymbol{x}^{(1)} \leftarrow \boldsymbol{I} + \Delta t\, \nabla\boldsymbol{u}^{(0)}$;
  $\boldsymbol{u}^{(1)}, \nabla\boldsymbol{u}^{(1)} \leftarrow \mathcal{L}(\boldsymbol{u}, \boldsymbol{x}^{(1)}), \mathcal{L}(\nabla\boldsymbol{u}, \boldsymbol{x}^{(1)})$;
  $\boldsymbol{x}^{(2)} \leftarrow \boldsymbol{x}^{(0)} + \frac{\Delta t}{4}\left(\boldsymbol{u}^{(0)} + \boldsymbol{u}^{(1)}\right)$;
  $\nabla\boldsymbol{x}^{(2)} \leftarrow \boldsymbol{I} + \frac{\Delta t}{4}\left(\nabla\boldsymbol{u}^{(0)} + \nabla\boldsymbol{x}^{(1)} \cdot \nabla\boldsymbol{u}^{(1)}\right)$;
  $\boldsymbol{u}^{(2)}, \nabla\boldsymbol{u}^{(2)} \leftarrow \mathcal{L}(\boldsymbol{u}, \boldsymbol{x}^{(2)}), \mathcal{L}(\nabla\boldsymbol{u}, \boldsymbol{x}^{(2)})$;
  $\boldsymbol{x}^{(3)} \leftarrow \boldsymbol{x}^{(0)} + \frac{\Delta t}{6}\left(\boldsymbol{u}^{(0)} + \boldsymbol{u}^{(1)} + 4\boldsymbol{u}^{(2)}\right)$;
  $\nabla\boldsymbol{x}^{(3)} \leftarrow \boldsymbol{I} + \frac{\Delta t}{6}\left(\nabla\boldsymbol{u}^{(0)} + \nabla\boldsymbol{x}^{(1)} \cdot \nabla\boldsymbol{u}^{(1)} + 4\nabla\boldsymbol{x}^{(2)} \cdot \nabla\boldsymbol{u}^{(2)}\right)$;
  /* Hermite interpolation */
  $\xi_i', \nabla\xi_i' \leftarrow \mathcal{H}(\xi, \nabla\xi, \boldsymbol{x}^{(3)})$;
  $\nabla\xi_i' \leftarrow \nabla\boldsymbol{x}^{(3)} \cdot \nabla\xi_i'$;

Alg. 2 shows specific steps of the proposed high-order semi-Lagrangian advection. Here the velocity field is assumed to be invariant during advection, and $\boldsymbol{I}$ stands for the second-order unit tensor. The procedures that return the reference map and its gradient as a pair by bicubic/tricubic Hermite interpolation are denoted $\mathcal{H}(\xi, \nabla\xi, \boldsymbol{x})$. The procedures that return velocity and its gradient by bilinear/trilinear interpolation are denoted $\mathcal{L}(\boldsymbol{u}, \boldsymbol{x})$ and $\mathcal{L}(\nabla\boldsymbol{u}, \boldsymbol{x})$.

4.1.3 Updating the level set and its gradients. Following the idea of Eqs. (7) and (8), we can restore the current level-set function and its gradients by

$$\begin{cases} \varphi(\boldsymbol{x}) = \varphi^*(\xi(\boldsymbol{x})), & (9) \\ \nabla\varphi(\boldsymbol{x}) = \nabla\varphi^*(\xi) \cdot \nabla\xi(\boldsymbol{x}). & (10) \end{cases}$$

Here, the first equation is used to generate the current level-set value, and the second equation is useful for evaluating the normal and curvature at a given point. Both equations take the updated reference map and use it to map the level set and its gradients from the reference frame. Again, we want to highlight the geometric connection between Eqs. (7), (8) and Eqs. (9), (10), which tackles different fields and their gradients both from a reference-mapping perspective. The difference lies in that we use a larger time interval $(t - t_0)$ (typically around $10\Delta t$ in our implementation) to back-trace φ and $\nabla\varphi$.

4.2 Extrapolation

4.2.1 Problem: reference-map extrapolation. In a traditional level-set advection procedure, a narrow band of grid cells outside the zero-level set is maintained to avoid reading any incorrect velocity values outside the fluid domain. Velocity values are extrapolated from the cells on the fluid interface to the cells in the narrow band.

In contrast to a pure level-set approach, a traditional reference map should be maintained in the full space, and the velocity values should be continuous everywhere. In a free-surface flow simulation, only those velocity values inside the narrow-band are valid, causing the result ξ and $\nabla\xi$ to be inaccurate and discontinuous. By interpolation on these values when we update the level-set values, the discontinuity can be further amplified and induce fluid creation (if the reference-mapped level set value is incorrectly negative) and annihilation (if the reference-mapped level-set value is incorrectly positive). Therefore, an extrapolation is needed to alleviate the disastrous consequences of the discontinuous reference map. We shall maintain two properties of the extrapolated reference map within the entire fluid domain plus its narrow band:

(1) ξ and $\nabla\xi$ should be continuous before an advection process;
(2) Interpolated level-set values φ should not change their sign after the extrapolation.

4.2.2 Algorithm: extended reference-map extrapolation. We devise a new reference-map extrapolation algorithm to realize this goal. Our key idea is to maintain a further extended narrow band outside the original narrow band (for velocity extrapolation) to support the advection of the reference map and its gradients. The ξ value in the extended narrow band is updated based on a constant extrapolation of ξ from the boundary of the original narrow band. The $\nabla\xi$ value can be updated using the finite difference of the extrapolated ξ in the extended narrow band. In practice, we find extrapolating only ξ but not $\nabla\xi$ is good enough to maintain a robust reference map for level-set advection.

We illustrate the key steps of our reference-map extrapolation algorithm in Fig. 9. We denote the fluid region as Ω, the fluid region plus its narrow band as Ω_a (which is named advection region), and

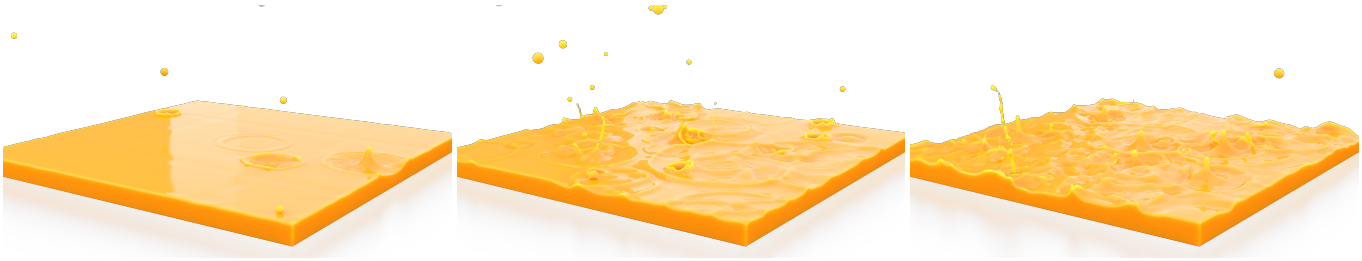

Fig. 8. Dropping of orange juice. Frame numbers from left to right: 175, 275, 375.

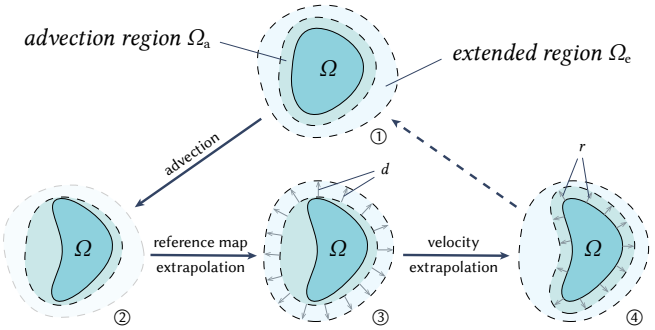

Fig. 9. The extrapolation process.

the advection region plus the extended narrow band as Ω_e (which is named extended region). In each time step, our algorithm includes three key steps. ①–② We perform level-set advection within Ω_a to update the fluid domain; then ②–③ we update the extended narrow band outside Ω_a to get the new Ω_e and perform reference-map extrapolation in this extended narrow band; ③–④ we update the narrow band outside the fluid domain to get the new Ω_a and perform velocity extrapolation in the new narrow band. We employ a CFL condition on the extrapolation band size d and r as $d > \Delta t|\boldsymbol{u}_{\max}|$ and $r > \Delta t|\boldsymbol{u}_{\max}|$ respectively, with $|\boldsymbol{u}_{\max}|$ denoting the estimated maximum speed of fluid. We summarize the algorithm pseudocode in Alg. 3.

ALGORITHM 3: Extended Extrapolation for the Reference Map

Input: the reference map $\boldsymbol{\xi}$ and its gradient $\nabla\boldsymbol{\xi}$, the velocity field $\boldsymbol{u}$, the liquid region Ω, the advection region Ω_a, and the extrapolation radii r and d.
Output: the extrapolated fields $\boldsymbol{\xi}'$, $\nabla\boldsymbol{\xi}'$, and $\boldsymbol{u}'$, and the new advection region Ω_a'.

```
for every cell center x_i do
    if ξ_i ∈ Ω_a then
        ξ'_i ← ξ_i;
        ∇ξ'_i ← ∇ξ(x_i);
/* Fig. 9 ②–③ reference map extrapolation */
for every cell center x_i do
    if ξ_i ∉ Ω_a and GetDistance(x_i, Ω_a) ≤ d then
        ξ'_i ← ξ(GetClosestPosition(x_i, Ω_a));
        ∇ξ'_i ← ∇_s ξ(GetClosestPosition(x_i, Ω_a));
/* Fig. 9 ③–④ velocity extrapolation */
Ω'_a ← ∅;
for every face center x_i do
    if x_i ∈ Ω then
        u'_i ← u(x_i);
        Ω'_a ← Ω'_a ∪ GetNeighborCells(x_i);
    else if GetDistance(x_i, Ω) ≤ r then
        u'_i ← u(GetClosestPosition(x_i, Ω));
        Ω'_a ← Ω'_a ∪ GetNeighborCells(x_i);
    else
        u'_i ← 0;
```

4.3 Reinitialization

4.3.1 Motivation. A conventional level-set method performs reinitialization regularly to redistance the level-set field. This step introduces extra numerical errors because the interface's position (0-isocontour) is affected by reinitialization due to its low-order accuracy. For instance, among the popular choices for reinitialization, the fast-marching method (FMM) [Sethian 1996] is first-order accurate, and even a generally fifth-order weighted essentially non-oscillatory (WENO) scheme [Jiang and Peng 2000] is only second-order accurate around the interface, as noted by Gibou and Fedkiw [2005]. This reinitialization-induced error becomes more dominant if numerical errors introduced by other steps are reduced by high-order schemes, e.g., the reference mapping [Narita and Ando 2022]. In particular, in our setting, numerical errors stem from two sources — advection and reinitialization. After reducing the reference-map error by incorporating gradients and reference map, a natural next step to improve the algorithm's overall accuracy is to reduce the numerical errors in reinitialization.

We practice two strategies toward this goal, by (1) employing a dynamic reinitialization criterion based on measuring the reference map stretching and (2) devising an improved quasi-Newton method to stabilize the 0-level set during reinitialization. After conducting these two new strategies, our reinitialization step manifests fourth-order accuracy (as evidenced by our numerical experiments in §6.1.2), which matches our fourth-order advection step. Next, we will introduce these two steps in detail. We summarize the entire algorithm in Alg. 4.

4.3.2 Reinitialization criterion. The presence of $\boldsymbol{\xi}$ makes it possible to reduce the call of reinitialization, because the reference map allows us to check if a level-set function is over-distorted, in which

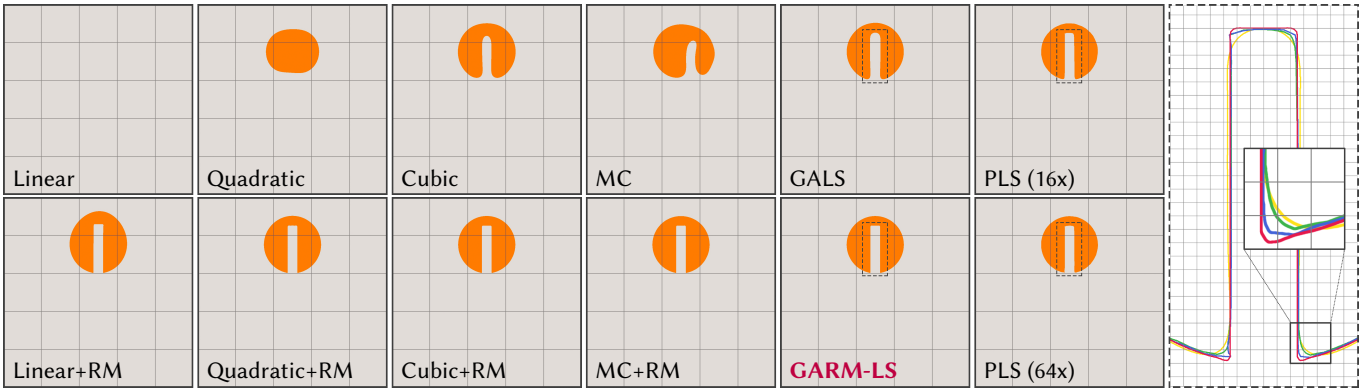

Fig. 10. Zalesak's disk (2D). A slotted circle is driven by a rotation velocity field. The left images with grey background are the simulation results when the circles perform two complete rotations ($t = 12.56$ s) with different advection algorithms annotated on the bottom left corner of each image. Every grid cell in these images represents a 20×20 block in the simulation. For a clearer comparison, the contours in the dashed line enclosed regions of GALS, GARM-LS, PLS (16x), and PLS (64x) are plot with different colors in the right most figure. A zoomed-in view around one contour corner is also provided. The grid cells on the white background coincide with those used in the simulation.

case bijectivity of the reference map is nearly lost. In 2D, lost bijectivity means that there must be one contour line of $\partial\xi/\partial x$ that intersects one contour line of $\partial\xi/\partial y$ more than once. The same conclusion applies to the three-dimensional case. Following Bellotti and Theillard [2019], we define a reinitialization criterion according to angles between partial derivatives of $\boldsymbol{\xi}$:

$$\max_{\boldsymbol{x}\in U(\partial\Omega,\varepsilon_\mathrm{r})} \max_{i,j\in[1,\mathrm{d}],i\neq j} \left(\frac{\boldsymbol{\nabla}\xi_i}{|\boldsymbol{\nabla}\xi_i|} \cdot \frac{\boldsymbol{\nabla}\xi_j}{|\boldsymbol{\nabla}\xi_j|} \right) > \cos\theta_\mathrm{crit}, \tag{11}$$

with $U(\partial\Omega, \varepsilon_\mathrm{r})$ standing for the ε_r-neighborhood of $\partial\Omega$ and d denoting the number of dimensions. Different from the cited work, which acquires these partial derivatives through numerical difference, we leverage the advected reference-map gradients to obtain a more reliable criterion. Every time when the reinitialization criterion is met, we perform a *restart* process to reinitialize the reference level-set function and then to set $\boldsymbol{\xi}$ and $\boldsymbol{\nabla}\boldsymbol{\xi}$ at every cell center $\boldsymbol{x}_i$ as follows:

$$\begin{cases} \varphi^*(\boldsymbol{\xi}) = \varphi(\boldsymbol{x}), & (12) \\ \boldsymbol{\xi}(\boldsymbol{x}_i) = \boldsymbol{x}_i, & (13) \\ \boldsymbol{\nabla}\boldsymbol{\xi}(\boldsymbol{x}_i) = \boldsymbol{I}. & (14) \end{cases}$$

4.3.3 Reinitialization algorithm. We propose a gradient-augmented reinitialization scheme based on Anumolu and Trujillo [2013], which, to the best of our knowledge, is the state-of-the-art result for gradient-augmented level-set reinitialization. For completeness, we briefly introduce their framework in the following three steps:

(1) The simulation region is divided into 2 subregions, namely Ω_i for the interfacial region and Ω_o for the rest region, where the interfacial region is chosen so that all cells inside it are close to the surface by a distance of several cell length, as illustrated in Fig. 11.
(2) A quasi-Newton method is applied to find the nearest point $\boldsymbol{x}$ on $\partial\Omega$ for every grid point $\boldsymbol{x}_i$ in Ω_i. We update $\boldsymbol{x}$ by iteratively perform a gradient-descent step and a perpendicular-moving step. The gradient-descent step finds the closet point on $\partial\Omega$ along current $\boldsymbol{\nabla}\varphi$. While, the perpendicular-moving step makes the point move perpendicularly with $\boldsymbol{\nabla}\varphi$ to further align $(\boldsymbol{x}-\boldsymbol{x}_i)$ with $\boldsymbol{\nabla}\varphi$. When converged, the found point is used to update the level-set value and gradient.
(3) Given that φ and $\boldsymbol{\nabla}\varphi$ in Ω_i are reinitialized, the level-set values are propagated from the interface to Ω_o by solving the Eikonal equation with a semi-Lagrangian advection.

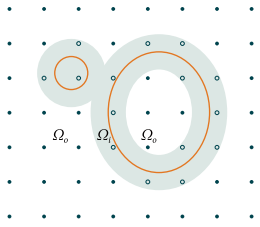

Fig. 11. An illustration for the interfacial region Ω_i and the rest region Ω_o.

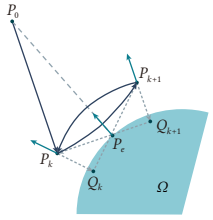

Fig. 12. An illustration of the jumping phenomenon in the original reinitialization algorithm.

We improve Anumolu and Trujillo [2013]'s scheme in three aspects including initial guess, back-and-forth trial, and fast-marching postprocess, to obtain robust results in different simulation settings. We briefly introduce each of these aspects.

Initial guess. In our method, after several time steps without a restart, the level-set function values and gradients evaluated by back-tracing the reference frame, can deviate from a signed distance field. Only those cells that are close to the interpolated surface can keep the gradient direction with enough precision. For cells being simultaneously close to different pieces of the surface, the gradient can even point to the further piece of the surface instead of the nearest. We obtain an initial guess by conducting several WENO iterations, allowing the quasi-Newton iteration to search for the nearest position on the surface. That allows a narrow band of cells around the interface to be correctly reinitialized.

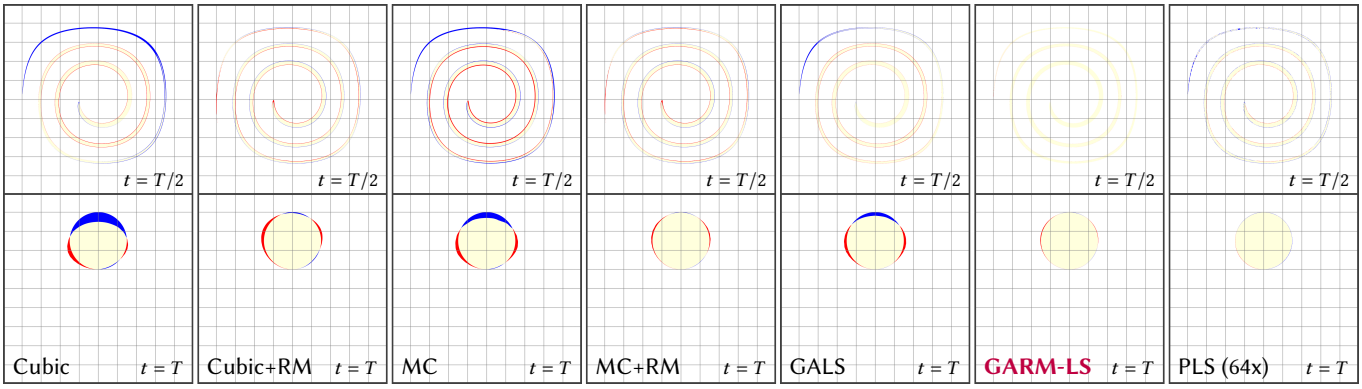

Fig. 13. LeVeque's circle (2D). The two figures in each column are generated by the same advection scheme that is annotated in the bottom left corner. Every grid cell in the figures represents a 20×20 simulation block. Given $T = 12.56$ s, we examine here the results of the algorithms at two particular time instances. The first is $t = T/2$ when the fluid is deformed to to the maximum (the top row), and the second is $t = T$ when the fluid shape is supposed to turn back into a circle (the bottom row). In each figure, we compare the simulated fluid region with the ground truth (which is generated by Cubic on a 4000×4000 grid). The areas added due to numerical errors are colored red, while the missing areas are colored blue. The light yellow area indicates the part where the ground truth overlaps the simulation result.

Back-and-forth trial. To capture small droplets whose radius of curvature is comparable to the size of one grid cell, we have to deal with the jumping phenomenon during the quasi-Newton iterations. Fig. 12 demonstrates an example of such jumping. The target cell center is denoted by P_0, and the fluid lies inside the region Ω. We expect the iteration point to converge at P_e. By some iterations, the actual iteration point goes to P_k, where the level-set function value and the gradient are almost correct. However, in the next step, the point not only goes through the gradient-descent step but also goes through the perpendicular-moving step to make the gradient point P_0. That makes the next iteration point P_{k+1} instead of Q_k, which further causes the iteration point to oscillate between P_{k+1} and P_k. We add a back-and-forth trial to detect such oscillation and choose the next iteration point to be the middle point of P_k and P_{k+1}. Our experiments show this process successfully eliminates jumping in our quasi-Newton iterations.

Fast-marching post-processing. A semi-Lagrangian propagation was used in the work of Anumolu and Trujillo [2013] to fill the rest region, requiring a non-zero gradient of the level-set function to be available in the entire rest region, which is not always satisfied in our setting. We apply a standard fast-marching method to propagate the level set from the interface to the rest region. Then, we calculate the level-set gradients with finite differences in the rest region. This change makes the propagation robust without affecting level set values near the interface.

5 TIME INTEGRATION

In each time step, we first advect the reference map, its gradient, and the velocity field according to the velocity field. After advection, we check the restart criterion and decide whether to perform a restart process. Next, we project the fluid velocity field for the conservation of volume. Finally, we extrapolate the projected velocity and reference map into the air. The full algorithm in a time step from t to $t + \Delta t$ reads as follows. For the completeness of the framework, all necessary components in general fluid simulation are included here.

ALGORITHM 4: Improved Reinitialization for GARM-LS

Input: the reference level set and its gradient φ^* and $\nabla\varphi^*$, the reference map and its gradient $\boldsymbol{\xi}$ and $\nabla\boldsymbol{\xi}$, the prescribed interfacial region Ω_i, the rest region Ω_o, and the tolerance ε.
Output: the reinitialized $\varphi^{*\prime}$ and $\nabla\varphi^{*\prime}$.
for *every cell center* $\boldsymbol{x}_i$ *in* Ω_i **do**
  $\boldsymbol{x} \leftarrow \texttt{GuessByWENO}(\boldsymbol{x}_i)$;
  $\boldsymbol{g} \leftarrow \tilde{\nabla}\varphi^*(\boldsymbol{\xi}) \cdot \nabla\boldsymbol{\xi}(\boldsymbol{x})$;
  $\Delta\boldsymbol{x}' \leftarrow 0$;
  repeat
    $\Delta\boldsymbol{x} \leftarrow -\varphi^*(\boldsymbol{\xi}(\boldsymbol{x}))\,\boldsymbol{g}/|\boldsymbol{g}|^2$;
    $\Delta\boldsymbol{x} \leftarrow \Delta\boldsymbol{x} + (\boldsymbol{x} - \boldsymbol{x}_i) - (\boldsymbol{g} \cdot (\boldsymbol{x} - \boldsymbol{x}_i))\,\boldsymbol{g}/|\boldsymbol{g}|^2$;
    if $|\Delta\boldsymbol{x} + \Delta\boldsymbol{x}'| < \varepsilon$ **then** $\Delta\boldsymbol{x} \leftarrow \Delta\boldsymbol{x}/2$;
    $\boldsymbol{x} \leftarrow \boldsymbol{x} + \Delta\boldsymbol{x}$;
    $\boldsymbol{g} \leftarrow \tilde{\nabla}\varphi^*(\boldsymbol{\xi}) \cdot \nabla\boldsymbol{\xi}(\boldsymbol{x})$;
    $\Delta\boldsymbol{x}' \leftarrow \Delta\boldsymbol{x}$;
  until $\Delta\boldsymbol{x} < \varepsilon$;
  $\varphi_i'^* \leftarrow |\boldsymbol{x}_i - \boldsymbol{x}| \cdot \operatorname{sgn}\varphi^*(\boldsymbol{\xi}(\boldsymbol{x}_i))$;
  $\nabla\varphi_i'^* \leftarrow (\boldsymbol{x}_i - \boldsymbol{x})/|\boldsymbol{x}_i - \boldsymbol{x}| \cdot \operatorname{sgn}\varphi^*(\boldsymbol{\xi}(\boldsymbol{x}_i))$;
$\varphi'^* \leftarrow \texttt{FastMarching}\,(\varphi'^*, \Omega_\mathrm{i}, \Omega_\mathrm{o})$;
for *every cell center* $\boldsymbol{x}_i \in \Omega_\mathrm{o}$ **do**
  $\nabla\varphi_i'^* \leftarrow \texttt{FiniteDifference}\,(\varphi'^*, \boldsymbol{x}_i)$;

The viscosity term is omitted in all of our demos to produce more dramatic animations. The presence of gravity and surface tension depends on the particular configuration.

(1) Advect the reference map $\boldsymbol{\xi}$, the reference map gradient $\nabla\boldsymbol{\xi}$, and the velocity field $\boldsymbol{u}$ in the advection region Ω_a with the super-consistent semi-Lagrangian method (Alg. 2). Record the level-set value on each grid cell by tracking back to the reference frame.
(2) (Optional) Add external fluid sources by unioning the level set with external sources. If the value of the level set field is changed somewhere, switch the level-set gradient to that of the external

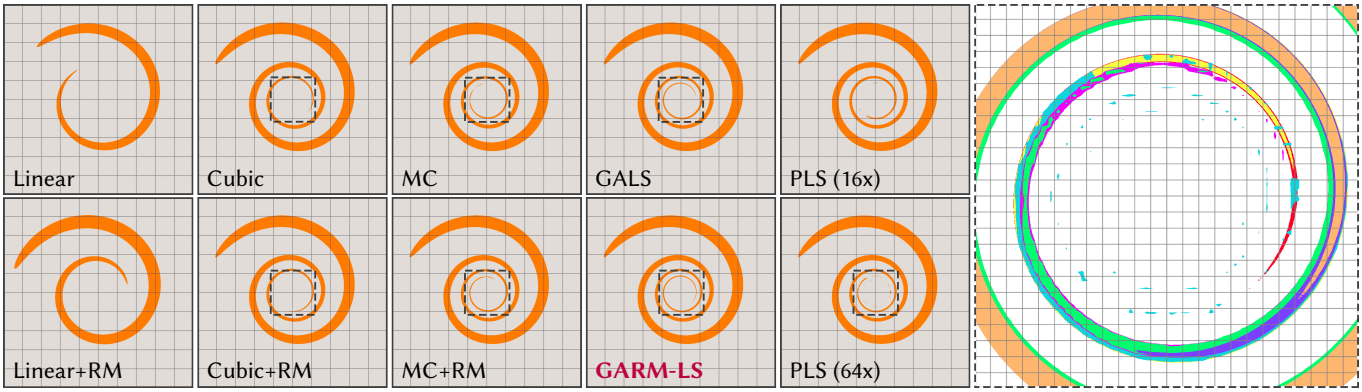

Fig. 14. Distorted circle (2D). The advection algorithm is annotated on each image with gray background showing the circle after extreme distortion ($t = 6$ s). Every grid cell in the visualization represents a 20×20 block in the simulation. Two algorithms, namely Quadratic and Quadratic+RM, are omitted here for conciseness. Quadratic and Quadratic+RM visually behaves in between Linear+RM and Cubic, with Quadratic+RM giving a longer spiral than Quadratic (see the supplementary video). In the rightmost figure with a white background, we plot results of seven algorithms together for comparison purposes. Each result corresponds to a unique color as follows: Cubic, Cubic+RM, MC, MC+RM, GALS, GALS+RM, and PLS (64x). The area drawn is framed with dotted lines in the images. There are grid cells indicating 2×2 block used in the simulation on the white background.

fluid sources, then reset the reference map and the reference map gradient there.

(3) Check whether the reinitialization criterion (11) is met. If so, the level-set function φ and its gradient $\nabla\varphi$ is reinitialized by Alg. 4, and the reference map is reset to an identity mapping (Eqs. (13) and (14)).

(4) (Optional) Apply external forces directly to velocity field, such as gravity: $\boldsymbol{u} \leftarrow \boldsymbol{u} + \Delta t \boldsymbol{g}$.

(5) Project $\boldsymbol{u}$ by solving a standard fluid Poisson's equation with boundary conditions as follows:

$$\begin{cases} \nabla \cdot \left(\boldsymbol{u} - \dfrac{\Delta t}{\rho} \nabla p \right) = 0, & \text{in the fluid region,} \quad (15a) \\ p = p_0, & \text{on the free surface,} \quad (15b) \\ \dfrac{\Delta t}{\rho} (\boldsymbol{n} \cdot \nabla p) = \boldsymbol{n} \cdot (\boldsymbol{u} - \boldsymbol{u}_{\text{solid}}), & \text{on the solid-liquid interface.} \quad (15c) \end{cases}$$

To retain the smoothness of the solid-liquid boundaries, we apply the method of Batty et al. [2007] which adds modifications to the matrix of projection solve. After solving the system for p we update the velocity by $\boldsymbol{u} \leftarrow \boldsymbol{u} - \Delta t \nabla p / \rho$.

(6) (Optional) Apply the capillary surface tension on the interface with a semi-implicit method [Zheng et al. 2006]. The linear system reads

$$\frac{\boldsymbol{u}' - \boldsymbol{u}}{\Delta t} = -\frac{\sigma}{\rho} \delta(\varphi) \left[\Delta t \nabla^2 \boldsymbol{u}' - \kappa \boldsymbol{n} - \Delta t \left(\frac{\partial^2 \boldsymbol{u}}{\partial \boldsymbol{n}^2} + \kappa \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \right) \right], \quad (16)$$

with the operators of partial derivatives defined by $\partial / \partial \boldsymbol{n} = \boldsymbol{n} \cdot \nabla$, and $\partial^2 / \partial \boldsymbol{n}^2 = \boldsymbol{n} \cdot \nabla^2 \cdot \boldsymbol{n}$. After solving this system for $\boldsymbol{u}'$, the velocity is updated by $\boldsymbol{u} \leftarrow \boldsymbol{u}'$.

(7) Re-apply the projection step to ensure a divergence-free velocity field.

(8) Extrapolate the reference map and the velocity field by Alg. 3. Meanwhile, yield the new advection region for the next time step.

Even though all of our examples do not consider viscosity, there should be no hurdle to integrate the conventional viscosity handling into the current framework after Step (5), in which a Poisson system

$$\boldsymbol{u}' = \boldsymbol{u} + \frac{\mu \Delta t}{\rho} \nabla^2 \boldsymbol{u}' \quad (17)$$

is solved for $\boldsymbol{u}'$, and then a velocity update $\boldsymbol{u} \leftarrow \boldsymbol{u}'$ is applied. We refer the readers to the work of Batty and Bridson [2008] for boundary conditions and other implementation details.

In the current implementation, the standard ghost fluid boundary conditions [Gibou et al. 2002] are applied for setting up the boundary condition in velocity projection step. Under this scheme, a cell is considered as a liquid cell as long as its center is inside the liquid. High-order velocity projection scheme is preferred to better track sub-grid features. We will come back to this limitation in §7.

6 EXPERIMENTAL RESULTS

6.1 Validation

We first validate our proposed algorithms upon specially designed scenarios. In order to reveal the capability of our algorithm on reducing the level-set error introduced by advection and reinitialization, we test these two procedures in §6.1.1 and §6.1.2 separately. We further show the practical results on the combination of these two procedures as well as the local extrapolation scheme in §6.2.

For the purpose of quantitative analysis, the area/volume of the level-set liquid is calculated as follows:

- In 2D, the liquid area is the sum of the areas of triangles, which are generated by marching squares on a grid that is four times finer than the original grid in each dimension;
- In 3D, the liquid volume is evaluated using the fast marching-cubes-style scheme, proposed by Takahashi and Batty [2022].

We also define the average absolute error (AAE) of the level-set function as

$$\text{AAE} = \frac{1}{N} \sum_{i=1}^{N} |\tilde{\varphi}(\boldsymbol{x}_i) - \varphi(\boldsymbol{x}_i)| \quad (18)$$

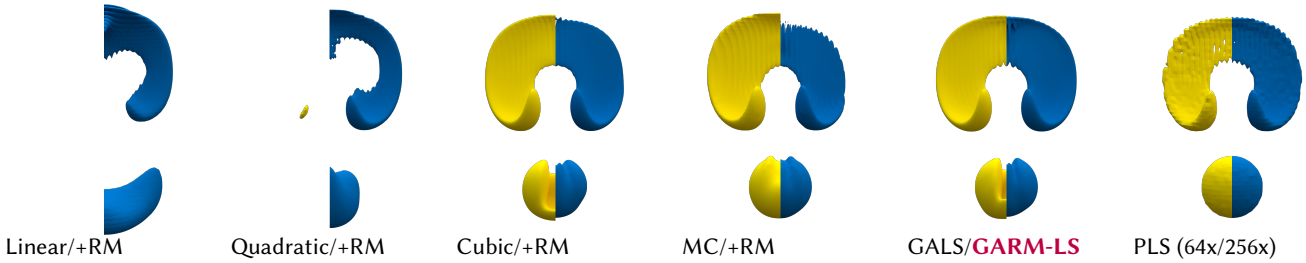

Fig. 15. LeVeque's sphere (3D). Each image is composed of two halves, namely the yellow half and the blue half, which are generated by relevant but different algorithms. Images in the top row show intermediate results at $t = 5$ s when the fluid is deformed to the maximum; those in the bottom row show the final counterparts at $t = 10$ s when the fluid shape is desired to turn back into a sphere. Images in the same column share the same advection algorithms, which are annotated below. The algorithm descriptions of the yellow half and the blue half are separated by a slash '/', where "~/~+RM" is abbreviated as "~/+RM".

Table 2. List of used acronyms of advection schemes throughout the paper.

Acronym†	Method
Linear	Semi-Lagrangian [Stam 1999]
Quadratic	Semi-Lagrangian with quadratic interpolation‡
Cubic	Semi-Lagrangian with cubic interpolation‡
MC	Modified MacCormack [Selle et al. 2008]
GALS	Gradient-augmented level-set [Nave et al. 2010]
PLS	Particle level-set [Enright et al. 2005]

† Acronyms of the reference-map modified versions of advection algorithms are denoted "~+RM", e.g., Linear+RM, Quadratic+RM, and Cubic+RM.

‡ The quadratic interpolation follows the Eqs. (12) and (13) in the work of Min and Gibou [2007]; the cubic one implements §3.5 of Bridson's book [2015].

to facilitate the quantitative evaluation of the level-set function distribution and the convergence of the algorithm. In AAE, $\tilde{\varphi}(\boldsymbol{x})$ and $\varphi(\boldsymbol{x})$ denote the numerical and the ground truth level-set function value at sample $\boldsymbol{x}_i$, respectively. A unified predefined point set, which consists of N (more than 10,000) randomly sampled points around the ground-truth interface, is used in each comparison.

6.1.1 Advection. We validate our approach in three standard and two original advection tests. All these tests are carried out in a simulation domain that satisfies $x, y, z \in [-5\,\text{m}, 5\,\text{m}]$ with a time step size $\Delta t = 0.02$ s. The comparisons are made among the mainstream grid-based level-set advection algorithms (listed in Table 2), their reference-map modified versions, and the particle level-set (PLS) method. For better accuracy, as recommended by Enright et al. [2005], we disabled reseeding in PLS during the following tests. The number of particles initialized in each near-surface grid cell is indicated in brackets, such like "PLS (16x)". Reinitialization is disabled for all the tests here to extract the level-set function error generated by pure advection, except for the PLS method, which requires reinitialization to maintain surface smoothness. For fairness reasons, whenever we back-trace grid points, a third-order Runge–Kutta method is used.

Zalesak's disk (2D). We normalize the classical Zalesak's disk [Selle et al. 2008] setting to fit in our simulation domain. A slotted circle with a radius of 1.5 m is initially centered at $(0, 2.5\,\text{m})$. The slot size is 0.5 m by 2.5 m. From $t = 0$ to $t = 12.56$ s, a rotational velocity field

$$\boldsymbol{u} = \frac{\pi}{157\Delta t}(-y, x)$$

is applied to drive the circle rotate. We show the advected results simulated on a 100×100 grid in Fig. 10 and plot the volume evolution curves in Fig. 16.

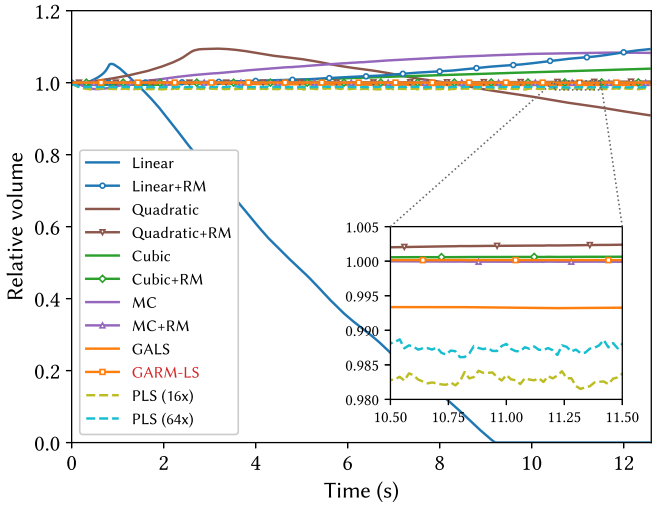

Fig. 16. Volume evolution curves of Zalesak's disk (2D). All reference-map augmented algorithms excluding Linear+RM preserve the volume well in this setting. In order to distinguish their curves, a zoomed-in view is attached.

After two complete rotations, the conventional (linear) semi-Lagrangian scheme loses all volume, while the quadratic-interpolated semi-Lagrangian scheme (9.05% variation in volume) results in an excessively smooth interface. These outcomes demonstrate that low-order algorithms cannot maintain surface detail over time. The higher-order schemes, namely the cubic-interpolated semi-Lagrangian (3.88% variation) and modified MacCormack (8.27% variation) schemes, are better at conserving volume and shape. However, the modified MacCormack scheme can destroy reflection symmetry. Among all the reference-map-free Eulerian methods, the GALS (6.52‰ variation) is most effective at preserving the sharp corners of the circle. The reference-map augmented versions of all

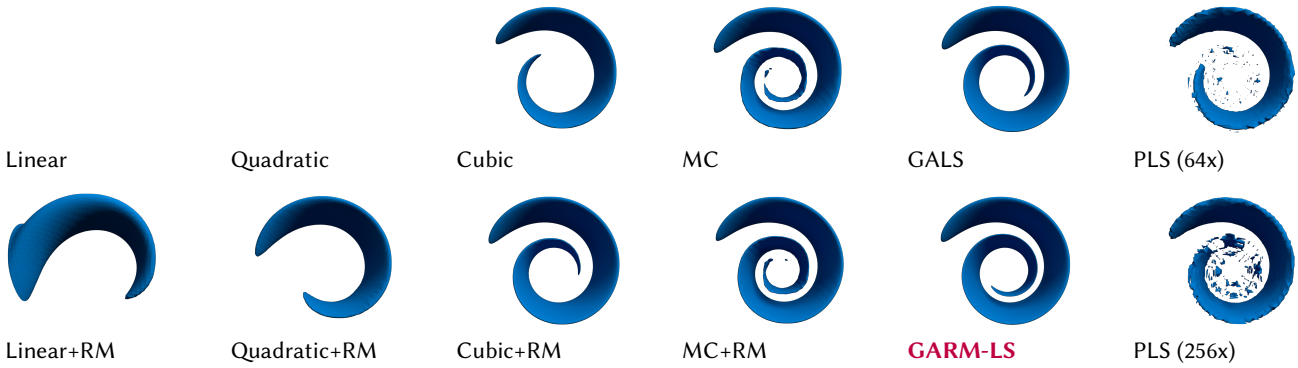

Fig. 17. Distorted sphere (3D). Images here show the final results at $t = 6$ s with the advection algorithms annotated bellow. Some of them are empty because the corresponding simulation loses all the volume at this moment. We point out that Linear+RM experiences a numerical drift that causes the fluid to hit the simulation boundary incorrectly.

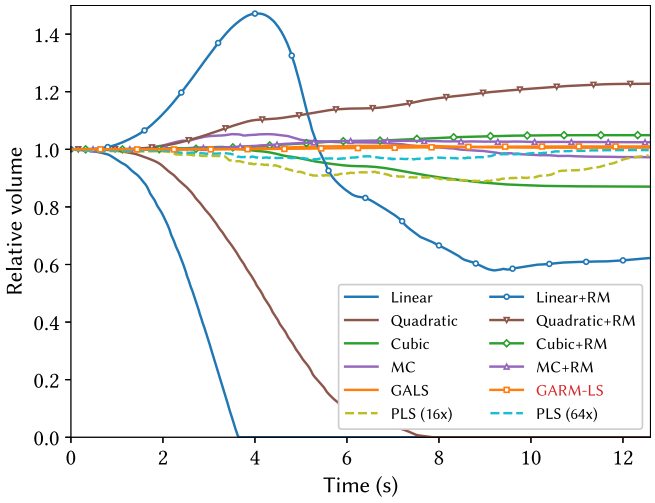

Fig. 18. Volume evolution curves of LeVeque's circle (2D). Some algorithms experience both increases and decreases in volume, which accidentally neutralizes volume variation.

the above-mentioned algorithms outperform their original counterparts on the volume and shape conservation. In this specific scenario, the Quadratic+RM (3.01‰ variation), Cubic+RM (0.755‰ variation), MC+RM (0.214‰ variation), and GARM-LS (0.163‰ variation) schemes achieve almost identical results. Even the conventional semi-Lagrangian scheme can largely maintain shape with the aid of reference mapping (9.31% variation).

As a hybrid method, PLS needs to correct the level set with particle information every time step. This mechanism improves the algorithm's ability on shape conservation but falls short in maintaining the interface smoothness (see the zoomed-in view in Fig. 16) and the temporal continuity (see the supplementary video). Although Enright et al. [2005] suggested generating 16 particles per cell initially for 2D cases, we find that the above issues can be somehow addressed by increasing the particle number. As shown in Fig. 16,

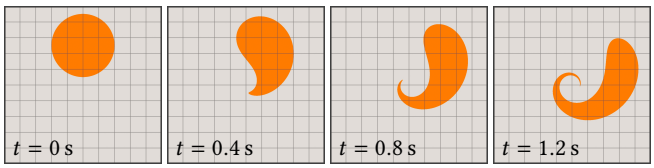

Fig. 19. A circle is being distorted by a divergence-free rotational velocity field. The closer to the center, the greater the angular velocity. The four frames shown are given by GARM-LS on a 200×200 grid. Every grid cell in the images corresponds to a 20×20 block in the simulation.

PLS (64x) is second only to GARM-LS in preserving sharp corners. However, the roughness of the PLS interface still results in global volume variation. According to quantitative analysis, PLS (16x) and PLS (64x) suffer from 1.83% and 1.20% volume loss, respectively.

LeVeque's circle (2D). A circle with radius $r = 1.5$ m is centered at $(0, 2.5\,\text{m})$ initially. Unlike the original LeVeque's circle test [1996], the circle here is deformed by a modified divergence-free velocity field [Nave et al. 2010]

$$\boldsymbol{u} = u_0 \cos\frac{\pi t}{628\Delta t}\left(\cos^2\frac{\pi x}{a_0}\sin\frac{2\pi y}{a_0}, -\sin\frac{2\pi x}{a_0}\cos^2\frac{\pi y}{a_0}\right),$$

in which $u_0 = 10\,\text{m}\,\text{s}^{-1}$ and $a_0 = 10$ m. At $t = 6.28$ s, the circle is deformed to maximum, and gradually restore to the initial shape at $t = 12.56$ s as the velocity reverses in the second half of the time interval. We conduct this experiment on a 200×200 simulation grid with different advection algorithms. Their simulation results are demonstrated in Fig. 13, and the volume evolution curves are plotted in Fig. 18.

At $t = 6.28$ s, GARM-LS is the best to predict the correct fluid shape with only 4.30‰ volume variation. All the other schemes result in non-negligible shape incorrectness. The algorithms that lead to volume loss are Linear (100% loss), Quadratic (91.3%), Linear+RM (16.5%), PLS (16x, 7.99%), Cubic (5.86%), and PLS (64x, 2.47%), and those lead to volume growth are Quadratic+RM (14.2% variation), MC+RM (2.92%), Cubic+RM (2.91%), MC (2.45%), and GALS (1.15%). In the second half of the simulation when the velocity field reverses,

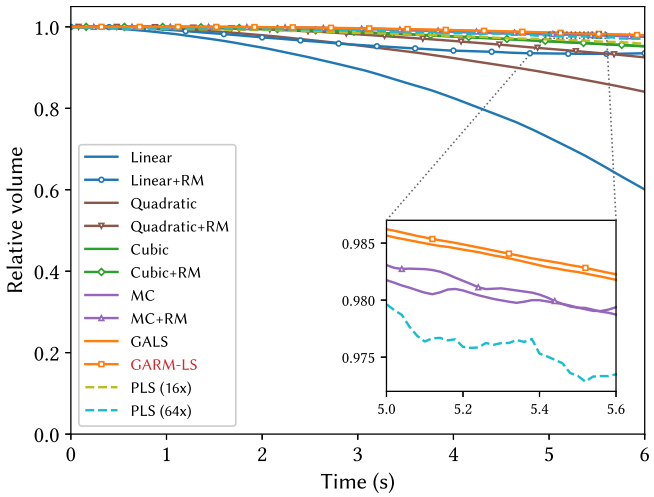

Fig. 20. Volume evolution curves of the distorted circle (2D). Every algorithm exhibits volume loss during the period of time. The subtle differences on volume loss of some algorithms are demonstrated in the zoomed-in view.

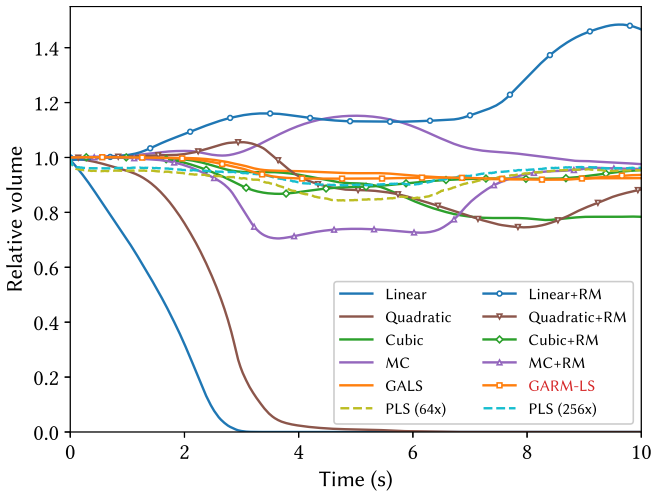

Fig. 21. Volume evolution curves of LeVeque's sphere (3D). Some algorithms accidentally neutralizes volume variation, similar to those in the 2D case.

some algorithms may introduce errors in the opposite direction, which coincidentally neutralizes the volume error. These algorithms are MC+RM (2.46% variation) and GALS (5.72‰ variation). Despite this, compared to other purely Eulerian methods, GARM-LS (9.90‰ variation) still outputs the most correct shape as shown in Fig 13. It is worth noting that such a back-and-forth scenario is well-suited for algorithms that can make use of the initial information. That is why reference-map augmented algorithms recovers circle better than the original versions, and why particle-based methods can return to the initial state more easily. When the number of particles per cell is increased to 64, the PLS method can generate a perfect circle with only 2.85‰ variation in volume. After all, more particles mean more initial information.

Distorted circle (2D). As illustrated in Fig. 19, a circle centered at $(0, 2.5\,\mathrm{m})$ with radius $r = 2\,\mathrm{m}$ is distorted as time flows by a well designed divergence-free rotational velocity field, which reads

$$\boldsymbol{u} = \frac{u_0}{\sqrt{x^2 + y^2} + r_0}(y, -x)$$

where $u_0 = 4\,\mathrm{m/s}$ and $r_0 = 1.0\,\mathrm{m}$. After 6 seconds, the circle has undergone severe distortion and turns into a spiral. The inner end of the spiral becomes so thin that even a 200×200 grid cannot prevent the sub-cell features from disappearing, as shown in Fig. 14. The corresponding volume curves are plotted in Fig. 20.

In this test, each algorithm suffers from volume loss. Sorted by the proportion of volume loss at $T = 6\,\mathrm{s}$ in a descending manner, they are Linear (39.9%), Quadratic (15.9%), Quadratic+RM (7.49%), Linear+RM (6.49%), Cubic (4.80%), Cubic+RM (4.73%), PLS (16x, 4.17%), PLS (64x, 2.89%), MC+RM (2.46%), MC (2.19%), GALS (2.09%), and GARM-LS (2.06%). In most cases, the reference-map augmented algorithms preserve more sub-cell geometric features (longer spirals) than their original versions. Even taking advantage of the reference map, low-order algorithms cannot outperform high-order algorithms. The PLS methods fails to acquire a continuous interface owing to its low-order interpolation scheme. Although particles are adequate around the inner end, they cannot be used to recover the level set due to the limited grid size. In addition, there is a radial offset in the simulation of MC and MC+RM. This reflects the shortcomings of the modified MacCormack method in preserving symmetry.

LeVeque's sphere (3D). As inspired by Nave et al. [2010], a sphere of radius $r = 1.5\,\mathrm{m}$ is centered at $(-1.5\,\mathrm{m}, -1.5\,\mathrm{m}, -1.5\,\mathrm{m})$ initially. It is then deformed by a divergence-free velocity field $\boldsymbol{u} = (u, v, w)$:

$$\begin{cases} u = 2u_0 \cos\dfrac{\pi t}{500\Delta t} \cos^2\dfrac{\pi x}{a_0} \sin\dfrac{2\pi y}{a_0} \sin\dfrac{2\pi z}{a_0}, \\ v = -u_0 \cos\dfrac{\pi t}{500\Delta t} \sin\dfrac{2\pi x}{a_0} \cos^2\dfrac{\pi y}{a_0} \sin\dfrac{2\pi z}{a_0}, \\ w = -u_0 \cos\dfrac{\pi t}{500\Delta t} \sin\dfrac{2\pi x}{a_0} \sin\dfrac{2\pi y}{a_0} \cos^2\dfrac{\pi z}{a_0}, \end{cases}$$

in which $u_0 = 2.5\,\mathrm{m\,s^{-1}}$ and $a_0 = 10\,\mathrm{m}$. The sphere is deformed to the maximum at $t = 5\,\mathrm{s}$ and returns to its initial shape at $t = 10\,\mathrm{s}$. As displayed in Fig. 15, we conduct experiments on a 50^3 grid and render the exported meshes at the mentioned two time instances. The volume evolution curves are plotted in Fig. 21.

Similar to the 2D case, algorithms with higher orders acquire smoother and more accurate shapes at $t = 5\,\mathrm{s}$, and algorithms that make better use of initial information recover more perfect spheres at $t = 10\,\mathrm{s}$. As illustrated in Fig. 21, such a forth-and-back scenario may cause algorithms to coincidentally cancel volume loss in the second half of the time, which is not true for our method. Besides, for the PLS methods in 3D cases, we generate 64 and 256 particles initially in each cell, while the recommended number is 32 [Enright et al. 2005].

Distorted sphere (3D). Similar to the distorted circle (Fig. 19), a 3D version was designed accordingly. At $t = 0$, a sphere of radius $r = 2\,\mathrm{m}$ is centered at $(0, 2.5\,\mathrm{m}, 0)$, and then it is distorted by a

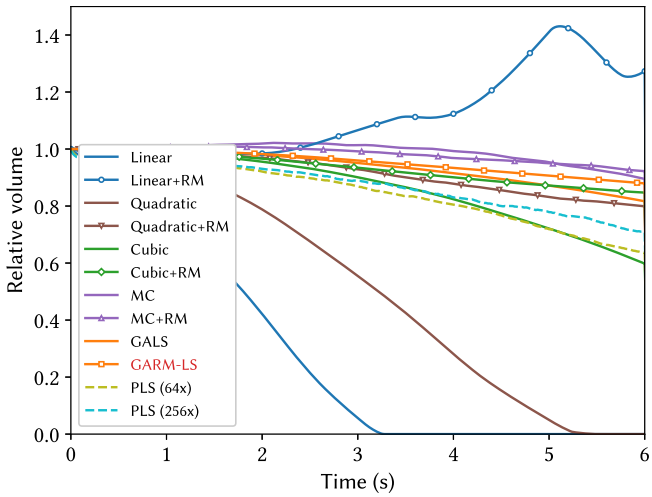

Fig. 22. Volume evolution curves of the distorted sphere (3D). Every algorithm, excluding Linear+RM, suffers from volume loss here. Though MC and MC+RM seems to preserve volume well, their results are discontinuous (see Fig. 17 and the supplementary video).

rotational velocity field

$$\boldsymbol{u} = \frac{u_0}{\sqrt{x^2+y^2}+r_0}(y,-x,0)$$

where $u_0 = 4\,\mathrm{m/s}$ and $r_0 = 1.0\,\mathrm{m}$. From $t = 0$ to $t = 6\,\mathrm{s}$, the sphere undergoes severe distortion and turns into a spiral. We test different algorithms upon a 50^3 grid, whose final results are rendered in Fig. 17, and volume evolution curves are plotted in Fig. 22.

We point out that this scene is not just a 3D extension of the 2D version in the z-direction. The circle there is replaced by a sphere rather than a cylinder, which is more challenging to capture after distortion. Among all the algorithms, Linear and Quadratic fail to preserve any volume after 6 s. Although reference mapping can improve the results, such low-order algorithms still suffer from severe numerical drifts. Algorithms with high-order accuracy, like MC and MC+RM, perform well in this scenario. However, they cannot provide spatially and temporally continuous surfaces without extrema clamping [Selle et al. 2008]. If the clamping strategy is applied, the volume conservation will become worse. In addition, similar to the 2D version, the PLS method fails to reconstruct a smooth surface. Our method outperforms all these schemes because it has not only high accuracy order but also stability in thin-sheet capturing.

Convergence. We perform Zalesak's disk and the distorted circle tests on simulation grids of different resolutions in order to analyze the algorithm convergence. In each test, we execute 20 time steps ($t = 0.4\,\mathrm{s}$) and compare the simulation result with the ground truth by Eq. (18). As plotted in Fig. 23, different algorithms exhibit similar orders of accuracy in the tests of Zalesak's disk. Based on the least-squares fit, the orders of Linear, Quadratic, and Cubic are 1.47, 1.91, 1.83, respectively, while that of MC is 1.75. The worst algorithm is PLS (64x) whose accuracy order is 1.29. Only GALS (2.41) and reference-map augmented algorithms (2.52) converge at orders of more than two. This phenomenon may be caused by the non-smooth

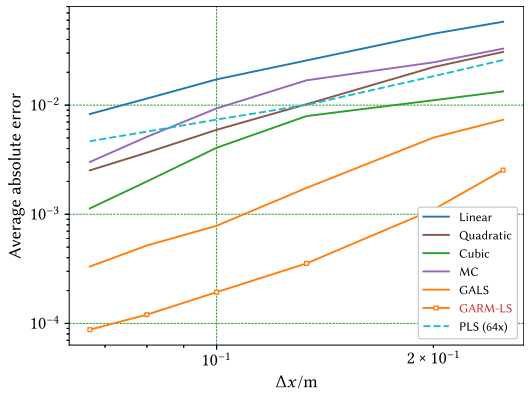

Fig. 23. Convergence curves of algorithms upon Zalesak's disk. The tested grid resolutions are 40^2, 50^2, 75^2, 100^2, 125^2, and 150^2. The plotted curves of Quadratic+RM, Cubic+RM, and MC+RM are all overlapped by that of GARM-LS, which are omitted in the figure.

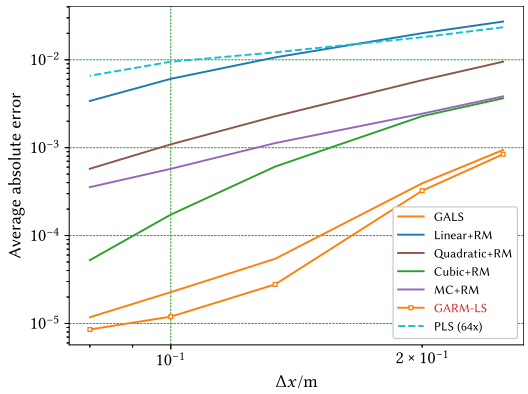

Fig. 24. Convergence curves of algorithms upon the distorted circle. The tested grid resolutions are 40^2, 50^2, 75^2, 100^2, and 125^2. Here we only plot the results of reference-map augmented Linear, Quadratic, Cubic, and MC for simplicity. Any original algorithm is slightly less accurate than its augmented version.

level-set field around the slot. For the distorted circle scenario, the difference in the order of accuracy between algorithms is more pronounced. Sorted by the accuracy order from lowest to highest, the algorithms in Fig. 24 are PLS (64x, 1.06), Linear+RM (1.79), MC+RM (2.08), Quadratic+RM (2.45), Cubic+RM (3.70), GALS (3.94), and GARM-LS (4.26), which is more consistent with intuition.

Summary on advection. According to the above results, the combination of high-order advection schemes with reference-map enhancement are the key factors for the volume and shape preservation. As a purely Eulerian method, our GARM-LS meets the above two necessary requirements and does demonstrate the corresponding advantageous in all the quantitative test.

6.1.2 Reinitialization. As for the reinitialization step, the validations mainly focus on two aspects: one is the order of accuracy and the other is the robustness of the algorithm.

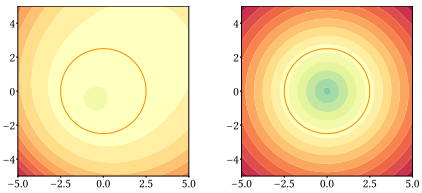

Fig. 25. The first test case of reinitialization, with the left one showing the distorted level-set function and the right one showing the corresponding SDF. Higher values are mapped to warmer colors.

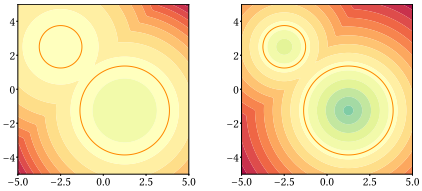

Fig. 26. The second test case of reinitialization, with the left one showing the distorted level-set function and the right one showing the corresponding SDF. Higher values are mapped to warmer colors.

Convergence. The comparison of the order of accuracy is conducted among our GARM-LS, fast marching, ENO and WENO algorithms on two static analytical tests.

As demonstrated in the left image of Fig. 25, the distorted level-set function of the first test with a single sphere is

$$\varphi(x,y) = \left((x-1)^2 + (y-1)^2 + 0.1\right)\left(\sqrt{x^2+y^2} - 1\right),$$

whose corresponding SDF is

$$\varphi(x,y) = \sqrt{x^2+y^2} - 1,$$

as shown in the right image of Fig. 25. As demonstrated in the left image of Fig. 26, the distorted level-set function of the second test with two separated spheres is

$$\varphi(x,y) = \min\left\{e^{(x-x_0)^2+(y-y_0)^2} - e^{r_0^2}, e^{(x-x_1)^2+(y-y_1)^2} - e^{r_1^2}\right\},$$

whose corresponding SDF is

$$\varphi(x,y) = \min\left\{\sqrt{(x-x_0)^2+(y-y_0)^2} - r_0, \sqrt{(x-x_1)^2+(y-y_1)^2} - r_1\right\},$$

as illustrated in the right image of Fig. 26.

In each test, the distorted level-set function is initially given and reinitialized with aforementioned 4 different algorithms. For each individual algorithm, the difference between the derived field and the corresponding SDF is quantitatively evaluated using Eq. (18). The convergence curves of these two tests are plotted in Fig. 27 and Fig. 28 respectively. From these two results, we conclude that the fast-marching method is first-order accurate around the interface, while the accuracy orders of ENO and WENO are both nearly 1.5th order. Our method is fourth-order accurate.

Robustness. To further verify the validity of our reinitialization scheme in practical fluid simulation, we perform a comparison experiment with the algorithm of Anumolu and Trujillo [2013] on a

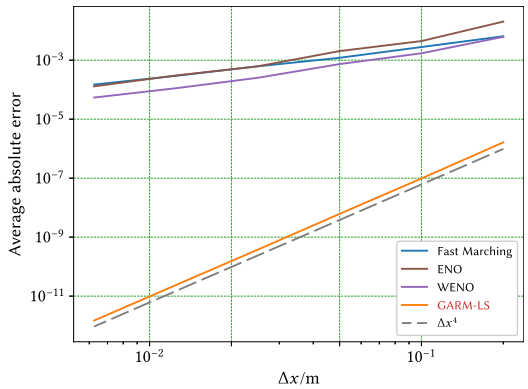

Fig. 27. Convergence curves of reinitialization algorithms upon the first analytical scene. The tested grid resolutions are 50^2, 100^2, 200^2, 400^2, 800^2, and 1600^2.

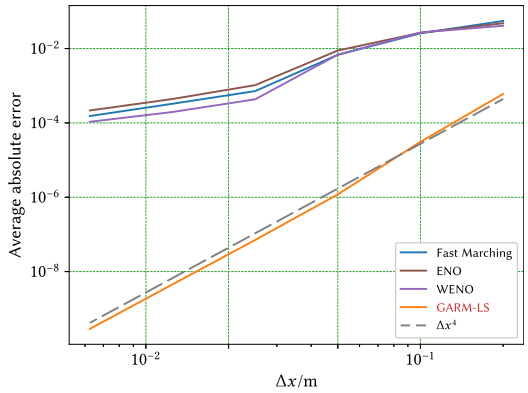

Fig. 28. Convergence curves of reinitialization algorithms upon the second analytical scene. The tested grid resolutions are 50^2, 100^2, 200^2, 400^2, 800^2, and 1600^2.

4-way dam break scene(§6.2). All the other procedures are identical with gradient and reference-map augmented. As shown in Fig. 29, as time flies, the dynamic simulation with Anumolu and Trujillo [2013]'s reinitialization shows severe instability issue: the volume of fluid grows uncontrollably due to non-convergence of the quasi-Newton method. Since our reinitialization scheme enhances the original algorithm of Anumolu and Trujillo [2013] with *initial guess*, *back-and-forth trial*, and *fast-marching post-processing* (see §4.3.3), the simulation result with GARM-LS shows more stable and robust behaviors.

Summary on reinitialization. We have verified that our method, namely GARM-LS, possesses a fourth-order accuracy around the interface in the reinitialization step, as its original algorithm [2013]. Besides the high order accuracy, all the enhancements proposed in §4.3.3 have been proofed to be indispensable for the overall convergence of the dynamic fluid simulation, in which fluid surface usually undergoes large distortion, and the nearest point frequently exhibits ambiguity/singularity issue.

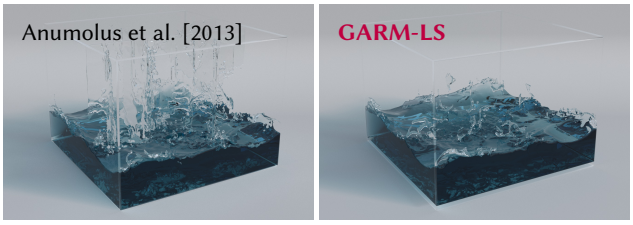

Fig. 29. Comparison of the original reinitialization algorithm and our improved version upon practical dynamic scenario of 4-way dam break.

6.2 Examples

For 3D simulations, we have designed several experiments to demonstrate the efficacy of our framework in simulating high-resolution liquids and preserving small fluid volumes. A GPU-based preconditioned conjugate gradient solver is adopted in our projection stage, while the other parts of the pipeline run on CPU. Statistics on all experiments (except for the high-resolution dam break) are collected from a desktop machine with an Intel i7-8700 @ 3.20GHz CPU, 16GB RAM, and an NVIDIA GeForce GTX 1060 graphics card, as listed in Tab. 3. The high-resolution dam break (case 1 in the table) is tested on a server machine with an Intel Xeon W-3175X @ 3.10GHz CPU, 160GB RAM, and an NVIDIA Quadro RTX 8000 graphics card. The narrow-band FLIP [Ferstl et al. 2016] implementation is taken from Houdini on another server.[1]

For simplicity, hereafter unless otherwise specified, we will refer to a specific simulation method using the acronym of advection scheme listed in Table. 2 with "+FMM" suffix whenever fast-marching reinitialization scheme is adopted. (For PLS and GARM-LS, the reinitialization scheme is inherently included by themselves.)

4-way dam break. In this example, bulks of incompressible liquid fall from 4 separate breaking dam without viscosity and surface tension, forming thin sheets of fluid and splashes.

We simulate this scenario with six different methods, including Linear+FMM, Cubic+FMM, Cubic+RM+FMM, PLS(128x), our GARM-LS, and narrow-band FLIP. Fig. 2 shows the intermediate frame when the thin sheets are present. Although all the methods (except Linear+FMM) manage to show some kinds of thin sheet, only PLS, narrow-band FLIP, and GARM-LS could preserve the splashes at the center of fluid region. Furthermore, Linear+FMM and Cubic+FMM also suffer from numerical diffusion, which cause the liquid looks excessively viscous, see the supplementary video. Fig. 30 depicts the volume evolution of all the methods. Same as we saw in §6.1, the simulations enhanced by the reference map perform much better in preserving the volume. We point out that the FLIP method is skilled at generating vivid splashes, but its ability to maintain the total volume and surface smoothness is limited.

Dam break. This classical 3D example is initialized with a cylindrical obstacle located at the center of the square. The water from a breaking dam pours down and hit the obstacle, causing splashes into the air. The liquid surface has undergone extreme distortion in response to gravity, surface tension, and collision.

[1]The performance of the narrow-band FLIP method is not statistically available due to its large differences in implementation.

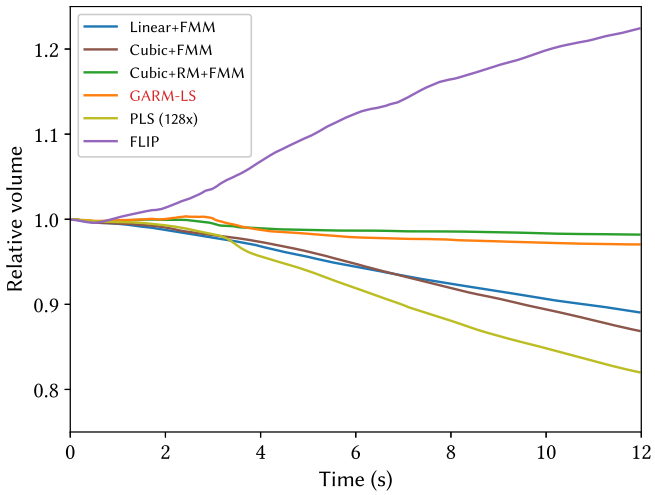

Fig. 30. Volume evolution curves of the 4-Way Dam Break test. Among the methods, only the volume of FLIP is increasing; all the others have experienced varying degrees of volume loss. The volume variation of Cubic+RM+FMM is the smallest, because this method is also enhanced with high-order reference mapping but does not produce the same pronounced splash effects as GARM-LS.

We simulate this scenario with three different methods, including Cubic+FMM (here we use a high-resolution level-set method with cubic interpolation scheme and fast marching for redistancing), Cubic+RM+FMM, PLS(128x), and GARM-LS method, as seen in Fig. 31. Compared to the high-resolution level-set method and PLS, the methods enhanced by the reference map perform much better in preserving the volume of fluid, which is supported by Fig. 32. Nevertheless, in Cubic+RM+FMM, the splashes that stem from the collision quickly smear out due to the reinitialization procedure, while PLS and GARM-LS manage to capture the water droplets, and produce vibrant outcomes. Among all the tested methods, only GARM-LS shows superior merit in both volumetric fluid simulation and splash simulation.

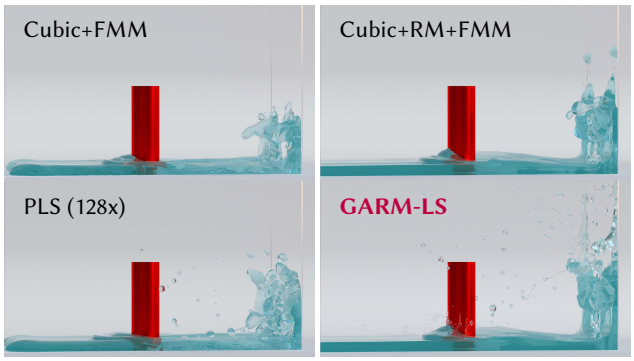

Fig. 31. Comparisons of water falling from a breaking dam among different interface-tracking methods at frame 100.

Table 3. Performance and statistics. (1-6) 4-Way Dam Break (without surface tension); (7-11) Dam break (with surface tension); (12) Galton board (high surface tension); (13) Galton board (low surface tension); (14) Droplet collides (high surface tension); (15) Droplet collide (low surface tension); (16) Diagonal collide; (17) Raining on juice; (18) Rinsing a bunny.

Case	Figure	Algorithm	Resolution	Cell size Δx/mm	Frame interval Δt/s	CFL	Simulation time/step†				
							Total	Advect.	Reinit.	Proj.	Extra.
1	Fig. 2	Linear+FMM	$192 \times 144 \times 192$	5.32	1/150	1.0	6.27 s	1.91 s	1.31 s	1.29 s	1.75 s
2	Fig. 2	Cubic+FMM	$192 \times 144 \times 192$	5.32	1/150	1.0	9.36 s	4.81 s	1.30 s	1.29 s	1.96 s
3	Fig. 2	Cubic+RM+FMM	$192 \times 144 \times 192$	5.32	1/150	1.0	11.58 s	5.81 s	1.29 s	1.33 s	3.15 s
4	Fig. 2	PLS (128x)	$192 \times 144 \times 192$	5.32	1/150	1.0	35.40 s	30.90 s	1.29 s	1.37 s	1.84 s
5	Fig. 2	Narrow-Band FLIP	$192 \times 144 \times 192$	5.32	1/150	1.0					
6	Fig. 2	GARM-LS	$192 \times 144 \times 192$	5.32	1/150	1.0	15.24 s	8.99 s	1.93 s	1.33 s	3.00 s
7	Fig. 31	Cubic+FMM	$256 \times 307 \times 512$	3.91	1/100	1.0	122.48 s	10.06 s	3.35 s	104.55 s	4.53 s
8	None‡	Cubic+FMM	$192 \times 230 \times 384$	3.91	1/100	1.0	66.26 s	14.77 s	1.42 s	43.44 s	6.64 s
9	Fig. 31	Cubic+RM+FMM	$192 \times 230 \times 384$	5.32	1/100	1.0	73.60 s	18.41 s	1.43 s	42.51 s	11.25 s
10	Fig. 31	PLS (128x)	$192 \times 230 \times 384$	5.32	1/100	1.0	52.94 s	25.68 s	1.40 s	19.20 s	6.66 s
11	Fig. 31	GARM-LS	$192 \times 230 \times 384$	5.32	1/100	1.0	83.14 s	28.39 s	2.36 s	41.42 s	10.97 s
12	Fig. 3	GARM-LS	$256 \times 256 \times 51$	23.8	1/100	1.0	18.34 s	5.81 s	0.41 s	9.32 s	2.81 s
13	Fig. 3	GARM-LS	$256 \times 256 \times 51$	23.8	1/100	1.0	18.39 s	5.81 s	0.51 s	9.74 s	2.33 s
14	Fig. 6	GARM-LS	$256 \times 256 \times 256$	23.8	1/100	1.0	81.97 s	28.47 s	1.88 s	40.56 s	11.05 s
15	Fig. 6	GARM-LS	$256 \times 256 \times 256$	23.8	1/100	1.0	81.81 s	28.97 s	1.88 s	40.57 s	11.09 s
16	Fig. 7	GARM-LS	$256 \times 64 \times 128$	23.8	1/400	1.0	12.54 s	3.66 s	0.32 s	7.05 s	1.51 s
17	Fig. 8	GARM-LS	$256 \times 256 \times 256$	3.97	1/200	1.0	99.76 s	28.65 s	2.66 s	57.95 s	10.50 s
18	Fig. 33	GARM-LS	$192 \times 230 \times 192$	4.26	1/150	1.0	42.62 s	14.74 s	1.11 s	20.78 s	5.99 s

† The collected simulation time is averaged. Abbreviations Advect., Reinit., Proj., and Extra. denote the advection time, reinitialization time (including restart time), projection time (including time for applying surface tension), and extrapolation time respectively.

‡ This setup is not rendered since it performs worse than the existing version with higher resolution. The case is listed here for the comparison of performance with other algorithms.

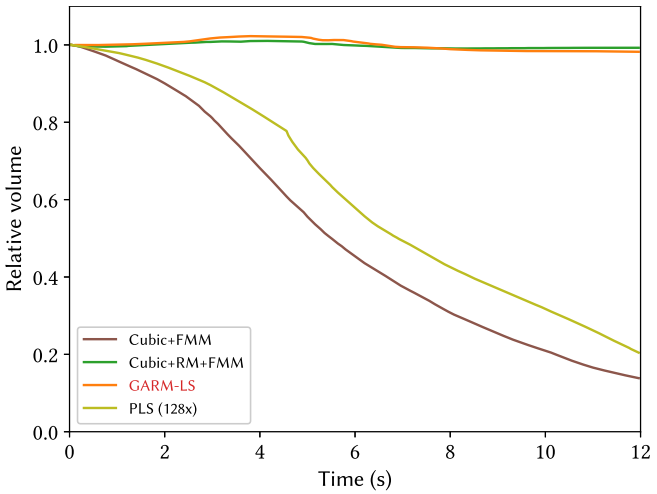

Fig. 32. Volume evolution curves of the Dam Break test.

Galton board. In this set of examples, a staggered array of cylindrical obstacles is placed in a flat box, forming a Galton board. Instead of a coin, a fluid ball falls through the board and split into pieces as a result. Fig. 3 shows 2 different setups for this example: one for the high-surface-tension fluid and one for the low surface tension. Splitting and merging happen consistently in the simulation. Since fluid-structure interaction is beyond the scope of this paper, our current implementation cannot guarantee completely divergence-free velocities at solid boundaries. This imperfection can occasionally lead to grid artifacts, such as fluid flickering and shrinkage on the cylindrical obstacle in this example. We will discuss this later in §7.

Droplet collide. These examples showcase our method's ability to track fluids with low dissipation. Two spheres are dyed blue and green respectively at the initial phase. As shown in Fig. 6, the two droplets collide and drop onto the ground, with their color gradually mixed with each other. Here, the reference map is further exploited for the color advection in the same way as it is used for level-set function. In the advection step, the transport equation of color c

$$\frac{\partial c}{\partial t} + (\boldsymbol{u} \cdot \boldsymbol{\nabla})c = 0 \tag{20}$$

is solved through backtracking the reference map

$$c(x_i, t) = c(\boldsymbol{\xi}(x_i, t), t_0) \tag{21}$$

to obtain the interpolated color at t_0, which refers to the last time the reference map restarted. With the aid of the reference map, we are able to acquire lower numerical diffusion and a clearer separation of the fluids.

Diagonal collide. We simulate the collision of 2 droplets in diagonal directions with noticeable surface tension. In Fig. 7, most of the droplets just go past each other, leaving a line-style piece of water in the middle of the scene. Finally, driven by surface tension, the piece of water contracted to form a sphere. Our method successfully keeps a smooth surface during the collision.

Fig. 33. Rinsing a bunny with a water jet. Frame numbers from left to right: 50, 150, 250.

Dropping juice. Fig. 8 shows droplets of orange juice dropping into a layer of juice and splashing. The radii of the droplets in this example are only several times the size of a cell. Our method manages to preserve the volume of every droplet until it drops into the layer. The same method can apply to similar scenes including raining and spraying.

Here we use the standard marching cubes algorithm to reconstruct the surface for rendering. However, the expressiveness of GARM-LS is beyond that of the marching cubes algorithm, hence some sub-cell fluid is omitted from the reconstructed surface, causing jagged artifacts on the edge of splash crown. A finer surface can be reconstructed by simply making a refined level-set function, but to solve the problem thoroughly, a higher-order reconstruction/rendering algorithm is called for.

Rinsing a bunny. As shown in Fig. 33, a bunny is rinsed by a water jet. Splashes from the point of impact fly over distances and finally fall apart from the viewport. The result showcases the ability of our method by interacting with complex boundaries and producing small, pinched-off droplets.

7 CONCLUSIONS AND FUTURE WORK

This paper presents GARM-LS, a novel level-set method based on gradient augmentation and reference map, for tracking complex interface dynamics in interfacial flow simulations. The central piece of our method is a gradient augmentation scheme to convect level set values and their gradients on a reference map. We build a full pipeline for GARM-LS interface tracking by devising novel algorithms for advection, interpolation, extrapolation, and reinitialization, which enable accurate and robust interface tracking results outperforming various state-of-the-art methods.

Compared to existing work incorporating a high-resolution level-set with the low-resolution physics [Bojsen-Hansen and Wojtan 2013; Goldade et al. 2016], our method focuses more on characterizing interfacial geometry with a high-order representation, i.e. the gradient-augmented scheme. As is pointed out by Nave et al. [2010], a high-order surface tracker not only better preserves fluid volume and fluid thin structures during a dynamic simulation, but also enables a high-order approximation of the local interface curvature, which is crucial for obtaining various surface tension driven flow behaviors in a physically accurate manner. On the other hand, this high-order surface tracking scheme as well as the smooth nature of a implicit surface make our method distinct from particle-based methods (e.g. PIC/FLIP and SPH). While the particle-based methods have achieved success in simulating large-scale liquids, our method exhibits more benefits for an accurate representation of interfacial fluids at small scales where surface tension dominates. We believe our work opens up a new direction towards fully physically accurate small-scale fluid animation and hence raises opportunities for future improvement, which we will discuss in the following paragraphs.

Coupling with solids. Our method does not include a mechanism for fully accurate interpolation around the solid boundaries. Due to grid-level errors in interpolation and projection, small droplets may flicker or shrink when they are close to the solid boundaries. It is beneficial to combine our method with a more accurate boundary interpolation scheme, such as the cut-cell approach [Azevedo et al. 2016]. From the perspective of strict volume preservation, coupling our method with some VOF methods is also beneficial.

Full high-order solver. Although our method exhibits outstanding capability in high-order interface tracking, the velocity projection and the semi-implicit surface tension scheme in the traditional Navier-Stokes solver are not modified to accommodate the finer interface representation. Therefore, the derived velocities may not be pointwise divergence-free, which in turn leads to some incorrect fluid behavior. For instance, a thin thread of fluid will not break up properly into droplets due to the lack of sub-cell level surface tension discretization. With a high-order projection and surface tension scheme, we expect a totally fourth-order accurate solver for free-surface fluids.

Rendering. In this work we reconstruct the fluid surface from level sets by the standard marching-squares/cubes algorithm. However, since these algorithms are not aware of the sub-grid details of the fluid (i.e., the high-order information of the level set), jagged artifacts can be observed on the reconstructed surface. A higher-order surface reconstruction algorithm for implicit geometry representations is desired to avoid this bottleneck lying in the rendering stage. One interesting possibility is a customized ray marching algorithm for our GARM-LS method to enable a smooth and precise surface rendering.

Performance. It is worth noting that the computational merits of using a Cartesian grid were not fully explored in GARM-LS. Discretizing our GARM-LS method on sparse/adaptive data structures (e.g., [Narita and Ando 2022]) remains an unexplored topic. The parallelization of our fluid solver also deserves further investigation.

ACKNOWLEDGMENTS

Xingqiao Li, Xingyu Ni, and Baoquan Chen acknowledge the funding support by National Key R&D Program of China (2022ZD0160801) and Shenzhen Collaborative Innovation Program (CJGJZD2021048092601003). We credit the Houdini Education licenses for the narrow-band FLIP simulation and video generations. The bunny mesh is courtesy of the Stanford Computer Graphics Laboratory.

REFERENCES

Lakshman Anumolu and Mario F. Trujillo. 2013. Gradient augmented reinitialization scheme for the level set method. *International Journal for Numerical Methods in Fluids* 73, 12 (2013), 1011–1041.

Vinicius C. Azevedo, Christopher Batty, and Manuel M. Oliveira. 2016. Preserving Geometry and Topology for Fluid Flows with Thin Obstacles and Narrow Gaps. *ACM Trans. Graph.* 35, 4 (jul 2016), 12 pages.

Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A Fast Variational Framework for Accurate Solid-Fluid Coupling. In *ACM SIGGRAPH 2007 Papers* (San

Diego, California) *(SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA, 100–es.

Christopher Batty and Robert Bridson. 2008. Accurate Viscous Free Surfaces for Buckling, Coiling, and Rotating Liquids. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Dublin, Ireland) *(SCA '08)*. Eurographics Association, Goslar, DEU, 219–228.

Thomas Bellotti and Maxime Theillard. 2019. A coupled level-set and reference map method for interface representation with applications to two-phase flows simulation. *J. Comput. Phys.* 392 (2019), 266–290.

Morten Bojsen-Hansen and Chris Wojtan. 2013. Liquid Surface Tracking with Error Compensation. *ACM Trans. Graph.* 32, 4, Article 68 (jul 2013), 13 pages.

Robert Bridson. 2015. *Fluid simulation for computer graphics* (2nd ed.). AK Peters/CRC Press, Boca Raton, FL, USA.

Jingyu Chen, Victoria Kala, Alan Marquez-Razon, Elias Gueidon, David A. B. Hyde, and Joseph Teran. 2021. A Momentum-Conserving Implicit Material Point Method for Surface Tension with Contact Angles and Spatial Gradients. *ACM Trans. Graph.* 40, 4, Article 111 (jul 2021), 16 pages.

Nuttapong Chentanez and Matthias Müller. 2014. Mass-Conserving Eulerian Liquid Simulation. *IEEE Transactions on Visualization and Computer Graphics* 20, 1 (Jan 2014), 17–29.

David L. Chopp. 2001. Some Improvements of the Fast Marching Method. *SIAM Journal on Scientific Computing* 23, 1 (2001), 230–244.

Ricardo Cortez. 1995. *Impulse-based methods for fluid flow*. Ph.D. Dissertation. University of California.

Douglas Enright, Ronald Fedkiw, Joel Ferziger, and Ian Mitchell. 2002a. A Hybrid Particle Level Set Method for Improved Interface Capturing. *J. Comput. Phys.* 183, 1 (2002), 83–116.

Douglas Enright, Frank Losasso, and Ronald Fedkiw. 2005. A fast and accurate semi-Lagrangian particle level set method. *Computers & Structures* 83, 6 (2005), 479–490.

Douglas Enright, Stephen Marschner, and Ronald Fedkiw. 2002b. Animation and Rendering of Complex Water Surfaces. *ACM Trans. Graph.* 21, 3 (jul 2002), 736–744.

Florian Ferstl, Ryoichi Ando, Chris Wojtan, Rüdiger Westermann, and Nils Thuerey. 2016. Narrow Band FLIP for Liquid Simulations. *Computer Graphics Forum* 35, 2 (2016), 225–232.

Nick Foster and Ronald Fedkiw. 2001. Practical Animation of Liquids. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 23–30.

Frédéric Gibou and Ronald Fedkiw. 2005. A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem. *J. Comput. Phys.* 202, 2 (2005), 577–601.

Frederic Gibou, Ronald P. Fedkiw, Li-Tien Cheng, and Myungjoo Kang. 2002. A Second-Order-Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains. *J. Comput. Phys.* 176, 1 (2002), 205–227.

Ryan Goldade, Christopher Batty, and Chris Wojtan. 2016. A Practical Method for High-Resolution Embedded Liquid Surfaces. *Computer Graphics Forum* 35, 2 (2016), 233–242.

Toshiya Hachisuka. 2005. Combined Lagrangian-Eulerian Approach for Accurate Advection. In *ACM SIGGRAPH 2005 Posters* (Los Angeles, California) *(SIGGRAPH '05)*. Association for Computing Machinery, New York, NY, USA, 114–es.

Simone E. Hieber and Petros Koumoutsakos. 2005. A Lagrangian particle level set method. *J. Comput. Phys.* 210, 1 (2005), 342–367.

Jeong-Mo Hong, Tamar Shinar, and Ronald Fedkiw. 2007. Wrinkled Flames and Cellular Patterns. *ACM Trans. Graph.* 26, 3 (July 2007), 47–es.

David A. B. Hyde, Steven W. Gagniere, Alan Marquez-Razon, and Joseph Teran. 2020. An Implicit Updated Lagrangian Formulation for Liquids with Large Surface Energy. *ACM Trans. Graph.* 39, 6, Article 183 (nov 2020), 13 pages.

Guang-Shan Jiang and Danping Peng. 2000. Weighted ENO Schemes for Hamilton–Jacobi Equations. *SIAM Journal on Scientific Computing* 21, 6 (2000), 2126–2143.

Petr Karnakov, Sergey Litvinov, and Petros Koumoutsakos. 2020. A hybrid particle volume-of-fluid method for curvature estimation in multiphase flows. *International Journal of Multiphase Flow* 125 (2020), 103209.

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jarek Rossignac. 2005. FlowFixer: Using BFECC for Fluid Simulation. In *Eurographics Workshop on Natural Phenomena*. The Eurographics Association, Dublin, Ireland, 6 pages.

Doyub Kim, Oh-young Song, and Hyeong-Seok Ko. 2009. Stretching and Wiggling Liquids. *ACM Trans. Graph.* 28, 5 (dec 2009), 7 pages.

Theodore Kim, Jerry Tessendorf, and Nils Thürey. 2013. Closest Point Turbulence for Liquid Surfaces. *ACM Trans. Graph.* 32, 2, Article 15 (April 2013), 13 pages.

Randall J. LeVeque. 1996. High-Resolution Conservative Algorithms for Advection in Incompressible Flow. *SIAM J. Numer. Anal.* 33, 2 (1996), 627–665.

Frank Losasso, Ronald Fedkiw, and Stanley Osher. 2006. Spatially adaptive techniques for level set methods and incompressible flow. *Computers & Fluids* 35, 10 (2006), 995–1010.

Frank Losasso, Frédéric Gibou, and Ron Fedkiw. 2004. Simulating Water and Smoke with an Octree Data Structure. *ACM Trans. Graph.* 23, 3 (aug 2004), 457–462.

Frank Losasso, Jerry Talton, Nipun Kwatra, and Ronald Fedkiw. 2008. Two-Way Coupled SPH and Particle Level Set Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics* 14, 4 (2008), 797–804.

Miles Macklin and Matthias Müller. 2013. Position Based Fluids. *ACM Trans. Graph.* 32, 4, Article 104 (jul 2013), 12 pages.

Olivier Mercier, Xi-Yuan Yin, and Jean-Christophe Nave. 2020. The Characteristic Mapping Method for the Linear Advection of Arbitrary Sets. *SIAM Journal on Scientific Computing* 42, 3 (2020), A1663–A1685.

Chohong Min and Frédéric Gibou. 2007. A second order accurate level set method on non-graded adaptive cartesian grids. *J. Comput. Phys.* 225, 1 (2007), 300–321.

Patrick Mullen, Alexander McKenzie, Yiying Tong, and Mathieu Desbrun. 2007. A Variational Approach to Eulerian Geometry Processing. *ACM Trans. Graph.* 26, 3 (jul 2007), 66–es.

Mohammad Sina Nabizadeh, Stephanie Wang, Ravi Ramamoorthi, and Albert Chern. 2022. Covector Fluids. *ACM Trans. Graph.* 41, 4, Article 113 (jul 2022), 16 pages.

Fumiya Narita and Ryoichi Ando. 2022. Tiled Characteristic Maps for Tracking Detailed Liquid Surfaces. *Computer Graphics Forum* 41, 8 (2022), 231–242.

Jean-Christophe Nave, Rodolfo Ruben Rosales, and Benjamin Seibold. 2010. A gradient-augmented level set method with an optimally local, coherent advection scheme. *J. Comput. Phys.* 229, 10 (2010), 3802–3827.

Xingyu Ni, Bo Zhu, Bin Wang, and Baoquan Chen. 2020. A Level-Set Method for Magnetic Substance Simulation. *ACM Trans. Graph.* 39, 4, Article 29 (aug 2020), 15 pages.

Stanley Osher and James A Sethian. 1988. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* 79, 1 (1988), 12–49.

J.-P. Pons, G. Hermosillo, R. Keriven, and O. Faugeras. 2006. Maintaining the point correspondence in the level set framework. *J. Comput. Phys.* 220, 1 (2006), 339–354.

Ziyin Qu, Xinxin Zhang, Ming Gao, Chenfanfu Jiang, and Baoquan Chen. 2019. Efficient and Conservative Fluids Using Bidirectional Mapping. *ACM Trans. Graph.* 38, 4, Article 128 (jul 2019), 12 pages.

Takahiro Sato, Takeo Igarashi, Christopher Batty, and Ryoichi Ando. 2017. A Long-Term Semi-Lagrangian Method for Accurate Velocity Advection. In *SIGGRAPH Asia 2017 Technical Briefs* (Bangkok, Thailand) *(SA '17)*. Association for Computing Machinery, New York, NY, USA, Article 5, 4 pages.

Robert Saye. 2014. High-order methods for computing distances to implicitly defined surfaces. *Communications in Applied Mathematics and Computational Science* 9, 1 (2014), 107–141.

Andrew Selle, Ronald Fedkiw, ByungMoon Kim, Yingjie Liu, and Jarek Rossignac. 2008. An Unconditionally Stable MacCormack Method. *Journal of Scientific Computing* 35, 2 (June 2008), 350–371.

J. A. Sethian. 1996. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences* 93, 4 (1996), 1591–1595.

Oh-Young Song, Hyuncheol Shin, and Hyeong-Seok Ko. 2005. Stable but Nondissipative Water. *ACM Trans. Graph.* 24, 1 (jan 2005), 81–97.

Jos Stam. 1999. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., USA, 121—-128.

Andrew Staniforth and Jean Côté. 1991. Semi-Lagrangian Integration Schemes for Atmospheric Models – A Review. *Monthly Weather Review* 119, 9 (1991), 2206–2223.

Mark Sussman and Elbridge Gerry Puckett. 2000. A Coupled Level Set and Volume-of-Fluid Method for Computing 3D and Axisymmetric Incompressible Two-Phase Flows. *J. Comput. Phys.* 162, 2 (2000), 301–337.

Mark Sussman, Peter Smereka, and Stanley Osher. 1994. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *J. Comput. Phys.* 114, 1 (1994), 146–159.

Tetsuya Takahashi and Christopher Batty. 2022. Fast Marching-Cubes-Style Volume Evaluation for Level Set Surfaces. *Journal of Computer Graphics Techniques (JCGT)* 11, 2 (30 June 2022), 30–45.

Andre Pradhana Tampubolon, Theodore Gast, Gergely Klár, Chuyuan Fu, Joseph Teran, Chenfanfu Jiang, and Ken Museth. 2017. Multi-Species Simulation of Porous Sand and Water Mixtures. *ACM Trans. Graph.* 36, 4, Article 105 (jul 2017), 11 pages.

Jerry Tessendorf and Brandon Pelfrey. 2011. The Characteristic Map for Fast and Efficient VFX Fluid Simulations. In *Computer Graphics International Workshop on VFX, Computer Animation, and Stereo Movies*. Computer Graphics Society, Ottawa, Ontario, Canada, 8 pages.

Maxime Theillard. 2021. A volume-preserving reference map method for the level set representation. *J. Comput. Phys.* 442 (2021), 110478.

Hui Wang, Yongxu Jin, Anqi Luo, Xubo Yang, and Bo Zhu. 2020. Codimensional Surface Tension Flow Using Moving-Least-Squares Particles. *ACM Trans. Graph.* 39, 4, Article 42 (aug 2020), 16 pages.

David C. Wiggert and Evan Benjamin Wylie. 1976. Numerical predictions of two-dimensional transient groundwater flow by the method of characteristics. *Water Resources Research* 12 (1976), 971–977.

Chris Wojtan, Matthias Müller-Fischer, and Tyson Brochu. 2011. Liquid Simulation with Mesh-Based Surface Tracking. In *ACM SIGGRAPH 2011 Courses* (Vancouver,

British Columbia, Canada) *(SIGGRAPH '11)*. Association for Computing Machinery, New York, NY, USA, Article 8, 84 pages.
Chris Wojtan, Nils Thürey, Markus Gross, and Greg Turk. 2009. Deforming Meshes That Split and Merge. *ACM Trans. Graph.* 28, 3, Article 76 (jul 2009), 10 pages.
Wen Zheng, Jun-Hai Yong, and Jean-Claude Paul. 2006. Simulation of Bubbles. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Vienna, Austria) *(SCA '06)*. Eurographics Association, Goslar, DEU, 325–333.
Wen Zheng, Bo Zhu, Byungmoon Kim, and Ronald Fedkiw. 2015. A new incompressibility discretization for a hybrid particle MAC grid representation with surface tension. *J. Comput. Phys.* 280 (2015), 96–142.
Yongning Zhu and Robert Bridson. 2005. Animating Sand as a Fluid. *ACM Trans. Graph.* 24, 3 (jul 2005), 965–972.

A CELL-BASED HERMITE INTERPOLATION

Suppose that we are interpolating values of a function f within a two-dimensional axis-aligned cell, the bicubic Hermite scheme suggests us with

$$f(x,y) = \sum_{p,q,r,s\in\{0,1\}} \omega_{pq,rs}\left(\frac{x-x_{\min}}{\Delta x}, \frac{y-y_{\min}}{\Delta y}\right) f_{pq,rs}, \tag{22}$$

where lengths of the two cell sides are defined by $\Delta x = x_{\max} - x_{\min}$ and $\Delta y = y_{\max} - y_{\min}$, respectively.

Here, $f_{pq,rs}$ represents the original function value, the first partial derivatives, and the second order mixed partial derivative, in which $f_{00,rs}$, $f_{01,rs}$, $f_{10,rs}$, and, $f_{11,rs}$ are the values respectively sampled at the lower-left, lower-right, upper-left, and upper-right nodes of the cell. Besides, r and s denote the order of partial derivatives w.r.t. x and y, respectively. The weight function takes the form of

$$\omega_{pq,rs}(\theta_1,\theta_2) = w_{p,r}(\theta_1)\, w_{q,s}(\theta_2), \tag{23}$$

where $\omega_{p,r}$, as well as $\omega_{q,s}$, is a polynomial of at most degree 3, defined as follows

$$\omega_{p,r}(\theta) = \begin{cases} f(\theta), & p=0 \wedge r=0, \quad (24a)\\ f(1-\theta), & p=1 \wedge r=0, \quad (24b)\\ g(\theta), & p=0 \wedge r=1, \quad (24c)\\ -g(1-\theta), & p=1 \wedge r=1, \quad (24d)\end{cases}$$

where $f(\theta) = 2\theta^3 - 3\theta^2 + 1$ and $g(\theta) = \theta^3 - 2\theta^2 + \theta$ hold.

Since the hybrid derivative $\partial f^2/\partial x\partial y$ is not maintained during gradient augmentation, before performing the above interpolation, we evaluate the hybrid derivative by a central difference method as follows:

$$\partial^2_{xy} f_{ij} = \frac{\partial_y f_{i+1,j} - \partial_y f_{i-1,j}}{2\Delta x} + O(\Delta x^2). \tag{25}$$

It is proved that a second-order accurate scheme is sufficient here for fourth-order interpolation.

In a three-dimensional case, the tricubic Hermite interpolation is similar to that described above, except that the number of hybrid derivatives is four, namely $\partial^2_{xy} f$, $\partial^2_{xz} f$, $\partial^2_{yz} f$, and $\partial^3_{xyz} f$, which should all be acquired by central differences.