Journal of Computational Physics 515 (2024) 113262

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A high-order fully Lagrangian particle level-set method for dynamic surfaces

Lennart J. Schulze [a,b,c], Sachin K.T. Veettil [a,b,c], Ivo F. Sbalzarini [a,b,c,*]

[a] Dresden University of Technology, Faculty of Computer Science, Nöthnitzer Str. 46, 01187 Dresden, Germany
[b] Max Planck Institute of Molecular Cell Biology and Genetics, Pfotenhauerstr. 108, 01307 Dresden, Germany
[c] Center for Systems Biology Dresden, Pfotenhauerstr. 108, 01307 Dresden, Germany

ARTICLE INFO

Dataset link: https://git.mpi-cbg.de/mosaic/software/parallel-computing/openfpm/openfpm_numerics

MSC:
65D18
76M28
76T99
35Q30

Keywords:
Level-set methods
Particle methods
Geometric computing
Multi-phase flow
Closest point transform
Dynamic surfaces

ABSTRACT

We present a fully Lagrangian particle level-set method based on high-order polynomial regression. This enables meshfree simulations of dynamic surfaces, relaxing the need for particle-mesh interpolation. Instead, we perform level-set redistancing directly on irregularly distributed particles by polynomial regression in a Newton-Lagrange basis on a set of unisolvent nodes. We demonstrate that the resulting particle closest-point (PCP) redistancing achieves high-order accuracy for 2D and 3D geometries discretized on irregular particle distributions and has better robustness against particle distortion than regression in a monomial basis. Further, we show convergence in classic level-set benchmark cases involving ill-conditioned particle distributions, and we present an example application to multi-phase flow problems involving oscillating and dividing droplets.

1. Introduction

The numerical representation of non-parametric surfaces is a key part of many spatio-temporal simulations, e.g., in additive manufacturing [1], geology [2], and biology [3]. This has motivated research into geometric computing algorithms that can achieve high accuracy for representing non-parametric surfaces at low computational cost. Ideally, the algorithms should also be parallelizable in order to leverage high-performance-computing and GPU resources.

Due to their geometric expressiveness and parallelizability, level-set methods [4] have emerged as a popular approach to geometric computing for non-parametric surfaces. They have been successfully applied to real-world problems involving complex geometries, such as the growth, division, and reconnection of tumors [5] and the diffusion of membrane proteins on the highly curved endoplasmic reticulum [3]. In level-set methods, a two-dimensional surface embedded in a three-dimensional space is described implicitly as the (usually zero) level-set of a scalar function over the embedding space. Often, the level-set function is chosen to be the signed distance

* Corresponding author.
E-mail addresses: lschulze@mpi-cbg.de (L.J. Schulze), sthekke@mpi-cbg.de (S.K.T. Veettil), sbalzarini@mpi-cbg.de (I.F. Sbalzarini).

https://doi.org/10.1016/j.jcp.2024.113262
Received 9 June 2023; Received in revised form 26 June 2024; Accepted 2 July 2024

Available online 8 July 2024

function (SDF) to the surface. This guarantees that the level-set function is smooth and continuously differentiable near the surface [6], and it simplifies surface computations as the level-set function value directly represents the shortest distance to the surface.

Level-set methods also generalize well to dynamically moving and deforming surfaces, as changes in the shape of the surface amount to advecting the level-set function. When advecting the level-set function, however, the signed-distance property is in general not conserved. Indeed, for deformation velocity fields that do not describe rigid-body motion, the advection of the level-set function in the embedding space destroys the signed-distance property [7]. This can be avoided by correcting the advection velocity away from the surface, for example by variational penalties [8,9] or Lagrange multipliers [10]. These approaches, however, are often inaccurate or accumulate advection errors over time. The most popular approach, therefore, is to recompute the SDF whenever the surface has deformed. This is known as "level-set redistancing", where the SDF in the embedding space is recomputed from the reconstructed current location of the surface.

Several conceptually different methods for level-set redistancing are available: Sussman et al. [7] performed redistancing by evolving an auxiliary Partial Differential Equation (PDE) in pseudo-time, which has the SDF as a steady-state solution. Their approach achieves high accuracy on regular Cartesian grids and avoids numerical instabilities of the pseudo-time evolution by using higher-order ENO/WENO finite-difference schemes [11]. The resulting algorithm, however, is computationally expensive as it amounts to evolving a PDE to steady state in the embedding space with sometimes stringent time-step limitations [12]. The point of computational efficiency has been addressed by fast marching or sweeping methods [13–15], which propagate the level-set values from the surface outward as a moving front. While this is computationally more efficient, it is limited to lower-order finite-difference schemes for which the resulting algebraic equations can be analytically solved, and it creates data dependencies that hamper parallelization [16]. A third approach therefore aims to directly compute the distance to the surface independently for all points in the embedding space by finding for each query point the closest point on the surface [17]. This closest-point (CP) transform has mainly found application in the numerical solution of surface PDEs [18,19] and has since been extended to level-set redistancing using higher-order polynomial regression [20]. The resulting method achieves high orders of accuracy, is computationally efficient, and parallelizable. While originally formulated for regular Cartesian grids, it has recently also been demonstrated on unstructured grids [21] and triangulations [22], confirming the versatility of the CP approach.

Despite the efficiency and versatility of CP redistancing, however, algorithms that achieve high orders of convergence are so far limited to connected meshes and discretize level-set advection in an Eulerian frame of reference by evolving the level-set function values at the mesh nodes. In such an Eulerian mesh-based approach, fulfilling conservation laws becomes nontrivial, and the numerical stability and adaptivity of the overall framework is limited by the CFL condition. Both limitations can be relaxed when discretizing level-set advection in a Lagrangian frame of reference, where the discretization points move with the local advection velocity and preserve their level-set function values. Lagrangian level-set methods have been shown to be highly geometry-adaptive with excellent numerical stability [12] while maintaining the general conservation properties [23] of Lagrangian particle methods.

In Lagrangian particle methods, however, particle distributions become increasingly irregular as particles move with the advection velocity. This hampers level-set redistancing, for which so far only first-order methods exist that aim to regularize the level-set function by renormalization [24,25]. Previous higher-order approaches exclusively operate on grids, which requires interpolation of the level-set function values from the Lagrangian particles to the grid nodes before redistancing [26,12]. This introduces additional computational cost and interpolation errors. While high-order particle-to-mesh interpolation schemes exist [27], they are based on conservation laws for the moments of the represented function, which do not apply to level-set functions. While again renormalization approaches have been proposed to address the problem [28], their convergence is limited to linear order.

Here, we present a fully Lagrangian particle CP level-set redistancing method that achieves higher-order convergence without requiring interpolation to a structured or unstructured intermediate mesh. The method directly operates on Lagrangian particles, maintaining their conservation properties and stability, while simplifying level-set advection for dynamic surfaces by inheriting the accuracy, computational efficiency, and parallelizability of CP redistancing. In the present method, particles in a narrow band around the surface carry and advect the level-set function values. After advection, the SDF is recomputed directly on the irregularly distributed particles by finding their respective closest points on the surface. We do this using high-order polynomial regression in a Newton-Lagrange basis on unisolvent nodes. The analytical form of the local regression polynomials enables straightforward computation of derivative geometric quantities, such as surface normals and local curvatures. The regression nodes are a suitably chosen subset of particles in the narrow band around the surface. We show that a clever choice of local unisolvent nodes maintains high-order convergence even on highly irregular particle distributions in the narrow band. We discuss accuracy, stability, and performance of the method on benchmark geometries with analytically known SDF, a vortex flow problem, and a multi-phase hydrodynamics application involving oscillating and dividing droplets. In the latter, we compare the present approach with a Smoothed Particle Hydrodynamics (SPH) method [29].

2. Level-set method

Level-set methods describe an evolving surface Γ_t implicitly as the zero level-set of a scalar function ϕ:

$$\Gamma_t = \{\mathbf{x} \,:\, \phi(\mathbf{x}, t) = 0\}, \tag{1}$$

with $\mathbf{x} \in \mathbb{R}^{n_d}$ being the coordinates in the n_d-dimensional space of real numbers, and t being the time. Due to favorable properties in accuracy and volume conservation, as well as simplified computations of surface-geometric quantities such as normals and curvatures, a popular choice for the level-set function is a signed distance towards the surface:

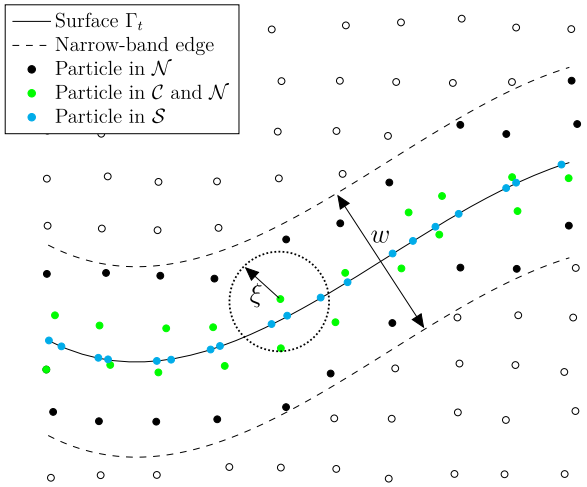


Fig. 1. Nomenclature of the method shown for a 2D domain containing a piece of a surface (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

$$\phi(\mathbf{x}) = \pm\|\mathbf{cp}(\mathbf{x}) - \mathbf{x}\|_2, \tag{2}$$

in which $\mathbf{cp}(\mathbf{x})$ is the closest-point function that yields the closest point of a given location on the surface, measured in L_2-distance. The sign of the level-set function (2) is chosen as positive if outside, and negative if inside of a closed surface. A natural property of the SDF in Eq. (2) is that its gradient has unit length,

$$\|\nabla\phi(\mathbf{x})\|_2 = 1. \tag{3}$$

Level-set functions also allow for computing derivative fields associated with the surface, such as the surface normal field

$$\mathbf{n}(\mathbf{x}) = \frac{\nabla\phi(\mathbf{x})}{\|\nabla\phi(\mathbf{x})\|_2} \tag{4}$$

and the mean curvature field in its fluid-mechanical definition (from now on simply referred to as "curvature")

$$\kappa(\mathbf{x}) = \nabla\cdot\mathbf{n}(\mathbf{x}). \tag{5}$$

The surface Γ_t can move and deform with velocity $\mathbf{u}(\mathbf{x},t)$ over time t. After any movement, material points lying on the surface remain on the surface:

$$\frac{\mathrm{D}\phi}{\mathrm{D}t} = 0. \tag{6}$$

Eq. (6) is formulated using the material derivative $\frac{\mathrm{D}(\cdot)}{\mathrm{D}t} = \frac{\partial(\cdot)}{\partial t} + \mathbf{u}\cdot\nabla(\cdot)$ and generally only holds for material points on the surface, i.e. on the zero level-set. Material points surrounding the surface can either approach to or recede from the surface, which is not accounted for by Eq. (6), as the velocity field $\mathbf{u}$ is embedded in the 3D space. If changes in distance towards the surface are not taken into account, the level-set function ceases to be the SDF and the property (3) is lost, hampering the computation of surface normals and local curvatures. Therefore, level-set redistancing is used to restore the SDF property.

3. Redistancing on irregular particle distributions

We represent the surface Γ_t using a finite set of n_p discrete Lagrangian particles P_i, $i \in \{1,\dots,n_p\}$. These particles store and advect the level-set function values ϕ_i in addition to their position. At the beginning of a simulation, the particles are usually seeded uniformly or uniformly at random with inter-particle spacing h in a tubular neighborhood of diameter w around the surface, which we call "narrow band" in accordance with the usual level-set terminology. Then, the particles move with the deformation velocity $\mathbf{u}$ of the surface, and the particle distribution can become arbitrary.

This Lagrangian formulation offers a number of benefits: rigid-body movements of the surface are simulated exactly [23], aside from time integration errors, and the method has better time-integration stability as the Lagrangian CFL number is larger than the Eulerian CFL number [30], and no grid data structure needs to be allocated and maintained.

We extend the CP redistancing proposed by Saye [20] to irregularly distributed particles in a narrow band of width w around the surface, as illustrated in Fig. 1. The set containing all particles of the narrow band is $\mathcal{N} = \{\mathsf{P}_i : |\phi(\mathbf{x}_i)| < w/2\}$.

Before redistancing, an additional set of *sample particles* $\mathcal{S}$ that lie precisely on the surface is created. For this, particles immediately adjacent to the surface are used as starting points (set $\mathcal{C}$ in Fig. 1). Using local polynomial regression over neighborhoods of such

near-surface particles, the level-set function $\phi(\mathbf{x})$ is locally approximated. The polynomial approximation is then used to project the location of the particle onto the zero level-set, yielding the corresponding sample particle.

For redistancing, the following steps are then computed for all particles in $\mathcal{N}$: First, we find the closest sample particle using a cell-list acceleration data structure [31], which serves as an initial guess for solving a constrained optimization problem. This problem minimizes the distance between the query point and another point under the constraint that the polynomially approximated level-set function at the location of the other point is zero. This is optimized over the same local regression polynomial used to generate the sample particles and yields the closest point on the zero level-set of the local regression polynomial. The distance between the so-computed closest point and the query particle is the corrected level-set value that restores the SDF.

This procedure crucially hinges on the numerical properties of the polynomial regression scheme used. In the spirit of particle methods, we perform polynomial regression in a local radial neighborhood of a particle. The radius r_c of this neighborhood needs to include at least as many particles as are required to determine the coefficients of the regression polynomial for a given polynomial degree. Further, the spatial arrangement of the particles within the regression neighborhood cannot be dependent in a way that renders the Vandermonde matrix of the regression problem singular. In the next section, we therefore pay particular attention to how polynomial regression is done here.

3.1. Local polynomial approximation of the level-set function

The sample particles provide a coarse estimate of the surface and are obtained by first finding a set $C \subset \mathcal{N}$ containing particles in the narrow band that are close to the surface. A particle is in C if it has another particle with opposite sign of the level-set function value within a certain radius ξ. This radius is a parameter of the method. Larger thresholds create more sample particles, thus improving the sampling of the surface, but causing higher computational cost. In this paper, we use a threshold radius of $\xi = 1.5h$ throughout. Then, C is a small subset of $\mathcal{N}$.

For each particle $\mathsf{P}_i \in C$, we approximate the level-set function as a continuous polynomial obtained by local least-squares regression. As we show in Sec. 4.1.1, this is simpler and more accurate than directly using a particle-function approximation of the level-set function. Using n_c monomials $M_k(\mathbf{x})$, the local approximation of the level-set function reads:

$$\phi(\mathbf{x}) \approx p_i(\mathbf{x}) = \sum_{k=0}^{n_c-1} c_k M_k(\mathbf{x}) . \tag{7}$$

The choice of the number and type of monomials can be made depending on the requirements of the application.

We determine the coefficients c_k of $p_i(\mathbf{x})$ by iterating over the neighborhood of a particle $\mathsf{P}_i \in C$ that contains n_n particles including P_i itself. In this neighborhood, we assemble the regression matrix $\mathbf{A} \in \mathbb{R}^{n_n \times n_c}$, the unknown coefficient vector $\mathbf{c} \in \mathbb{R}^{n_c}$, and the right-hand side vector containing the level-set function values $\boldsymbol{\phi} \in \mathbb{R}^{n_n}$. Then, the linear system of equations

$$\mathbf{A}\mathbf{c} = \boldsymbol{\phi} \tag{8}$$

is solved using orthogonal decomposition. If $n_n = n_c$ and $\mathbf{A}$ has full rank, the polynomial $p_i(\mathbf{x})$ is the interpolation polynomial. If $n_n > n_c$, $p_i(\mathbf{x})$ is the least-squares solution for the polynomial regression problem.

The condition number of the regression matrix depends on the distribution of particles and the choice of basis $M_k(\mathbf{x})$. We do not have control on the former, but we can choose the polynomial basis $M_k(\mathbf{x})$. As a basis, we use the Lagrange polynomials $L_k(\mathbf{x})$ determined on a Chebyshev-Lobatto grid $G \subseteq [-1,1]^{n_d}$ in the regression neighborhood, i.e., $L_k(\mathbf{q}_l) = \delta_{k,l}$, $\mathbf{q}_l \in G$, where $\delta_{k,l}$ is the Kronecker delta [32]. Therefore, the points at which the polynomial basis is computed (i.e., the nodes of a local Chebyshev-Lobatto grid) in general differ from the points $p_i(\mathbf{x})$ at which the regression polynomial is evaluated. We use basis polynomials of total degree (ℓ_1-degree) [32]. The regression matrix is constructed by evaluating the Newton form of the Lagrange basis polynomials on the regression nodes. This choice of basis has been shown to effectively regularize regression over randomly distributed points for a large class of analytic functions [33]. We refer to this regression approach as *minter* regression.

Since Eq. (6) holds on the surface after any movement, solving Eq. (8) for each particle in C yields a polynomial representation of the surface as the zero level-set of the local regression polynomials $p_i(\mathbf{x})$. Using this representation in the proximity of the "center" particle $\mathsf{P}_i \in C$, the locations of the sample particles S can be determined. This is done by iterative projection onto the zero level-set of $p_i(\mathbf{x})$, using $\mathsf{P}_i \in C$ as a starting point and iterating

$$\mathbf{x}^{k+1} = \mathbf{x}^k - p_i(\mathbf{x}^k) \frac{\nabla p_i(\mathbf{x}^k)}{\|\nabla p_i(\mathbf{x}^k)\|_2^2}, \qquad k = 0, 1, 2, \ldots \tag{9}$$

The iteration is stopped as soon as $|p_i\left(\mathbf{x}^k\right)| < \varepsilon$ for a user-defined tolerance ε.

Doing so for all particles in C completes the sample particle set S, as illustrated in Fig. 1. The sample particles provide starting points for the subsequent search for the closest point of any query point (also between particles). They further act to store the regression polynomials and therefore the local geometry of the surface. The resolution with which S samples the surface results from the particle distribution around the surface: Each particle in C, from both sides of the surface, creates one sample particle. For example, a straight 1D surface of length l embedded in 2D space, in which the domain has been discretized using an inter-particle spacing of h, would contain $2l/h$ sample particles.

3.2. Finding the closest point on the surface for a given particle

From the sample particles $\mathcal{S}$ and the associated polynomials, the distance of any given location towards the surface can be computed. The query point may also lie between particles or outside of the narrow band. In level-set redistancing, however, typically the query points are the particles $\mathcal{N}$ within the narrow band.

For each query particle $\mathsf{P}_q \in \mathcal{N}$ at position $\mathbf{x}_q$, we look for the closest sample particle $\mathsf{P}_s \in \mathcal{S}$, located at position $\mathbf{x}_s$. In a well-sampled surface, we know that the closest point of P_q lies in a neighborhood of its closest sample particle. Hence, the zero level-set of the regression polynomial p_s is used as a local approximation of the surface. To find the closest point $\mathbf{x}$ of P_q on the approximated surface, we solve the constrained optimization problem

$$\arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{x}_q\|_2^2, \quad \text{s.t. } p_s(\mathbf{x}) = 0, \tag{10}$$

minimizing the distance under the constraint that the solution lies on the zero level-set of the regression polynomial. We reformulate this problem using a Lagrange multiplier λ and the associated Lagrangian

$$\mathcal{L}(\mathbf{x},\lambda) = \frac{1}{2}\|\mathbf{x}-\mathbf{x}_q\|_2^2 + \lambda p_s(\mathbf{x}). \tag{11}$$

Stationary points of the Lagrangian fulfill

$$\nabla_{\mathbf{x},\lambda}\mathcal{L} = \begin{pmatrix} \mathbf{x}-\mathbf{x}_q + \lambda\nabla p_s(\mathbf{x}) \\ p_s(\mathbf{x}) \end{pmatrix} = \mathbf{0} \tag{12}$$

and are found using the Newton method. The subscripts $\mathbf{x}$ and λ indicate differential operators with respect to both variables. As an initial guess $\mathbf{x}^0$ for the iterative Newton method, we use the location of the closest sample particle: $\mathbf{x}^0 = \mathbf{x}_s$. The initial Lagrange multiplier is $\lambda^0 = \left(\mathbf{x}_q - \mathbf{x}^0\right)\cdot\nabla p_s(\mathbf{x}^0)/\|\nabla p_s(\mathbf{x}^0)\|_2^2$. Subsequently, we iterate

$$\begin{pmatrix} \mathbf{x}^{k+1} \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{x}^{k} \\ \lambda^{k} \end{pmatrix} - \left(\mathrm{H}_{\mathbf{x},\lambda}\mathcal{L}(\mathbf{x}^k,\lambda^k)\right)^{-1}\nabla_{\mathbf{x},\lambda}\mathcal{L}(\mathbf{x}^k,\lambda^k), \tag{13}$$

where H is the Hessian, computed as

$$\mathrm{H}_{\mathbf{x},\lambda}\mathcal{L}(\mathbf{x},\lambda) = \begin{pmatrix} I + \lambda\mathrm{H}p_s(\mathbf{x}) & \nabla p_s(\mathbf{x}) \\ \nabla p_s(\mathbf{x})^\top & 0 \end{pmatrix}. \tag{14}$$

We perform the iterations in Eq. (13) until the L_2-norm of the gradient in Eq. (12) falls below the tolerance ε, or a maximum number of iterations $k_{\max}$ is reached.

If the Newton iterations stray out of the support neighborhood of the sample point P_s, this can be detected and a new iteration can be started from the corresponding neighboring sample point. In all benchmarks presented in this paper, however, we did not encounter such a case.

Following the definition in Eq. (2), we use the resulting approximation of the closest point $\mathbf{x} \approx \mathbf{cp}(\mathbf{x}_q)$ to update the level-set function value of the query particle P_q:

$$\phi_q = \mathrm{sgn}(\phi_q)\|\mathbf{x}-\mathbf{x}_q\|_2. \tag{15}$$

Here, sgn denotes the sign function that ensures that the sign of the new SDF value is the same as the sign of the old level-set function value.

3.3. Derivative surface quantities

From the closest point of a given query particle and the polynomial approximation of the level-set function in the vicinity of the closest point, it is straightforward to compute derivative surface quantities, such as the surface normal and the local curvature at the closest point.

We compute the surface normal at the closest point of a query particle P_q as

$$\mathbf{n}_q = \frac{\nabla p_s\left(\mathbf{cp}(\mathbf{x}_q)\right)}{\|\nabla p_s\left(\mathbf{cp}(\mathbf{x}_q)\right)\|_2}, \tag{16}$$

and the local curvature as

$$\kappa_q = \nabla\cdot\mathbf{n}_q. \tag{17}$$

Note that as during the previously outlined redistancing method, the gradient and Hessian of the polynomial are known analytically. Evaluating the derivative at the closest point on the surface also yields a constant normal extension of surface normals and curvature values into the narrow band.

3.4. Narrow-band updates

During level-set advection, particles may enter or exit the narrow band of width w. We assign particles to the narrow-band set $\mathcal{N}$ or its complement $\mathcal{N}^c$ depending on their level-set function value:

$$\mathsf{P}_i \in \begin{cases} \mathcal{N}, & \text{if } |\phi_i| \leq \frac{w}{2} \\ \mathcal{N}^c, & \text{if } |\phi_i| > \frac{w}{2} \end{cases}. \tag{18}$$

As the correct signed-distance function value of a particle is not known before redistancing, we estimate it as the distance to the closest sample particle. We efficiently find the closest sample particle using a cell list of edge length $w/2$ over all sample particles [31]. For particles far away from the surface, all surrounding cells are empty. If the cells contain at least one sample particle, the closest sample particle is found by computing the distances. If the distance to the closest sample particle is less than $w/2$, the query particle is assigned to the narrow band and its exact distance to the surface needs to be recomputed using the approach outlined in Sec. 3.2.

3.5. Implementation

We implement the presented Particle Closest-Point (PCP) method in C++ in the scalable scientific computing framework *OpenFPM* [34] using the *minter* package [35] and the *Eigen* library [36] for polynomial regression. Parallelization is transparently provided by using the internally distributed data structures of *OpenFPM* as described [34,37]. No further parallelization primitives or communication routines are implemented in this work. The implementation is available as open-source from https://git.mpi-cbg.de/mosaic/software/parallel-computing/openfpm/openfpm_numerics.

The implementation provides a PCP class of which an object can be initialized by passing the set of particles containing the level-set values. The class contains a method for initialization, and a main method that can subsequently be called for any set of query points. The initialization routine performs the following two steps: (1) Build an internal particle list by iterating through the user's particle list to find all particles in the narrow band whose smallest distance to a particle of opposite level-function sign is less than $r_c + \xi$. If the smallest distance is less than ξ, the particle is simultaneously flagged as a *close particle*. (2) Iterate through the internal particle list and solve the local regression problem for each close particle using the *minter* and *Eigen* libraries. Store the polynomial coefficients as an array-valued property of the respective particle. Subsequently, project onto the surface as described in Eq. (9) to find the sample particles, which are also stored as an array-valued property of the close particle.

Once initialization has been performed, the main method of the class can be called for any set of query points. The set of query points can be identical to, a subset of, or disjoint from the set of particles used for initialization. The main method iterates through the list of query points to find for each query point the closest sample particle in the internal particle list constructed during initialization. It then solves the constrained optimization problem for this sample particle to compute the results (closest point on the surface, SDF value, surface normal and curvature), and it returns the results for each query point as particle properties.

Creating and working on the (smaller and temporary) internal particle list avoids memory access to the (larger) particle list of the user program. It also facilitates integration into existing user code, since the user only needs to provide those particle properties that actually store the geometric information of interest. The separate internal particle list also allows for the PCP class to internally use a different cell list in order to not impose additional ghost layers onto the user code. The cutoff radius that the user code uses to determine particle interactions can therefore be different (and generally smaller) from the one used by PCP for determining candidate sample particles for a given query particle.

4. Results

We characterize the accuracy and efficiency of the method in benchmarks ranging from basic geometries with known analytical SDF over standard dynamic-surface test cases to a multi-phase flow problem with interfacial effects and changes in topology.

4.1. Basic geometries

In his work on mesh-based closest-point redistancing, Saye [20] presents results for elementary geometries such as 2D ellipses and 3D ellipsoids. To demonstrate the utility of a method tailored to data on irregularly distributed particles, we first highlight the problems of the apparent alternative of interpolating particle data to mesh nodes and subsequently applying a mesh-based method. Then, we compare it with the proposed PCP method directly on the particles.

Irregular particle distributions for the benchmarks are obtained by randomly shifting the nodes $\mathbf{m}_{ij}$ of a regular Cartesian mesh with spacing h:

$$\mathbf{x}_p = \mathbf{m}_{ij} + \mathbf{X}, \tag{19}$$

with random shifts

$$X_d = \alpha h \mu, \tag{20}$$

for $d = \{1, \ldots, n_d\}$. The pseudo-random variables $\mu \sim \mathcal{U}[-1, 1]$ are *i.i.d.* from the uniform distribution over the interval $[-1, 1]$. The shift amplitude α is always chosen < 0.5 to ensure that no two particles coincide.

Table 1
Smoothing factors ϵ/h used for the Wendland C2 and Gaussian kernels in the remeshing convergence test.

h	1/32	1/64	1/128	1/256	1/512	1/1024	1/2048
ϵ/h	1.3	2.0	3.0	4.0	5.0	6.0	6.0

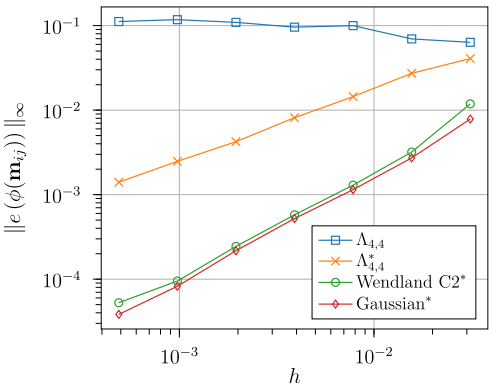


Fig. 2. Convergence of different particle-mesh interpolation schemes (symbols, inset legend) applied to the ellipse Eq. (21) discretized on irregularly ($\alpha = 0.3$) distributed particles. The star in the legend entries indicates that the respective remeshing formulation is renormalized by the particle volumes.

4.1.1. Remeshing followed by closest-point redistancing

As a baseline, we first characterize the classic approach of interpolating the particle values to a grid, followed by CP redistancing [20]. For this, we consider an ellipse in the 2D domain $[-1,1]\times[-1,1]$ discretized by particles with spacing h and shift amplitude $\alpha = 0.3$ covering the entire domain with periodic boundary conditions. The level-set values at the particles are initialized to

$$\phi(x,y) = 1 - \sqrt{\frac{x^2}{A^2} + \frac{y^2}{B^2}}\,, \tag{21}$$

for an ellipse with semi-major axis $A = 0.75$ and semi-minor axis $B = 0.5$. The zero level-set of Eq. (21) coincides with the zero level-set of the SDF of the ellipse. Away from the surface, however, this is not a SDF. We compute a SDF approximation by remeshing followed by mesh-based CP redistancing according to Saye [20].

We compare a variety of remeshing schemes as described in Appendix A. We compute mesh-node values using the $\Lambda_{4,4}$ kernel in both the basic formulation (A.1) and the renormalized formulation (A.2), as well as using the renormalized Wendland C2 and Gaussian kernel functions (A.5). We compare the remeshing results with the analytically known exact values ϕ_{exact} obtained by evaluating Eq. (21) at the mesh nodes. The absolute remeshing error on each mesh node is then computed as $e\left(\phi(\mathbf{m}_{ij})\right) = |\phi(\mathbf{m}_{ij}) - \phi_{\text{exact}}(\mathbf{m}_{ij})|$. To characterize the convergence behavior, we increase the number of particles by decreasing the spacing h. For the Wendland and Gaussian kernels, we simultaneously increase the fraction ϵ/h according to Table 1 as required [38]. The maximum error over all mesh nodes in a narrow band of radius $\frac{w}{2} = \frac{1}{16}$ around the zero level-set is plotted in Fig. 2.

As expected, the $\Lambda_{4,4}$ kernel in the basic formulation (Eq. (A.1)) does not converge with its theoretical order of four, as it is derived from moment conservation laws that do not hold for level-set functions. In fact, it diverges slowly, which is expected since the maximum error occurs in the most irregular particle neighborhood with an irregularity proportional to the total number of randomly perturbed particles in the domain. Convergence is restored with order 0.81 (still far from the theoretical 4) when renormalizing function values by the amount of contributions they experienced. If in addition to renormalizing function values, the contributions from individual particles are weighted by respective particle volumes, and the smoothing lengths of the kernel functions are increased sufficiently, convergence of order 1.3 is achieved for the Wendland C2 and Gaussian kernels. For these two, however, there is a noticeable effect from the irregular particle distribution, as they are expected to converge with order two for particles distributed on a regular Cartesian grid.

We next test if CP redistancing applied to the interpolated mesh values can restore an overall higher order of convergence. We therefore plot the maximum error in the approximated SDF over the same narrow band in Fig. 3. The reference SDF values were obtained using the method outlined in Ref. [39]. CP redistancing is done using fourth-order polynomials with a monomial basis of total degree (referred to as "Taylor 4" in Ref. [20]), such that we expect fifth-order convergence overall [20]. Due to the intermediate remeshing step, however, the errors converge slower than if redistancing was based on error-free mesh data. When remeshing with the standard or renormalized $\Lambda_{4,4}$ method, the overall convergence is of order 0.95 and 0.96, respectively. When remeshing with the Wendland C2 and Gaussian kernels, overall convergence in the SDF error is of order 1.3.

In all cases where the remeshing itself converges, the overall redistancing accuracy is therefore limited by the particle-mesh interpolation error. An interesting exception is the $\Lambda_{4,4}$ kernel without renormalization, which does not converge by itself but convergence is restored by CP redistancing. This is because CP redistancing uses a fixed number of grid layers around the surface, rather than a

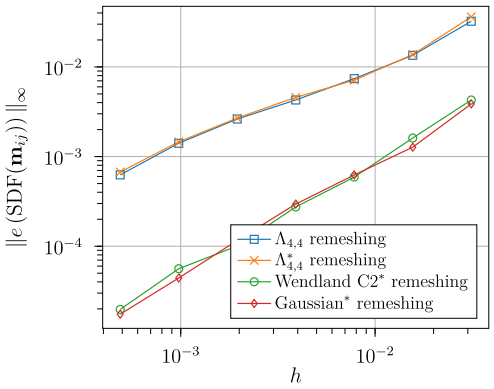


Fig. 3. Convergence of the maximum error in the SDF after mesh-based CP redistancing of the remeshed level-set values of the 2D ellipse using different remeshing schemes. The star in the legend entries indicates that the respective remeshing formulation is renormalized.

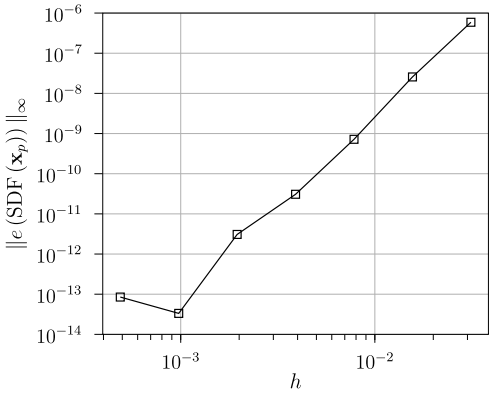


Fig. 4. Convergence of the SDF computed by the present particle closest-point (PCP) method for the 2D ellipse case with irregularly ($\alpha = 0.3$) distributed particles and fourth-order minter regression polynomials. The observed convergence order is 4.8 (theoretical: 5) with a numerical solver tolerance of 10^{-14}.

fixed narrow-band width. If the absolute remeshing error is evaluated in a fixed number of grid layers, rather than a fixed-size narrow band, convergence of order 0.9 is also seen for the remeshing alone. This is because the level-set function has smaller absolute values closer to the surface. The relative error per mesh node nevertheless remains constant.

In summary, first interpolating from Lagrangian particles to a regular Cartesian mesh prevents mesh-based CP redistancing from reaching its design order of convergence as particle-mesh interpolation limits the overall accuracy.

4.1.2. Particle closest-point redistancing

We compare the above results with the present PCP redistancing method for the same ellipse example with the same irregular particle distribution ($\alpha = 0.3$). We use a fourth-order minter basis and a cutoff radius of $r_c = 2.5h$. We set the tolerance $\varepsilon = 10^{-14}$ and the maximum number of iterations $k_{\max} = 1000$. For the simple ellipse geometry, both the iterative projections and the Newton algorithm converge quickly, i.e. in 2 to 4 iterations, such that $k_{\max}$ has no effect on the results.

Fig. 4 shows the convergence in the SDF. In absence of any grid, the errors are now evaluated directly on the query particles P_q in the entire narrow band. Even for coarse resolutions the error is orders of magnitude smaller than any accuracy achieved by remeshed CP redistancing. The theoretical fifth-order convergence is approximately reached, with a measured order of 4.8, until the error plateaus near the set tolerance $\varepsilon = 10^{-14}$.

We next assess the convergence of the PCP method for computing derivative surface quantities in a 3D narrow band around an ellipsoid with semi-major axis $A = 0.75$ and both semi-minor axes $B = C = 0.5$. We discretize the level-set function on irregularly distributed particles ($\alpha = 0.3$) within a narrow band of width $w = 12h$ and use a tolerance of $\varepsilon = 10^{-14}$. The maximum error is reported over the entire narrow band in Fig. 5 for fourth- and fifth-order minter regression. For the fourth-order polynomials a cutoff radius of $r_c = 2.4h$ is used, for the fifth-order polynomials $r_c = 2.6h$.

The theoretical convergence orders are almost achieved when using fourth-order minter regression: the SDF converges with order 4.9, the closest-point transform $\mathbf{cp}(\mathbf{x}_p)$ with 4.8, the surface normals $\mathbf{n}(\mathbf{x}_p)$ with 3.8, and the local curvature $\kappa(\mathbf{x}_p)$ with 3.0 until they plateau at the highest resolution. The SDF and closest-point function converge with the same order and are separated by a constant offset. Since the normal field and local curvatures are computed as the derivatives of the SDF, they are third- and second-order polynomials, respectively, with theoretical convergence orders of 4 and 3.

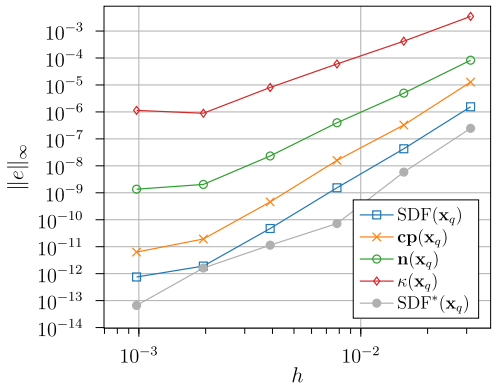


Fig. 5. Convergence of the SDF (observed: 4.9, theoretical: 5), locations of the closest points (observed: 4.8, theoretical: 5), surface normals (observed: 3.8, theoretical: 4), and local curvatures (observed: 3.0, theoretical: 3) computed using the PCP method for a 3D ellipsoid discretized on irregular particles ($\alpha = 0.3$). SDF* is computed using fifth-order polynomials (observed order until $h = 1/128$: 5.9, theoretical: 6), whereas all other results use fourth-order polynomials.

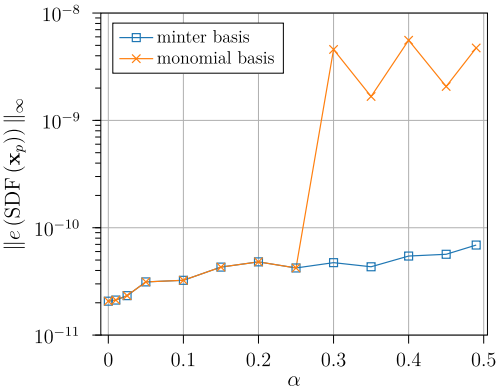


Fig. 6. Maximum error of the SDF for the 3D ellipsoid case with $h = \frac{1}{256}$ for increasing irregularity of the particle distribution from a regular grid $\alpha = 0$ to the maximum possible irregularity. Results are computed using fourth-order bases for both minter regression and regression using monomial basis functions.

Using fifth-order polynomials increases both accuracy and convergence order (to 5.9) especially for the lower resolutions between $h = \frac{1}{32}$ until $h = \frac{1}{128}$ where round-off errors do not dominate. The convergence order of the other quantities computed with fifth-order PCP is also close to optimal (not shown in the figure to avoid clutter): 6.4 for $\mathbf{cp}(\mathbf{x}_p)$, and 5.2 for $\mathbf{n}(\mathbf{x}_p)$, and 4.0 for $\kappa(\mathbf{x}_p)$.

To test the robustness of the PCP method to distortions in the particle distribution, we vary the shift amplitude α and compare the results computed using a minter basis with baseline results obtained using a monomial basis. For both, we use fourth-order polynomials of total degree and a cutoff radius of $r_c = 2.4h$. We consider the same ellipsoid as before with resolution fixed to $h = \frac{1}{256}$. Fig. 6 shows the maximum error of the SDF within the narrow band for α from 0 (i.e., regular Cartesian grid) to 0.49. As expected, the errors grow for increasing irregularity of the particle distribution. Comparing the minter basis with the monomial basis, the errors start to differ at $\alpha = 0.25$. Beyond this point, the error on the monomial basis grows by two orders of magnitude as the regression problem becomes ill-conditioned. This demonstrates that indeed the minter basis and Chebyshev-Lobatto subgrids regularize the regression problem sufficiently to allow convergent redistancing on highly distorted particle distributions.

Finally, we confirm that the PCP method behaves as expected when applied to non-smooth surfaces. For this, we perform PCP redistancing for a rounded rectangle (C^1) and a square (C^0). On irregular particle distributions ($\alpha = 0.3$), the SDF computed using the PCP method converges with order 2 for the rounded rectangle and order 1 for the square. These are the same convergence orders as for the mesh-based CP method [20], agnostic towards the chosen regression basis and order (> 0), since a smooth polynomial cannot describe the jump in the curvature (rounded rectangle) or the normals (square). Hence, the leading error term is always limited by the smoothness of the shape itself.

4.1.3. Parallel scalability

We test how well our *OpenFPM* implementation of PCP redistancing scales to multiple processor (CPU) cores. For this, we again use the 3D ellipsoid test case with fixed $h = \frac{1}{512}$, $w = 12h$, and $\varepsilon = 10^{-14}$ with a fourth-order minter basis and $r_c = 2.4h$. We measure the wall-clock time $t_{\text{cpu}}\left(n_{\text{cpu}}\right)$ for different numbers of processor cores n_{cpu} on an AMD Ryzen Threadripper 3990X CPU with 64 cores and 256 GB of shared memory. We report the parallel speedup

$$S(n_{\text{cpu}}) = \frac{t_{\text{cpu}}(1)}{t_{\text{cpu}}(n_{\text{cpu}})} \tag{22}$$

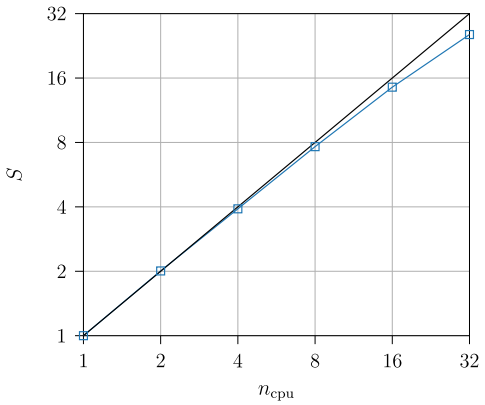


Fig. 7. Speedup S over number of processor (CPU) cores n_{cpu}. The solid black line marks the ideal linear speedup.

for this strong scaling (i.e., fixed problem size) in Fig. 7, showing near-optimal scalability. For $n_{\text{cpu}} = 32$, the parallel efficiency $E(n_{\text{cpu}}) = S(n_{\text{cpu}})/n_{\text{cpu}}$ is about 80%. The 20% communication overhead is consistent with the volume of the ghost layers required to localize the closest sample particle. Another likely reason is an uneven distribution of load amongst the processors. While we ensured that all processors have the same amount of particles, this does not necessarily result in the same surface area, and therefore $|\mathcal{C}|$, per processor. Hence there is a possible load imbalance during local polynomial regression and surface sampling.

4.2. Spiraling vortex

Moving beyond basic geometric shapes, and into dynamically deforming surfaces, we next consider a classic level-set benchmark case in which a circle is stretched and spiraled by an advection velocity field $\mathbf{u}(\mathbf{x})$. The spiraling vortex reaches a state of maximum distortion, and subsequently the velocity field is reversed. At the end, an ideal method would recover the initial circle. In practice, however, numerical errors in the advection of the level-set function and the periodic redistancing accumulate in a final geometry that differs from the initial circle. This difference can be used to compare methods.

For Lagrangian particle methods without remeshing, this test case also introduces another challenge: the particle distribution becomes increasingly distorted and inhomogeneous. The PCP redistancing method is thus confronted with excess information along some directions, and a lack of information along others. This makes an ideal test case to assess the robustness of the PCP method on ill-conditioned particle distributions.

The initial circle of radius $R = 0.15$ centered at $(0.5, 0.75)$ in the 2D square domain $[0,1] \times [0,1]$ is discretized with a narrow band ($w = 40h$) of particles. The particles then move with the advection velocity

$$\mathbf{u}(\mathbf{x}, t) = 2\cos\left(\tfrac{\pi t}{8}\right)\begin{pmatrix} -\sin^2(\pi x)\sin(\pi y)\cos(\pi y) \\ \sin^2(\pi y)\sin(\pi x)\cos(\pi x) \end{pmatrix}. \tag{23}$$

Time integration uses the explicit fourth-order Runge-Kutta scheme with time-step size $\Delta t = \frac{1}{30}$. The velocity field is reversed at $t = 4.0$, such that the simulation ends at $t_{\text{end}} = 8.0$. At $t = 0$, the level-set function values on all particles are initialized to the exact analytical distance to the circle. PCP redistancing with fourth-order minter regression is done after each time step of the advection with $\varepsilon = 10^{-10}$, $r_c = 15h$, and $k_{\max} = 100$. The reduced maximum number of iterations is required in this case, as the solver is unable to reach the desired tolerance for some of the most distorted particle distributions.

The large cutoff radius reduces the accuracy in the early and late time steps of this example, yet it is beneficial during steps involving extremely ill-conditioned particle distributions, where the particle density along one direction differs significantly from the density in the other direction. The flow field causing particles to align poses severe limitations to polynomial regression approaches and also causes the approximated zero level-set to lose its smoothness during the simulation. The most ill-conditioned particle distribution (at $t = 4.0$) is visualized in Fig. 8. The zero level-set of the SDF is shown as an iso-contour as determined by Paraview [40].

As can be seen in Fig. 8, the particle distribution is particularly sparse in the direction orthogonal to the surface. This is challenging for any redistancing scheme, since the SDF solely depends on information in the orthogonal direction. Another challenging aspect of this particle distribution is that multiple zero level-sets are present in each r_c-neighborhood, as shown exemplary for one particle in the closeup in Fig. 8, dotted circle.

Running the simulation until the final time $t_{\text{end}} = 8.0$, we can observe how well the PCP method with redistancing at each time step is able to recover the initial circle. Fig. 9 visualizes the final iso-contours reconstructed by Paraview [40] for three different resolutions. As expected, simulations with fewer particles suffer more. For $h = \frac{1}{64}$, the final level-set resembles a circle on average, but has noticeable errors. These errors do, however, converge with increasing numbers of particles, and the final shape for $h = \frac{1}{2048}$ is visually indistinguishable from the original circle. This can also be seen in the convergence of the average and maximum errors in the SDF and the enclosed area A ($e(A) = |(A - A_{\text{exact}})|/A_{\text{exact}}$) of the level-set as reported in Table 2. The average error in the SDF converges with order 1.9, the maximum error converges with order 1.5, and the area converges with order 1.8.

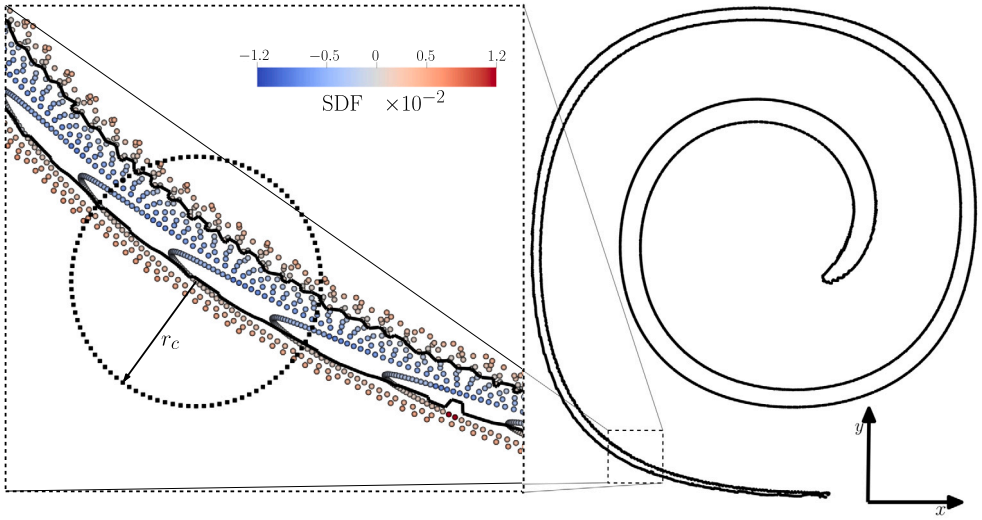


Fig. 8. The most ill-conditioned particle distribution in the spiraling vortex case with $h = \frac{1}{512}$ at $t = 4.0$. Spheres represent particle positions and are colored level-set function values. The dotted circle is an exemplary regression neighborhood of a particle with $r_c = 15h$. The solid line represents the reconstructed surface Γ_t.

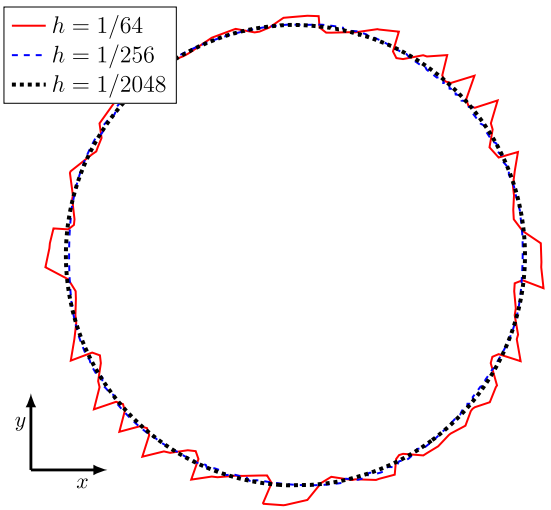


Fig. 9. Visualization of the final zero level-set at $t_{\text{end}} = 8.0$ for three different inter-particle spacings (line styles, see inset legend).

Table 2
Errors in the SDF values and the enclosed area A for the final state of the spiraling vortex.

h	$e(A)$	$\|e(\text{SDF})\|_2$	$\|e(\text{SDF})\|_\infty$
1/32	5.85×10^{-2}	3.60×10^{-2}	8.8×10^{-2}
1/64	3.83×10^{-2}	1.42×10^{-2}	2.4×10^{-2}
1/128	1.18×10^{-2}	2.90×10^{-3}	8.1×10^{-3}
1/256	1.11×10^{-3}	1.10×10^{-3}	2.2×10^{-3}
1/512	3.24×10^{-4}	3.98×10^{-4}	1.1×10^{-3}
1/1024	2.99×10^{-4}	6.90×10^{-5}	4.4×10^{-4}
1/2048	4.04×10^{-5}	1.62×10^{-5}	1.8×10^{-4}

This confirms that the PCP method converges (albeit with sub-optimal order) despite the severely ill-posed particle distributions occurring during the spiraling vortex dynamics.

4.3. Oscillating droplet with multi-phase hydrodynamics

We next consider the more "real-world" example of droplet dynamics in multi-phase hydrodynamics in both 2D and 3D. In this test case, a closed surface is coupled to two surrounding fluid phases, one inside and one outside. The curved interface between the two fluids has a surface tension and thus exerts forces on the fluids that depend on its curvature and cause dynamic flows, which in turn again advect and shape the interface. Both fluids are modeled using the incompressible Navier-Stokes equations

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} = -\rho\nabla\cdot\mathbf{u}\,, \tag{24}$$

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\eta\Delta\mathbf{u} + \mathbf{F}^{(s)}\,, \tag{25}$$

where density is denoted by ρ, pressure by P, η is the dynamic viscosity, and Δ is the Laplace operator. The surface tension effect is modeled through a continuum surface force [41]. Here, a volumetric surface force is active in the smoothed out interface, or transition region, which can be described with a smooth surface delta function δ. The surface force then reads

$$\mathbf{F}^{(s)} = -\frac{\tau}{\rho}\kappa\mathbf{n}\delta \tag{26}$$

and acts in the normal direction of the surface and is proportional to both the surface tension τ and the local curvature κ.

The continuity Eq. (24) and momentum Eq. (25) are complemented by the Cole equation of state [42]

$$P(\rho) = \frac{c^2\rho_0}{\gamma}\left(\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right), \tag{27}$$

linking the pressure P to the density ρ via a reference density ρ_0, a speed of sound c and the polytropic index of the fluid γ. We choose the speed of sound $c = 100$ at least one order of magnitude larger than the maximum flow velocity $\|\mathbf{u}_{\max}\|_2$ to ensure Mach numbers $Ma := \|\mathbf{u}_{\max}\|_2/c \lesssim 0.3$ remain in the region of incompressible flow. We further set $\gamma = 7$ and consider two identical fluids with $\rho_0 = 1.0$ and $\eta = 0.5$, separated by an interface with $\tau = 50.0$.

We solve the Navier-Stokes equations inside and outside the droplet without the interfacial forces $\mathbf{F}^{(s)}$ using a weakly compressible Smoothed Particle Hydrodynamics (SPH) approach [43]. Further details of the numerical method and the test case are given in Appendix B.1. The interfacial forces $\mathbf{F}^{(s)}$ in Eq. (26) are computed in two ways for comparison: (1) Using the present PCP method to compute the SDF, the surface normals, and the curvatures, resulting in the force given by Eq. (B.4). We use a fourth-order minter basis and threshold parameters $\varepsilon = 10^{-12}$ and $k_{\max} = 1000$, a cutoff radius of $r_c = 2.8h$, a sample-particle threshold of $\xi = 1.5h$, and a narrow-band width of $w = 13h$.[1] (2) Using a colorfield function and SPH operators [29], resulting in the force given by Eq. (B.10). For the quantitative analysis we first consider a 2D model. Here, we perform simulations for three different resolutions, $h = \frac{1}{32}$, $\frac{1}{64}$, and $\frac{1}{128}$ with a smoothing factor of $\epsilon/h = 3$. The time steps for the three different resolutions are $\Delta t = 2\times10^{-4}$, $\Delta t = 1\times10^{-4}$, and $\Delta t = 5\times10^{-5}$.

Initially, at time $t = 0$, both fluid phases are at rest (zero velocity everywhere) and the interface is an ellipse (Eq. (21)) with $A = 0.75$ and $B = 0.5$ embedded in the 2D domain $(-1.22, 1.22)\times(-1.22, 1.22)$ with periodic boundary conditions in both directions. Interface regions of higher curvature exert greater force than those of lower curvature, causing oscillating deformation of the incompressible droplet, eventually subsiding with the shape of minimal surface area, a circle. For the 3D case, this is visualized in the Supplementary Video 2 of the web version of this article. We run the simulations until $t_{\text{end}} = 1$. Fig. 10 visualizes the particle distribution, the computed curvature values, and the zero level-set reconstructed from the advected level-set values at $t \approx 0.15$, when the droplet assumes its maximum vertical elongation.

Table 3 compares the computational times of the entire 2D simulation using the present PCP method and using colorfield-SPH for different resolutions h with the same $\Delta t = 5\times10^{-5}$. Computing geometric quantities involves more computations per particle in the PCP method than in the colorfield-SPH approach, which is why the computational time is not smaller for PCP at low spatial resolution, even though the computations are confined to the narrow band. However, as the fraction of particles that are inside the narrow band decreases with decreasing average inter-particle spacing h, the computational cost of PCP reduces for finer resolutions compared to SPH.

A similar trend is seen when comparing the wall-clock times of computing the force balance using the SPH operators with those required by PCP (Table 4). For the coarsest resolution, the geometric computing accounts for the majority of the computational time, whereas this changes with increasing resolution. The ratio of the two times increases linearly with h, which is expected for any narrow-band method.

A key advantage of a closest-point approach can be seen in Figs. 10 and C.15 (Appendix C): geometric quantities are computed at the closest point, i.e. on the surface itself, and then extended along the surface normal to particles in the embedding space. For finite h, this differs from the colorfield-SPH approach, which computes geometric quantities of the iso-contours of the level-set function on

[1] The narrow-band width is determined by the maximum distance from the surface at which any geometric information is of importance. For the problem at hand, this is either the polynomial regression domain, which extends at most $r_c + \xi = 2.8h + 1.5h = 4.3h$ into the bulk phases, or the length scale of the smoothed surface tension. This length scale is given by the kernel function in Eq. (B.3) with a cutoff radius of $2\epsilon = 6h$. The larger value is $6h$ to which we add a margin of $0.5h$ for grid sampling effects, resulting in $w/2 = 6.5h$.

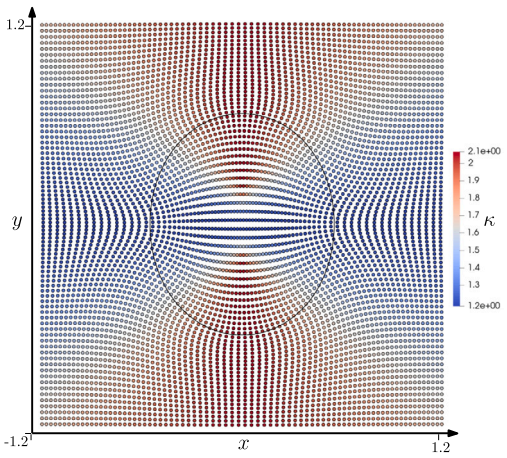


Fig. 10. Computed curvature values of the closest points (color bar) at $t \approx 0.15$ for $h = \frac{1}{32}$ for the two-phase droplet hydrodynamics. The zero level-set in solid black was reconstructed by Paraview [40] from the SDF values computed using the PCP method.

Table 3
Wall-clock times in minutes required for the entire 2D two-phase flow simulation with different spatial resolutions h using a fixed time resolution of $\Delta t = 5 \times 10^{-5}$ and two different geometric computing approaches for the interface dynamics: the colorfield-SPH approach (SPH) and the present particle closest-point (PCP) method. All simulations are performed on a single processor core of an AMD Ryzen Threadripper 3990X.

h	1/32	1/64	1/128
SPH	31	122	473
PCP	39	107	341

Table 4
Wall-clock times in milliseconds per time step for force-balance computations using SPH (t_{FB}) and for geometric computing using PCP (t_{PCP}). The ratio $t_{\text{FB}}/t_{\text{PCP}}$ is expected to increase linearly with resolution h. All simulations are performed on a single processor core of an AMD Ryzen Threadripper 3990X.

h	1/32	1/64	1/128	1/256
t_{FB}	20	79	314	1250
t_{PCP}	38	74	148	304
$t_{\text{FB}}/t_{\text{PCP}}$	0.53	1.07	2.12	4.11

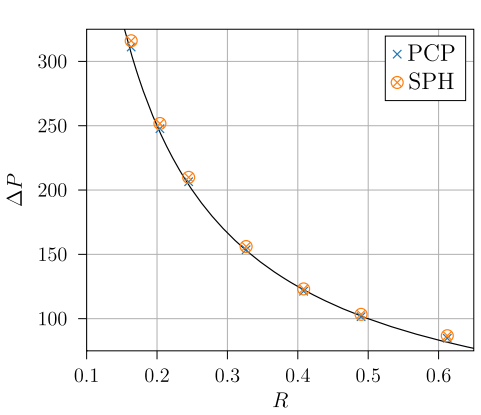
which the particles lie. Hence, it is unsurprising that in Fig. C.15 in Appendix C the standard deviation of the computed curvature values is smaller for the PCP method. The minimum standard deviation of the curvature values occurs when the interface becomes a circle. For the colorfield-SPH approach in the highest resolution, the smallest achieved standard deviation is 0.105, while the values range from 8×10^{-4} to 3×10^{-4} for the different resolutions in the PCP approach. This is orders of magnitude better than the colorfield approach and generally satisfying, given that the results are limited by the time resolution of the simulations, and the exact moment the geometry becomes circular may not be captured. A further limiting factor of the results can be identified in the accuracy of the SPH approximation of the remaining terms in the Navier-Stokes equations for the surrounding fluids.

The oscillation frequency of the droplet is determined by the surface tension, whereas the dynamics of the amplitude is determined by the viscosity of the fluid [44]. As shown in Fig. C.15, the frequency is in good agreement between the PCP approach and the colorfield-SPH approach. Since the droplet oscillation frequency of the colorfield-SPH approach has been validated against theoretical values for droplets with large density ratios [43] and against grid-based simulation results [29], this validates the PCP method.

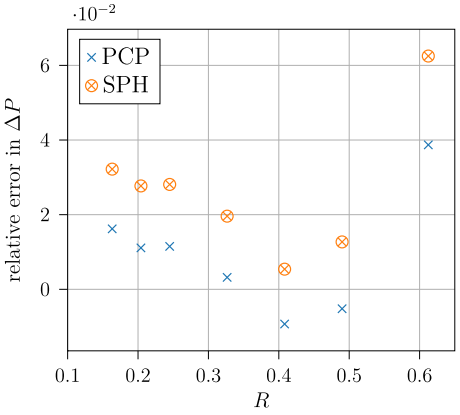
Looking at the average curvature across particles in the interface region (Fig. C.16, Appendix C) we notice that the PCP method is slightly biased towards higher curvatures than the exact value an ideally incompressible fluid would yield. The volume reconstructed from the computed SDF is consistent with the mean curvature. For $h = \frac{1}{128}$, the volume reconstructed from the SDF computed by PCP has a relative error of -0.5% at the end of the simulation after a total of 40,000 redistancing steps (computing the volume using colorfield SPH is not straightforward). Given that the results from the colorfield-SPH approach have a similar bias for lower resolutions, but are less accurate, the bias is probably due to the simulated fluid not being ideally incompressible in the weakly compressible SPH approach. In summary, PCP not only computes curvature values at the correct iso-contour, but the values are also less noisy than those from colorfield-SPH. This is because it does not use a binary indicator function, which heavily depends on the particle distribution in capturing the geometry.

To validate the overall multi-phase hydrodynamics, we compute the pressure difference ΔP across the interface and compare it to the capillary pressure difference expected from the Young–Laplace law

$$\Delta P = \tau \kappa \,. \tag{28}$$



(a) Computed pressure jump ΔP across the interface. The solid black line is the theoretical prediction from the Young–Laplace law (Eq. (28)).



(b) Relative error in the computed pressure difference ΔP. Positive values indicate overestimated differences.

Fig. 11. Pressure difference ΔP across the droplet interface at the final simulation time point for different resulting droplet radii R. Numerical results are computed using the PCP method (blue crosses) and the SPH-colorfield approach (orange circled crosses).

We numerically compute the pressure difference across the interface of droplets of different sizes at the final simulation time point, when the droplets are spherical with radius R. In order to exclude finite-resolution effects, we keep the ratio R/h constant across simulations.

We find that both geometric computing approaches, the PCP method and the colorfield-SPH formulation, correctly recover the inverse relation between the radius of curvature R of the final droplet and the pressure difference ΔP across the interface, as shown in Fig. 11a. However, PCP is more accurate. The relative error in ΔP, shown in Fig. 11b, is up to six-fold smaller for PCP than for colorfield-SPH (for radius $R = 0.327$: SPH error 0.02, PCP error 0.003), and SPH systematically overestimates the pressure difference. The error in the pressure difference has two parts: (1) The error in the computed interface curvature, and (2) the error in the discretization of the pressure gradient. The PCP method improves the former, but the pressure gradient is still computed using the same SPH operators. The SPH error then eventually becomes limiting.

In addition to its higher accuracy, the PCP method also captures a steeper pressure gradient in the transition from one fluid phase to the other (Fig. C.17, Appendix C). For a droplet with initial semi-major axes $A = 0.75$ and $B = 0.5$ and resolutions $h = 1/32$, $h = 1/64$, and $h = 1/128$ we find a factor of roughly 1.8 in the gradient magnitude. This shows that the PCP method better approximates the sharp interface than the colorfield approach, in which geometric quantities on surrounding iso-contours are computed. Computing geometric quantities on surrounding iso-contours also leads to overestimation of the pressure jump, as the curvature increases faster inside the droplet, on iso-contours with smaller radii. This systematic error is not observed in the PCP method.

To assess the compatibility of the PCP method with a particle method other than SPH, we alternatively discretize the differential operators in Eqs. (24) and (25) using discretization-corrected particle strength exchange (DC-PSE) [50] of second-order accuracy with a cutoff radius $r_c = 3h$. We use the same cutoff radius for the PCP method. All other parameters remain the same as in the SPH simulations. We consider the same resolutions $h \in \{1/32, 1/64, 1/128\}$ and compare the final average curvature $\bar{\kappa}$ in the narrow band with the theoretical value $\kappa_{\text{theory}} = 1/\sqrt{AB}$ for ideal incompressibility. We find decreasing errors $|\bar{\kappa} - \kappa_{\text{theory}}| \in \{0.0120, 0.0069, 0.0051\}$, indicating h-convergence. A visualization of the simulations can be found in Supplementary Video 1 of the web version of this article.

We also test the PCP multi-phase SPH fluid simulation for a 3D oscillating droplet with an initial shape of an ellipsoid with semi-major axis $A = 0.75$ and both semi-minor axes $B = C = 0.5$. We use the same parameters as in the 2D case, except for $\tau = 25.0$, and choose $h = 1/32$. To reduce the computational burden from the SPH operators, we lower the smoothing factor to $\epsilon/h = 2$ and consequently reduce the narrow-band width to $w = 9h$. Fig. 12 visualizes the particles of the inner fluid phase with their computed curvature values during the maximum vertical elongation of the droplet at $t \approx 0.19$. We also compute results for a lower viscosity value of $\eta = 0.3$, visualized in Supplementary Video 2 of the web version of this article.

Similarly as in the 2D simulations, the PCP method computes smooth curvature values of the dynamically deforming surface in 3D. Also in 3D, the PCP results are more accurate and computationally less expensive than the colorfield-SPH approach. This suggests that the PCP method is well suited for simulations involving dynamically deforming shapes.

4.4. Multi-phase hydrodynamics of a dividing droplet

Handling changes in the topology of the simulated surfaces is one of the key advantages of level-set methods. This property is naturally inherited by the PCP method. We illustrate this by simulating a droplet dividing into two smaller droplets.

Again, we consider a multi-phase setting with a fluid droplet immersed in a surrounding fluid phase. We simulate initially spherical droplets of radius $R_o = 0.25$ both in the 2D domain $(-1.0, -0.5) \times (1.0, 0.5)$ and in the 3D domain $(-0.75, -25/64, -25/64) \times$

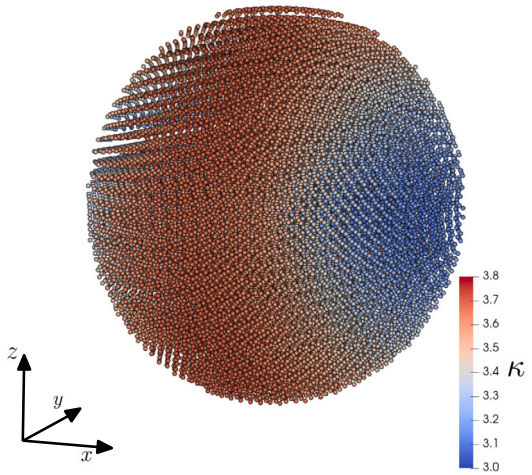


Fig. 12. Particles of the inner fluid phase of a 3D oscillating droplet simulation, color-coded according to the computed curvature values κ (color bar).

$(0.75, 25/64, 25/64)$. To induce droplet splitting, we impose external body forces in the x-direction on all particles of the inner fluid phase until $T = 0.5$ in 2D, and $T = 0.05$ in 3D. In 2D, the corresponding acceleration along x is:

$$a_x(x_p, y_p, t) = F_b \sin(2\pi \tfrac{t}{2T}) \sin(2\pi \tfrac{x_p}{2A}) \cos(2\pi \tfrac{y_p}{4B}), \tag{29}$$

and in 3D it is:

$$a_x(x_p, y_p, z_p, t) = F_b \sin(2\pi \tfrac{t}{2T}) \sin(2\pi \tfrac{x_p}{2A}) \cos(2\pi \tfrac{y_p}{4B}) \cos(2\pi \tfrac{z_p}{4C}), \tag{30}$$

with the body-force coefficient F_b set to 1000 in 2D and 5000 in 3D. We consider a fluid of viscosity $\eta = 0.5$ in both 2D and 3D. In order to reduce the forces opposing a change in topology, we choose a lower surface tension coefficient of $\tau = 5.0$ than in the previous sub-section. The artificial accelerations can lead to velocity fields with non-vanishing divergence if the magnitude F_b and duration T are not chosen carefully. Further, a breakup into multiple smaller droplets which subsequently fuse again should be avoided in the present surface tension model (see comment to Eq. (B.4)). The system behavior is sensitive to changes in acceleration magnitude and duration, surface tension, and viscosity.

We also use this test case to show applicability of the PCP method in conjunction with the popular particle shifting technique (PST) [45]; see Appendix B.2 for details. In 3D, we use an initial inter-particle spacing of $h = 1/32$, a smoothing length of $\epsilon = 2h$, and a time step of $\Delta t = 1.5 \cdot 10^{-4}$. In 2D, we use a time step of $\Delta t = 1.0 \cdot 10^{-4}$ and assess the convergence of the method for $h \in \{1/32, 1/48, 1/64, 1/80\}$. We simultaneously increase the number of neighbors per particle in the SPH operators through the smoothing length $\epsilon \in \{3.0h, 3.25h, 3.5h, 3.75h\}$ as required [38]. Both in 2D and in 3D, we use a cutoff radius for the regression problem in PCP of $r_c = 3.0h$ and compute third-order polynomials of total degree. All other parameters remain the same as in the simulations of the oscillating droplet.

Fig. 13 shows the results of the 2D simulation at different time points. The iso-surface of the interface is visualized using Paraview [40] based on the SDF values. Despite the large deformation of the surface and the change in topology, the computed curvature field remains smooth. The PCP method correctly computes negative curvatures before the pinch-off and large positive curvatures shortly after the pinch-off. Immediately before the pinch-off, the simulations show a wrinkling of the fluid stalk with alternating positive and negative curvature (see Fig. 13, $t = 0.3175$ inset). This is reminiscent of the Plateau-Rayleigh instability, providing a physical mechanism by which droplets break up [46]. Quickly, this unstable fluid stalk separates the droplets, as is correctly captured by the PCP method forming two separate zero-level sets with locally high positive curvature.

We confirm convergence of the average curvature in the narrow band, $\bar{\kappa}$, at the final time point $t = 1$ for the different resolutions h. For $h = 1/32$ we find $\bar{\kappa} = 5.39698$, for $h = 1/48$ we find $\bar{\kappa} = 5.61629$, for $h = 1/64$ we find $\bar{\kappa} = 5.56708$, and for $h = 1/80$ we find $\bar{\kappa} = 5.54885$. The difference in the computed average curvature becomes smaller with decreasing h, indicating h-convergence.

In the 3D simulation, as visualized in Fig. 14, the curvature is the sum of the two principal curvatures, $\kappa = \nabla \cdot \mathbf{n} = \kappa_1 + \kappa_2$. The distribution of curvature values is therefore different than in 2D. This becomes apparent in the results computed by the PCP method for the time points before division, when the initial droplet begins to elongate. In the 2D simulation at $t = 0.15$, negative curvature is observed at the central constriction (Fig. 13). In the 3D simulation, curvature remains positive everywhere at $t = 0.03$ (Fig. 14). This is due to the dominant positive principal curvature in the y-z plane. Negative curvatures appear only after division, when the pinched-off tips snap back into the separated droplets.

For a lower viscosity of $\eta = 0.15$, the snap-back causes visible capillary waves, characterized by curvature oscillating between negative and positive values over a short distance. The amplitude of the wave decreases over time as damped by the fluid viscosity, until both new droplets become spherical and stationary at the final time. Supplementary Video 3 of the web version of this article shows the full dynamics of the 3D simulation of the dividing droplets with $\eta = 0.15$.

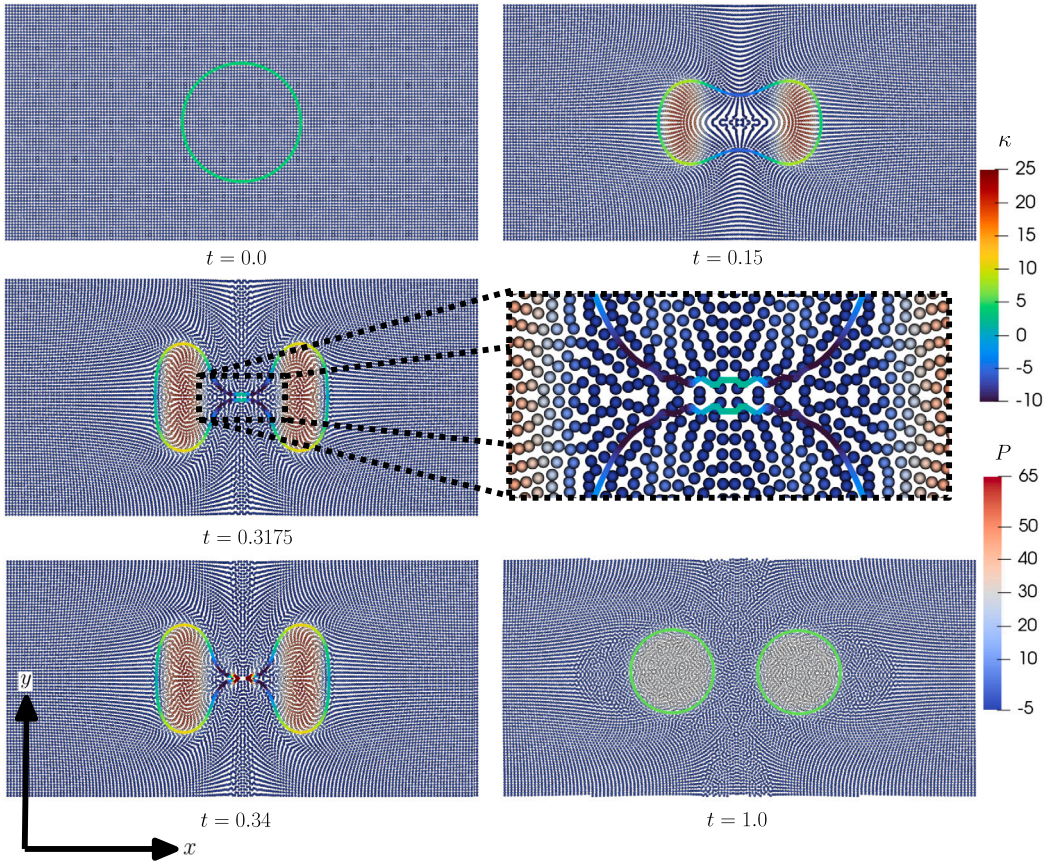


Fig. 13. Visualization of the 2D splitting droplet simulation for the highest resolution ($h = 1/80$) at different time points (panel labels). Particles are color-coded by pressure (bottom color bar), whereas the interface is color-coded by curvature (top color bar) as computed by the PCP method.

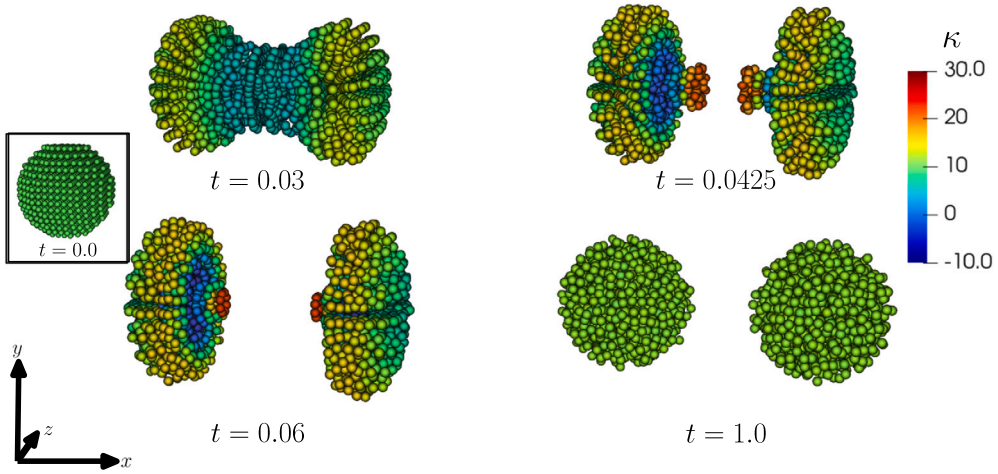


Fig. 14. Particles of the inner fluid phase at different time points (panel labels) of the 3D splitting droplet simulation with $\eta = 0.5$, color-coded by interface curvature (color bar) as computed by the PCP method. The inset on the left shows the initial condition.

5. Conclusions and discussion

We presented a higher-order redistancing scheme for fully Lagrangian particle level-set methods, extending closest-point redistancing [20] to irregularly distributed points. Unlike previous particle level-set methods [26,12], the proposed approach does not require interpolation from particles to a regular mesh, improving convergence for non-conserved level-set functions. The presented

Particle Closest-Point (PCP) method relies on minter regression on Chebyshev-Lobatto subgrids to achieve numerical robustness. We have shown that this renders the method more robust to distortion in the particle distribution than regression using monomial bases.

In the PCP method, the particles act as sample points of the level-set function, which is in contrast to colorfield approaches where particles directly represent the (smoothed) presence of a certain phase [29,43,47]. Our approach is purely geometric and allows for arbitrarily placed query points to be redistanced. Hence, the present approach can be used to initialize geometric quantities of new particles, e.g., in multi-resolution methods [48], after remeshing [49,12], and in particle shifting techniques.

We showed that the PCP method provides high-order convergent geometric quantities for basic geometries without requiring any connected mesh. We tested the robustness of the approach by studying highly irregular particle distributions and found that the high-order polynomial regression in Lagrange basis on unisolvent nodes outperforms monomial regression for irregular particle distributions. This enabled the PCP method to converge even for highly ill-conditioned particle distributions, as seen in the spiraling vortex case. We also applied the method to more complex problems involving multi-phase hydrodynamics as discretized by SPH and DC-PSE [50]. Specifically, we simulated oscillating and dividing droplets in both 2D and 3D, showing converging curvature values. This also demonstrated that the proposed particle level-set method can handle changes in interface topology. We have shown how the popular SPH particle shifting technique can be integrated with the PCP method in such a way that the shifting does not cause nonphysical interface deformation or wrinkling. Since the PCP method does not imply a specific differential-operator discretization, future work can explore its use in conjunction with other meshfree numerical methods, such as moving least squares [51] or generalized finite difference methods [52].

Overall, we found that the PCP method copes well with irregular particle distributions, yet we still expect it to require a certain homogeneity in the distribution as it benefited from larger smoothing lengths in the SPH operators. We also observed that some particles approached the zero contour and remained there. To avoid volume loss, we prevented particles from taking on a true 0 as a level-set value and from changing the sign of their level-set values [53]. This clearly is a limitation of the proposed approach and some form of particle distribution regularization will eventually be necessary. Another limitation is the non-uniform load distribution when implementing PCP methods on parallel computers. This is because the computational cost on a given processor not only depends on the number of particles it handles (which can be evenly distributed), but also on the local geometry of the surface (which is impossible to predict in a dynamic simulation), through the local regression problems. Nevertheless, the narrow-band character of the PCP approach confines the computational effort of the geometric computing framework to the proximity of the surface.

As with most level-set methods, the proposed PCP approach is suitable for continuous, orientable, closed surfaces that do not self-intersect. This is the case for any closed surface that possesses a tubular neighborhood. For non-smooth surfaces, the convergence order of the method is bounded by the smoothness class of the surface and the degree of the regression polynomials used, whichever is smaller. If the smoothness class of the surface is smaller than the used polynomial degree, the leading error term is then determined by the derivative order in which discontinuities first appear.

Future work could further optimize the numerical robustness of the method by testing different approaches to choosing particles that lie close to the unisolvent nodes of a Chebyshev-Lobatto subgrid. Further research could consider how Lagrangian formulations could incorporate interface velocity extensions [21] to reduce the required frequency of redistancing steps. Future work could also provide a GPU implementation of the PCP method. The current implementation in *OpenFPM* could make use of the GPU-accelerated data structures that *OpenFPM* provides [37] once the *Eigen* library supports GPU solvers.

Future applications of the PCP method can take advantage of the polynomial minter regression that is the defining feature of the method. This can, for example, include numerically solving PDEs on surfaces, where constant orthogonal extension is a popular solution approach [18,19]. There, the PCP method solves two of the main challenges: It accurately computes the CP transform using an orthogonal decomposition of the regression problem, which can straightforwardly be reused to also approximate the values of other fields at the computed closest point. Due to its computational efficiency, parallel scalability, and robustness against distortion in the particle distribution, the PCP method could therefore be useful in solving PDEs on dynamically deforming surfaces.

CRediT authorship contribution statement

Lennart J. Schulze: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation. **Sachin K.T. Veettil:** Writing – original draft, Validation, Software, Methodology, Investigation, Formal analysis, Data curation. **Ivo F. Sbalzarini:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The C++ implementation of the presented particle closest point method in *OpenFPM* is publicly available from https://git.mpi-cbg.de/mosaic/software/parallel-computing/openfpm/openfpm_numerics.

Acknowledgements

We thank Dr. Michael Hecht (Center for Advanced Systems Understanding, Görlitz), Alejandra Foggia, Johannes Pahlke, and Justina Stark (all Sbalzarini group) for discussions and proofreading, and Dr. Pietro Incardona (University of Bonn) for support with the *OpenFPM* implementation of the presented method. This work was supported by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF) in the joint project "6G-life" (ID 16KISK001K).

Appendix A. Particle-mesh interpolation

A popular kernel for particle-mesh interpolation is the $\Lambda_{4,4}$ kernel [27], which conserves the first four moments and produces C^4-regular results:

$$\Lambda_{4,4}(q) = \begin{cases} 1 - \frac{5}{4}q^2 + \frac{1}{4}q^4 - \frac{100}{3}q^5 + \frac{455}{4}q^6 - \frac{295}{2}q^7 + \frac{345}{4}q^8 - \frac{115}{6}q^9, & \text{if } 0 \le q < 1, \\ -199 + \frac{5485}{4}q - \frac{32975}{8}q^2 + \frac{28425}{4}q^3 - \frac{61953}{8}q^4 + \frac{33175}{6}q^5 - \frac{20685}{8}q^6 + \frac{3055}{4}q^7 - \frac{1035}{8}q^8 + \frac{115}{12}q^9, & \text{if } 1 \le q < 2, \\ 5913 - \frac{89235}{4}q + \frac{297585}{8}q^2 - \frac{143895}{4}q^3 + \frac{177871}{8}q^4 - \frac{54641}{6}q^5 + \frac{19775}{8}q^6 - \frac{1715}{4}q^7 + \frac{345}{8}q^8 - \frac{23}{12}q^9, & \text{if } 2 \le q < 3, \\ 0, & \text{else.} \end{cases}$$

It can be used to evaluate mesh node values as

$$\phi_{ij} = \sum_p \phi_p \Lambda_{4,4}\left(\frac{|x_i - x_p|}{h}\right) \Lambda_{4,4}\left(\frac{|y_j - y_p|}{h}\right), \tag{A.1}$$

in which i and j are the grid indices for x and y directions, respectively, and p is the index for the particles.

This classic approach to particle-mesh interpolation is designed for intensive fields of conserved quantities, such as density or concentration fields [28,54,49]. As level-set functions generally do not obey any conservation laws, the different amounts of contributions individual mesh nodes receive need to be accounted for as soon as the particle distribution becomes irregular. This is done by renormalizing the interpolated quantities according to

$$\phi_{ij} = \left(\sum_p \Lambda_{4,4}\left(\frac{|x_i - x_p|}{h}\right) \Lambda_{4,4}\left(\frac{|y_j - y_p|}{h}\right)\right)^{-1} \sum_p \phi_p \Lambda_{4,4}\left(\frac{|x_i - x_p|}{h}\right) \Lambda_{4,4}\left(\frac{|y_j - y_p|}{h}\right). \tag{A.2}$$

Similarly, the particle contributions can be scaled by individual volumes to yield the classic particle representation of an arbitrary field as

$$f(\mathbf{x}) \approx \sum_p f_p W_\epsilon\left(\|\mathbf{x} - \mathbf{x}_p\|_2\right) V_p, \tag{A.3}$$

where W is a local, symmetric, normalized kernel function with a smoothing length of ϵ. The smoothing length determines how many particles contribute to a field evaluation, and it strongly influences the convergence properties of particle methods. Generally, as the inter-particle spacing h becomes smaller, the smoothing length ϵ should also become smaller, yet the ratio $\frac{\epsilon}{h}$ should grow to ensure convergence of the scheme [38].

In Eq. (A.3), V_p is the volume associated with particle p, which can be computed as

$$V_q = \left(\sum_p W_\epsilon\left(\|\mathbf{x}_q - \mathbf{x}_p\|_2\right)\right)^{-1}. \tag{A.4}$$

The particle function approximation in Eq. (A.3) of a level-set function can then also be renormalized and subsequently evaluated at mesh nodes ij:

$$\phi_{ij} = \left(\sum_p W\left(\|\mathbf{m}_{ij} - \mathbf{x}_p\|_2, \epsilon\right) V_p\right)^{-1} \sum_p \phi_p W\left(\|\mathbf{m}_{ij} - \mathbf{x}_p\|_2, \epsilon\right) V_p. \tag{A.5}$$

In the main text, we compare this classic particle-mesh interpolation scheme with two different kernel functions popular in the SPH community. The first is the Wendland C2 kernel [55]

$$W_\epsilon(q) = \begin{cases} \sigma_{2D}\left(1 - \frac{q}{2}\right)^4 (1 + 2q) & \text{if } 0 \le q < 2, \\ 0 & \text{else,} \end{cases} \tag{A.6}$$

where the variable q is defined as: $q = \frac{\|\mathbf{x} - \mathbf{x}_p\|_2}{\epsilon}$, and $\sigma_{2D} = \frac{7}{4\pi\epsilon^2}$ is a normalization factor ensuring that the kernel integrates to one. The second popular SPH kernel is the Gaussian

$$W_\epsilon(q) = \begin{cases} \sigma_{2D} e^{-q^2} & \text{if } 0 \le q \le 3, \\ 0 & \text{else,} \end{cases} \tag{A.7}$$

with q defined as for the Wendland kernel, and $\sigma_{2D} = \frac{1}{\pi\epsilon^2}$ the normalization constant.

Appendix B. SPH formulation for multi-phase flow

We discretize both continuum fluids with a set of in total n_p particles, initialized on a regular Cartesian grid of spacing h covering the entirety of the simulation domain. The masses $m_i = M/n_p$ of the particles are computed by considering the total mass of the fluids M, resulting from the reference density and the occupied volume.

B.1. Oscillating droplet discretization

To estimate the density of a single particle i, density summation is performed as

$$\rho_i = m_i \sum_j W_{ij}\,, \tag{B.1}$$

where the Wendland C2 kernel from Eq. (A.6) is used as $W_{ij} = W_\epsilon(\|\mathbf{x}_i - \mathbf{x}_j\|_2)$. Simultaneously, the volume of each particle V_i is computed according to Eq. (A.4). Having computed the densities, the pressures are obtained by evaluating Eq. (27) for ρ_i. At $t_0 = 0$, all velocities are set to $\mathbf{u}_i = \mathbf{0}$. The change in velocity per particle is determined by the discrete momentum equation as [43]:

$$\frac{\mathrm{D}\mathbf{u}_i}{\mathrm{D}t} = -\frac{1}{m_i}\sum_j \left(V_i^2 + V_j^2\right)\frac{\rho_i P_j + \rho_j P_i}{\rho_i + \rho_j}\nabla W_{ij} + \frac{1}{m_i}\sum_j \eta\left(V_i^2 + V_j^2\right)\frac{\mathbf{u}_{ij}}{r_{ij}}\frac{\partial W}{\partial r_{ij}} + \mathbf{F}_i^{(s)}, \tag{B.2}$$

with $r_{ij} = \|\mathbf{r}_i - \mathbf{r}_j\|_2$ and $\mathbf{u}_{ij} = \mathbf{u}_i - \mathbf{u}_j$. Note that despite the absence of large reference density fractions, we include the smoothing effect of the inter-particle averaged pressure term.

To determine the volumetric surface force acting on each particle, we consider two different approaches: (1) based on the present PCP method and (2), an approach based on a colorfield function and SPH operators. For both approaches, we require a smooth surface representation that distributes the surface tension effect on particles surrounding the interface. To this end, we use the Wendland C2 kernel in 1D,

$$W_{\epsilon 1D}(q) = \begin{cases} \frac{5}{8\epsilon}\left(1 - \frac{q}{2}\right)^3 (1.5q + 1) & \text{if } 0 \le q < 2, \\ 0 & \text{else,} \end{cases} \tag{B.3}$$

in which $q = \frac{|\phi|}{\epsilon}$. The smoothing length can in principle be chosen independently from the rest of the SPH operators, but we choose them to be identical for convenience.

With the level-set SDF, the surface normals, and the curvatures, the discrete surface force on particle i is computed as

$$\mathbf{F}_i^{(s)} = -\frac{\tau}{\rho_i}\kappa_i \mathbf{n}_i W_{\epsilon 1D}\left(\phi_i\right)\,. \tag{B.4}$$

While this surface-tension model allows droplets to split, it cannot describe merging droplets. This is because the surface tension according to Eq. (B.4) points towards the interface for positive curvature. Since the PCP method computes geometric quantities of the zero iso-contour, this also holds for particles on surrounding iso-contours. This leads to ambient particles remaining between two nearby interfaces, preventing their fusion.

In the main text, we compare the level-set approach with the popular colorfield-SPH approach [29]. Colorfield-SPH identifies the transition region of the interface and computes normals and curvatures from a *color function* c assigning a unique color to each of the fluid phases. Following [56,29], a smooth color value of a particle is obtained as a convolution, or particle function representation, of the binary indicator field,

$$c_i = \sum_j c_j^{(ab)} W_{ij} V_j, \tag{B.5}$$

where $c_j^{(ab)}$ is the binary phase indicator taking values of either 1 on one side of the interface or 0 on the other side, W_{ij} here is the Wendland C2 kernel, and the volumes V_j are computed according to Eq. (A.4). The remainder of the colorfield-based geometric computing framework is as outlined in Ref. [29], where non-unit surface normals $\hat{\mathbf{n}}$ are obtained as

$$\hat{\mathbf{n}}_i = \sum_j \left(c_j - c_i\right)\nabla W_{ij}, \tag{B.6}$$

whose magnitude is used as an indicator whether a particle is part of the smoothed interface or not, based on

$$N_i = \begin{cases} 1 & \text{if } \|\hat{\mathbf{n}}_i\|_2 > 0.01/\epsilon \\ 0 & \text{else.} \end{cases} \tag{B.7}$$

Subsequently, narrow-banded unit normals can be obtained as

$$\mathbf{n}_i = \begin{cases} \hat{\mathbf{n}}_i / \|\hat{\mathbf{n}}_i\|_2 & \text{if } N_i = 1, \\ 0 & \text{else.} \end{cases} \tag{B.8}$$

Finally, the curvature is approximated by the SPH divergence of the unit surface normals from all "interface particles":

$$\kappa_i = \left(\sum_j \min\left(N_i, N_j\right) W_{ij} V_j \right)^{-1} \sum_j \min\left(N_i, N_j\right) \left(\mathbf{n}_j - \mathbf{n}_i\right) \cdot \nabla W_{ij} V_j \,, \tag{B.9}$$

where the pre-factor accounts for differing contributions due to particles qualifying as interface particles, or not. In Eq. (26), the gradient of the smoothed color function, $\hat{\mathbf{n}}_i$, is interpreted as the product of a regularized delta function and the surface normal, yielding the discrete surface force on particle i as computed using colorfield-SPH:

$$\mathbf{F}_i^{(s)} = -\frac{\tau}{\rho_i} \kappa_i \hat{\mathbf{n}}_i \,. \tag{B.10}$$

Regardless of the approach chosen to determine the interfacial forces, PCP or colorfield-SPH, we integrate the positions of the particles in time with a second-order predictor-corrector scheme as in Ref. [57] and apply the geometric computations at every predictor and every corrector step. We determine the time-step size Δt such that it fulfills the CFL-like conditions given in Refs. [23, 43,29]:

$$\Delta t \leq \min\left(0.25 \frac{\epsilon}{c + \|\mathbf{u}_{\max}\|_2}, \; 0.125 \frac{\rho \epsilon^2}{\eta}, \; 0.25 \sqrt{\frac{\rho_0 \epsilon^3}{2\pi\tau}} \right), \tag{B.11}$$

where we again use $\|\mathbf{u}_{\max}\|_2 = 3$.

B.2. Splitting droplet discretization

For the splitting droplet simulation we use the same discretization in time, but change to a different spatial discretization. Instead of performing density summation, we evolve the density over time with a diffusion term [58], yielding

$$\frac{\mathrm{D}\rho_i}{\mathrm{D}t} = \rho_i \sum_j \left(\mathbf{u}_i - \mathbf{u}_j\right) \cdot \nabla W_{ij} V_j + \hat{\xi} \epsilon c \sum_j 2(\rho_i - \rho_j) \frac{\mathbf{r}_{ij}}{r_{ij}} \cdot \nabla W_{ij} V_j \,. \tag{B.12}$$

The magnitude of the density diffusion term is scaled by the parameter $\hat{\xi}$, which we set to $\hat{\xi} = 0.05$ for all simulations.

For the momentum equation, we now use a different discretization of the pressure gradient and an artificial viscosity term [23], resulting in

$$\frac{\mathrm{D}\mathbf{u}_i}{\mathrm{D}t} = -m_i \sum_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \nabla W_{ij} \,, \tag{B.13}$$

with artificial viscosity term

$$\Pi_{ij} = -\frac{\hat{\alpha} \epsilon c}{0.5(\rho_i + \rho_j)} \left(\frac{\mathbf{u}_{ij} \cdot \mathbf{r}_{ij}}{r_{ij}^2 + \hat{\mu}\epsilon^2} \right) . \tag{B.14}$$

The parameter preventing zero denominators is chosen as $\hat{\mu} = 0.01$ for all simulations, and the artificial viscosity parameter $\hat{\alpha}$ can be linked to the physical viscosity η as $\hat{\alpha} = \frac{8\eta}{\rho_0 \epsilon c}$ in 2D and $\hat{\alpha} = \frac{10\eta}{\rho_0 \epsilon c}$ in 3D.

As the external body-force fields in Eqs. (29) and (30) result in velocity fields with non-vanishing divergence, the particle distribution would become ill-conditioned, with voids developing in the center of the domain. In order to regularize this, and to ensure that the local regression problems contain similar information from both sides of the splitting surface, we use the popular particle shifting technique (PST) [45]. For PST, the particle-concentration gradient is computed in each time step with a gradient discretization that avoids tensile instability [59]:

$$\nabla C_i = \sum_j \left(1 + 0.2 \left(\frac{W_{ij}}{W(h)} \right)^4 \right) \nabla W_{ij} V_j \,. \tag{B.15}$$

To regularize the particle distribution, the particles are subsequently shifted in the direction opposite to the particle-concentration gradient with a maximum displacement limited to 0.02ϵ. The shifting displacement on particle i is:

$$\Delta \mathbf{x}_i = \begin{cases} -0.05\epsilon^2 \nabla C_i & \text{if } 0.05\epsilon^2 \|\nabla C_i\| < 0.02\epsilon \\ -0.02\epsilon \frac{\nabla C_i}{\|\nabla C_i\|_2} & \text{else.} \end{cases} \tag{B.16}$$

In order to avoid that particle shifting introduces nonphysical deformation or wrinkling of the level-set interface, we compute the regression polynomials and create the sample particles at each time step before shifting the particles. After shifting, we redistance the shifted particle set using the regression polynomials and sample particles obtained for the unshifted level-set. This ensures that PST does not alter the embedded surface, while it still benefits the numerical stability of both the SPH operators and the PCP method by regularizing the particle distribution. The same approach can also be used when remeshing particle distributions [49,12].

Appendix C. Appendix figures

We provide additional figures as referred to and discussed in the main text.

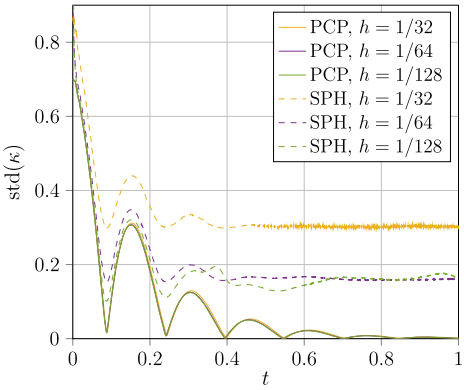


Fig. C.15. Standard deviation of the computed curvature values over the particles participating in continuum surface force approximation for different resolutions and methods.

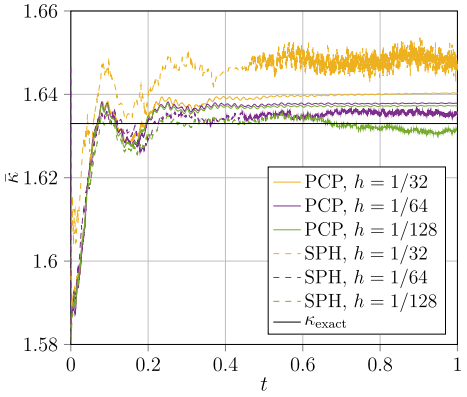


Fig. C.16. Mean of the computed curvature values over the particles participating in continuum surface force approximation for different resolutions and methods.

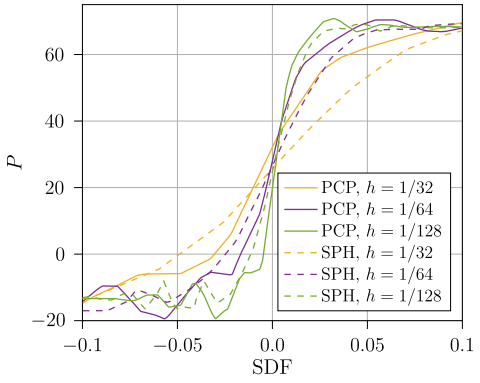


Fig. C.17. Pressure change across the interface of the stationary droplet at $t = 1.0$ as determined by PCP (solid lines) and colorfield-SPH (dashed lines) using different resolutions h (color, inset legend).

Appendix D. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jcp.2024.113262.

References

[1] Q. Chen, G. Guillemot, C.-A. Gandin, M. Bellet, Three-dimensional finite element thermomechanical modeling of additive manufacturing by selective laser melting for ceramic materials, Addit. Manuf. 16 (2017) 124–137.
[2] X. Li, H. Huang, P. Meakin, Level set simulation of coupled advection-diffusion and pore structure evolution due to mineral precipitation in porous media, Water Resour. Res. 44 (12) (2008).
[3] I.F. Sbalzarini, A. Hayer, A. Helenius, P. Koumoutsakos, Simulations of (an-) isotropic diffusion on curved biological surfaces, Biophys. J. 90 (3) (2006) 878–885.
[4] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[5] X. Zheng, S. Wise, V. Cristini, Nonlinear simulation of tumor necrosis, neo-vascularization and tissue invasion via an adaptive finite-element/level-set method, Bull. Math. Biol. 67 (2005) 211–259.
[6] J.A. Sethian, P. Smereka, Level set methods for fluid interfaces, Annu. Rev. Fluid Mech. 35 (1) (2003) 341–372.
[7] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159.
[8] C. Li, C. Xu, C. Gui, M.D. Fox, Level Set Evolution Without Re-Initialization: a New Variational Formulation, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, IEEE, 2005, pp. 430–436.
[9] C. Li, C. Xu, C. Gui, M.D. Fox, Distance regularized level set evolution and its application to image segmentation, IEEE Trans. Image Process. 19 (12) (2010) 3243–3254.
[10] V. Estellers, D. Zosso, R. Lai, S. Osher, J.-P. Thiran, X. Bresson, Efficient algorithm for level set method preserving distance function, IEEE Trans. Image Process. 21 (12) (2012) 4722–4734.
[11] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. 115 (1) (1994) 200–212.
[12] S.E. Hieber, P. Koumoutsakos, A Lagrangian particle level set method, J. Comput. Phys. 210 (1) (2005) 342–367.
[13] J.N. Tsitsiklis, Efficient algorithms for globally optimal trajectories, IEEE Trans. Autom. Control 40 (9) (1995) 1528–1538.
[14] J.A. Sethian, A fast marching level set method for monotonically advancing fronts, Proc. Natl. Acad. Sci. 93 (4) (1996) 1591–1595.
[15] H. Zhao, A fast sweeping method for Eikonal equations, Math. Comput. 74 (250) (2005) 603–627.
[16] S. Kim, An $\mathcal{O}(N)$ level set method for Eikonal equations, SIAM J. Sci. Comput. 22 (6) (2001) 2178–2193.
[17] S. Mauch, A fast algorithm for computing the closest point and distance transform, 2000.
[18] S.J. Ruuth, B. Merriman, A simple embedding method for solving partial differential equations on surfaces, J. Comput. Phys. 227 (3) (2008) 1943–1961.
[19] T. Marz, C.B. Macdonald, Calculus on surfaces with general closest point functions, SIAM J. Numer. Anal. 50 (6) (2012) 3303–3328.
[20] R. Saye, High-order methods for computing distances to implicitly defined surfaces, Commun. Appl. Math. Comput. Sci. 9 (1) (2014) 107–141.
[21] D. Henneaux, P. Schrooyen, L. Arbaoui, P. Chatelain, T.E. Magin, A high-order level-set method coupled with an extended discontinuous Galerkin method for simulating moving interface problems, in: AIAA AVIATION 2021 FORUM, 2021, p. 2742.
[22] L.C. Ngo, Q.-N. Dinh, H.G. Choi, High-order level set reinitialization for multiphase flow simulations based on unstructured grids, Comput. Math. Appl. 120 (2022) 60–77.
[23] J.J. Monaghan, Smoothed particle hydrodynamics, Rep. Prog. Phys. 68 (8) (2005) 1703.
[24] B. Engquist, A.-K. Tornberg, R. Tsai, Discretization of Dirac delta functions in level set methods, J. Comput. Phys. 207 (1) (2005) 28–51.
[25] G.-H. Cottet, E. Maitre, A level set method for fluid-structure interactions with immersed surfaces, Math. Models Methods Appl. Sci. 16 (03) (2006) 415–438.
[26] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (1) (2002) 83–116.
[27] G.-H. Cottet, J.-M. Etancelin, F. Pérignon, C. Picard, High order semi-Lagrangian particle methods for transport equations: numerical analysis and implementation issues, ESAIM: Math. Model. Numer. Anal. 48 (4) (2014) 1029–1060.
[28] M. Bergdorf, I.F. Sbalzarini, P. Koumoutsakos, A Lagrangian particle method for reaction–diffusion systems on deforming surfaces, J. Math. Biol. 61 (5) (2010) 649–663.
[29] J.P. Morris, Simulating surface tension with smoothed particle hydrodynamics, Int. J. Numer. Methods Fluids 33 (3) (2000) 333–353.
[30] M. Bergdorf, P. Koumoutsakos, A Lagrangian particle-wavelet method, Multiscale Model. Simul. 5 (3) (2006) 980–995.
[31] B. Quentrec, C. Brot, New method for searching for neighbors in molecular dynamics computations, J. Comput. Phys. 13 (3) (1973) 430–432.
[32] M. Hecht, K. Gonciarz, J. Michelfeit, V. Sivkin, I.F. Sbalzarini, Multivariate interpolation in unisolvent nodes–lifting the curse of dimensionality, arXiv preprint, arXiv:2010.10824, 2020.
[33] S.K.T. Veettil, Y. Zheng, U.H. Acosta, D. Wicaksono, M. Hecht, Multivariate polynomial regression of Euclidean degree extends the stability for fast approximations of Trefethen functions, arXiv:2212.11706, 2022.
[34] P. Incardona, A. Leo, Y. Zaluzhnyi, R. Ramaswamy, I.F. Sbalzarini, OpenFPM: a scalable open framework for particle and particle-mesh codes on parallel computers, Comput. Phys. Commun. 241 (2019) 155–177.
[35] S.K.T. Veettil, minter, https://git.mpi-cbg.de/mosaic/software/math/minter.
[36] G. Guennebaud, B. Jacob, et al., Eigen v3, http://eigen.tuxfamily.org, 2010.
[37] P. Incardona, A. Gupta, S. Yaskovets, I.F. Sbalzarini, A portable C++ library for memory and compute abstraction on multi-core CPUs and GPUs, Concurr. Comput., Pract. Exp. (2023) e7870.
[38] P.-A. Raviart, An analysis of particle methods, in: Numerical Methods in Fluid Dynamics: Lectures Given at the 3rd 1983 Session of the Centro Internationale Matematico Estivo (CIME), held at Como, Italy, July 7–15, 1983, Springer, 1985, pp. 243–324.
[39] D. Eberly, Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid, in: Geometric Tools, LLC, 2011.
[40] J. Ahrens, B. Geveci, C. Law, ParaView: an end-user tool for large data visualization, in: Visualization Handbook, Elsevier Inc., Burlington, MA, USA, 2005, pp. 717–731.
[41] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (2) (1992) 335–354.
[42] R.H. Cole, R. Weller, Underwater explosions, Phys. Today 1 (6) (1948) 35.
[43] S. Adami, X. Hu, N.A. Adams, A new surface-tension formulation for multi-phase SPH using a reproducing divergence approximation, J. Comput. Phys. 229 (13) (2010) 5011–5021.
[44] A. Aalilija, C.-A. Gandin, E. Hachem, On the analytical and numerical simulation of an oscillating drop in zero-gravity, Comput. Fluids 197 (2020) 104362.
[45] S.J. Lind, R. Xu, P.K. Stansby, B.D. Rogers, Incompressible smoothed particle hydrodynamics for free-surface flows: a generalised diffusion-based algorithm for stability and validations for impulsive flows and propagating waves, J. Comput. Phys. 231 (4) (2012) 1499–1523.
[46] J. Eggers, Nonlinear dynamics and breakup of free-surface flows, Rev. Mod. Phys. 69 (3) (1997) 865.
[47] S. Marrone, A. Colagrossi, D. Le Touzé, G. Graziani, Fast free-surface detection and level-set function definition in SPH solvers, J. Comput. Phys. 229 (10) (2010) 3652–3663.

[48] S. Reboux, B. Schrader, I.F. Sbalzarini, A self-organizing Lagrangian particle method for adaptive-resolution advection–diffusion simulations, J. Comput. Phys. 231 (9) (2012) 3623–3646.
[49] S.E. Hieber, J.H. Walther, P. Koumoutsakos, Remeshed smoothed particle hydrodynamics simulation of the mechanical behavior of human organs, Technol. Health Care 12 (4) (2004) 305–314.
[50] B. Schrader, S. Reboux, I.F. Sbalzarini, Discretization correction of general integral PSE operators for particle methods, J. Comput. Phys. 229 (11) (2010) 4159–4182.
[51] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Methods Appl. Mech. Eng. 139 (1–4) (1996) 3–47.
[52] T. Liszka, J. Orkisz, The finite difference method at arbitrary irregular grids and its application in applied mechanics, Comput. Struct. 11 (1–2) (1980) 83–95.
[53] D. Enright, Use of the Particle Level Set Method for Enhanced Resolution of Free Surface Flows, Stanford University, 2002.
[54] M.E. Bergdorf, Multiresolution particle methods for the simulation of growth and flow, Ph.D. thesis, ETH, Zurich, 2007.
[55] H. Wendland, Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, Adv. Comput. Math. 4 (1) (1995) 389–396.
[56] M. Williams, D. Kothe, E. Puckett, Accuracy and convergence of continuum surface tension models, Fluid Dyn. Interfaces (1998) 294–305.
[57] V. Zago, G. Bilotta, A. Hérault, R.A. Dalrymple, L. Fortuna, A. Cappello, G. Ganci, C. Del Negro, Semi-implicit 3D SPH on GPU for lava flows, J. Comput. Phys. 375 (2018) 854–870.
[58] M. Antuono, A. Colagrossi, S. Marrone, D. Molteni, Free-surface flows solved by means of SPH schemes with numerical diffusive terms, Comput. Phys. Commun. 181 (3) (2010) 532–549.
[59] J.J. Monaghan, SPH without a tensile instability, J. Comput. Phys. 159 (2) (2000) 290–311.