To appear in proceedings of PacificVis 2013

Visualization of Piecewise Linear Interface Calculation

Grzegorz K. Karch* Filip Sadlo* Christian Meister † Philipp Rauschenberger †
Kathrin Eisenschmidt † Bernhard Weigand † Thomas Ertl*

*VISUS, †ITLR, University of Stuttgart, Germany

Abstract

Piecewise linear interface calculation (PLIC) is one of the most widely employed reconstruction schemes for the simulation of multiphase flow. In this visualization paper we focus on the reconstruction from the simulation point of view, i.e., we present a framework for the analysis of this reconstruction scheme together with its implications on the overall simulation. By interpreting PLIC reconstruction as an isosurface extraction problem from the first-order Taylor approximation of the underlying volume of fluid field, we obtain a framework for error analysis and geometric representation of the reconstruction including the fluxes involved in the simulation. At the same time this generalizes PLIC to higher-order approximation. We exemplify the utility and versatility of our visualization approach on several multiphase CFD examples.

Index Terms: Simulation and Modeling [I.6.6]: Simulation Output Analysis; Physical Sciences and Engineering [J.2]: Physics

1 Introduction

A multitude of phenomena in science and engineering involve multiphase flow. From a global point of view, single-phase flow is rather the exception—liquids are typically not bounded by solids alone but are in contact with the gaseous phase as well. While in single-phase flow simulation the Navier-Stokes equations account for the advection of velocity and density, multiphase flow additionally requires the advection of the material distribution, i.e., the (liquid) phase. Advecting the material distribution in the same manner as the density field would, however, typically lead to non-sharp interfaces due to numerical diffusion caused by repeated interpolation over time. Thus, techniques are required that keep the interface sharp. A common approach is to track the boundary between liquid and gaseous flow for each time step—necessitating the reconstruction of this interface. Since this step has to be repeated for each time step of the simulation, the focus is on both its accuracy and its computational cost, advocating comparably simple techniques. Piecewise linear interface calculation (PLIC) due to Youngs [21, 22] is an approach that is widely used in computational fluid dynamics (CFD) for these reasons and is therefore the subject of this paper.

Flow visualization concentrated on single-phase flow so far. However, due to differentiation in research and increased compute resources, multiphase flow simulation is steadily gaining importance. This necessitates appropriate visualization techniques. In this paper we focus on the simulation side, i.e., it is not our primary focus to provide tools for investigating the space-time structure of multiphase flow results. Instead, we aim at providing a visualization tool that supports the researcher in the evaluation and assessment with respect to simulation modeling. In other words, we want to support the simulation community by providing simulation-oriented visualization tools that enable and give insights into what is "going on" in flow simulations and how it affects the outcome.

The field of simulation-oriented visualization necessitates algorithmic and numerical proximity to simulation codes. In the present approach we account for this by utilizing algorithms and numerical techniques from the respective solvers. On the one hand, this allows for accurate visualization of solver behavior, on the other hand, however, it makes the visualization techniques strongly dependent on the type of simulation. As a remedy, we advocate reuse of simulation code, preferably by linking against simulation libraries.

In Section 2 we cover works related to this paper. Section 3 provides simulation details relevant for our technique. After presenting our framework for visualization of the PLIC scheme in Section 4, we provide results obtained with our framework in Section 5. Section 6 concludes our work and provides an outlook to future work.

2 Related Work

In the simulation context there are basically two approaches to the computation of fluid interfaces. In Lagrangian schemes the interface is represented explicitly by the moving mesh that divides the domain into regions of different phases [6]. In this case the cells function as control volumes that move with the flow. In the Eulerian schemes, in contrast, the fluid configuration is discretized on a fixed grid, on which the interface must be tracked. For level-set methods [20], the distance to the interface is computed for each cell. In order to avoid smearing due to repeated advection and to assure mass conservation, the interface must be regularly reinitialized. This is in contrast to the volume of fluid (VOF) method with PLIC-reconstructed interfaces. There, the fluid is represented by a volume fraction for each cell [11]. The advantage of the VOF method over the Lagrangian schemes is that it can handle arbitrarily complex flow structures and changes in topology, since no explicit representation of the fluid interface is needed. Several schemes have been proposed for the reconstruction of the interface, with piecewise constant [11, 15], stair-stepped [11], piecewise linear (PLIC) [21, 22, 9, 18] (Section 3.2) approximation, and second-order reconstruction where the fluid surface is build of freely arrangeable planes within a cell [4, 9, 17, 18]. We refer the reader to [7, 8] for a detailed introduction to multiphase flow simulation.

Many visualization methods have been proposed for material interface reconstruction. Bonnell et al. [5] describe an algorithm that is able to reconstruct arbitrarily many interfaces within a single cell. However, the volume fractions enclosed within the reconstructed interfaces can deviate from the original ones. Meredith and Childs [14] developed a more accurate and smooth representation of interfaces with correct connectivity. The smoothness was also addressed in [2] where smoothing and volumetric forces are applied to obtain high quality surfaces. Anderson et al. [3] produce continuous interfaces across cell boundaries for time-varying and static data in arbitrary dimension with bounded error. Obermaier et al. [16] analyze the stability of reconstructed interfaces by comparing with time surfaces. As reconstruction of PLIC patches involves gradient estimation, a related work is that by Hossain et al. [12] presenting gradient estimation methods for field data on regular lattices. These methods were further improved in [1] where storage overhead was reduced. Some of these works visualize 3D PLIC

*e-mail:{karchgz|sadlo|ertl}@visus.uni-stuttgart.de
†e-mail:{christian.meister|philipp.rauschenberger|kathrin.eisenschmidt|bernhard.weigand}@itlr.uni-stuttgart.de

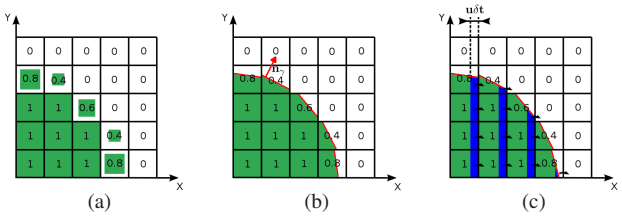


Figure 1: Volume of fluid method with PLIC interface reconstruction. (a) Exemplary distribution of the f-field. (b) PLIC reconstruction (gray) of the interface based on normals $\mathbf{n}_\gamma$. (c) Time integrals $\overline{\phi}_x$ (blue) of f-fluxes over time step δt in x-direction.

patches for comparison with their smooth interface reconstruction techniques. However, they focus on appropriate visualization of the resulting features while we concentrate on visualization of interface reconstruction with respect to the simulation process. Hence, our technique does not aim at providing a smooth representation.

3 Simulation

As our visualization technique is closely linked to the respective simulation, we describe here the relevant details of the simulation code that was used in our experiments. We refer the reader to [7] for a thorough description of these simulation techniques. In our visualization experiments we address multiphase simulations on Eulerian Cartesian grids where the interface between the phases is reconstructed using PLIC. The application of our visualization technique to any other PLIC-based solver code is straightforward, including, e.g., those operating on unstructured grids.[1]

3.1 Fundamentals

Let $\mathbf{u}(\mathbf{x},t) := (u(\mathbf{x}),v(\mathbf{x}),w(\mathbf{x}),t)^\top$ be the simulated flow velocity, with $\mathbf{x} := (x,y,z)^\top$ and time t, and $\rho(\mathbf{x},t)$ be the fluid density. A widely used approach to describe the fluid interface in multiphase flow simulations is the application of the VOF method, originally introduced by Hirt and Nichols [11]. Thereby, to be able to distinguish between the liquid and the gaseous phase, an additional field variable $f(\mathbf{x},t)$, also called VOF-field, describing the volume fraction of the liquid phase in the computational cells, is introduced: for gaseous phase $f = 0$, in the liquid phase $f = 1$, and in the cells with interfaces $0 < f < 1$. An illustration of the f-field in a typical two-phase flow situation is given in Figure 1(a). The temporal and spatial evolution of the f-field is described by the transport equation

$$\frac{\partial f}{\partial t} + \nabla \cdot (f\mathbf{u}) = 0\,, \tag{1}$$

where $\mathbf{u}$ denotes the advection velocity of the interface.

For improved numerics during simulation, u, v, and w are typically kept in staggered grid representation, i.e., u is discretized at the yz-face centers, v at the xz-face centers, and w at the xy-face centers, while the other quantities such as f and ρ are kept in cell-centered representation (e.g., denoted as $f_c := f(\mathbf{x}_c)$ with cell center $\mathbf{x}_c$). Typically, the simulation output of u, v, and w is interpolated to cell-centered representation for long-term storage and post-processing in standard visualization tools. For improved accuracy, our visualization framework supports the staggered representation of velocity, additional to the cell-centered. To this end the simulation experts extended their code for staggered representation output. Nevertheless, since the process pipeline in science and engineering is not yet adapted to staggered-grid representation and thus many simulation data are not available in staggered form, we used both representation types in our experiments.

As we will see in Section 3.2, PLIC (and hence also its visualization) requires the computation of ∇f at the cell center. In order to assure consistency with the simulation process in calculating ∇f, we make use of the respective gradient estimation code. The gradient of cell-centered quantities such as f_c is obtained by computing the partial derivatives using finite differencing at the centers of the cell faces between the respective cells, e.g., the partial derivative in x-direction is computed at the center of the yz-face. Then, the partial derivatives at the centers of 2×2 co-planar faces (in this case yz-faces) are averaged and stored at the node that is shared between those faces. Repeating this procedure in y- and z-direction provides the node-based gradient. Finally, it is interpolated to the cell center.

Beyond the gradient, our framework (Section 4) requires the estimation of higher partial derivatives—in our experiments we also need to compute second-order partial derivatives. To be consistent with the simulation code also with this respect, we use the finite difference scheme employed in the simulation of surface tension. However, since the curvature estimation in the simulation code is based on $\nabla(\nabla f/\|\nabla f\|)$, we could not directly call the respective routine but implemented the Hessian $\nabla(\nabla f)$ using the same scheme. Specifically, we compute it from the node-based gradient by computing the finite differences between neighboring nodes along the cell edges. This provides a partial x-derivative at the center of each x-edge. To obtain the cell-centered Hessian these partial derivatives (in fact its columns) at the four x-edges of a cell are averaged. Repeating this in y- and z-direction provides the full Hessian.

3.2 PLIC Reconstruction

In order to prevent the smearing of f across cells at the interface due to numerical diffusion during simulation, it is important to maintain a sharp interface and to know its exact position. This is typically accomplished by calculating the fluxes in Eq. 1 based on a piecewise linear reconstruction of the interface in each cell, known as PLIC.

As f is maintained in cell-centered representation, i.e., cell-wise constant (Figure 1(a)), it is not immediately clear whether gas, liquid, or a mixture of both is transported across a cell boundary. To determine the actual flux of f across the cell boundaries, the interface is reconstructed by a plane section within each interface cell, i.e., in each cell with $0 < f_c < 1$. In each of these cells the plane's normal vector $\mathbf{n}_\gamma := -\nabla f(\mathbf{x}_c)/\|\nabla f(\mathbf{x}_c)\|$ is determined from the f-field according to the gradient estimation described in Section 3.1. The remaining degree of freedom is the translation τ of the plane along $\mathbf{n}_\gamma$. This translation is chosen such that the volume enclosed between the cell's boundaries and the plane (we denote this as the PLIC polyhedron P) equals f_c, see Figure 1(b) for an illustration. For computational simplicity, this step is typically implemented by an iterative optimization. It is apparent in Figure 1(b) that a piecewise linear reconstruction is subject to C^0 and even C^{-1} discontinuities at the cell boundaries, i.e., between the PLIC surface patches.

3.3 Flux Computation

A central step in CFD is the advection of quantities between the cells of the computational grid. This is typically accomplished by the finite volume method [10], which determines the flux of the quantity through each cell face and integrates this flux over the duration of the time step to obtain the amount of the quantity that has to be moved between the two cells sharing the face. The flux of a quantity can be formulated as the surface integral of the normal component of a flow multiplied by the quantity. In CFD simulation, the flow is usually the velocity field and the surface is a cell face. Hence, the normal component of velocity is required at each cell face.

In the PLIC context, it is the flux of the f-field due to the u-field that has to be evaluated for every cell face. Since the advection is

[1] Our framework only requires derivatives, subdivision, and isosurface extraction using marching cubes [13], all available for unstructured grids.

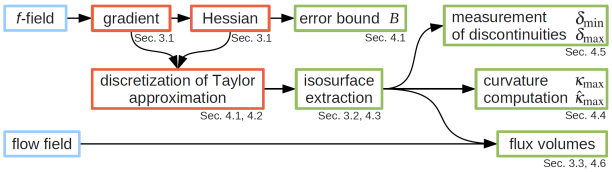


Figure 2: Overall pipeline of our framework with respect to data (blue), computation (orange), and results (green).

trivial in regions where $f = 0$ or $f = 1$ (where f does not change), the flux computation is carried out only for interface cells (where $0 < f_c < 1$). For each of these cells, the PLIC reconstruction is obtained according to Section 3.2. The flux is then computed in a sequential manner using operator splitting (or Strang-splitting [19]), where the order of flux computation in u-, v- and w-direction is reversed in every second time step. Hence, for even time steps, one computes $\overline{\phi}_x$, the time-step δt integral of the flux with respect to the velocity component u in x-direction, by "moving" the PLIC polyhedron by $u\delta t$ in x-direction and measuring the volume that has passed the cell face in (ux)-direction (Figure 1(c)). The f_c-values are updated to f_c' using the $\overline{\phi}_x$ of each cell face by subtracting $\overline{\phi}_x$ of the cell in upstream direction and adding $\overline{\phi}_x$ to the cell in downstream direction. Then a PLIC reconstruction of the f_c'-field is carried out, and the process is repeated in y-direction with respect to the v-component leading to f_c'', finally the procedure is repeated once more for the w-component and z-direction, resulting in f_c''', which are the f_c-values of the next time step. Note that ρ is derived from f in each time step for other purposes such as the advection of velocity, which is, however, of no importance for our technique.

4 Framework

The motivation for this paper is basically twofold. We want to supply techniques that enable the simulation community to judge the quality of the PLIC reconstruction both in terms of geometric approximation and with respect to the impact on the simulation result, in particular in terms of flux computation (Section 3.3). This shall also provide a means to guide the simulation modeling, e.g., with respect to necessary refinement of the simulation grid. We address all these goals by an integrated framework based on cell-wise Taylor approximation of the f-field. Interestingly, this framework at the same time generalizes PLIC to higher-order approximation.

Figure 2 provides an overview of the overall procedure and the organization of the paper. The core is the Taylor approximation of the f-field within each cell of the simulation grid separately, which necessitates the computation of derivatives at the center of each cell. These derivatives are also used for obtaining a per-cell error bound of interface approximation. The Taylor approximation is then discretized, i.e., evaluated at the nodes of the original cell in case of first-order approximation, or sampled on a subdivision of the original cell in case of higher-order approximation. Isosurface extraction from this discretization is finally the basis for different error measures and a visualization of the flux between cells.

4.1 Approximation of f

The PLIC patches (Section 3.2) are planar and perpendicular to $\nabla f(\mathbf{x}_c)$, which is also the case for isosurfaces of the first-order Taylor approximation $\tilde{f}^1(\mathbf{x}_c+\mathbf{h}) := f(\mathbf{x}_c)+(\nabla f(\mathbf{x}_c))\cdot\mathbf{h}$ of f, centered around the cell center $\mathbf{x}_c$, with $f(\mathbf{x}_c) = f_c$, see Figure 3(a). Hence, the PLIC patches can, to the best of our knowledge for the first time, be represented and obtained by marching cubes isosurface extraction from $\tilde{f}^1$, separately within each cell. In this formulation the "PLIC polyhedron volume property", i.e., the translation τ of the PLIC patch along $\nabla f(\mathbf{x}_c)$ (Section 3.2), simply corresponds to the selection of the isolevel σ of $\tilde{f}^1$, as illustrated in Figure 3(a).

Our new interpretation of PLIC reconstruction as an isosurface extraction problem from $\tilde{f}^1$ not only provides a visualization of PLIC patches—it also provides a basis for analyzing the involved approximation error. Knowing that PLIC reconstruction corresponds to isosurface extraction from the first-order approximation $\tilde{f}^1$ of f has several consequences. First, it allows us to bound the approximation error, i.e., how strongly f deviates from $\tilde{f}^1$, the approximation made implicitly by PLIC reconstruction. According to Taylor's theorem (in fact Hille's version for finite differences), the approximation error is bounded by the uniform estimate computed from the remainder of the Taylor approximation, given that the approximated function is continuous and bounded. Since the physically-correct interface is smooth, f is necessarily continuous and it is bounded within $[0,1]$ by definition. Second, it allows us to obtain higher-order approximations of the interface by isosurface extraction from higher-order Taylor approximations $\tilde{f}^k$ of f. Both applications require $\tilde{f}^k$. Generally, the multivariate k^{th}-order Taylor approximation of f, centered at the cell center $\mathbf{x}_c$ reads

$$\tilde{f}^k(\mathbf{x}_c+\mathbf{h}) := \sum_{|\boldsymbol{\alpha}|\le k} \frac{\partial^{\boldsymbol{\alpha}} f(\mathbf{x}_c)}{\boldsymbol{\alpha}!}\mathbf{h}^{\boldsymbol{\alpha}} \tag{2}$$

with $\mathbf{x}_c = (x_1,x_2,x_3)^\top, \mathbf{h} = (h_1,h_2,h_3)^\top \in \mathbb{R}^3$ and $\boldsymbol{\alpha} \in \mathbb{N}_0^3$, using *multi-index notation*. For $k=2$ this results in the second-order Taylor approximation

$$\begin{aligned}\tilde{f}^2(\mathbf{x}+\mathbf{h}) = f(\mathbf{x}) &\\ &+ \frac{\partial f(\mathbf{x})}{\partial x_1}h_1 + \frac{\partial f(\mathbf{x})}{\partial x_2}h_2 + \frac{\partial f(\mathbf{x})}{\partial x_3}h_3 \Big\}\, 1^{st}\\ &+ \frac{\partial^2 f(\mathbf{x})}{\partial x_1\partial x_2}h_1h_2 + \frac{\partial^2 f(\mathbf{x})}{\partial x_1\partial x_3}h_1h_3 + \frac{\partial^2 f(\mathbf{x})}{\partial x_2\partial x_3}h_2h_3\\ &+ \frac{\partial^2 f(\mathbf{x})}{2\partial x_1^2}h_1^2 + \frac{\partial^2 f(\mathbf{x})}{2\partial x_2^2}h_2^2 + \frac{\partial^2 f(\mathbf{x})}{2\partial x_3^2}h_3^2 \Big\}\, 2^{nd}\end{aligned} \tag{3}$$

with the respective zeroth, first, and second-order terms. We constrain our investigation to a maximum of order two, although any order is straightforward to use within our framework.

We will now derive a bound on the approximation error of $\tilde{f}^k$, which will lead to a first error measure of PLIC reconstruction. According to the multivariate version of Taylor's theorem, since f is $k+1$ times continuously differentiable (see above) within the cell volume C, there are $R_{\boldsymbol{\beta}} : \mathbb{R}^3 \to \mathbb{R}$ with $\boldsymbol{\beta} \in \mathbb{N}_0^3$ such that

$$f(\mathbf{x}_c+\mathbf{h}) = \sum_{|\boldsymbol{\alpha}|\le k} \frac{\partial^{\boldsymbol{\alpha}} f(\mathbf{x}_c)}{\boldsymbol{\alpha}!}\mathbf{h}^{\boldsymbol{\alpha}} + \sum_{|\boldsymbol{\beta}|=k+1} R_{\boldsymbol{\beta}}(\mathbf{x})\mathbf{h}^{\boldsymbol{\beta}}. \tag{4}$$

It is known that from this follows

$$|R_{\boldsymbol{\beta}}(\mathbf{x})| \le \frac{|\boldsymbol{\beta}|}{\boldsymbol{\beta}!} \max_{|\boldsymbol{\gamma}|=|\boldsymbol{\beta}|} \max_{\mathbf{y}\in C} |\partial^{\boldsymbol{\gamma}} f(\mathbf{y})| =: M \tag{5}$$

with $\mathbf{x} \in C$, leading to the uniform estimate that bounds the approximation error of the k^{th}-order Taylor approximation:

$$f(\mathbf{x}) - \tilde{f}^k(\mathbf{x}) \le Mc^{k+1} \tag{6}$$

within C with c being the maximum of the cell's x, y, and z-extent.

Hence, the upper bound B for the approximation error of f with respect to PLIC reconstruction, which represents a first-order approximation of f, is obtained by $k=1$ in Eq. 6 and inserting Eq. 5:

$$B := \frac{2}{2} \max_{|\boldsymbol{\gamma}|=2} \max_{\mathbf{y}\in C} |\partial^{\boldsymbol{\gamma}} f(\mathbf{y})| \cdot c^2 = \max_{\mathbf{y}\in C} \|\nabla(\nabla f(\mathbf{y}))\|_{\max} \cdot c^2 \tag{7}$$

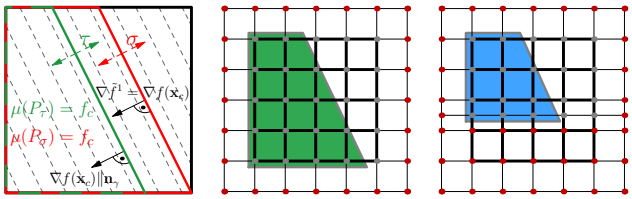


Figure 3: (a) PLIC as isosurface extraction problem in $\tilde{f}^1$. First-order approximation $\tilde{f}^1$ exhibits planar isosurfaces (dashed lines) perpendicular to $\nabla f(\mathbf{x}_c)$ (red), and so does PLIC by construction (green). Hence, instead of adjusting τ such that the volume $\mu(P_\tau)$ of the enclosed polyhedron P_τ equals f_c, one can adjust isolevel σ accordingly. (b) and (c) Discretization (gray nodes) of $\tilde{f}^k$ within a simulation cell (bold grid). (b) An isosurface (gray polygon) representing the PLIC polyhedron P (green) is obtained by setting all boundary nodes (red) of the supersampling grid to `-FLT_MAX`. (c) Cells are split at "reversely advected" cell face to generate flux volume (blue), $\tilde{f}^k$ is interpolated at the new (gray) nodes. Nodes below the face are set to `-FLT_MAX` to obtain the respective isosurface representation (gray).

with the maximum norm $\|A\|_{\max} = \max\{|a_{ij}|\}$. Thus, we need to determine the largest modulus of the elements of the Hessian of f within C and multiply it by the square of the largest cell extent.[2]

Since we build on finite differences that involve, depending on the order of the partial derivatives, the interpolation of derivatives from cell nodes, cell edges, or cell faces (Section 3.1), the variation of the Hessian within a cell is that of a tensor product linear interpolation. As detailed in Section 3.1 the columns of the Hessian are bilinearly interpolated at the cell center from those at the cell edges. Hence, the absolute maximum of the elements of the Hessian within the cell is the absolute maximum of the elements of the Hessian columns determined at the cell edges, which is easily evaluated and gives our error bound on the first-order approximation of f (i.e., with respect to the approximation implicated by PLIC).

Note that since PLIC reconstruction corresponds to first-order approximation of f, B not only provides a conservative bound on possible inaccuracies due to the advection of the f-field, it also indicates more general discretization problems of f, e.g., aliasing with respect to resolution and orientation of the simulation grid, as observed in Figure 4(e) and in particular in Figure 6. However, although B provides a measure for the approximation of f with respect to PLIC (including the offset of the PLIC patch along $\nabla f(\mathbf{x}_c)$) and can be multiplied by c to provide an estimate how far the isosurface moves along $\nabla \tilde{f}^k$ if the isovalue is varied by the value B (Figure 6), it is not able to directly provide insight into the deviation of the shape of the planar PLIC patch from the shape of the curved isosurface of $\tilde{f}^k$. To this end, we visualize the curvature of these isosurfaces (Section 4.4) and the discontinuities between the PLIC patches (Section 4.5). Since both require these isosurfaces, we describe next how they are obtained.

4.2 Discretization

As our framework requires the computation of isosurfaces of $\tilde{f}^k$, we have to subdivide each original cell of the simulation grid by a factor $n \geq 1$ into a new grid G_c consisting of n^3 cells and resample $\tilde{f}^k$ at the nodes of G_c. Isosurface extraction from this grid readily produces the PLIC patches if $k = 1$ and higher-order PLIC generalizations for $k > 1$. Since $\tilde{f}^1$ is linear, the traditional PLIC case requires no subdivision, (i.e., $n = 1$ applies). However, in order to determine the isovalue σ that produces the PLIC patch at the right offset along $\nabla f(\mathbf{x}_c)$ (Figure 3(a)), we need to determine the volume $\mu(P_\sigma)$ of the PLIC polyhedron (Section 3.2). Since we want to address generic cases with $k \geq 1$, a possible approach would be to apply triangulation to close the isosurface along the respective face parts of the original cell. This would be, however, a demanding task with respect to computational cost and implementation complexity. Instead, we make use of the marching cubes algorithm also for this task, which therefore would also allow for parallelization (nevertheless, we address parallelization on a GPU as future work, as the timings of our prototype (Section 5) already allow for productive operation). We achieve this by extending G_c with an additional layer of cells (which adds a layer of additional nodes N) along its boundary,[3] i.e., we obtain a uniform grid of $(n+2)^3$ cells where the inner n^3 cells are located within the original cell and the remaining cells are located outside, see Figure 3(b). By setting the value at each node in N to a very large negative value λ (in our implementation $\lambda =$ `-FLT_MAX`), any isosurface is closed along the face regions of the original cell, providing the triangulated representation of P_σ corresponding to isolevel σ. Note that the introduced inaccuracy is negligible (in the order of numerical accuracy) because the additional part(s) of the isosurface are located in the outer layer of cells of G_c but very close to the faces of the original cell since λ dominates the $\tilde{f}^k$-values. However, since this is only the case if the partial derivatives of f are reasonably bounded, i.e., below the order of λ, we test if the modulus of the values within G_c exceeds $-\lambda \cdot 10^{-3}$ and warn the user in such a case (which, however, never was the case in our experiments).

Since the triangles in the mesh of P_σ are oriented consistently by the marching cubes algorithm, the volume μ of P_σ can directly be measured by $\mu(P_\sigma) = \sum_{t \in P_\sigma} \mathbf{v}_1 \cdot (\mathbf{v}_2 \times \mathbf{v}_3)/6$, with $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ being the three vertices of triangle t.

The PLIC patches are obtained either from the isosurfaces of the n^3 grid directly, or from the closed isosurfaces of the $(n+2)^3$ grid by discarding the isosurface parts in the outer cell layer.

4.3 Determination of Isovalue

The remaining building block for cell-wise PLIC reconstruction by our framework is the determination of the isolevel σ such that $\mu(P_\sigma) = f_c$. As in traditional PLIC reconstruction (Section 3.2) we address the problem by iterative optimization, in our case we apply the bisection method to find σ, see Algorithm 1. For traditional PLIC reconstruction, $k = 1$ is used. However, using $k > 1$ provides higher-order interface approximation, which we use for comparison and visualization of the approximation error with respect to the curvature of the interface in this paper. We used $n_i = 10$ in our experiments, hence the resulting quantization of σ is $(f_{\max} - f_{\min})/2^{n_i+1}$.

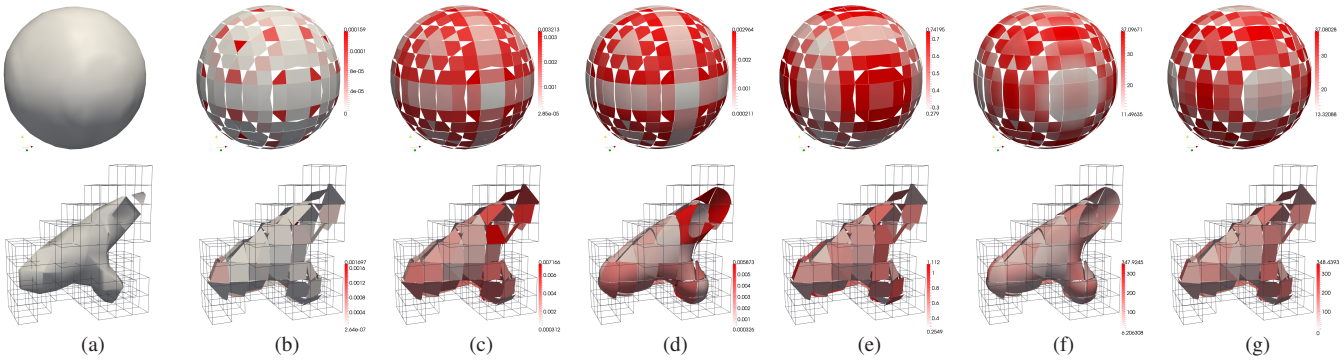
4.4 Measurement of Isosurface Curvature

While B (Section 4.1) bounds the approximation error with respect to the f-field in terms of PLIC reconstruction, we derive here a measure for the PLIC approximation error with respect to the shape of the interface. In compliance with the derivation of B, we compare the isosurface of $\tilde{f}^1$ at isolevel σ_1 with the corresponding (curved) isosurface of $\tilde{f}^2$ at isolevel σ_2. Both σ_1 and σ_2 are obtained according to Section 4.3, i.e., such that $\mu(P_1) = \mu(P_2) = f_c$.

The principal curvatures of an isosurface of $\tilde{f}^k$ at point $\mathbf{x}$ are given by the generally two nonzero eigenvalues λ_1 and λ_2 of $\nabla(\nabla \tilde{f}^k(\mathbf{x})/\|\nabla \tilde{f}^k(\mathbf{x})\|)$. We extract the isosurface of $\tilde{f}^2$ at isolevel σ_2 and evaluate $\kappa_{\max} := \max(|\lambda_1|, |\lambda_2|)$ at its vertices. These values are either directly visualized on the surface (see, e.g., Figure 4(f)), or the maximum $\hat{\kappa}_{\max}$ over the isosurface within the original cell is determined and this maximum is visualized by uniform color on the corresponding PLIC surface patch (see, e.g., Figure 4(g)). The

[2] For the extension to unstructured grids, one can use the cell's bounding box instead of C.

[3] For extension to unstructured grids, this step would need to make use of a simple meshing algorithm.



(a) (b) (c) (d) (e) (f) (g)

Figure 4: Our measures applied to the Sphere dataset (top row, rendered with backface culling to make discontinuity gaps better visible), and to a small part of a ligament (bottom row, without backface culling) in the last time step of Figure 8. Columns: Traditional isosurface (a), minimum discontinuity $\delta_{\min}$ (b), maximum discontinuity $\delta_{\max}$ on PLIC (c) and on second-order patches (d), bound B (Eq. 7) on approximation error with respect to f (e), second-order PLIC patches colored with surface curvature $\kappa_{\max}$ (f), and cell-wise maximum $\hat{\kappa}_{\max}$ of (f) colored on the PLIC patches (g). The radius of the sphere according to our PLIC reconstruction is $0.0505\,\mathrm{m}$ (initialized as $0.05\,\mathrm{m}$), hence the curvature should be ideally $20\,\mathrm{m}^{-1}$. While (b), (c), and (d) account for the displacement of the patches along $\nabla f(\mathbf{x})$, (e), (f), and (g) take into account only partial derivatives of f and are hence not directly dependent on the displacement.

Algorithm 1 Bisection to find isovalue σ such that $\mu(P_\sigma) = f_c$.

$f_{\min} \leftarrow \min_{\eta \in G_c \setminus N} \tilde{f}^k(\eta)$ {minimum of nodes η in G_c without N}
$f_{\max} \leftarrow \max_{\eta \in G_c \setminus N} \tilde{f}^k(\eta)$ {maximum of nodes η in G_c without N}
$\sigma_{\min} = f_{\min}$
$\sigma_{\max} = f_{\max}$
for $i = 0$ **to** n_i **do** {n_i bisection iterations}
  $\sigma \leftarrow (\sigma_{\min} + \sigma_{\max})/2$
  obtain P_σ by marching cubes
  measure $\mu(P_\sigma)$
  if $\mu(P_\sigma) < f_c$ **then**
    $\sigma_{\max} \leftarrow \sigma$
  else
    $\sigma_{\min} \leftarrow \sigma$
  end if
end for
$\sigma \leftarrow (\sigma_{\min} + \sigma_{\max})/2$

latter provides the maximum deviation of the PLIC patch from the second-order interface approximation in terms of curvature, while the former enables detailed inspection of this second-order surface approximation. If the absolute distance between the two isosurfaces is of interest, as in mesh resolution analysis (see, e.g., Figure 7), $\hat{\kappa}_{\max} \cdot c^2$ provides a conservative bound w.r.t. the cell extent c.

4.5 Measurement of C^{-1} Discontinuities

While the technique presented in Section 4.4 captures the deviation of the PLIC patch shape from that of a second-order approximation, it is not capable of measuring discontinuities between the patches. Regarding flux computation (Section 3.3), $C^{\geq 0}$ discontinuities are of lower impact than C^{-1} discontinuities because these represent gaps in the reconstructed interface. We complete our analysis stage for PLIC patches by providing here a respective technique.

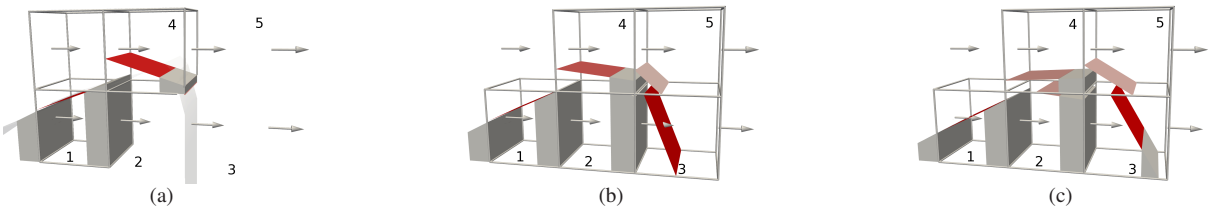
Our discontinuity measure is visualized per cell, i.e., by a uniform color applied to the respective isosurface of $\tilde{f}^k$ (it is applicable to both PLIC cases with $k = 1$ and higher-order approximations with $k > 1$). We obtain this cell value by extracting the boundary curves of the isosurface(s) within each original cell (note that multiple isosurface parts can result in case of $k > 1$) and by measuring the minimum Euclidean distance between these boundary curves and all other boundary curves of the other cells. We measure these distances by supersampling each edge of the current cell's mesh boundary polygons by r samples (in our experiments we used $r = 9$) and determining for each sample the shortest distance to all boundary curves residing in other cells. These other boundary curves do not need to be supersampled because the involved point-to-segment distance can be determined exactly. From the resulting minimum distances along the current cell's boundary polygons we take both their minimum $\delta_{\min}$ and their maximum $\delta_{\max}$. Both measures can be mapped to the isosurface(s) of the current cell using uniform color. The measure $\delta_{\min}$ provides a lower bound on the discontinuity of the current cell (i.e., the "best case") while $\delta_{\max}$ indicates the upper bound (i.e., the "worst case"). Please see Figure 4(b)–(d).

4.6 Determination of Flux Volumes

So far we address the approximation properties of the f-field (in terms of B from Section 4.1) and those of the PLIC reconstruction with respect to geometric approximation of the interface (Sections 4.4 and 4.5). Although they already give insight into the implications of PLIC reconstruction in simulation codes, they do not provide a direct approach to the impact on the simulation result. Therefore we supply our framework with the visualization of the involved fluxes. Although flux is an instantaneous measure, this is the only part of our framework that has to take into account the progress in time, however, only in the sense that the duration of the time step is required. All time steps of the simulation data, such as the $\mathbf{u}$- and f-fields are visualized independently. As discussed in Section 3.3, the finite volume method has to integrate the flux at a cell face over the duration of the time step to obtain the amount of the quantity that has to be moved between the respective cells. In our case f is the quantity and the transported amount is obtained by "moving" the PLIC polyhedron P sequentially by u, v, and w, and obtaining its part that passes the respective downstream cell face.

To obtain a geometric representation of this volume for visualization, we make use of the marching cubes approach that is used for obtaining the triangulation of P (Section 4.2). The only difference is that we split the cells (inserting new nodes) of G_c where the respective cell face intersects G_c if the face is advected in reverse direction, see Figure 3(c). We resample $\tilde{f}^k$ at the inserted nodes and set all nodes of G_c that are located on the other side of the advected face to `-FLT_MAX`, with the effect that the resulting isosurface (taken at isolevel σ that was determined for the reconstruction in this cell according to Section 4.3) represents the flux volume, i.e., the amount of f that is advected through the respective cell face.

As flux computation is accomplished using operator splitting,



(a) (b) (c)

Figure 5: Initial time evolution of Zalesak dataset. Interface cells shown as boxes, flux volumes as gray polyhedra, and Zalesak pattern by transparent isosurface. In the initial state (a), cell 2 is completely filled and cell 3 is empty. However, the gradient in cell 3 does not point straight to the right but has an upward component due to finite differencing. Hence, the resulting PLIC patch degenerates to the red strip visible at the upper left edge of cell 3. Adding the flux volumes to their downstream cells (and subtracting them from their current cells) updates the f_c values as reconstructed by the PLIC patches in (b). Second state (b): since f_c is now larger in cell 3, one can see the respective patch and its inclination. The flux volume at the right face of cell 2 is now larger than the flux volume in cell 1, hence f_c will drop in cell 2 in the next time step. Third state (c): f_c is now below one in cell 2, causing a PLIC patch in cell 2. However, due to steeper gradients in cell 3, its PLIC normal points slightly to the right. This is why the gaseous phase is located at the upper right instead of the upper left edge of cell 2. The visualization nicely illustrates a reconstruction error (which then implies an error in flux calculation) due to the normal vector calculation. Cell 3 has obtained more fluid and the overall f_c-distribution causes further inclination of the patch in cell 3, further deteriorating the Zalesak pattern. This process continues over time.

the visualization of the fluxes in u-direction is straightforward. For those in v-direction one needs to first compute the flux volumes in u-direction, update the f_c-field therefrom accordingly, repeat PLIC reconstruction on the updated f_c-field, and compute the flux volumes then in v-direction. For the flux volumes in w-direction, one needs to repeat this process once more in w-direction (Section 3.3). Note that typically only every m^{th} step is output during simulation and hence the spacing between simulation results is larger than the simulation step. Hence, either every time step has to be output for analysis by our technique or the step size has to be provided.

5 Results

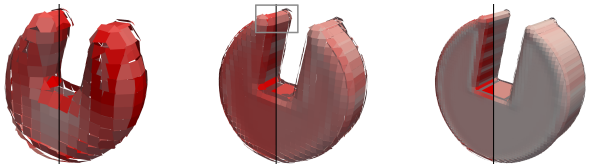
This section reports the results obtained with our approach together with the assessment by the CFD experts that have coauthored this paper. We applied our framework to several multiphase flow simulations in the context of droplet dynamics. Table 1 provides timing results of our implementation. The computation of the discontinuities $\delta_{\min}$ and $\delta_{\max}$ is clearly the most expensive step. However, we used only a basic search structure for the distance test and this step would lend itself to GPU acceleration. Performance is significantly reduced for second-order patches as the number of triangles per patch increases quadratically due to the involved supersampling. However, the implementation allows for productive operation.

5.1 Sphere Dataset

As a reference configuration that provides a wide spectrum of normal directions, a stationary sphere with known curvature $\kappa = 20\,\text{m}^{-1}$ was initialized using the simulation code. This serves for evaluation and illustration of our framework, and provides insight into the dependency on the sampling grid. We applied all visualizations of our framework to the 64^3 dataset, except for the flow volumes as these require velocity data. Figure 4(b)–(g) (top row) provide the respective results. Interestingly, each measure captures different discretization properties of PLIC. While the discontinuity measures $\delta_{\min}$ and $\delta_{\max}$ exhibit low values on the sphere where $(x = 0) \vee (y = 0) \vee (z = 0)$ with $\vee$ representing "or", the approximation-based measures exhibit high (in case of B) or low (in case of $\kappa_{\max}$) values at $x_{\min} \vee x_{\max} \vee y_{\min} \vee y_{\max} \vee z_{\min} \vee z_{\max}$ of the sphere. This complies with the flattened (low $\kappa_{\max}$) parts visible in Figure 4(a) (top) and the involved aliasing in general (large B). Hence, discontinuities are typically small at axis-aligned patches, while aliasing (B) is typically large in these cases, however, with the potential benefit of flattened interfaces and hence better approximation by PLIC. This behavior is expected in PLIC-based CFD.

Table 1: Timings (in seconds) for the datasets from Figure 8 (C1: 32^3, C2: 64^3, C3: 128^3) and from Figure 9 (M), QUAD: second-order surface, Flux: flux volumes. Total exectution time in brackets.

Data	PLIC	QUAD	Curvature	$\delta_{\max}$ PLIC	Flux PLIC	Flux QUAD
C1	0.27 (0.29)	3.28 (3.32)	0.12 (3.5)	3.89 (4.19)	0.61 (0.91)	6.84 (10.1)
C2	0.99 (1.19)	12.2 (12.5)	0.76 (13.4)	29.8 (31.0)	2.44 (3.66)	29.8 (42.4)
C3	2.84 (3.96)	35.9 (37.5)	3.91 (41.7)	236 (240)	7.83 (12.0)	98.4 (136)
M	10.4 (13.6)	128 (132)	7.27 (140)	419 (434)	26.9 (67.2)	312 (475)

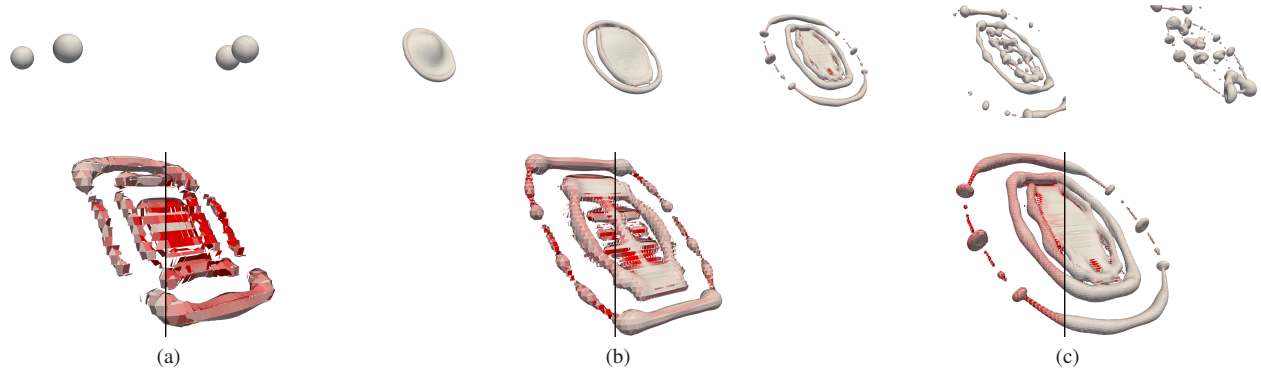


(a) (b) (c)

Figure 6: Zalesak dataset at time step 25 (time 0.0375 s) and simulation grid $32^2 \times 16$ (a), $64^2 \times 32$ (b), and $128^2 \times 64$ (c), visualized by PLIC patches. The coloring by B (left halves) and $c \cdot B$ (right halves) conveys approximation problems regarding the f-field, including aliasing. The gray box depicts the region analyzed in Figure 5, however for the onset of the simulation.

5.2 Zalesak Dataset

A 3D liquid disc with a cut out reminiscent of Zalesak's 2D disc is initialized in a Couette flow $\mathbf{u}(\mathbf{x}) = (ay, 0, 0)^\top$ with $a = 5\,\text{s}^{-1}$ with the lower domain boundary at rest and the upper one moving at a velocity of $1\,\text{m/s}$. The simulation is carried out at different resolutions of the simulation grid and consists of 109 simulation steps which are output at the frequency of a single simulation time step $\delta t = 0.0015$ s. Figure 6 visualizes the dataset using PLIC patches colored with B. It is apparent that both B and the robustness $c \cdot B$ (Section 4.1) of the PLIC patch with respect to σ are larger on the front face of the disc in (b) than in (a), although (b) was simulated at higher resolution. This is due to aliasing with respect to the resolution of the simulation grid, which is also visible as periodic pattern at the cutout in (c). The Zalesak shape is traditionally used to investigate the influence of numerical diffusion and reconstruction properties in general since it features surfaces of greatly varying curvature, questions that lend themselves to analysis based on B. It



(a) (b) (c)

Figure 7: Colliding Drops dataset. Visualization by PLIC patches with $\hat{\kappa}_{\max}$ (left halves) and $\hat{\kappa}_{\max} \cdot c$ (right halves), for same time step at resolution 32^3 (a), 64^3 (b), 128^3 (c). $\hat{\kappa}_{\max} \cdot c$ provides good estimate of approximation quality. Top: Time evolution by PLIC colored with $\hat{\kappa}_{\max}$.

is apparent in particular in (c) that the sharp edges at the top deteriorate during advection. In Figure 5 we demonstrate how our framework can be used to investigate the reasons for this process in terms of PLIC reconstruction and the involved fluxes. Interestingly, a reconstruction problem, which can lead to "multiple fronts" (Figure 5(c)), has been identified using our technique in this context.

5.3 Colliding Drops Dataset

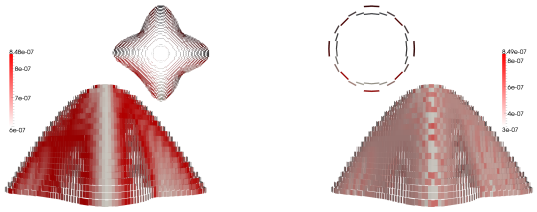
This dataset consists of a simulation of a peripheral droplet-droplet collision. Figure 7 (top row) provides an overview of the temporal evolution. The simulation has been conducted at different resolutions to investigate the impact of grid resolution. It turned out that $\kappa_{\max} \cdot c^2$ provides a good estimate of approximation quality. It is subject to future work to investigate this approach further, e.g., for adaptive mesh refinement during simulation. It is apparent that resolutions (a) and (b) are insufficient while (c) starts to exhibit physically correct behavior, consistent with the estimation $\kappa_{\max} \cdot c^2$. This can be explained by the fact that disintegration is closely related to surface tension and hence to the curvature of the interface.

5.4 Merging Drops Dataset

This case models the peripheral collision of two rain drops of different size discretized on a grid of $512 \times 256 \times 256$ cells, see Figure 8 for an overview of the time evolution. This represents a rather complex case due to droplet disintegration and formation of small ligaments that are resolved with only few cells and therefore exhibit large discontinuities in PLIC reconstruction. As visualized in Figure 9, the physically important breakup of the ligaments is essentially dominated by the f-fluxes. Our visualization in Figure 9(c) indicates that the instabilities leading to breakup of sheets are suppressed by the PLIC reconstruction, i.e., that the flux out of the breakup cell would be larger if second-order interface reconstruction (d) would be used (visible at the larger flux volume at the lower side of the breakup cell).

5.5 Freezing Drop Dataset

In this simulation the freezing process of supercooled water, i.e., pure water which may remain liquid for temperatures well below the freezing point of water, is investigated. The setup for a 2D simulation of an initially round particle is visualized in Figure 10. Freezing starts and the particle grows circularly. Since the ice-water interface is morphologically unstable, small perturbations can initiate dendrite growth. As possible causes include the variation in PLIC patch size, we visualized them. Water has a hexagonal molecular lattice and should exhibit a six-fold symmetry of the ice crystal. Currently no anisotropy effects are included in the model and therefore circular growth is expected. However, as can be seen in



(a) (b)

Figure 10: Freezing Drop 2D dataset stacked in space-time (time axis downward). View along time visualizes the growth of the crystal (upper right in (a)). The PLIC patch size (bottom of (a)) is smaller at the dendrites. Although, the crystal growth is faster in these areas. Discontinuities $\delta_{\max}$ are smaller at the dendrites (bottom of (b)), providing a possible explanation. Another unexpected result are two parallel PLIC patches per axis at initialization (upper left in (b)).

Figure 10, a four-fold symmetry is attained with the dendrites being aligned along the axes of the Cartesian grid. The visualization shows that the dendrites grow in directions where the PLIC patch size is smaller. A possible explanation, supported by our visualization (Figure 10(b)), is with respect to the discontinuities between the patches, which are smaller at the dendrites. Finally, our visualization technique revealed that there exist two parallel PLIC-planes in all four directions aligned with the grid axes at initialization, which might as well trigger dendritic growth.

5.6 Discussion

The investigated examples represent different research problems that require different visualization approaches. While all measures provide valuable insight in the Sphere dataset (Section 5.1), discretization problems of the f-field, such as in the Zalesak dataset, are best investigated with B. The investigation of the overall approximation quality, also with respect to physics, lends itself well to analysis based on $\kappa_{\max} \cdot c^2$ (Section 5.3). Our flux-based visualization approach is particularly well suited for understanding the simulation behavior in small regions of interest, such as the tip of the Zalesak dataset or the ligament breakup in the Merging Drops dataset. As mentioned in Section 4.5, C^{-1} discontinuities have a high impact on flux computation and hence are useful in related investigations as shown in Figures 5 and 9. Finally, the Freezing Drop dataset represents an example where the size of the PLIC patches correlate with simulation behavior. However, we can provide only exemplary insight into solver behavior as a thorough application of our tools together with detailed examination and interpretation of any of our examples would already exceed the scope of this paper.

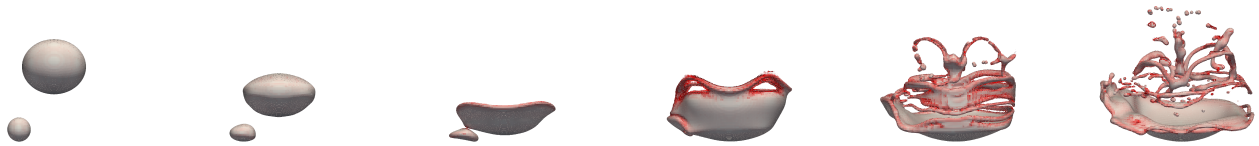
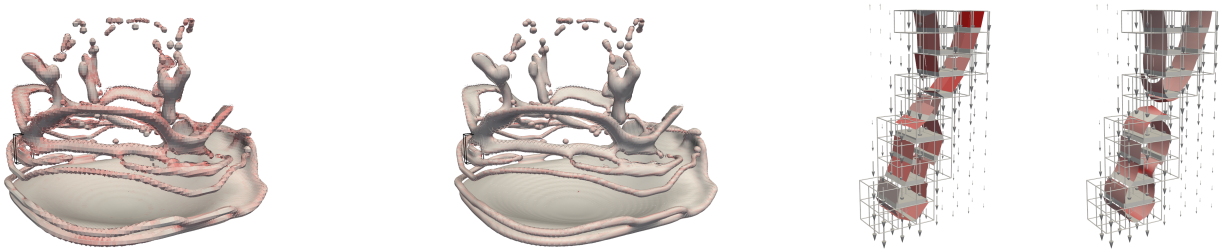


Figure 8: Time sequence of Merging Drops dataset. Visualization by PLIC patches colored by $\delta_{\max}$, in frame of reference moving with the drops.



(a) (b) (c) (d)

Figure 9: Merging Drops dataset. PLIC patches colored by discontinuity $\delta_{\max}$ (a), by curvature $\kappa_{\max}$ on second-order surface reconstruction (b), and a closeup thereof visualized using surface reconstruction and flux volumes based on first-order (PLIC) approximation (c), and second-order approximation (d). As shown in (d), disconnection should occur due to larger flux volume in the lower region. In (c) PLIC suppresses the breakup.

6 Conclusion

We presented a technique for the simulation-oriented visualization of multiphase flow simulations based on piecewise linear interface calculation (PLIC). By identifying PLIC reconstruction as an isosurface extraction problem from the first-order Taylor approximation of the volume of fluid (VOF) field we obtained a versatile framework. On the one hand it allows us to derive error bounds on the implicit approximation of the VOF-field, on the other hand it provides several geometry-based error measures with respect to the shape of the reconstruction and the discontinuities at cell boundaries. It also provides geometric representations of the volume enclosed by PLIC as well as the flux of the VOF-field. Finally, it readily generalizes PLIC reconstruction to higher-order approximation. Next, we plan to examine the utility of our generalization and our error measures for the simulation of multiphase flow.

Acknowledgements

This work was supported in part by the Cluster of Excellence in Simulation Technology (EXC 310/1) and the Collaborative Research Centre SFB-TRR 75 at the University of Stuttgart.

References

[1] U. Alim, T. Möller, and L. Condat. Gradient estimation revitalized. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1495–1504, 2010.

[2] J. Anderson, C. Garth, M. Duchaineau, and K. Joy. Smooth, volume-accurate material interface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):802–814, 2010.

[3] J. C. Anderson, C. Garth, M. A. Duchaineau, and K. I. Joy. Discrete multi-material interface reconstruction for volume fraction data. *Computer Graphics Forum*, 27(3):1015–1022, 2008.

[4] N. Ashgriz and J. Poo. FLAIR: Flux line-segment model for advection and interface reconstruction. *Journal of Computational Physics*, 93(2):449–468, 1991.

[5] K. S. Bonnell, M. A. Duchaineau, D. R. Schikore, B. Hamann, and K. I. Joy. Material interface reconstruction. *IEEE Transactions on Visualization Computer Graphics*, 9:500–511, 2003.

[6] W. Cao, W. Huang, and R. D. Russell. A moving mesh method based on the geometric conservation law. *SIAM Journal on Scientific Computing*, 24(1):118–142, 2002.

[7] D. Fuster, G. Agbaglah, C. Josserand, S. Popinet, and S. Zaleski. Numerical simulation of droplets, bubbles and waves: state of the art. *Fluid Dynamics Research*, 41(6):065001, 2009.

[8] H. Gomaa, N. Roth, J. Schlottke, and B. Weigand. DNS Calculations for the Modeling of Real Industrial Applications. *Atomization and Sprays*, 20(4):281–296, 2010.

[9] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, and S. Zaleski. Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows. *Journal of Computational Physics*, 152(2):423–456, July 1999.

[10] C. Hirsch. *Numerical computation of internal & external flows*. John Wiley & Sons, Ltd., 2007.

[11] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–225, Jan. 1981.

[12] Z. Hossain, U. Alim, and T. Möller. Toward high-quality gradient estimation on regular lattices. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):426–439, 2011.

[13] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, 1987.

[14] J. S. Meredith and H. Childs. Visualization and Analysis-Oriented Reconstruction of Material Interfaces. *Computer Graphics Forum*, 29:1241–1250, 2010.

[15] W. Noh and P. Woodward. SLIC (simple line interface calculation). volume 59 of *Lecture Notes in Physics*, pages 330–340. 1976.

[16] H. Obermaier, F. Chen, H. Hagen, and K. I. Joy. Visualization of material interface stability. In *IEEE Pacific Visualization (PacificVis 2012)*, pages 225–232, 2012.

[17] J. E. Pilliod Jr. and E. G. Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *Journal of Computational Physics*, 199(2):465–502, 2004.

[18] W. J. Rider and D. B. Kothe. Reconstructing Volume Tracking. *Journal of Computational Physics*, 141(2):112–152, 1998.

[19] G. Strang. On construction and comparison of difference schemes. *Siam Journal On Numerical Analysis*, 5(3):506–517, 1968.

[20] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114(1):146–159, 1994.

[21] D. L. Youngs. Time-dependent multi-material flow with large fluid distortion. *Numerical Methods for Fluid Dynamics*, pages 273–285, 1982.

[22] D. L. Youngs. An interface tracking method for a 3D Eulerian hydrodynamics code. Technical Report 44/92/35, AWRE, 1984.