A review of level-set methods for two-phase flows

Arthur Kerst

Supervisor: Dennis den Ouden-van der Horst

February 28, 2020

Contents

1 Introduction

Multiphase flow simulations are fundamental tools in many industrial applications and natural processes. Examples are rain drops in the air, free surface flows in the ocean, the dispersion of two immiscible fluids into each other to create emulsions, liquid phase sintering and inkjet printing [7, 24]. The flow field can be modelled with the Navier-Stokes equation, but the difficulty of modelling a two-phase flow lies in accurately tracking or capturing the interface.

In the last three decades several methods have been developed for solving the two-phase flow. Each method has its advantages and disadvantages. While some methods suffer from poor mass conservation properties, other methods suffer from the inability to handle complex domains with unstructured meshes. Popular methods are the front-tracking method, marker-and-cell method, level-set method and volume-of-fluid method. Over the years steady improvements have been made by proposing various variations to the existing methods.

The goal for this project is to develop a method using a continuous Galerkin approach for solving the level-set equation on triangular control volumes. Ideally, the developed method should conserve volume exactly, give a continuous description of the interface and track the interface accurately. Furthermore, this method must be able to handle unstructured triangular grids. In order to achieve this, we first investigate the available methods and study their advantages and disadvantages.

This report is organized as follows. In Section 2 an overview of methods is given. In Section 3 we describe discretization methods for the level-set method and volume-of-fluid method. Some results are presented in Section 4, which is followed by a conclusion in Section 5.

2 Overview of methods

2.1 Types of methods for modelling two-phase flow

Numerical simulations of two-phase flow are difficult and far more challenging than a single phase flow. Besides demanding mass conservation, accurately modelling the interface, such that it remains smooth and sharp, is also very important. An accurate interface is essential, especially when the normal vector and curvature are utilized for computing the surface tension [17]. The interface separating the two fluids must be either tracked or captured in the same time step as the flow field evolution [24]. Interface-tracking methods are Lagrangian, that is, the interface is explicitly represented by the mesh. Whenever the interface moves or deforms, the interface has to be rebuild [9]. These methods are very accurate and efficient for moving interfaces with small deformation, but they do not handle disconnecting and reconnecting of the interface well [12]. Interface-capturing methods are Eulerian, where the interface is described by an implicit function on a fixed mesh [12]. Although these methods are robust and have a wide range of applicability, they usually require a higher mesh resolution.

The interface-tracking techniques either employ a deforming mesh that fits to the interface, such as arbitrary Lagrangian-Eulerian (ALE) methods, or explicitly track the interface, which is done in marker-and-cell (MAC) methods. For interface-capturing techniques an auxiliary function is needed. Examples are volume-of-fluid (VoF) methods, where volume fraction function is used and level-set (LS) methods, where a signed-distance function is used.

Several methods and variations to existing methods have been introduced for solving the two-phase flow. Approaches where the reconstruction techniques rely on the Cartesian control volumes are not able to handle geometrically complex domains. On the other hand, approaches that are applicable for triangular control volumes suffer from mass loss and/or fragility [17] due to their complexity.

Generally, the following four approaches are used for modelling the two-phase flow: front-tracking methods, marker-and-cell methods, level-set method and volume-of-fluid method. Each of these methods have there advantages and disadvantages based on mass conservation, ease of implementation and ability for handling complex geometries. Only the VoF and LS method can be used when handling interface movement without any geometrical restrictions [17]. All other proposed methods are derived from one or a combination of the mentioned methods. These four approaches are discussed in Sections 2.2-2.5.

2.2 Front-tracking method

Front-tracking methods are based on the movement of the surface $\Gamma(t)$. Suppose that the velocity $\mathbf{u}(x(t), t)$ is known for every point $x(t) \in \Gamma(t)$. Then we get

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\mathbf{x}(t) = \mathbf{u}(\mathbf{x}(t), t). \tag{1}$$

This is known as the the Lagrangian formulation of the interface evolution equation [9]. Front-tracking methods can be categorized into Lagrangian and Eulerian methods. For Lagrangian methods, the mesh is constructed such that one of the boundaries corresponds to the interface. This means that the interface needs to be rebuild whenever it moves or deforms. Eulerian front-tracking methods utilize an additional data structure for describing the interface. In two dimensions line segments are used, where the vertices are called markers. So the movement of the interface is determined by the movement of these markers.

In the next part, a brief overview of an Eulerian front-tracking method is given. Suppose that the initial locations of the markers are given. The new positions are computed according to the Lagrangian formulation, Eq. (1), from which the interface can be obtained. The new set of connected markers is called the interface grid. So, the front-tracking method uses a stationary grid on the whole domain and a moving interface grid for the interface. Since the velocity is given for the stationary grid, it has to be interpolated for moving the interface.

During the computation, the interface grid changes, which can result in loss of connectivity. Furthermore, it can occur that line segments degenerate or that markers accumulate in one region.

In order to avoid these problems, it is necessary to reconstruct the interface grid. In this process line segments and markers are added or removed if necessary. In [22], ideas of adding, removing, redistributing and reconstructing can be found. However, these approaches will cause large computational costs.

Furthermore, numerical solutions can be unstable and may need to be stabilized by smoothing the corners of the interface. Another disadvantage is its inability to handle merging and splitting of two interface parts automatically and it is complicated to deal with these topological changes. The advantage of the front-tracking method is that the interface is sharp and explicitly given.

2.3 Marker-and-cell method

Marker-and-cell (MAC) methods or marker particle methods utilize marker particles. In contrast to front-tracking methods, this method marks the volume occupied by a phase rather than the interface itself [9]. The new particle location is computed by moving with the fluid velocity according to the Lagrangian formulation, Eq. (1). A cell containing particles while having at least one neighbour cell without any particles, is called an interface cell. The distribution of particles in interface cells is used to construct the interface.

By using particle markers, there is no need for ordering and well-defined neighbour markers [9]. This is an advantage for the MAC method, since it is able to handle merging and splitting of interface parts easily.

A drawback of this method, is that a large amount of the particles is needed to give an accurate approximation of the interface. When only using few particles, reconstruction of the interface is sensitive to small errors [9]. Numerical errors are being made when transporting the particles, resulting in a blurred interface. In order to obtain a sharp interface, far more particles than cells are needed. Furthermore, a balanced distribution of particles is required in the interface cells. Therefore, adding and removing of particles is necessary during this process. This will lead to a large computational cost.

2.4 Level-set method

In the level-set method, the interface is captured by a signed-distance function ϕ defined by

$$\phi(\mathbf{x},t) = \begin{cases} \min\limits_{y\in\Gamma(t)} \|\mathbf{x}-\mathbf{y}\| & \text{if } \mathbf{x}\in\Omega_1(t), \\ 0 & \text{if } \mathbf{x}\in\Gamma(t), \\ -\min\limits_{y\in\Gamma(t)} \|\mathbf{x}-\mathbf{y}\| & \text{if } \mathbf{x}\in\Omega_2(t), \end{cases} \tag{2}$$

where $\Gamma(t)$ is the interface between the two phases $\Omega_1(t)$ and $\Omega_2(t)$ [2]. This function is advected according to the advection equation

$$\frac{\partial\phi}{\partial t} + \mathbf{u}\cdot\nabla\phi = 0, \tag{3}$$

with $\mathbf{u}$ the velocity of the flow. The derivation is given in Section A.1.

The signed-distance function satisfies $\|\nabla\phi\| = 1$, where $\nabla\phi$ is defined, which is known as the Eikonal equation. This property can be used to redistance the function, which will be discussed in Section 2.4.1.

2.4.1 Reinitialization of level-set function

In general, the level-set function ϕ does not remain a signed-distance function when ϕ is advected, which means that too steep and flat gradients can occur. This results in errors when derivatives are taken for computing the interface normal vector and instability problems [12]. Hence it is desirable to maintain ϕ as a signed-distance function. Therefore reinitialization is needed to modify the level-set function [24]. There are several techniques for reinitializing ϕ as a signed-distance function [12], such as the fast marching method and the fast sweeping method. Both methods solve the Eikonal equation $\|\nabla\phi\| = 1$ and retain the location of the interface. Another method solves a first-order partial-differential equation in pseudo-time, given by

$$\frac{\partial \phi}{\partial \tau} + S(\phi_0)(\|\nabla\phi\|) - 1) = 0, \tag{4}$$

where ϕ_0 is the level-set function from the previous time step, $S(\phi_0)$ is the smoothed sign function $S(\phi_0) = \phi_0/\sqrt{\phi_0^2 + \alpha^2}$ and α is a function of the grid size. However, numerical diffusion when solving the equation can result in a shifted interface.

Reinitialization of the level-set function is not always necessary, and is only required for stability reasons which causes ϕ to diverge from the signed-distance function, or when a constant interface thickness is needed for smoothing discontinuities. If this is not the case, one should avoid reinitialization since it is computationally expensive and can be difficult to implement [12].

2.4.2 Properties of level-set method

The advantage of the level-set method is its ability to handle merging and breaking of the interface automatically, without having to explicitly reconstruct the interface. Furthermore, geometrical information can be easily calculated, since ϕ is smooth near the interface

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}, \quad \kappa = \nabla \cdot \mathbf{n},$$

where $\mathbf{n}$ is the unit outward normal and κ the curvature.

The disadvantage is poor volume conservation, which can come from two aspects [28]. Firstly, numerical dissipation from discretization of the level-set equation, since a smooth distance function is advected rather than a conserved physical quantity, such as mass or volume. Therefore the level-set method has no mass conservation by design. And secondly, mass gain/loss from the reinitialization process, where it can occur that the interface shifts, because the signed-distance function does not provide an accurate approximation of the exact location [28].

2.5 Volume-of-fluid method

The volume-of-fluid (VoF) method uses a marker function, denoted by ψ, indicating the fractional volume of a certain fluid [23]. The definition is given by

$$\psi_k(t) = \frac{1}{|\Omega_k|}\int_{\Omega_k} c(\mathbf{x}, t)\,\mathrm{d}\Omega, \quad |\Omega_k| = \int_{\Omega_k} \mathrm{d}\Omega. \tag{5}$$

where the control volume is denoted by Ω_k, and the colour function $c(\mathbf{x}, t) : \mathbb{R} \to \{0, 1\}$ is defined by

$$c(\mathbf{x}, t) = \begin{cases} 0 & \text{if phase 0 is present,} \\ 1 & \text{if phase 1 is present.} \end{cases}$$

This volume fraction function gives the ratio of phase 1 to the total ratio, which means ψ is 0 and 1 when the cell only consists of phase 0 and 1, respectively, and lies between 0 and 1 when both phases are present. In each time step, the volume fraction is advected according to

$$\frac{\partial \psi}{\partial t} + \mathbf{u} \cdot \nabla\psi = 0. \tag{6}$$

2.5.1 Interface reconstruction

Since the volume-of-function ψ is discontinuous across the cells, the exact interface location is unknown and has to be approximated in each cell, where the volume fraction function takes values between 0 and 1 [17]. This procedure is called 'interface reconstruction'. Common methods used for interface reconstruction are the Simple Line Interface Calculation (SLIC) and the Piecewise Linear Interface Calculation (PLIC). This is computationally expensive, which makes it a disadvantage for VoF [28]. Furthermore, the reconstructed interface is discontinuous between cells [10].

2.5.2 Properties of volume-of-fluid method

The VoF method conserves volume really well except for small over and under shoots. Besides the computationally intensive interface reconstruction, other drawbacks of this method are ejection of non-physical fluid, parasitic currents and the inability to compute geometric information, such as unit normal vector and curvature, accurately [10]. This is caused by the step-like behaviour of ψ [23]. Numerical diffusion arises by application of the numerical scheme for the advection equation. This results in numerical errors which are replaced by inaccurate mass motion due to mass conservation.

2.6 Improvements to the level-set method

Several techniques can be used to solve the level-set equation, Eq. (3), in space and time. However, numerical methods may become unstable due to steep or flat gradients of the level set function. One solution to this problem is reinitialization. But, according to [12], “the major drawbacks of the reinitialization process is the difficulty in not moving the original location of the interface, often leading to breakdown in the conservation of mass.” Furthermore, the level-set method is not inherently mass-conserving. So even when the reinitialization procedure has no mass loss/gain, mass conservation is still not guaranteed.

Various approaches have been proposed to make the level-set method stable and mass conserving. These techniques can be divided into four methodologies [4]:

1. Improving LS discretization and reinitialization; by using a higher order scheme, minimizing displacement during reinitialization or refining the mesh near the interface, numerical errors are reduced. However, the level-set method remains inherently non-conservative [4]. In [16], two alternative FEM implementations are discussed.

2. Coupling the LS method with conservative methods, such as VoF or Lagrangian particles; by combining two methods mass can be conserved exactly. However, these hybrid method can be very complex to implement.

3. Adding a volume constraint in the level-set (LS) or Navier-Stokes (NS) formulation or introducing a volume/mass correction procedure.

4. Modifying the LS definition, such as the (improved) conservative level-set method [28] or the assumed-gradient level-set method [13].

In Section 2.7 the available hybrid methods are discussed. Volume-correction procedures are reviewed in Section 2.8, followed by the adapted LS formulations in Section 2.9.

2.7 Hybrid methods

A popular approach to get a mass-conserving method is to combine the level-set method with the volume-of-fluid method. This combination can be roughly divided into three subcategories.

1. The coupled level-set and volume-of-fluid (CLSVOF) method takes full advantage of both methods. The VoF method is utilized for interface reconstruction and the LS method for normal vector and curvature computation.

2. The coupled volume-of-fluid and level-set (VOSET) method uses the VoF approach to capture the interface. The level-set function is iteratively constructed for accurately computation of the geometric properties.

3. The mass-conserving level-set (MCLS) method adopts the LS method to capture the interface. The VoF funtion is constructed for retaining mass conservation.

All methods with a coupling between the LS and VoF method share the same idea as one of these three methods. In Sections 2.7.1-2.7.5 several approaches with a coupling between LS and VoF are discussed.

2.7.1 Adaptive coupled level-set volume-of-fluid (ACLSVOF) method

In [24], the level-set method and the volume-of-fluid method are combined to obtain advantages from both methods. The adaptive coupled level-set volume-of-fluid (ACLSVOF) method is based on the CLSVOF method, introduced in [20]. The level-set function and the volume-of-fluid method are advected using a discontinuous Galerkin and Lagrangian-Eulerian method, respectively. The ACLSVOF method employs the level-set function for calculating the interface normal vector, and the volume-of-fluid function for the line constant in order to maintain mass conservation. Since ϕ is continuous, it is easy and accurate to compute the unit normal, and by the volume-conserving property of ψ, volume can be conserved accurately. In each time step the interface needs to be reconstructed for which [24] developed an analytic piecewise linear interface reconstruction. This method is developed for unstructured triangular grids, where the grid can be adapted in order to resolve complex changes in interface topology.

The procedure of this method is as follows [24]. Functions ϕ and ψ are initialized according to the interface position, and the velocity is initialized by solving the governing flow equations. Then the first step of the two-stage RKDG Eulerian level-set advection is performed, resulting in the intermediate function $\phi^{n+1/2}$. The intermediate Lagrangian grid is obtained by performing the first step of the two-stage Runge-Kutta projection. Then the intermediate Eulerian velocity is computed by solving the governing flow equations at time step $t^{n+1/2}$. For the next step, the intermediate Lagrangian velocity is required, which is computed from the intermediate Eulerian velocity through interpolation. The second steps of the RKDG level-set advection and Runge-Kutta grid projection are executed to obtain ϕ^{n+1} and the Lagrangian grid, respectively. Then the interface is reconstructed on the Lagrangian mesh utilizing both ϕ^{n+1} and ψ^{n+1}, where interpolation is used to get ϕ^{n+1} on the Lagrangian mesh. The new values for the volume-of-fluid function on the Eulerian mesh are obtained by remapping, and the level-set function is obtained as the signed-distance function. This procedure is repeated until the end time is reached.

This method is based on the interaction between the VoF and LS functions for interface reconstruction and the interplay between the Eulerian and Lagrangian mesh for advection, making it a complex method to implement [26]. Furthermore, this method suffers from computationally expensive interface reconstruction. According to the authors of [26], "development of a more efficient and accurate algorithm for CLSVOF method is still needed".

2.7.2 Coupled volume-of-fluid and level-set method (VOSET)

In [8], a coupled volume-of-fluid and level-set method (VOSET) is introduced which combines the volume-of-fluid and level-set method on arbitrary polygon meshes. The LS function is geometrically constructed from the VoF function in order to accurately compute the geometric properties and physically sharp interface without discontinuous oscillations [26]. Since ϕ is constructed from ψ, the level-set field does not need to be advected. The proposed VOSET method can track the interface accurately and can be applied for complex geometries.

The algorithm is summarized as follows. The level-set function is constructed from the volume fractions. Then the smoothed Heaviside function is calculated from the level-set function. Using this function, the density and viscosity fields, and the interface curvature and surface tension are computed. Then the velocity field can be solved. Finally, the new volume fractions are computed with the incremental remapping approach, a method similar to the Lagrangian-Eulerian method discussed in Section 2.7.1.

2.7.3 Mass-conserving level-set (MCLS) method

Unlike the ACLSVOF method, the mass-conserving level-set (MCLS) method [17, 23] is not a combination of two or multiple methods, but uses all available information from the level-set function ϕ, rather than coupling with the VoF method. The VoF function, however, is utilized, but without applying the computationally expensive interface reconstruction step. This results in a mass-conserving method, which is easy to implement and gives a explicit interface position.

In [23], an explicit relation between the level-set function ϕ and the VoF function ψ is introduced, which is the following

$$\psi = f(\phi, \nabla\phi), \tag{7}$$

where the assumption of piecewise linear interfaces within a computational cell is made. This relation makes advection of ψ easy and ϕ can be easily obtained from ψ. For a computational cell Ω_k, the Heaviside function is used for the color function, which becomes $c = H(\psi)$. This results in the following connection

$$\psi_k = \frac{1}{|\Omega_k|}\int_{\Omega_k} H(\psi)\,\mathrm{d}\Omega. \tag{8}$$

Because of this relation, the level-set field has mass conserving properties of the volume-of-fluid field and the interface is exactly defined by the level-set field.

As already discussed, a drawback of the level-set method is that conservation of ϕ does not imply conservation of mass, in contrast to the VoF method, where mass is conserved when ψ is conserved [23]. Therefore, MCLS method uses the fractional volume ψ to make corrections to the level-set function ϕ. First ϕ is advected and reinitialized, where the result is denoted by $\phi^{n+1,*}$. Corrections to $\phi^{n+1,*}$ for mass conservation are divided in the following three steps:

1. computation of the VoF function from the level-set function using the Heaviside step function and linearisation: $\psi = f(\phi, \nabla\phi)$;
2. conservatively advection of the VoF function with a finite volume approach, which yields ψ^{n+1};
3. corrections to $\phi^{n+1,*}$ using the inverse function of f and Picard-iterations, such that $f(\phi^{n+1}, \nabla\phi^{n+1}) = \psi^{n+1}$ holds.

The original formulation of the MCLS method in [23] is for Cartesian quadrilateral control volumes, but in [17] an extension has been made for unstructured triangular control volumes for geometrical flexibility. Here, the level-set field is advected with a discontinuous Galerkin (DG) finite element method for space discretization, which has high accuracy near boundaries and the ability to handle arbitrary geometrical complexity [17]. Runge-Kutta is used for time discretization. The advection of the VoF field is done simultaneously. This is difficult, since utilizing 'standard' schemes can result in strong oscillations in the solution of unacceptable smearing of the interface [17], which possibly leads to mass loss/gain. The approach Lagrangian-Eulerian VoF evolution is used for advection, discussed in Section 2.7.1.However, this is performed without the computationally intensive interface reconstruction procedure. Instead, the interface position from the LS field is used. The advected function is then compared to the constructed VoF function. If the error is large, the level-set function is adapted according to the advected VoF function.

In order to conserve mass exactly, mass correction needs to be applied for fully filled and partially filled cells. The area of a cell can change on the Lagrangian grid with the cell on the Eulerian grid. This change in mass needs to be adjusted by adding/removing mass in the partially filled cells near the interface. The interface (at the partially filled cells) is adapted, such that the mass corresponds to the mass in Eulerian grid plus the change in mass from the fully filled cells. After the advection, the VoF function is computed from the LS function; $\psi^* = f(\phi_k^*, \nabla\phi_k^*)$. When the difference between ϕ_k^{n+1} and ψ^* is small, no correction is made, but when the difference is significant, the LS function is computed by the inverse of f.

We will shortly discuss the volume correction procedure introduced in [23]. In each cell a modification to ϕ^{n+1} is sought, such that

$$|f(\phi_k^{n+1}, \nabla\phi_k^{n+1}) - \psi_k^{n+1}| \leq \epsilon, \quad \forall k = 1, 2, \ldots,$$

which implies that mass is conserved in that cell. For this procedure the inverse function g of f is used to make the coupling between the mass-conserved ψ^{n+1} and the to be adapted level-set function

$$f(g(\psi, \nabla\phi), \nabla\phi) = \psi.$$

The updated level-set function is obtained by employing Picard-iterations for

$$\phi_k^{n+1,m+1} = g(\psi_k^{n+1}, \nabla\phi_k^{n+1,m}), \quad \forall k = 1,2,\ldots, \tag{9}$$

where m denotes the m-th iteration of the Picard iteration.

2.7.4 Curvature-based mass-conserving level-set method

In [11], a mass-conserving level-set method with a mass remedy procedure based on the local curvature is introduced. This method utilizes volume fractions for mass conservation by constructing the VoF function. Mass is conserved by adapting the level-set function based on the curvature of each cell at the interface. These corrections are found by considering the volume fractions in combination with the change in volume. The proposed method is able to accurately capture the complex interface by reducing the mass loss without a lot of extra computational cost.

The outline of this approach is as follows. First, the advection step for the LS function is performed, in which mass loss/gain occurs. Then the volume fractions are computed in each cell using the analytical formula from [23]. The total volume loss is computed

$$d\psi_{\text{tot}} = \psi_{\text{tot,init}} - \psi_{\text{tot}}^{n+1,*}, \tag{10}$$

where $\psi_{\text{tot,init}}$ and $\psi_{\text{tot}}^{n+1,*}$ are the initial volume at time step 0 and the current volume at time step $n+1$. The change in volume is then distributed to the cells at the boundary, for which $0 < \psi_{\text{tot}}^{n+1,*} < 1$ holds. This can be done in a trivial manner, however, any local property of the level-set function will then be ignored. So, this is taken into account by considering the curvature. The volume is distributed according to the following formula

$$d\psi_p = d\psi_{\text{tot}} \cdot \frac{\frac{\kappa_p}{\kappa_{\max}}}{\sum_{p=1}^{m} \frac{\kappa_p}{\kappa_{\max}}}. \tag{11}$$

This approach is used rather than simply multiplying with the local curvature κ_p, since mass loss tends to occur in regions with large curvatures. After the volume fraction remedy, mass is conserved for ψ_{n+1}. Now, the level-set function $\phi_{n+1,*}$ needs to be adapted, such that it corresponds to ψ_{n+1}. This is obtained by stating

$$|f(\phi_k^{n+1}, \nabla\phi_k^{n+1}) - \psi_k^{n+1}| \leq \epsilon, \quad \forall k = 1,2,\ldots,$$

with initial guess $\phi_{n+1,*}$. A solution is found by using a Newton iteration.

Note that this method does not advect the VoF function, but only uses the volume fractions for the mass remedy procedure in order to obtain mass conservation properties.

2.7.5 Volume-preserving continuous Galerkin level-set approach

In [2] a volume-preserving method is proposed, where a Galerkin level-set formulation based on linear triangles is coupled with the volume-of-fluid method on star-shaped polygonal finite-volume meshes. Volume preservation during advection is ensured by employing a volume-correction algorithm. This method uses a continuous Galerkin approach to ensure the continuity of the level-set function.

The advantage of this method is that the volume is preserved due to the coupling between the LS and VoF function and the volume correction step. Furthermore, the interface is continuous, since the interface is defined by the LS function, which is continuous since ϕ is defined as the signed-distance and advected using a continuous Galerkin approach. Finally, no costly interface construction has to be performed, which is a drawback of the VoF method.

This method defines a level-set function $\phi(\mathbf{x}, t)$ on a triangular primal mesh and a volume-of-fluid method $\psi(\mathbf{x}, t)$ on a star-shaped polytopal dual mesh. Barycentres of adjacent triangular cells are used to construct the dual mesh. The elements of the primal mesh $\mathcal{T}$ are denoted by $K \in \mathcal{T}$, and the control volumes of the dual mesh by $\mathcal{K}_{\mathbf{v}}$, with $\mathbf{v}$ a vertex in $\mathcal{T}$.

The volume-of-fluid method is defined as follows

$$\psi_\phi(\mathbf{v}, t) = \frac{1}{|\mathcal{K}_{\mathbf{v}}|} \int_{\mathcal{K}_{\mathbf{v}}} H(\phi(\mathbf{x}, t))\, \mathrm{d}\mathbf{x}. \tag{12}$$

This approach advects both the level-set function and the volume-of-fluid function. After advection the discrete volume-of-fluid functions $\psi_h(\mathbf{x}, t)$ and $\psi_{\phi,h}(\mathbf{x}, t)$ are compared and a volume correction algorithm is applied to adapt the level-set function $\phi(\mathbf{x}, t)$.

The volume correction procedure is executed as follows. The volume fraction function is constructed from the level-set function. Corrections to ϕ are sought such that ϕ corresponds to ψ. These corrections are made locally. However, since ϕ is continuous, other locally corrected values for ϕ effect each other, which causes the corresponding ψ values to differ from the desired value. So in order to get the obtained result, this procedure must be performed iteratively. The corrections are found simply by adding a value to the current value ϕ. The iterative process stops when the corrections 'converge' to zero.

2.7.6 Particle level-set (PLS) method

The particle level-set (PLS) method combines the accuracy of Lagrangian front-tracking with the simplicity and efficiency of the level-set method [10]. Lagrangian marker particles are used to rebuild the level-set function in regions, where mass loss/gain has occurred. The advantage of this method is that the reconstructed interface still has the properties from the level set method, such as continuity, smoothness and robustness.

The procedure is as follows. Positive particles are placed at the $\phi > 0$ and negative particles at the $\phi < 0$ side. Besides the level-set function, thsse particles are also advected with the flow and used to adapt the level-set function in order to restore mass conservation, based on the principles that particles should not cross the interface.

2.8 Volume/mass correction techniques

The methods discussed in this section employ a mass/volume correction procedure, rather than coupling the level-set method with another method, in order to obtain mass conservation.

2.8.1 Global mass correction (GMC) procedure

As discussed, the level-set function ϕ does not remain a signed-distance function in general. Therefore, reinitialization is needed. However, properly redistancing function ϕ does not solve the loss/-gain in mass. In [25], a global mass correction (GMC) procedure is proposed and adapted in [27]. In this approach a steady-state solution is obtained by solving

$$\frac{\partial \tilde{\phi}}{\partial \tau} = M_{\text{cor}}, \tag{13}$$

where M_{cor} is the mass correction factor and τ a pseudo-time. The reinitialized level-set function is denoted by $\tilde{\phi}$. The definition of the mass correction factor is given by

$$M_{\text{cor}} = \text{sgn}(\phi_{\text{ref}}) \frac{M_d - M_c}{M_d}, \tag{14}$$

with M_d the desired mass and M_c the current mass. For M_d the original mass is taken. A reference phase is chosen, from which the original and current mass is computed according to

$$M = \begin{cases} \sum \rho_{\text{ref}} H(\phi_{\text{ref}}) \Delta V & H_{\text{ref}} = 1, \\ \sum \rho_{\text{ref}} (1 - H(\phi_{\text{ref}})) \Delta V & H_{\text{ref}} = 0, \end{cases} \tag{15}$$

where the summation is performed on the entire domain. The Heaviside function is denoted by H, and the area of a cell is denoted by ΔV.

When the equation reaches the steady-state solution, the mass correction factor is equal to zero and mass is conserved. The level-set function is modified on the entire domain.

2.8.2 Improved interface-preserving level-set method

The improved interface-preserving level-set method, introduced in [26], utilizes a mass correction term in order to retain the mass conservation property. This term is added to the reinitialization equation, such that the level-set function remains a signed-distance function and conserves mass.

The smoothed Heaviside error, defined by

$$\mathcal{H}_{\text{error}} := \int \mathcal{H}(\phi, t=0)\,\mathrm{d}\Omega - \int \mathcal{H}(\phi, t)\,\mathrm{d}\Omega, \tag{16}$$

is introduced, with $\mathcal{H}(\phi, t)$ the smoothed Heaviside function. A new Heaviside function is introduced

$$\mathcal{H}(\phi_{\text{new}}, t) = \begin{cases} \mathcal{H}(\phi, t) + \frac{|\mathcal{H}_{\text{error}}|\operatorname{sgn}(\mathcal{H}_{\text{error}})}{N} & 0 < \mathcal{H}(\phi, t) < 1, \\ \mathcal{H}(\phi, t) & \mathcal{H}(\phi, t) = 0, \quad \mathcal{H}(\phi, t) = 1, \end{cases} \tag{17}$$

where N denotes the number of nodal points in the smooth layer. From the new Heaviside function, the new level-set function is computed. This procedure makes sure mass is conserved.

The level-set function is reinitialized by solving the following equation in pseudo-time:

$$\frac{\partial \phi}{\partial \tau} = \operatorname{sgn}(\phi_{\text{new}})(1 - |\nabla\phi|) + \lambda\delta(\phi)|\nabla\phi|. \tag{18}$$

This re-distances the level-set function, while retaining mass conservation. Here, δ is the smoothed delta function and λ a parameter, which can be determined.

2.8.3 Interface-correction level-set (ICLS) method

The interface-correction level-set (ICLS) method makes small corrections near the interface by solving a PDE similar to the advection equation in order to conserve mass. Let the volume of Ω_1 be denoted by V. Without loss of generality we assume $\phi < 0$ in Ω_1 and $\phi > 0$ in Ω_2. The rate of change of V can be written as the following integral

$$\frac{\delta V}{\delta t} = \int_\Gamma \mathbf{n} \cdot \mathbf{u}_c \,\mathrm{d}\Gamma, \tag{19}$$

where $\mathbf{n}$ is the outward-pointing normal on the interface, and $\mathbf{u}_c$ the normal velocity. According to [4], "if $-\frac{\delta V}{\delta t}$ corresponds to the mass loss over an arbitrary period of time, then $\mathbf{u}_c$ can be thought as a surface velocity that corrects the volume by an amount $\frac{\delta V}{\delta t}$, hence compensating the mass loss". So when the normal velocity $\mathbf{u}_c$ is known, the mass loss can be restored over δt be solving

$$\frac{\partial \phi}{\partial t} + \mathbf{u}_c \cdot \nabla\phi = 0. \tag{20}$$

The derivation of $\mathbf{u}_c$ can be found in [4]. The surface velocity is spread by a smoothed Heaviside function, which allows an easy handling of the interface location and enhances the numerical stability. Note that this method smooths the correction-velocity rather than the interface, therefore the interface remains sharp.

2.9 Adapted level-set methods

The methods reviewed in this section use a modified formulation of the level-set method.

2.9.1 Improved conservative level-set (ICLS) method

Several approaches have been proposed to improve the level-set method for mass conservation properties, but generally these techniques are complex and ruin the simplicity of the original method. However, the conservative level-set (CLS) method proposed by [14] improves the mass conservation while retaining the simplicity [28].

The CLS method replaces the signed-distance function by employing the Heaviside function

$$H(\phi) = \begin{cases} 1 & \phi > 0 \\ 1/2 & \phi = 0 \\ 0 & \phi < 0 \end{cases} \tag{21}$$

with ϕ the signed-distance function. However, using this definition will result in numerical instabilities due to strong discontinuities at the interface [28]. Therefore, the signed-distance is often replaced with a hyperbolic tangent profile, where the hyperbolic tangent function is is given by

$$H(\phi) = \frac{1}{2}\left(1 + \tanh\left(\frac{\phi}{2\epsilon}\right)\right), \tag{22}$$

where ϵ is used as the spreading width of H.

When H is advected, the interface will be deformed which is caused by numerical errors and artificial diffusion [28]. Therefore after advection a reinitialization step is needed in order to maintain the shape and width of the interface, which is given by

$$\frac{\partial H}{\partial \tau} + \nabla \cdot (H(1-H)\mathbf{n}) = \nabla \cdot (\epsilon(\nabla \cdot \mathbf{n})\mathbf{n}), \tag{23}$$

where $\mathbf{n}$ is the normal vector at the beginning of reinitialization. The normal vector in Eq. (23) can be calculated by

$$\mathbf{n} = \frac{\nabla H}{\|\nabla H\|} \tag{24}$$

as has been done in [14]. However, this approach is sensitive to spurious oscillations, since the gradient of H is very small when H is close to 0 or 1, resulting in completely false normal vectors.

While the CLS method conserves mass well and retains the simplicity of the original level-set method, the results are usually worsened by inaccurate interface normal vectors. Therefore, an improved strategy is introduced by [28]. First, both functions ϕ and H are advected. Then reinitialization must be applied. However, as discussed in [12], the interface can shift during the reinitialization step. To avoid this problem, points close to the interface are not updated until the end of the time step. Points are defined to be close to the interface when their value $|\phi|$ is smaller than a chosen value δ. Their normal vectors are obtained from the Heaviside function, and the normal vectors from the other points are obtained from the reinitialized function ϕ. Then the reinitialization step of H can be executed until a steady state of Eq. (23) is reached. Finally, the removed points can be updated by calculating the inverse of Eq. (22), which is possible, since for these value H is not close to 0 or 1.

2.9.2 Assumed-gradient level-set method

In [13], a simpler version of the level-set equation is formulated by assuming that the level-set function remains a signed-distance function. The advection velocity is replaced by extensional velocity u_e, such that $\|\nabla\phi\| = 1$ is maintained. This level-set equation then becomes

$$\frac{\partial \phi}{\partial t} + u_e = 0, \tag{25}$$

which reduces to an ordinary differential equation when u_e is independent of $\nabla\phi$. It has been shown that using a standard Galerkin formulation results in sufficient accuracy.

While this method is easy to implement, building the extensional velocity field and reinitializing the level-set field are computationally expensive.

3 Discretization

3.1 Spatial discretization methods for the level-set method

We wish to discretize the level-set method with a finite element approach for its geometrical flexibility. However, we will not use the standard continuous Galerkin approach, since it is a well-known fact that naive finite-element implementations are unsatisfactory [1, 5, 16]. Standard finite-element discretizations suffer from instability issues, and in order to stabilize the numerical solution a certain amount of numerical viscosity has to be added [16]. While adding artificial diffusion solves this problem, it must be taken into account that this also effects the accuracy of the solution, which may result in mass loss.

3.1.1 Continuous Galerkin (CG) method

Let us define the following function spaces

$$\begin{aligned} W &:= \{w \in L_2(\Omega) : \mathbf{u} \cdot \nabla w \in L_2(\Omega)\}, \\ W_0 &:= \{w \in W : w|_{\partial\Omega_{in}} = 0\}, \\ W_g &:= \{w \in W : w|_{\partial\Omega_{in}} = g_D\}, \end{aligned} \tag{26}$$

where $\partial\Omega_{in}$ denotes the inlet boundary, that is, the part of the boundary where the flow enters the domain.

Let $\mathcal{T}_h$ be a triangulation of the domain Ω, and let $\mathcal{P}_k$ be the space of polynomials of degree $\leq k$. The approximation of ϕ is denoted by ϕ_h. The space of continuous piecewise polynomial functions [9] is then defined by

$$\begin{aligned} V_h^k &:= \{v_h \in \mathcal{C}(\bar{\Omega}) : v_h|_K \in \mathcal{P}_k \quad \forall K \in \mathcal{T}_h\}, \\ V_{h,g}^k &:= \{v_h \in V_h^k : v_h|_{\partial\Omega_{\mathrm{in}}} = g_D\}. \end{aligned} \tag{27}$$

for $k \geq 1$. Note that V_h^k is a subspace of W.

The weak form of the standard Galerkin method is as follows. For all $t \in [0, T]$ find $\phi_h(t) \in V_h^k$ such that

$$\begin{cases} \displaystyle\int_\Omega \frac{\partial \phi_h}{\partial t} v_h \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi_h) \, v_h \, \mathrm{d}\Omega = 0, \quad \forall v_h \in V_h^k, \\ \phi_h(0) = \phi_{0,h}. \end{cases} \tag{28}$$

However, this formulation will result in instabilities near the inlet boundary. Therefore we need to impose Dirichlet boundary conditions on the inlet boundary. The weak form of the standard Galerkin method for Dirichlet boundary conditions is given by: for all $t \in [0, T]$ find $\phi_h(t) \in V_{h,g}^k$ such that

$$\begin{cases} \displaystyle\int_\Omega \frac{\partial \phi_h}{\partial t} v_h \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi_h) \, v_h \, \mathrm{d}\Omega = 0, \quad \forall v_h \in V_{h,0}^k, \\ \phi_h(0) = \phi_{0,h}. \end{cases} \tag{29}$$

Results from Figure 7 show that the standard Galerkin approach leads to instabilities, which confirms what has been found in literature.

3.1.2 Streamline upwind Petrov-Galerkin (SUPG) method

The streamline upwind Petrov-Galerkin (SUPG) stabilizes the standard Galerkin method by adding diffusion, which smooths the solution. Instead of simply adding a diffusion term to the level-set equation, diffusion is added in the streamline direction. In this way, less diffusion is produced. The diffusion is incorporated into the equation in such a way, that the exact solution is also a solution to the stabilized problem [9].

Note that $\mathbf{u} \cdot \nabla v \in L_2(\Omega)$ for any function $v \in W$, which means that

$$\int_\Omega \frac{\partial \phi}{\partial t} (\mathbf{u} \cdot \nabla v) \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi) (\mathbf{u} \cdot \nabla v) \, \mathrm{d}\Omega = 0 \tag{30}$$

holds for the exact solution in the weak sense and for any $v \in W$. This is added δ times to standard variational formulation, resulting in

$$\int_\Omega \frac{\partial \phi}{\partial t} (v + \delta \mathbf{u} \cdot \nabla v) \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi) (v + \delta \mathbf{u} \cdot \nabla v) \, \mathrm{d}\Omega = 0, \tag{31}$$

with δ a positive constant.

When we added the diffusion term, we changed the test space. This means that the trial and test are not the same any more, therefore this method is a Petrov-Galerkin method [9]. The SUPG test function is defined by

$$w_h(v_h) := v_h + \delta_K \mathbf{u} \cdot \nabla v_h, \tag{32}$$

where $v_h \in V_h^k$. The parameter δ_K is non-negative and defined element-wise on every $K \in \mathcal{T}_h$. The SUPG test space is defined by

$$V_h^{\text{SUPG}} := \{w_h(v_h) : v_h \in V_h^k\}. \tag{33}$$

Substituting this into the standard variational equations and summing over all elements $K \in \mathcal{T}_h$ gives the desired problem. For all $t \in [0, T]$ find $\phi_h(t) \in V_{h,g}^k$ such that

$$\begin{cases} \displaystyle\sum_{K \in \mathcal{T}_h} \int_{\Omega_K} \frac{\partial \phi_h}{\partial t} w_h \, \mathrm{d}\Omega + \int_{\Omega_K} (\mathbf{u} \cdot \nabla \phi_h) \, w_h \, \mathrm{d}\Omega = 0, \quad \forall w_h \in V_{h,0}^{\text{SUPG}}, \\ \phi_h(0) = \phi_{0,h}. \end{cases} \tag{34}$$

A difficulty of the SUPG method is to choose an appropriate value of δ_K. Usually, the parameter is chosen to be proportional to the grid size [9]. For this problem δ_K is taken as

$$\delta_K = s \frac{h_K}{\max\{\epsilon, \|\mathbf{u}\|_{\infty,K}\}}, \tag{35}$$

with $\epsilon > 0$, and $s = \mathcal{O}(1)$ a scaling factor.

Another stabilizing finite-element approach is the Galerkin Least Squares (GLS) finite-element method. However, the variational formulation is identical to the SUPG method for the hyperbolic case, or for piecewise linear elements in the general case [6]. Therefore we will not consider the GLS method.

3.1.3 Subgrid edge stabilization method

As shown in the results and discussed in the literature, the standard Galerkin method will be insufficient for pure convection equations and will lead to instabilities. Therefore a suitable stabilization is required. The subgrid edge stabilization method [1, 16] adds a term to the standard Galerkin formulation, which penalizes the jump of the gradient over internal element boundaries. In [16], a local version of the interior penalty stabilization is considered.

A quasi-uniform triangulation, denoted by $\mathcal{T}_H$, where H characterizes the mesh refinement, is considered. This method also uses a finer triangulation with half the mesh size $h = H/2$. Only the jump of the gradient over the subtriangles is penalized. The subtriangles for each triangle K_H are defined by

$$\mathcal{E}_i(K_H) := \{e_i \in \partial K_h | e_i \notin \partial K_h, \forall K_h \in K_H\}.$$

Then the stabilized continuous finite-element discretization is given by: for all $t \in [0, T]$ find $\phi_h(t) \in V_{h,g}^k$ such that

$$\begin{cases} \displaystyle\int_\Omega \frac{\partial \phi_h}{\partial t} v_h \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi_h) \, v_h \, \mathrm{d}\Omega + j(\phi_h, v_h) = 0, \quad \forall v \in V_{h,0}^k, \\ \phi_h(0) = \phi_{0,h}, \end{cases} \tag{36}$$

with the stabilization term $j(\phi_h, v_h)$ defined by

$$j(v_h, \phi_h) := \sum_{K_H} \sum_{e_i \in \mathcal{E}_i} \int_{e_i} h_{e_i}^2 [\nabla \phi_h][\nabla v_h] \, \mathrm{d}s, \tag{37}$$

where $[f] = f^+ - f^-$ denotes the standard jump operator.

In contrast to Petrov-Galerkin type stabilizations, such as SUPG, approaches with interior penalization have complete freedom in choosing time discretization. Furthermore, this method has good mass conservation properties, since the stabilization is only present on the finest scale. Therefore the method will have the same conservation properties as the standard Galerkin method [15].

3.1.4 Stabilized finite element method without reinitialization

In [21], a new stabilized finite-element method is proposed. A term depended on the local residual of the Eikonal equation is added to the SUPG formulation of the level-set equation. This improves the interface resolution without having to reinitialize the level-set function. Furthermore, it is claimed that with the penalty term simpler and more efficient numerical schemes can be used. And it has been shown that this method enhances the numerical behaviour of the SUPG method.

The following modified level-set equation is proposed

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi - \nabla \cdot \left(p \frac{\nabla \phi}{|\nabla \phi|} \right) = 0, \tag{38}$$

where p is the projection of the residual $(|\nabla \phi| - 1)$.

The weak form of the modified level-set equation using SUPG is given by: for all $t \in [0, T]$ find $\phi_h(t) \in V_{h,g}^k$ such that

$$\begin{cases} \displaystyle\int_\Omega \left[\left(\frac{\partial \phi_h}{\partial t} + \mathbf{u} \cdot \nabla \phi_h \right) (v_h + \tau \mathbf{u} \cdot \nabla v_h) + p \frac{\nabla \phi_h}{|\nabla \phi_h|} \cdot \nabla v_h \right] \mathrm{d}\Omega = 0, \\ \phi_h(0) = \phi_{0,h}, \end{cases} \tag{39}$$

where $\tau = \alpha \frac{h}{2|\mathbf{u}|}$ with $0 \leq \alpha \leq 1$.

3.1.5 Discontinuous Galerkin (DG) method

As discussed, we wish to use a continuous discretization approach, since we want to maintain the continuity of the level-set function. However, the discontinuous Galerkin (DG) method is often used, and therefore we will briefly review method.

The most popular methods for approximating the solution of partial differential equations are the finite-volume method (FVM) and the finite-element method (FEM). Finite-volume methods are able to accurately approximate discontinuous solutions since the finite-volume approximations are discontinuous on the interelement interface [3], but on the other hand, FVM is of low-order accuracy. Finite-element methods can yield accurate numerical solutions because of the high polynomial-degree approximations, but spurious oscillations can occur due to steep gradients or discontinuities. The DG method utilizes ideas and techniques from both methods, resulting in a numerical scheme with advantages from FVM and FEM. Interior and boundary penalty are introduced to mimic the continuity of the approximate solution in a weaker sense. The interior penalty is required in standard conforming finite-element methods and the boundary penalty is employed for the Dirichlet boundary condition [3].

The Runge-Kutta discontinuous Galerkin (RK-DG) method is a popular numerical technique thanks to the fact it is accurate, compact, robust and it can handle complex geometries. Furthermore, discontinuous Galerkin finite-element approximations requires much less artificial viscosity than stabilization techniques applied to continuous Galerkin approximations [16]. For more information on this subject, we refer to [3].

3.2 Temporal discretization methods for the level-set method

For the temporal discretization of the semi-discrete system we consider the Crank-Nicolson method. The semi-discrete system is given by

$$M\mathbf{c}'(t) = S(t)\mathbf{c}(t) + \mathbf{f}(t). \tag{40}$$

3.2.1 Crank-Nicolson method

Using the Crank-Nicolson method, this is discretized in time as follows

$$\begin{aligned} M\frac{\mathbf{c}^{n+1} - \mathbf{c}^n}{\Delta t} &= \frac{1}{2}S^n(\mathbf{c}^n + \mathbf{c}^{n+1}) + \mathbf{f}^n, \\ \implies \mathbf{c}^{n+1} &= (M - \tfrac{1}{2}\Delta t S^n)^{-1}\left[(M + \tfrac{1}{2}\Delta t S^n)\mathbf{c}^n + \Delta t\mathbf{f}^n\right]. \end{aligned} \tag{41}$$

3.3 Discretization methods for the volume-of-fluid method

Solving Eq. (6) is not easy, since standard numerical scheme can easily diffuse the interface due to the discontinuity of function ψ. A remedy to overcome this problem is to first reconstruct the interface before advecting ψ [24]. The most common interface construction procedures are simple line (piecewise constant) interface reconstruction (SLIC) and piecewise linear interface reconstruction (PLIC). An overview of reconstruction methods is given in [18]. Several advection methods have been developed which utilize the reconstructed interface. For structured rectangular meshes it would be an obvious choice to discretize the VoF method with a Finite Volume approach. However, such schemes for rectangular elements cannot be easily adapted for triangular elements [9, 19, 24]. The implementation for unstructured triangular grids is challenging due to the complexity of computing edge fluxes and corner fluxes. Instead, in [17, 19, 24], the volume fraction advection is performed using a Lagrangian-Eulerian method, which is applicable for both structured rectangular grids and unstructured triangular grids. In [8], the incremental remapping approach is employed. In this section, a brief overview of both methods is given.

3.3.1 Lagrangian-Eulerian advection method

The Lagrangian-Eulerian advection scheme consists of three stages: Lagrangian projection, reconstruction and remapping [24]. In the first stage the volume fraction is projected, which is equivalent to the solving Eq. (6) using a Lagrangian approach. The volume fraction is solely transported with the flow, so for an element K at time t^n we have

$$\tilde{\psi}_K^{n+1} = \psi_K^n, \quad \forall K \in \mathcal{T}_h, \tag{42}$$

where $\tilde{\psi}_K^{n+1}$ is the volume fraction after the Lagrangian projection.

If we assume that the velocity is piecewise linear over every finite element, then it is sufficient to only find the Lagrangian position for every vertex of the element. So for a vertex $\mathbf{x}$ the new position is found by solving

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}. \tag{43}$$

A second-order Runge-Kutta (RK) scheme is used to obtain the Lagrangian points

$$\begin{aligned} \tilde{\mathbf{x}}_i^{n+\frac{1}{2}} &= \mathbf{x}_i^n + \frac{\Delta t}{2}\mathbf{u}(\mathbf{x}_i^n, t^n), \quad \text{for } i = 1, \dots, n, \\ \tilde{\mathbf{x}}_i^{n+1} &= \mathbf{x}_i^n + \Delta t\mathbf{u}(\tilde{\mathbf{x}}_i^{n+\frac{1}{2}}, t^{n+\frac{1}{2}}), \quad \text{for } i = 1, \dots, n, \end{aligned} \tag{44}$$

with n the total grid nodes. The projected grid is called the Lagrangian grid. Note that in the second RK step the velocity $\mathbf{u}(\tilde{\mathbf{x}}_i^{n+\frac{1}{2}}, t^{n+\frac{1}{2}})$ on the Lagrangian grid is required, so this needs to be interpolated from the Eulerian grid. The projected element may deform but remains a triangle.

Theoretically, the Lagrangian projection conserves the volume in each element due to the incompressibility of the fluid. However, numerically, this may not be the case. According to [24] the volume of fluid can not be exactly conserved due to: the numerical velocity not being exactly conserved, the velocity being approximated by piecewise linear basis functions, the velocity being interpolated from the Eulerian mesh and the numerical errors from Eq. (6). Results from [24] show that the volume loss due to the last three reasons is exceedingly small.

After the Lagrangian projection, the volume fraction in each Lagrangian cell is known. In order to map the volume fraction back to the Eulerian mesh, we need to know which part is occupied with fluid 1 and which part is occupied with fluid 2. Therefore the interface is needed. It is common to reconstruct the interface as a line segment of the form $\mathbf{n} \cdot \mathbf{x} = \alpha$, where $\mathbf{n}$ is the unit normal vector, $\mathbf{x}$ a point on the interface and α a line constant. The unit normal vector is calculated as the gradient of the volume fraction function. This function is discontinuous, so most method are less than second order accurate or very expensive. An available method for computing the normal vector is the differential least squares (DLS) method, discussed in [19]. The parameter α is determined by enforcing volume conservation, which is usually done iteratively. However, in [24], an analytic method is proposed for computing this line constant.

In the final stage, the new advected volume fraction ψ_K^{n+1} is obtained by mapping the projected elements back to the Eulerian grid, which is achieved by performing polygon-polygon clippings. Note that only cells near the interface need to be clipped, other cells simply remain 0 or 1.

3.3.2 Incremental remapping approach

Just like the Lagrangian-Eulerian advection method, the interface is reconstructed. After this step, the volume fractions are advected with the incremental remapping approach. The calculation procedure is as follows [8]. We consider the phase transportation in an element K. For the vertices of element K, for which the velocities are known, corresponding virtual points are determined, which are expected to reach the locations of the vertices in Δt. These points are calculated according to

$$\mathbf{x}_i' = \mathbf{x}_i - \mathbf{u}_i^{\text{node}} \Delta t, \tag{45}$$

where $\mathbf{x}_i$ is the location of the ith point, and $\mathbf{u}_i^{\text{node}}$ the given node velocity. The polygon corresponding to the virtual points is denoted as the *departure element*. The volume fraction at the next time step for element K can be estimated as the current volume fraction in the departure element

$$\psi_K^{n+1} = \frac{F_{\text{DE}}}{S_{\text{DE}}}, \tag{46}$$

with F_{DE} and S_{DE} the phase volume and entire volume of the departure element. For calculating F_{DE}, element K, its neighbours and the constructed interface must be taken into account. For element K and its neighbour elements the following possible situations are considered. If $\psi = 0$, no volume is added to F_{DE}. If $\psi = 1$, the intersection between the element and the departure element is determined and its area added to F_{DE}. If $0 < \psi < 1$, the intersection between the element and the departure element is determined and the polygon is then clipped by the reconstructed interface.

4 Results

4.1 Comparison of standard Galerkin and SUPG for level-set method

In this section we will compare the standard Galerkin discretization with the SUPG discretization for the level-set method. All cases are simulated without reinitialization. For this comparison, we will consider a uniform, rotating and reversing vortex velocity field. Both systems are discretized in time according to the Crank-Nicolson scheme. For all simulation we use 2500 nodes, with $n_x = 50$ and $n_y = 50$. The volumes are scaled with the volume at $t = 0$, so the initial volume is always 1. With the standard Galerkin approach we obtain the results shown in Figures 1, 2 and 3.

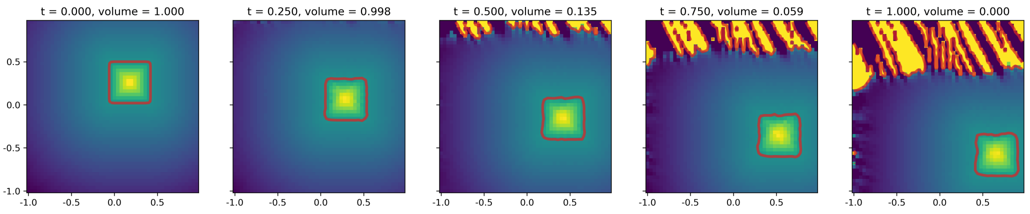

Figure 1: Uniform flow with standard Galerkin approach.

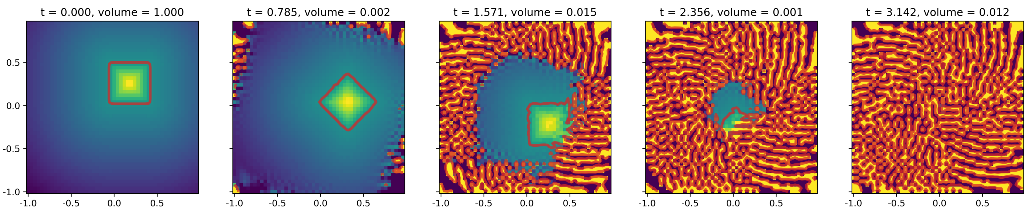

Figure 2: Rotating flow with standard Galerkin approach.

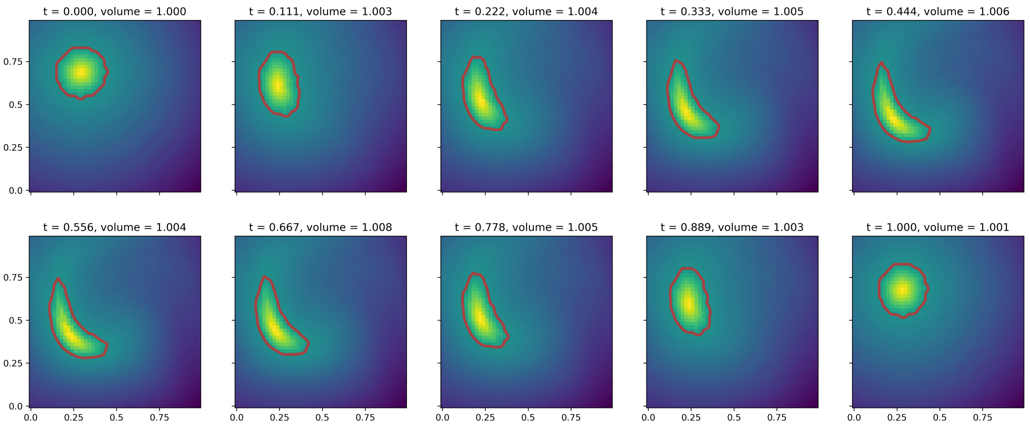

Figure 3: Reverse vortex flow with standard Galerkin approach.

Results clearly show that instabilities occur, especially at the inlet boundary. Therefore we choose to impose Dirichlet boundary conditions at the inlet boundary. This is achieved as follows. When the signed-distance function is initialized, the function is flattened by setting $\phi = 5 \cdot \Delta x$ for $|\phi| > 5 \cdot \Delta x$. This results in constant values at the inlet boundary. In this way, we can impose the Dirichlet condition $g_D = 5 \cdot \Delta x$ at the inlet boundary. Results are displayed in figures 4, 5 and 6.

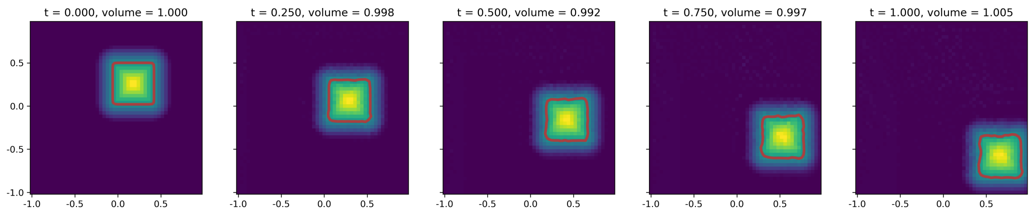

Figure 4: Uniform flow with standard Galerkin approach and Dirichlet boundary conditions.

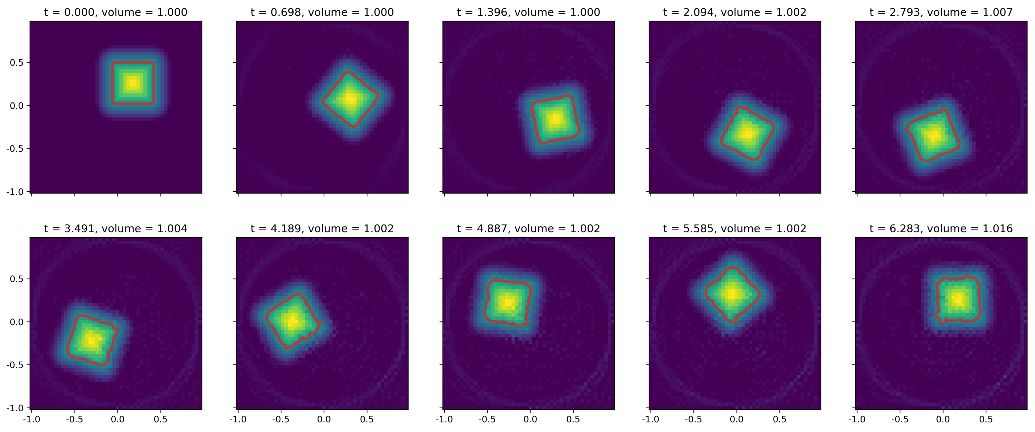

Figure 5: Rotating flow with standard Galerkin approach and Dirichlet boundary conditions.

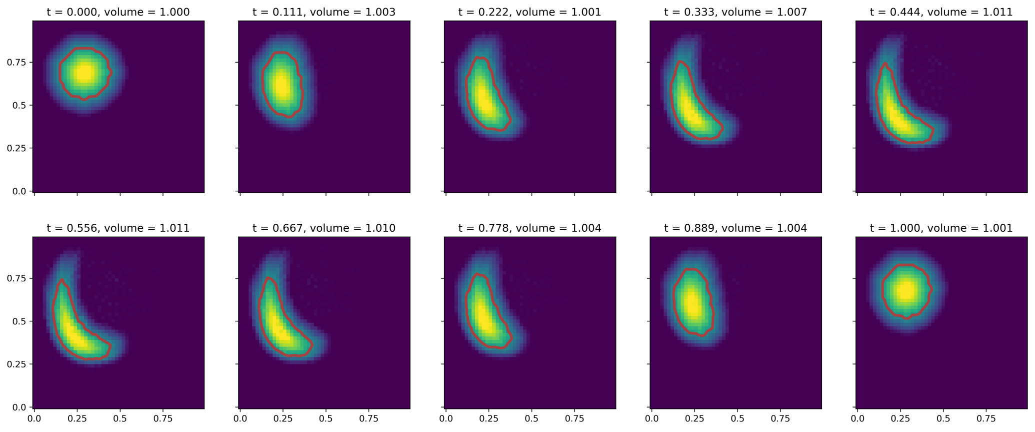

Figure 6: Reverse vortex flow with standard Galerkin approach and Dirichlet boundary conditions.

The instabilities at the inlet boundary have vanished thanks to the Dirichlet boundary conditions. However, we notice small instabilities start to occur at the interface. When we simulate the rotating flow with the square from Figure 5 twice as long, this becomes more visible, see Figure 7.

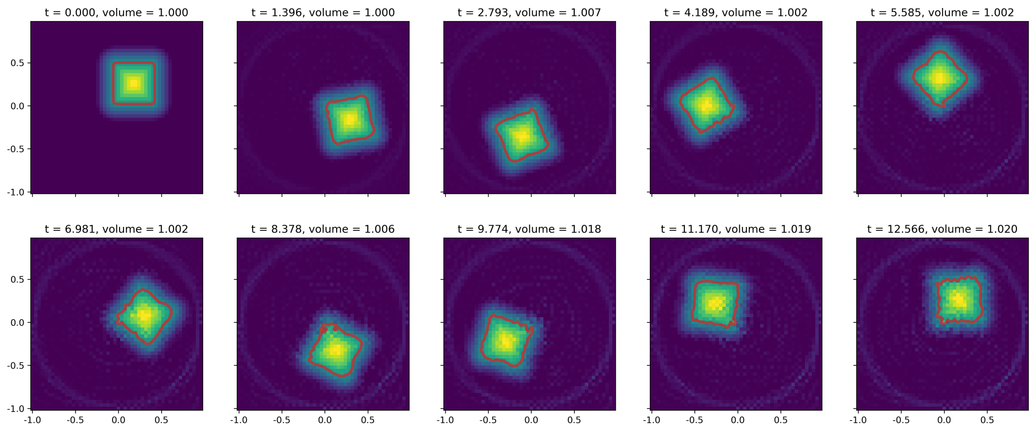

Figure 7: Rotating flow with standard Galerkin approach and Dirichlet boundary conditions.

This confirms what has been found in literature. Indeed, stabilization is needed for a continuous Galerkin approach, since the naive implementation will lead to instabilities. With the SUPG method we get the results, shown in Figures 8, 9 and 10.

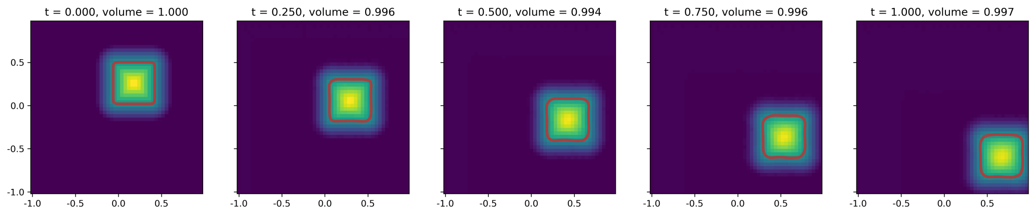

Figure 8: Uniform flow with SUPG method and Dirichlet boundary conditions.

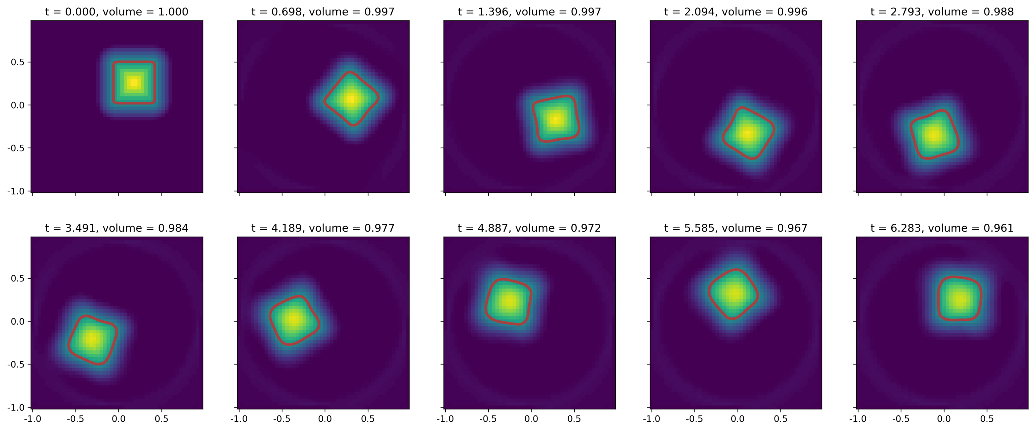

Figure 9: Rotating flow with SUPG method and Dirichlet boundary conditions.

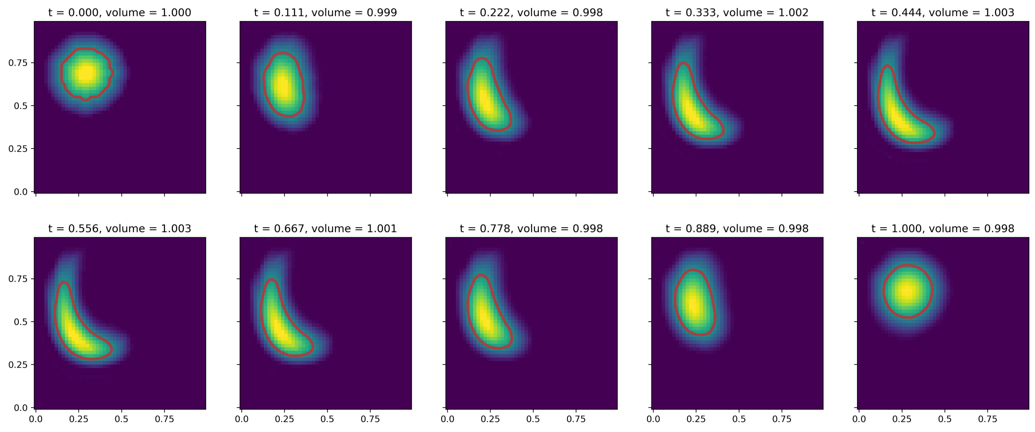

Figure 10: Reverse vortex flow with SUPG method and Dirichlet boundary conditions.

Although the instabilities at the inlet boundary and the interface have vanished completely, volume is still not conserved. Therefore an extra procedure is needed in order to obtain good mass conservation properties.

4.2 Reinitialization

In this section we will investigate how incorporating reinitialization will effect the stability. In every time step the level-set function is reinitialized using the fast marching method.

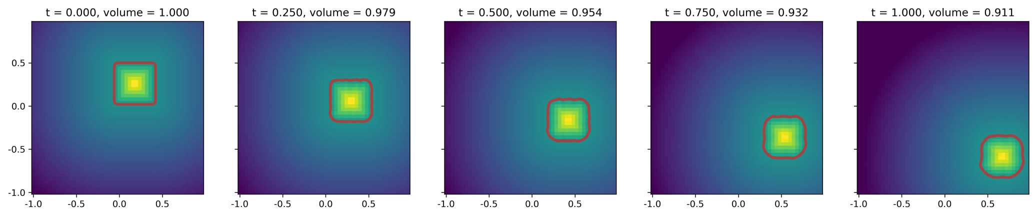

Figure 11: Uniform flow with standard Galerkin and reinitialization

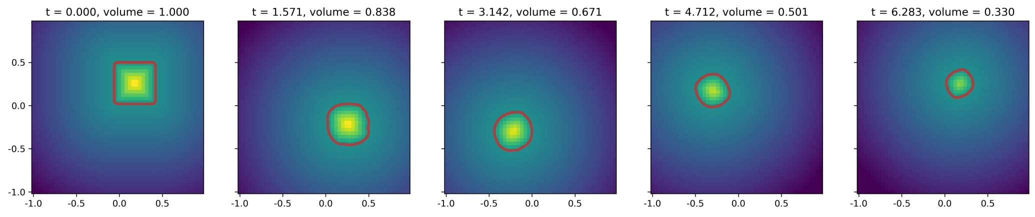

Figure 12: Rotating flow with standard Galerkin and reinitializations.

Results displayed in Figure 11 and 12 show that reinitialization resolve the instabilities. However, the mass conservation property is extremely bad and the shape of the original form is not conserved. So we can conclude that only incorporating reinitialization does not lead to a desired result.

In Figure 13 reinitialization is only applied once at the end of the simulation. The most noticeable difference is the transformation of the granulated ϕ-field to the smooth field. We also notice that a slightly deformation of the interface occures. Furthermore, volume is not conserved exactly during this process.

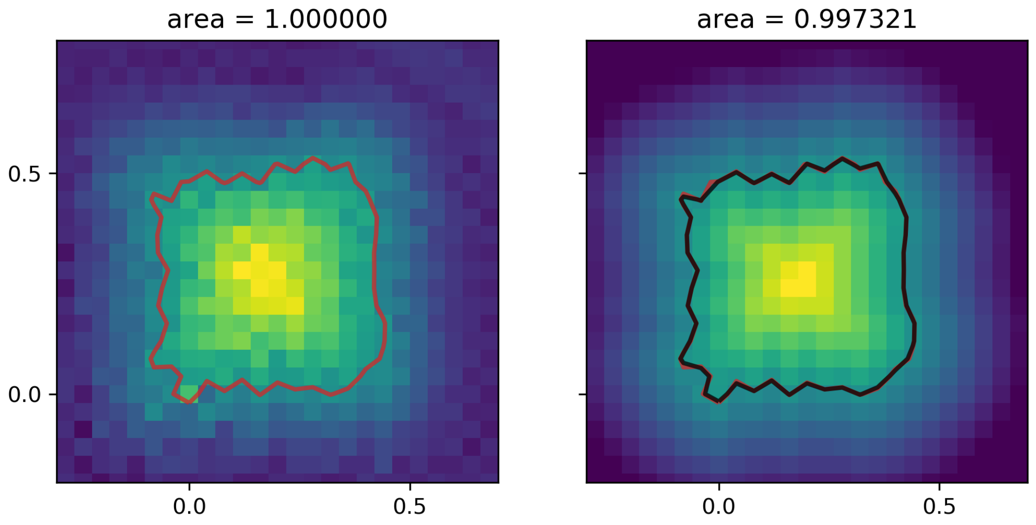

Figure 13: Results of simulation at $t = 4\pi$ before and after reinitialization, where the redistanced interface is displayed as the black contour.

5 Conclusion and discussion

5.1 Conclusion

In this literature study we looked at the available methods for modelling the two-phase flow, and discussed their advantages and disadvantages. In particular, improvements to the original formulation of the level-set method were reviewed. Furthermore, several spatial discretization methods for the level-set method were covered, where the standard Galerkin method and the SUPG method were compared for stability properties. Finally, two methods for the volume-of-fluid advection were discussed.

There exist four methodologies for modelling the two-phase flow, where only the VoF and LS method can be used for handling interface movement without any geometrical restrictions. The advantage of the VoF method is its mass conservation property, but the drawback is the discontinuity and the computationally intensive interface reconstruction. The LS method gives a continuous description of the interface, such that geometrical information can be easily calculated. The drawback is that this method inherently does not conserve volume.

Various methods have been proposed to enhance the behaviour of the LS method, which can be divided into the following approaches: hybrid methods, volume correction methods and modified level-set formulation methods. The most popular approach for hybrid methods is to couple the LS method with the VoF method. This methodology can be roughly divided into three subcategories: CLSVOF, MCLS and VOSET. The CLSVOF method advects both the LS and VoF function, where the LS field is used for interface reconstruction and the volume fractions for mass conservation. The MCLS method uses all available information from the LS method. The VoF function is constructed from the LS function, and is advected without interface reconstruction. The VOSET method only advects the VoF field, and the LS function is constructed for computing the interface normal and curvature.

Results of the comparison of the standard Galerkin method and the SUPG method for the level-set method showed that the naive implementation leads to instabilities. The SUPG resolved this problem, but mass loss still occurred, which reinitialization could not solve either. Therefore an extra procedure is required to conserve mass exactly.

5.2 Further research

The goal for this project is to develop a method, which should conserve volume exactly, give a continuous description of the interface and track the interface accurately. The continuous description of the interface can be achieved with the level-set method. In order to conserve mass we couple this with the VoF method. The VoF method advects a physical quantity, whereas the methods which only utilize a volume correction procedure or modify the formulation of the level-set method are not always physically justified.

CLSVOF is complex and computationally expensive because of the interface reconstruction, and VOSET is based around the VoF function. So MCLS meets our requirements the best. We want to maintain the continuity of the level-set function by advecting the LS field with a continuous Galerkin approach. For geometrical flexibility we wish to develop a method which is able to handle triangular control volumes. However, existing MCLS methods only handle structured rectangular grids or use discontinuous discretization methods. A method for unstructured triangular meshes with a continuous discretization approach still needs to be developed. In further research, we will try to implement a coupled level-set method and investigate what is necessary to successfully and efficiently implement this with a continuous Galerkin approach.

While the SUPG method shows promising results, its accuracy needs to be further evaluated and how well this combines with the VoF method. Other stabilization methods can be considered when the SUPG underperforms. Furthermore, a correction procedure is needed to successfully couple the level-set method with the volume-of-fluid method. Both [2] and [17] proposed a correction algorithm for this particular problem, and inspiration can be taken from other correction schemes. It should also be considered whether it is necessary to reinitialize the level-set function, for which the fast marching method or the solving the differential equation from Eq. (4) can be used.

In this work, we used the Crank-Nicolson scheme for temporal discretization. It may be necessary to switch to an alternate temporal discretization method, due to the implicitness of the scheme. For now, we assume that the velocity field is given for each time step, but when we couple the method to a Navier-Stokes solver, this might be too expensive.

A Level-set method

A.1 Derivation of level-set method

In this part we will give the derivation of the level-set equation, Eq. (3). Consider a point $\mathbf{x}(t)$ on the interface $\Gamma(t)$ where $\phi = 0$. Now assume we have $\phi = \phi(\mathbf{x}(t), t)$ and consider the equation

$$\phi(\mathbf{x}(t), t) = 0.$$

Differentiating this equation with respect to t yields

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = 0.$$

Under the assumption that $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}$, we get

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \mathbf{u} = 0 \quad \text{on } \Gamma(t).$$

This can be generalized to the entire domain Ω resulting in Eq. (3).

A.2 Spatial discretization

A.2.1 Spatial discretization with standard Galerkin

Consider the level-set equation, Eq. (3), the weak form is obtained by multiplying with a test function $v \in V_h^k$. This gives

$$\int_\Omega \frac{\partial \phi}{\partial t} v \, \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla \phi) \, v \, \mathrm{d}v = 0, \quad \forall v \in V_h^k. \tag{47}$$

With Galerkin's method the following approximation is introduced

$$\phi(\mathbf{x}, t) \approx \phi^n(\mathbf{x}, t) = \sum_{j=1}^{n} c_j(t) v_j(\mathbf{x}), \tag{48}$$

where $v_j(\mathbf{x})$ for $j = 1, \ldots, n$ are piecewise polynomials with property $v_j(\mathbf{x}_i) = \delta_{ij}$ and $\sum_j v_j(\mathbf{x}) = 1$.

Substituting this into Eq. (28) yields

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_{j=1}^{n} c_j(t) \int_\Omega v_j v_i \, \mathrm{d}\Omega = \sum_{j=1}^{n} -c_j(t) \int_\Omega \left(\mathbf{u} \cdot \nabla v_j\right) v_i \, \mathrm{d}\Omega, \quad i = 1, \ldots, n, \tag{49}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_{j=1}^{n} M_{ij} c_j(t) = \sum_{j=1}^{n} S_{ij} c_j(t), \quad i = 1, \ldots, n, \tag{50}$$

with

$$\begin{aligned} M_{ij} &= \int_\Omega v_j v_i \, \mathrm{d}\Omega \\ &= \sum_{p=1}^{n_T} \int_{e_p} v_j v_i \, \mathrm{d}\Omega \\ &= \sum_{p=1}^{n_T} M_{ij}^{e_p}, \end{aligned} \tag{51}$$

and

$$\begin{aligned} S_{ij} &= -\int_\Omega \left(\mathbf{u} \cdot \nabla v_j\right) v_i \, \mathrm{d}\Omega \\ &= \sum_{p=1}^{n_T} -\int_{e_p} \left(\mathbf{u} \cdot \nabla v_j\right) v_i \, \mathrm{d}\Omega \\ &= \sum_{p=1}^{n_T} S_{ij}^{e_p}, \end{aligned} \tag{52}$$

where e_p denotes a triangular element with vertices $\mathbf{x}_{p_1}$, $\mathbf{x}_{p_2}$ and $\mathbf{x}_{p_3}$, and n_T the number of elements.

For the next part the formula of Holand and Bell is used. Let e be a triangle with vertices $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ and v_1, v_2 and v_3 linear on e, and let $m_1, m_2, m_3 \in \mathbb{N}$, then

$$\int_e v_1^{m_1} v_2^{m_2} v_3^{m_3} \, \mathrm{d}\Omega = \frac{|\Delta e| m_1! m_2! m_3!}{(2 + m_1 + m_2 + m_3)!}, \tag{53}$$

where the area of e is equal to $|\Delta e|/2$ with

$$|\Delta e| = \left\| \begin{matrix} x_3 - x_1 & x_2 - x_1 \\ y_3 - y_1 & y_2 - y_1 \end{matrix} \right\| = \left\| \begin{matrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{matrix} \right\| .$$

Then with the Holand and Bell formula we find

$$\begin{aligned} M_{ij}^{e_p} = \int_{e_p} v_j v_i \, \mathrm{d}\Omega &\overset{\text{H-B}}{=} \begin{cases} \dfrac{|\Delta e_p|}{24} & i \neq j \\ \dfrac{|\Delta e_p|}{12} & i = j \end{cases} \\ &= \frac{1}{24}(1 + \delta_{ij})|\Delta e_p|, \quad i, j \in \{p_1, p_2, p_3\}. \end{aligned} \tag{54}$$

So the element mass matrix of e_p is given by

$$M^{e_p} = \frac{|\Delta e_p|}{24} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} .$$

For the piecewise polynomial $v_j(\mathbf{x})$ we can write $\phi_j(\mathbf{x}) = \alpha_j + \beta_j x + \gamma_j y$. Using $\mathbf{u} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^\top$, we can write

$$\begin{aligned} S_{ij}^{e_p} &= -\int_{e_p} \left(\mathbf{u} \cdot \nabla v_j \right) v_i \, \mathrm{d}\Omega \\ &= -\left(\beta_j \int_{e_p} u_1 v_i \, \mathrm{d}\Omega + \gamma_j \int_{e_p} u_2 v_i \, \mathrm{d}\Omega \right). \end{aligned} \tag{55}$$

We evaluate the first integral using the Newton-Cotes quadrature rule

$$\begin{aligned} \int_{e_p} u_1 v_i \, \mathrm{d}\Omega &\overset{\text{N-C}}{=} \frac{|\Delta e_p|}{6} \sum_{l \in \{p_1, p_2, p_3\}} u_1(\mathbf{x}_l) v_i(\mathbf{x}_l) \\ &= \frac{|\Delta e_p|}{6} \sum_{l \in \{p_1, p_2, p_3\}} u_1(\mathbf{x}_l) \delta_{il} \\ &= \frac{|\Delta e_p|}{6} u_1(\mathbf{x}_i), \quad i \in \{p_1, p_2, p_3\}. \end{aligned}$$

Hence, for $S_{ij}^{e_p}$ we have

$$S_{ij}^{e_p} = -\frac{|\Delta e_p|}{6} \left(\beta_j u_1(\mathbf{x}_i) + \gamma_j u_2(\mathbf{x}_i) \right), \quad i, j \in \{p_1, p_2, p_3\}. \tag{56}$$

So the element stiffness matrix of e_p is given by

$$\begin{aligned} S^{e_p} &= -\frac{|\Delta e_p|}{6} \begin{bmatrix} [\beta_1 \ \gamma_1]^\top \cdot \mathbf{u}(\mathbf{x}_1) & [\beta_2 \ \gamma_2]^\top \cdot \mathbf{u}(\mathbf{x}_1) & [\beta_3 \ \gamma_3]^\top \cdot \mathbf{u}(\mathbf{x}_1) \\ [\beta_1 \ \gamma_1]^\top \cdot \mathbf{u}(\mathbf{x}_2) & [\beta_2 \ \gamma_2]^\top \cdot \mathbf{u}(\mathbf{x}_2) & [\beta_3 \ \gamma_3]^\top \cdot \mathbf{u}(\mathbf{x}_2) \\ [\beta_1 \ \gamma_1]^\top \cdot \mathbf{u}(\mathbf{x}_3) & [\beta_2 \ \gamma_2]^\top \cdot \mathbf{u}(\mathbf{x}_3) & [\beta_3 \ \gamma_3]^\top \cdot \mathbf{u}(\mathbf{x}_3) \end{bmatrix} \\ &= -\frac{|\Delta e_p|}{6} \left(\begin{bmatrix} u_1(\mathbf{x}_1) \\ u_1(\mathbf{x}_2) \\ u_1(\mathbf{x}_3) \end{bmatrix} \begin{bmatrix} \beta_1 & \beta_2 & \beta_3 \end{bmatrix} + \begin{bmatrix} u_2(\mathbf{x}_1) \\ u_2(\mathbf{x}_2) \\ u_2(\mathbf{x}_3) \end{bmatrix} \begin{bmatrix} \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \right). \end{aligned}$$

Finally, the semi-discrete system is obtained

$$M \frac{\mathrm{d}\mathbf{c}}{\mathrm{d}t} = S\mathbf{c}. \tag{57}$$

A.2.2 Spatial discretization with standard Galerkin and Dirichlet boundary conditions

Now we will also incorporate Dirichlet boundary conditions at the inlet boundaries

$$\begin{cases} \dfrac{\partial \phi}{\partial t} + \mathbf{u}(\mathbf{x}) \cdot \nabla \phi = 0 & \mathbf{x} \in \Omega, \\ \phi = g(\mathbf{x}) & \mathbf{x} \in \partial\Omega_{\text{inlet}}. \end{cases} \tag{58}$$

With Galerkin's method we use the following approximation

$$\phi(\mathbf{x}, t) \approx \phi^n(\mathbf{x}, t) = \sum_{j \in \text{Ind}} c_j(t) v_j(\mathbf{x}) + \sum_{j \in \text{Dir}} g(\mathbf{x}_j) v_j(\mathbf{x}), \tag{59}$$

where $v_j(\mathbf{x})$ for $j = 1, \ldots, n$ are piecewise polynomials with property $v_j(\mathbf{x}_i) = \delta_{ij}$. The independent nodes are denoted by Ind and the Dirichlet nodes by Dir.

Substituting this into Eq. (28) yields

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \sum_{j \in \text{Ind}} c_j(t) \int_\Omega v_j v_i \, \mathrm{d}\Omega = & \sum_{j \in \text{Ind}} -c_j(t) \int_\Omega \big(\mathbf{u} \cdot \nabla v_j\big) \, v_i \, \mathrm{d}\Omega \\ & - \sum_{j \in \text{Dir}} g(\mathbf{x}_j) \left(\int_\Omega v_j v_i \, \mathrm{d}\Omega + \int_\Omega \big(\mathbf{u} \cdot \nabla v_j\big) \, v_i \, \mathrm{d}\Omega \right), \quad \forall i \in \text{Ind}. \end{aligned} \tag{60}$$

This can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_{j \in \text{Ind}} M_{ij} c_j(t) = \sum_{j \in \text{Ind}} S_{ij} c_j(t) + \sum_{j \in \text{Dir}} (S_{ij} - M_{ij}) g(\mathbf{x}_j), \quad \forall i \in \text{Ind}. \tag{61}$$

With the same derivation as in Section A.2.1 we obtain the semi-discrete system for Dirichlet boundary conditions at the inlet boundary

$$M \frac{\mathrm{d}\mathbf{c}}{\mathrm{d}t} = S\mathbf{c} + \mathbf{f}. \tag{62}$$

A.2.3 Spatial discretization with SUPG

The discretization is similar to the discretization from Section A.2.2, but instead the test function from Eq. 32 is used. This gives

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \sum_{j \in \text{Ind}} c_j(t) \int_\Omega v_j (v_i + \delta_K \mathbf{u} \cdot \nabla v_i) \, \mathrm{d}\Omega = & \\ & \sum_{j \in \text{Ind}} -c_j(t) \int_\Omega \big(\mathbf{u} \cdot \nabla v_j\big) \, (v_i + \delta_K \mathbf{u} \cdot \nabla v_i) \, \mathrm{d}\Omega \\ & - \sum_{j \in \text{Dir}} g(\mathbf{x}_j) \int_\Omega v_j (v_i + \delta_K \mathbf{u} \cdot \nabla v_i) \, \mathrm{d}\Omega \\ & - \sum_{j \in \text{Dir}} g(\mathbf{x}_j) \int_\Omega (\mathbf{u} \cdot \nabla v_j)(v_i + \delta_K \mathbf{u} \cdot \nabla v_i) \, \mathrm{d}\Omega, \quad \forall i \in \text{Ind}. \end{aligned} \tag{63}$$

This can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_{j \in \text{Ind}} \tilde{M}_{ij} c_j(t) = \sum_{j \in \text{Ind}} \tilde{S}_{ij} c_j(t) + \sum_{j \in \text{Dir}} (\tilde{S}_{ij} - \tilde{M}_{ij}) g(\mathbf{x}_j), \quad \forall i \in \text{Ind}. \tag{64}$$

with

$$\begin{aligned}\tilde{M}_{ij} &= \int_\Omega v_j(v_i + \delta_p \mathbf{u}\cdot\nabla v_i)\,\mathrm{d}\Omega \\ &= \frac{\Delta e_p}{24}\left(1 + \delta_{ij} + 4\delta_p(\beta_i u_1(\mathbf{x}_j) + \gamma_i u_2(\mathbf{x}_j)) + \sum_{l\in\{p_1,p_2,p_3\}} u_1(\mathbf{x}_l)\right).\end{aligned} \tag{65}$$

and

$$\begin{aligned}\tilde{S}_{ij} &= -\int_\Omega \left(\mathbf{u}\cdot\nabla v_j\right)\left(v_i + \delta_K \mathbf{u}\cdot\nabla v_i\right)\mathrm{d}\Omega \\ &= -\frac{\Delta e_p}{6}\Big(\beta_j u_{1,p}(\mathbf{x}_i) + \gamma_j u_{2,p}(\mathbf{x}_i) + \delta_p((u_1^2)_p\beta_i\beta_j + (u_1u_2)_p\gamma_i\beta_j + (u_1u_2)_p\beta_i\gamma_j) + (u_2^2)_p\gamma_i\gamma_j\Big),\end{aligned} \tag{66}$$

with

$$u_p = \sum_{l\in\{p_1,p_2,p_3\}} u(\mathbf{x}_l).$$

The mass matrix element is computed as follows

$$\tilde{M}^{e_p} = \frac{|\Delta e_p|}{24}\begin{bmatrix} 2 + 4\delta_p[\beta_1\ \gamma_1]^\top\cdot\mathbf{u}(\mathbf{x}_1) & 1 + 4\delta_p[\beta_1\ \gamma_1]^\top\cdot\mathbf{u}(\mathbf{x}_2) & 1 + 4\delta_p[\beta_1\ \gamma_1]^\top\cdot\mathbf{u}(\mathbf{x}_3) \\ 1 + 4\delta_p[\beta_2\ \gamma_2]^\top\cdot\mathbf{u}(\mathbf{x}_1) & 2 + 4\delta_p[\beta_2\ \gamma_2]^\top\cdot\mathbf{u}(\mathbf{x}_2) & 1 + 4\delta_p[\beta_2\ \gamma_2]^\top\cdot\mathbf{u}(\mathbf{x}_3) \\ 1 + 4\delta_p[\beta_3\ \gamma_3]^\top\cdot\mathbf{u}(\mathbf{x}_1) & 1 + 4\delta_p[\beta_3\ \gamma_3]^\top\cdot\mathbf{u}(\mathbf{x}_2) & 2 + 4\delta_p[\beta_3\ \gamma_3]^\top\cdot\mathbf{u}(\mathbf{x}_3)\end{bmatrix}.$$

For the stiffness matrix we will only show how one entry is computed

$$\begin{aligned}\tilde{S}_{12} = -\frac{\Delta e_p}{6}\big(&\beta_2 u_{1,p}(\mathbf{x}_1) + \gamma_2 u_{2,p}(\mathbf{x}_1) \\ &+ \delta_p((u_1^2)_p\beta_1\beta_2 + (u_1u_2)_p\gamma_1\beta_2 + (u_1u_2)_p\beta_1\gamma_2) + (u_2^2)_p\gamma_1\gamma_2\big).\end{aligned}$$

Then we obtain the semi-discrete SUPG system for Dirichlet boundary conditions at the inlet boundary

$$\tilde{M}\frac{\mathrm{d}\mathbf{c}}{\mathrm{d}t} = \tilde{S}\mathbf{c} + \tilde{\mathbf{f}}. \tag{67}$$

References

[1] E. Burman. Consistent supg-method for transient transport problems: Stability and convergence. *Computer Methods in Applied Mechanics and Engineering*, 199(17):1114 – 1123, 2010.

[2] D. den Ouden-van der Horst and M. Möller. *Volume-Preserving Continuous Galerkin Level-Set Approach for Linear Finite Elements.* Delft University of Technology, Netherlands, 2 2020.

[3] V. Dolejsí and M. Feistauer. *Discontinuous Galerkin Method: Analysis and Applications to Compressible Flow.* Springer Publishing Company, Incorporated, 1st edition, 2015.

[4] Z. Ge, J.-C. Loiseau, O. Tammisola, and L. Brandt. An efficient mass-preserving interface-correction level set/ghost fluid method for droplet suspensions under depletion forces. *Journal of Computational Physics*, 353:435 – 459, 2018.

[5] S. Gross and A. Reusken. *Numerical Methods for Two-phase Incompressible Flows.* Springer, Berlin, Heidelberg, 1st edition, 2011.

[6] T. J. Hughes, L. P. Franca, and G. M. Hulbert. A new finite element formulation for computational fluid dynamics: Viii. the galerkin/least-squares method for advective-diffusive equations. *Computer Methods in Applied Mechanics and Engineering*, 73(2):173 – 189, 1989.

[7] D. Kennedy. Level set methods for two-phase flows with fem. Master's thesis, Uppsala University, Department of Information Technology, 2014.

[8] K. Ling, S. Zhang, P.-Z. Wu, S.-Y. Yang, and W.-Q. Tao. A coupled volume-of-fluid and level-set method (voset) for capturing interface of two-phase flows in arbitrary polygon grid. *International Journal of Heat and Mass Transfer*, 143:118565, 2019.

[9] E. Loch and A. Reusken. The level set method for capturing interfaces with applications in two-phase flow problems. In *PhD Thesis, RWTH Aachen*, 2013.

[10] F. Losasso, R. Fedkiw, and S. Osher. Spatially adaptive techniques for level set methods and incompressible flow. *Computers & Fluids*, 35(10):995 – 1010, 2006.

[11] K. Luo, C. Shao, Y. Yang, and J. Fan. A mass conserving level set method for detailed numerical simulation of liquid atomization. *Journal of Computational Physics*, 298:495 – 519, 2015.

[12] E. Marchandise, J.-F. Remacle, and N. Chevaugeon. A quadrature-free discontinuous galerkin method for the level set equation. *Journal of Computational Physics*, 212(1):338 – 357, 2006.

[13] H. M. Mourad, J. Dolbow, and K. Garikipati. An assumed-gradient finite element method for the level set equation. *International Journal for Numerical Methods in Engineering*, 64(8):1009–1032, 2005.

[14] E. Olsson and G. Kreiss. A conservative level set method for two phase flow. *Journal of Computational Physics*, 210(1):225 – 246, 2005.

[15] N. Parolini. Computational fluid dynamics for naval engineering problems. *EPFL scientific publications*, page 184, 2004. Sélectionné pour le "Prix EPFL de doctorats 2004" avec mention spéciale - "EPFL doctorate award 2004" distinction nominee.

[16] D. A. D. Pietro, S. L. Forte, and N. Parolini. Mass preserving finite element implementations of the level set method. *Applied Numerical Mathematics*, 56(9):1179 – 1195, 2006. Numerical Methods for Viscosity Solutions and Applications.

[17] F. Raees, D. R. van der Heul, and C. Vuik. A mass-conserving level-set method for simulation of multiphase flow in geometrically complicated domains. *International Journal for Numerical Methods in Fluids*, 81(7):399–425, 2016.

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

[19] K. Shahbazi, M. Paraschivoiu, and J. Mostaghimi. Second order accurate volume tracking based on remapping for triangular meshes. *Journal of Computational Physics*, 188(1):100 – 122, 2003.

[20] M. Sussman and E. G. Puckett. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. *Journal of Computational Physics*, 162(2):301 – 337, 2000.

[21] M. K. Touré and A. Soulaimani. Stabilized finite element methods for solving the level set equation without reinitialization. *Computers & Mathematics with Applications*, 71(8):1602 – 1623, 2016.

[22] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A front-tracking method for the computations of multiphase flow. *Journal of Computational Physics*, 169(2):708 – 759, 2001.

[23] S. P. van der Pijl, A. Segal, C. Vuik, and P. Wesseling. A mass-conserving level-set method for modelling of multi-phase flows. *International Journal for Numerical Methods in Fluids*, 47(4):339–361, 2005.

[24] X. Yang, A. J. James, J. Lowengrub, X. Zheng, and V. Cristini. An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids. *Journal of Computational Physics*, 217(2):364 – 394, 2006.

[25] Y. F. Yap, J. C. Chai, T. N. Wong, K. C. Toh, and H. Y. Zhang. A global mass correction scheme for the level-set method. *Numerical Heat Transfer, Part B: Fundamentals*, 50(5):455–472, 2006.

[26] C. Yu, Z. Ye, T. W. Sheu, Y. Lin, and X. Zhao. An improved interface preserving level set method for simulating three dimensional rising bubble. *International Journal of Heat and Mass Transfer*, 103:753 – 772, 2016.

[27] Y. Zhang, Q. Zou, and D. Greaves. Numerical simulation of free-surface flow using the level-set method with global mass correction. *International Journal for Numerical Methods in Fluids*, 63:651 – 680, 01 2009.

[28] L. Zhao, J. Mao, X. Bai, X. Liu, T. Li, and J. Williams. Finite element implementation of an improved conservative level set method for two-phase flow. *Computers & Fluids*, 100:138 – 154, 2014.