A SEMI-LAGRANGIAN PARTICLE LEVEL SET FINITE ELEMENT METHOD FOR INTERFACE PROBLEMS∗

RODOLFO BERMEJO† AND JUAN LUIS PRIETO‡

Abstract. We present a quasi-monotone semi-Lagrangian particle level set (QMSL-PLS) method for moving interfaces. The QMSL method is a blend of first order monotone and second order semi-Lagrangian methods. The QMSL-PLS method is easy to implement, efficient, and well adapted for unstructured, either simplicial or hexahedral, meshes. We prove that it is unconditionally stable in the maximum discrete norm, \( \| \cdot \|_{h, \infty} \), and the error analysis shows that when the level set solution \( u(t) \) is in the Sobolev space \( W^{r+1, \infty}(D) \), \( r \geq 0 \), the convergence in the maximum norm is of the form \( (KT/\Delta t)^{\min(1, \alpha)} h^{p} + h^{q} \), \( p = \min(2, r+1) \), and \( q = \min(3, r+1) \), where \( \alpha \) is a velocity. This means that at high CFL numbers, that is, when \( \Delta t > h \), the error is \( O((1 - \alpha) h^{p-1} + h^{q-1}) \), whereas at CFL numbers less than 1, the error is \( O((1 - \alpha) h^{p} + h^{q}) \). We have tested our method with satisfactory results in benchmark problems such as the Zalesak’s slotted disk, the single vortex flow, and the rising bubble.

Key words. level set, semi-Lagrangian, reinitialization, finite elements, incompressible flows, interfaces

AMS subject classifications. 65M12, 65M25, 65M60

DOI. 10.1137/110830587

1. Introduction. The level set method is a front capturing technique proposed in [23] to calculate the motion of fluid interfaces as well as of curves and surfaces whose speeds depend on local curvatures. The books [22] and [25] are general references of the method and its applications in many engineering and science problems in which the motion of interfaces and fronts are an important component of the solution. The technique uses a fixed (Eulerian) mesh and finds the front as a particular level set (moving with time) of a scalar function. In this paper, we shall consider the front as the zero level set of the signed distance function to the interface satisfying a time dependent advection equation. The problem with this approach is that the numerical solution of the linear advection equation loses its distance character and because of numerical errors also the conservation of volume property (known as mass conservation as well). To overcome these drawbacks, some authors use an adaptive approach combined with discontinuous Galerkin method as, for instance, [11] and [14], or spectral methods as [13], to calculate the level set solution; however, [28] and [26] have devised different schemes under the name of reinitialization or redistancing, which consist of solving, until reaching the steady state, a pseudo time dependent nonlinear transport equation using as initial condition the solution of the linear advection equation. Standard shock capturing Eulerian schemes, such as WENO/ENO schemes for Hamilton–Jacoby equations (HJ-(W)ENO) combined with TVD-based high order
Runge–Kutta schemes, have been used to numerically solve these equations. The issues we face with the use of these Eulerian schemes are twofold: first, they must satisfy a stability CFL criterion that limits the length of the time step $\Delta t$; and second, in underresolved regions where there is a lack of information about the structure of the characteristic curves of the equation, they may not process the information about the characteristics in the right way and, therefore, are not able to reconstruct accurately enough the zero level set in such regions. To remedy this deficiency, [16] proposes the use of Lagrangian marker particles, randomly distributed in a narrow tube around the interface, to measure and correct the numerical errors by counting the number of particles crossing the interface, rebuilding in this way the zero level set in underresolved regions more accurately and, consequently, improving the mass conservation property. The new method is named hybrid particle level set method (HPLSM). Later on, [17] demonstrates that the combination of Lagrangian marked particles with the so-called Courant–Isaacson–Rees (CIR) scheme, which is a first order semi-Lagrangian scheme, applied to the level set computations yields a method that is much more efficient, in terms of CPU time versus accuracy, than the HPSLM. One reason for this is the fact that the CIR scheme is stable in the maximum norm, so that there is no CFL restriction on the size of the time step. The CIR scheme was introduced by Courant, Isaacson, and Rees [12] to integrate hyperbolic equations using the method of characteristics backward in time and interpolating the solution at the feet of the characteristics by piecewise linear polynomials; this procedure yields a monotonicity preserving upwind scheme with the least truncation error. Strain [30, 29] has also used the semi-Lagrangian approach with high order interpolators of ENO type to integrate level set equations in adaptive quadtree meshes. Dupont and Liu [15] combine the backward error compensation technique with the semi-Lagrangian CIR scheme to achieve a second order semi-Lagrangian scheme to integrate level set equations.

In this paper we present a quasi-monotone semi-Lagrangian particle level set (QMSL-PLS) method for level set interface computations. Our method is developed in the framework of finite elements and is specifically designed to be used in simplicial meshes; nevertheless, it can also be used in simplicial finite volume methods. The idea of using simplicial meshes is twofold. First, considering that in general the level set approach is a component of a more general problem formulated in domains with no simple shapes, it is recognized that in this case simplicial meshes yield a better representation of the geometrical features. Second, we think of the level set method formulated in an adaptive framework: for this case the simplicial meshes are more flexible than the quadrilateral meshes. Our treatment of the level set problem consists of solving by the QMSL scheme of [8] and [4] the advection equation for the level set function, the solution of which is corrected at the zero level set by the particle method of [17], followed then by a reinitialization procedure that is solved by a two-step scheme. In the first step we calculate directly (by a geometrical or minimization method) the distance to the zero level set of the mesh points inside a narrow band around it, and in the second step we solve using the QMSL method the pseudo time dependent nonlinear transport problem outside that band. The nice properties of this scheme are the following: (1) in the reinitialization stage and for $\Delta t$ small, the characteristics of the pseudo time dependent nonlinear transport equation will, in general, not cross the interface and (2) the solution of the nonlinear problem becomes a piecewise interpolation problem at the feet of the characteristics curves. The scheme is fast, accurate, and easy to implement.

We introduce some notation about the functional spaces we use in this paper. For $s \geq 0$ real and real $1 \leq p \leq \infty$, $W^{s,p}(D)$ denotes the real Sobolev spaces defined
on $D$ for real scalar-valued functions. $\| \cdot \|_{W^{s,p}(D)}$ and $| \cdot |_{W^{s,p}(D)}$ denote the norm and semi-norm, respectively, of $W^{s,p}(D)$. When $s = 0$, $W^{0,p}(D) := L^p(D)$. The corresponding spaces of real vector-valued functions are denoted by $W^{s,p}(D)^d := \{ v : D \to \mathbb{R}^d : v_i \in W^{s,p}(D) \}$. Let $X$ be a real Banach space $(X, \| \cdot \|_X)$. If $v : [0, T] \to X$ is a strongly measurable function with values in $X$, we set $\| v \|_{L^p(0,t;X)} = \left( \int_0^t \| v(\tau) \|_X^p \, d\tau \right)^{1/p}$ for $1 \leq p < \infty$, and $\| v \|_{L^\infty(0,t;X)} \equiv \esssup_{0 \leq \tau \leq t} \| v(\tau) \|_X$; when $t = T$, we shall write, unless otherwise stated, $\| v \|_{L^p(X)}$. We also use the space of continuous and bounded functions $C(D)$ and the space

$$l^\infty(0,T;X) := \left\{ v : [0,t_1, t_2, \ldots, t_N = T] \to X : \max_{1 \leq i \leq N} \| v(t_i) \|_X < \infty \right\},$$

and write $l^\infty(0,T;X)$ as $l^\infty(X)$.

The layout of the paper is as follows. In section 2 we introduce the level set formulation and the reinitialization problem. Section 3 is devoted to the presentation of two subdomains, say, $D$ and $\partial D$ in $\mathbb{R}^n$, convenient to choose $\Gamma$: any time $t \in [0,T]$, $\Gamma(t)$ is a one-dimensional manifold separating the domains $D$ and $\partial D$ undergoing a time dependent motion; therefore, we write $\Gamma(t)$, where $t \in [0,T]$ and $T > 0$ is a real number. $\Gamma(t)$ is called free interface, or simply interface. Assuming that at time $t = 0$, $\Gamma(0)$ is known, the level set method is a technique to describe the motion of $\Gamma(t)$ considering it as the zero level set of a function $u(t) : D \to \mathbb{R}$. Specifically, at any time $t \in [0,T]

\begin{equation}
\tag{2.1}
\Gamma(t) := \{ x \in D : u(x,t) = 0 \}.
\end{equation}

On any level set of $u(x,t)$, i.e., $u(x,t) = C$, it follows that $\frac{D}{Dt} = \frac{\partial u}{\partial t} + v \cdot \nabla u = 0$ in $D \times (0,T)$, where $v(x,t) = \frac{\partial u}{\partial x}$ is a velocity field defined in $D$; therefore, when $x(t) \in \Gamma(t)$, $v(x)$ represents the velocity of the points of the interface. There may be many functions $u(x,t)$ for which $\Gamma(t)$ is a zero level set, but for many purposes it is convenient to choose $u(x,t)$ as the signed normal distance function to $\Gamma(t)$, i.e.,

\begin{equation}
\tag{2.2}
u(x,t) = \pm \min_{y \in \Gamma(t)} |x - y|, \quad x \in D,
\end{equation}

where $|x - y|$ denotes, unless otherwise stated, the Euclidean distance between $x$ and $y$. Therefore, for $t \in [0,T]$, $u(x,t)$ is characterized by the following properties:

1. The initial value problem,

\begin{equation}
\tag{2.3a}
\begin{cases}
\frac{Du}{Dt} = \frac{\partial u}{\partial t} + v \cdot \nabla u = 0 \text{ in } D \times (0,T), \\
u(x,0) = \pm \min_{y \in \Gamma(0)} |x - y|, \quad x \in D,
\end{cases}
\end{equation}
(2) The distance property,
\[ |\nabla u| = 1, \]
and
(3)
\[ u(x, t) \begin{cases} > 0 & \text{if } x \in D_1, \\ = 0 & \text{if } x \in \Gamma_0(t), \\ < 0 & \text{if } x \in D_2. \end{cases} \]

Hence, \( u(x, t) \) is a Lipschitz continuous function with respect to \( x \), i.e., for \( t \) in \([0, T]\), \( u(x, t) \in W^{1,\infty}(D) \). Actually, \( u \) is many times continuously differentiable in \( D_1 \) and \( D_2 \). It is interesting to note that (2.3a) moves the interface \( \Gamma_0(t) \) at the correct velocity, but \( u(x, t) \) as solution of (2.3a) will no longer remain a signed distance function because, in general, \( u \) will not satisfy \( |\nabla u| = 1 \) and may become irregular or flat after a few time steps. To remedy these drawbacks, [28] proposes a procedure, called reinitialization or redistancing, that restores to \( u(x, t) \) the character of signed distance.

A nice feature of the level set formulation is that geometric quantities such as the unit normal vector to the level set \( u(x, t) = C \),
\[ n = \frac{\nabla u}{|\nabla u|}, \]
the curvature
\[ \kappa = -\nabla \cdot n, \]
and the integral
\[ |D_2| = \int_D H(-u)dx \]
are easy to calculate in terms of \( u \). Here, \( |D_2| \) denotes the measure (area in two dimensions (2D) or volume in three dimensions (3D)) of \( D_2 \), and \( H(u) \) is the graph of Heaviside, i.e.,
\[ H(u) = \begin{cases} 1 & \text{if } u > 0, \\ [0, 1] & \text{if } u = 0, \\ 0 & \text{if } u < 0. \end{cases} \]

2.1. Reinitialization. The reinitialization is the process of replacing \( u(x, t_n) \) at time \( t_n \) by a signed distance function \( d(x, t_n) \) that has the same zero level set and better smoothness properties, and then \( d(x, t_n) \) is taken as the new datum to advance the solution of (2.3a) until the new round of reinitialization. Now, the question is on the legitimacy of this process in the calculation of the interface \( \Gamma_0(t) \). Theoretically, this is justified in [18], where it is proved that \( \Gamma_0(t) \) does not depend on the particular choice of the initial condition \( u(x, 0) \) as long as the zero level set coincides with \( \Gamma_0(0) \).

To make the solution \( u(x, t) \) to (2.3a) satisfy \( |\nabla u| = 1 \), [28] proposes the method that consists of solving for the function \( d : D \times [0, T^*] \to \mathbb{R} \), up to reaching the steady state, the first order nonlinear hyperbolic problem
\[ \begin{cases} \frac{\partial d}{\partial \tau} + w \cdot \nabla d = \text{sign}(u) & \text{in } D \times (0, T^*], \\ d(x, 0) = u(x, t), \end{cases} \]
where \( \tau \) is a pseudo time variable, \( u(x, t) \) is the solution to (2.3a) at time \( t \) whose zero level set is the interface, and \( w \) is a real vector-valued function that plays the role of an advection velocity, the expression of which is

\[
(2.5b) \quad w = \text{sign}(u) \frac{\nabla d}{|\nabla d|} = \text{sign}(u)n.
\]

Several items must be noted. First, Sussman and Fatemi [26] calculate the solution of (2.5a)-(2.5b) by the method of characteristics and show that there is a time \( t^* \) such that in the domain where \( u(x, t) \) is positive

\[
(2.6a) \quad d(x, \tau) = \begin{cases} \tau + u(X_w(x, \tau; 0), t) & \text{if } \tau \leq t^*, \\ t^* & \text{if } \tau > t^*, \end{cases}
\]

\( t^* \) being the shortest distance from \( x \) to the zero level set \( u(x, t) = 0 \). Similarly, in the domain \( D_2 \), where \( u(x, t) \) is negative,

\[
(2.6b) \quad d(x, \tau) = \begin{cases} -\tau + u(X_w(x, \tau; 0), t) & \text{if } \tau \leq t^*, \\ -t^* & \text{if } \tau > t^*. \end{cases}
\]

Assuming that \( u(x, t) \) is of class \( C^2 \) in \( D_1 \) and \( D_2 \), for \( \tau \) sufficiently small such that \( \tau \max |\kappa_{\Gamma_0}| < 1 \), we can extend the solution in a neighborhood of \( \Gamma_0(t) \) by virtue of the implicit function theorem; specifically, from \( x_0 \in \Gamma_0(t) \) the solution is extended by the formula

\[
(2.6c) \quad d(x_0 \pm \tau n(x_0)) = \pm \tau,
\]

where \( \kappa_{\Gamma_0} \) denotes the curvature at the points of \( \Gamma_0(t) \) and \( n(x_0) \) is the unit normal vector at \( x_0 \). For details see [26]. In the above formulae, \( X_w(x, s, \tau) \) denotes the characteristic curves of the operator \( \frac{\partial}{\partial \tau} + w \cdot \nabla \), which are a solution of the initial value problem

\[
(2.6d) \quad \begin{cases} \frac{dX_w(x, s, \tau)}{d\tau} = w(X_w(x, s, \tau), \tau) & \text{in } D \times (0, T^*], \\ X_w(x, s, s) = x. \end{cases}
\]

Second, \( w \) is zero on the interface \( \Gamma_0(t) \) pointing away from it with modulus equal to 1; this means that the characteristics of (2.5a) propagate away from the interface \( \Gamma_0(t) \) maintaining fixed its position. Third, the new level set function at time \( t \) is then

\[
(2.7) \quad u(x, t) = d_{\text{steady}}(x, t^*) \quad \text{in } D.
\]

Fourth, there must be a conservation of volume conventionally known as mass conservation, that is, for all \( t \)

\[
(2.8) \quad \int_D H(-u) dx = |D_2| = \text{Constant}.
\]

The numerical solutions to (2.5a)-(2.5b) do not satisfy the fourth item because of the numerical errors; consequently, the conservation property is lost. In order to remedy or alleviate this fact, new numerical procedures have been devised. We shall consider the so-called PLS method combined with a second order QMSL scheme.
3. Numerical method. Following the semi-Lagrangian-level set finite element formulation for incompressible Navier–Stokes equations with free surface of [20], the type of finite element we use to approximate the level set function is the so-called \( P_1 \)-iso\( P_2 \) element. To do so, the domain \( D \) is tessellated to yield a regular quasi-uniform partition \( D_H \) of \( d \)-simplexes \( T_j \), such that if \( NE1 \) is the number of elements of such a partition, then

\[
D \cup \partial D = \bigcup_{j=1}^{NE1} T_j.
\]

Applying a red-green refinement to the partition \( D_H \) we obtain the partition \( D_h \) whose number of elements \( NE2 = 4NE1 \). Figure 1 illustrates graphically the method of generating the partition \( D_h \) from \( D_H \) for a two-dimensional domain \( D \). We associate finite elements spaces \( V_h \) and \( V_H \) with the partitions \( D_h \) and \( D_H \), respectively, defined as

\[
V_h := \{v_h \in C^0(\overline{D}) : v_h|_{T_j} \in P_1(T_j), \ 1 \leq j \leq NE2\},
\]

\[
V_H := \{w_H \in C^0(\overline{D}) : w_H|_{T_k} \in P_2(T_k), \ 1 \leq k \leq NE1\},
\]

where \( P_m(T) \) denotes the set of polynomials of degree less than or equal to \( m \) defined on the simplex \( T \). Thus, any function \( v_h \in V_h \) can be expressed by the formula

\[
v_h = \sum_{i=1}^{NN} V_i \psi_i,
\]

where \( NN \) denotes the number of nodes of the partition \( D_h \), \( V_i = v(x_i) \), \( x_i \) being the \( i \)-th node of \( D_h \), and \( \{\psi_i\}_{i=1}^{NN} \) is the set of global basis functions of the space \( V_h \).

Similarly, any function \( w_H \in V_H \) is of the form

\[
w_H = \sum_{i=1}^{NN} W_i \psi_i,
\]

where \( NN \) denotes the number of nodes of the partition \( D_h \). 

\[\text{Fig. 3.1. Partitions } D_H \text{ and } D_h.\]
where \( \{ \varphi_i \}_{i=1}^{N} \) is the set of global basis functions of \( V_H \). Hereafter, unless otherwise stated, we shall use \( h \) and \( H \) to denote the largest diameter of the elements in the partitions \( D_h \) and \( D_H \), respectively. To calculate the numerical solutions of both the initial boundary value problem (2.3a) and the reinitialization problem (2.5a)–(2.5b), we define in the time interval \([0, T]\) a uniform partition of step \( \Delta t \), \( \mathcal{P}_{\Delta t} := 0 = t_0 < t_1 < \cdots < t_N = T \), such that for all \( n \), \( \Delta t = t_n - t_{n-1} \), and consider the numerical solution of (2.3a) (resp., (2.5a)–(2.5b)) as the mapping \( u_h^n : \mathcal{P}_{\Delta t} \rightarrow V_h \). The procedure to calculate \( u_h^n \) consists of the following steps:

1. Apply the QMSL scheme to calculate \( u_h^n \) as an approximation to the solution of (2.3a).
2. Apply the PLS method to correct \( u_h^n \).
3. Apply the QMSL scheme to calculate the numerical solution of the reinitialization problem (2.5a)–(2.5b).

Next, we describe in detail these steps.

### 3.1. The QMSL method for the level set transport equation.

The semi-Lagrangian method to calculate the numerical solution to (2.3a) is based on the observation that for each time interval \([t_{n-1}, t_n]\), the null variation of the function \( u \) along the characteristics of the material derivative operator, i.e.,

\[
\frac{D u}{D t} = 0, \quad t_{n-1} \leq t \leq t_n,
\]

implies that

\[
u(x, t_n) = u(X(x, t_n; t_{n-1}), t_{n-1}),
\]

where \( X(x, t_n; t_{n-1}) \in \overline{D} \) denotes the position at time \( t_{n-1} \) of a point that moving with velocity \( v \) will reach the point \( x \in \overline{D} \) at time \( t_n \). The points \( X(x, t_n; t_{n-1}) \) are called departure points or feet of the characteristics at time \( t_{n-1} \). Therefore, for each \( x \in \overline{D}, \) \( X(x, t_n; t) \) is the backward solution of the initial value problem

\[
\begin{cases}
\frac{dX(x, t_n; t)}{dt} = v(X(x, t_n; t), t), & t_{n-1} \leq t < t_n, \\
X(x, t_n, t_n) = x.
\end{cases}
\]

Assuming that \( v \in L^\infty(0, T; W^{1,\infty}(D)^d) \), (3.2c) has a unique solution which can be expressed as

\[
X(x, t_n; t) = x - \int_t^{t_n} v(X(x, t_n; \tau), \tau) d\tau.
\]

Setting \( x = x_i \) (the vertices of the elements of the partition \( D_h \)) in (3.2b), we have that the values of the numerical solution at the nodes \( \{ x_i \} \) are then

\[
u_h^n(x_i) = u_h^{n-1}(X_h(x_i, t_n; t_{n-1})),
\]

where \( X_h(x_i, t_n; t_{n-1}) \) denotes the calculated numerical approximation to the exact \( X(x_i, t_n; t_{n-1}) \). Therefore, the calculation of the numerical solution to the pure transport equation (3.2a) by a semi-Lagrangian method consists of two steps: the first one is the calculation for each mesh point \( x_i \) of the departure points \( X_h(x_i, t_n; t_{n-1}) \) by integrating numerically (3.2c) via a Runge–Kutta method of order \( \geq 2 \), or a fixed point method as described in [20]; the second step is the calculation of \( u_h^n(x_i) = u_h^{n-1}(X_h(x_i, t_n; t_{n-1})) \).
3.1.1. Calculation of the solution at the departure points. In \cite{8} and \cite{4} a method is proposed to calculate $u_h^{-1}(X_h(x_i,t_n; t_{n-1}))$, such that this value is close to the one obtained by a piecewise interpolation of order 2 (in this paper) or higher in regions of sufficiently smoothness, and close to that of the piecewise linear interpolation in regions of low smoothness. In this way we have a scheme that satisfies a local maximum principle and is basically oscillation free. This compound scheme is then a blend of piecewise linear and higher order interpolations and is called a discrete QMSL scheme. We say that a numerical method $L$ is discrete monotone if given the mesh functions $U^n := (U^n_i)_{i=1}^{NN}$ and $V^n := (V^n_i)_{i=1}^{NN}$ at time $t_n$, such that for all $i$, $U^n_i \leq V^n_i$, it then holds that at time $t_{n+1}$, $U^n_{i+1} \leq V^n_{i+1}$ for all $i$, where $U^{n+1} := LU^n$ and $V^{n+1} := LV^{n+1}$, respectively; consequently, a monotone method satisfies a maximum principle such as

$$\|V^{n+1}\|_{h,\infty} \leq \|U^{n+1}\|_{h,\infty},$$

where $\|\cdot\|_{h,\infty}$ is the so-called discrete maximum norm defined as $\|U\|_{h,\infty} = \max_i |U_i|$. The QMSL scheme is described as follows.

At time $t_n$, we calculate the values $u_h^n(x_i) := U^n_i$, $1 \leq i \leq NN$, by the formula

\begin{equation}
U^n_i = (1 - \beta^n_i) I_h u_h^{-1}(X_h(x_i,t_n; t_{n-1})) + \beta^n_i I_H u_h^{-1}(X_h(x_i,t_n; t_{n-1})),
\end{equation}

where $I_h : C(\overline{T}) \to V_h$ and $I_H : C(\overline{T}) \to V_H$ are the interpolation operators defined as follows: for all $f(x) \in C(\overline{T})$, $I_h f(x) \in V_h$ such that $I_h f(x_i) = f(x_i)$, $x_i$ being any mesh point; similarly, $I_H f(x) \in V_H$ and $I_H f(x_i) = f(x_i)$. Therefore, since the solution $u_h^n \in V_h$ is by definition a function of $\overline{T}$, then

\begin{equation}
I_h u_h^{-1}(X_h(x_i,t_n; t_{n-1})) = \sum_{i=1}^{NN} U^n_{i-1} \phi_i(X_h(x_i,t_n; t_{n-1}))
\end{equation}

and

\begin{equation}
I_H u_h^{-1}(X_h(x_i,t_n; t_{n-1})) = \sum_{i=1}^{NN} U^n_{i-1} \phi_i(X_h(x_i,t_n; t_{n-1})).
\end{equation}

$I_H u_h^{-1}(X_h(x_i,t_n; t_{n-1}))$ is calculated (in this paper) by a piecewise second degree polynomial so that these values may have an oscillatory behavior with amplitude $O(h^\alpha)$, $0 < \alpha \leq 1$, in a neighborhood of points of strong variation of the solution. The coefficients $\beta^n_i$ are limiting coefficients that suppress the oscillations of $I_H u_h^{-1}$ while trying to maintain the convergence of the piecewise quadratic interpolation in regions where the solution is smooth. The limiting coefficients $\beta^n_i$ are calculated as follows: for each $i$, $1 \leq i \leq NN$, find the elements $T_j \in D_h$ and $T_k \in D_H$, $T_j \subset T_k$, that contain the departure point $X_h(x_i,t_{n+1}; t_{n-1})$, and then calculate

\begin{equation}
\left\{
\begin{array}{l}
U^+ = \max u_h^{-1} |_{Nodes(T_k)} \text{ and } U^- = \min u_h^{-1} |_{Nodes(T_k)}, \\
Q^\pm = U^\pm - I_h u_h^{-1}(X_h(x_i,t_n; t_{n-1})), \\
P = I_H u_h^{-1}(X_h(x_i,t_n; t_{n-1})) - I_h u_h^{-1}(X_h(x_i,t_n; t_{n-1})),
\end{array}
\right.
\end{equation}

where $u_h^{-1} |_{Nodes(T_k)}$ denotes the set of values of $u_h^{-1}$ at the nodes of $T_k$. Note that
if $P \neq 0$, $Q^+ > 0$ and $Q^- < 0$. Next, we set

\[
(3.6b) \quad \beta_i^{n-1} = \begin{cases} 
\min \left(1, \frac{Q^+}{P}\right), & \text{if } P > 0, \\
\min \left(1, \frac{Q^-}{P}\right), & \text{else if } P < 0, \\
1, & \text{else if } P = 0.
\end{cases}
\]

By construction, it is easy to realize, see [4], that $U^0_i$ is giving by the formula

\[
U^0_i = \begin{cases} 
U^+ & \text{if } I_H u_h^{n-1}(X_h(x_i, t_n; t_{n-1})) > U^+, \\
U^- & \text{if } I_H u_h^{n-1}(X_h(x_i, t_n; t_{n-1})) < U^-, \\
I_H u_h^{n-1}(X_h(x_i, t_n; t_{n-1})) & \text{otherwise}.
\end{cases}
\]

Finally, we set

\[
(3.8) \quad u_h^n = \sum_{i=1}^{NN} U^n_i \psi_i.
\]

From a computational point of view the calculation of $U^0_i$ by (3.7) is very economical because it is not necessary to calculate explicitly $\beta_i^{n-1}$ or $I_H u_h^n(X(x_i, t_{n+1}; t_{n-1}))$. The algorithmic version of the QMSL scheme is the following.

**QMSL algorithm.**

Given $\{U_i^{n-1}\}$, $\Delta t$ and $v(x, t)$:

For $i = 1, 2, \ldots, NN$

1. Calculate the departure points $X_h(x_i, t_n; t_{n-1})$ by solving numerically (3.2c), and using a search-locate algorithm as described in [2] to identify the elements $T_k \in D_H$, where $X_h(x_i, t_n; t_{n-1})$ are located.
2. For each $T_k$, calculate $I_H u_h^{n-1}(X_h(x_i, t_n; t_{n-1}))$, and $U^+$ and $U^-$. 
3. Calculate $U^n_i$ by (3.7) and $u_h^n(x)$ by (3.8).

**3.2. The PLS method.** For the sake of completeness we furnish a brief description of the method based on the one presented in [16]. At the initial time $t_0$, two sets of massless marked particles are randomly distributed in a narrow band $\Sigma_\beta$ of radius $\beta h$, $1 \leq \beta \leq 3$, around (the numerically approximated) interface $u_h^0(x) = 0$; positive (negative) particles are those which are located in the region $u_h^0 > 0$ ($u_h^0 < 0$). At each time instant $t$, the band $\Sigma_\beta$ is defined by

\[
\Sigma_\beta = \{x \in D : 0 \leq \min_{y \in T_{h0}} |x - y| \leq \beta h\}.
\]

The elements of $\Sigma_\beta$ will have a number $n_\rho$ of particles of each sign, usually, $n_\rho = 32$ (64) for two-dimensional (three-dimensional) problems with uniform squared grids. At each time instant $t_n$, a particle is characterized by its position $x^n_\rho$ and its radius $r^n_\rho$.

The position $x^n_\rho$ is calculated by integrating the initial value problem

\[
(3.9) \quad \begin{cases} 
\frac{dx_\rho(t)}{dt} = v(x_\rho(t), t), & t_{n-1} < t \leq t_n, \\
x_\rho(t_{n-1}) = x_\rho^{n-1} \text{ is a datum},
\end{cases}
\]
and the radius $r^n_p$ is defined as
\begin{equation}
(3.10) \\
r^n_p = \begin{cases} 
  r_{\text{max}} & \text{if } s^n_p u^n_h(x^n_p) > r_{\text{max}}, \\
  s^n_p u^n_h(x^n_p) & \text{if } r_{\text{min}} \leq s^n_p u^n_h(x^n_p) < r_{\text{max}}, \\
  r_{\text{min}} & \text{otherwise},
\end{cases}
\end{equation}

where $s^n_p = \text{sign } u^n_h(x^n_p)$, i.e., $s^n_p = 1$ if $u^n_h(x^n_p) > 0$, $s^n_p = -1$ if $u^n_h(x^n_p) < 0$, and $s^n_p = 0$ if $u^n_h(x^n_p) = 0$; $r_{\text{min}} = 0.01h$ and $r_{\text{max}} = 0.05h$. An important concept of the method is that of an escaped particle. An escaped particle is one that crosses the interface by a distance larger than its radius $r^n_p$. To each escaped particle is assigned a particle level set function $u^n_{hp}(x)$, which is used for the error correction on the interface, defined as
\begin{equation}
(3.11) \\
u^n_{hp}(x) = s^n_p (r^n_p - |x - x^n_p|).
\end{equation}

$u^n_{hp}(x)$ is locally computed at the vertices of the element that contains the escaped particle. These local values of $u^n_{hp}(x)$ are the particle predictions of the values of the overall level set function $u^n_h(x)$ at such vertices.

Let $E^+$ and $E^-$ be the sets of escaped positive and negative particles, respectively, at time instant $t_n$, and an estimate of the corrected level set function in the $u^n_h(x) > 0$ region is the following:
\begin{equation}
(3.12a) \\
u^n_+(x) = \max_{p \in E^+} \left( u^n_{hp}(x), u^n_h(x) \right).
\end{equation}

Similarly for $u^n_h(x) < 0$,
\begin{equation}
(3.12b) \\
u^n_-(x) = \min_{p \in E^-} \left( u^n_{hp}(x), u^n_h(x) \right).
\end{equation}

$u^n_+$ and $u^n_-$ in the above two equations are initialized with $u^n_h$. Finally, the corrected level set function is
\begin{equation}
(3.12c) \\
u^n_h(x) = \begin{cases} 
  u^n_+(x) & \text{if } |u^n_+(x)| \leq |u^n_-(x)|, \\
  u^n_-(x) & \text{if } |u^n_+(x)| > |u^n_-(x)|.
\end{cases}
\end{equation}

The particles which remain escaped have their radius set to $r_{\text{min}}$; the latter operation is called radii adjustment.

The PLS method has the following algorithmic formulation.

**PLS algorithm.**

Choose the parameters $\beta$, $n_p$, $r_{\text{min}}$, and $r_{\text{max}}$

1. At $t = 0$:
   1.1. Distribute randomly $n_p$ particles in a band of radius $\beta h$ around the zero level set $u_h(x, 0) = 0$.
   1.2. For each particle $p$ find its position $x^n_p$, calculate its radius $r^n_p$ using (3.10), and identify whether it is $+$ or $-$.

2. At $t_n$, assuming $u^n_h$ is known:
   2.1. For each particle $p$ calculate $x^n_p$ by integrating numerically (3.9) and $r^n_p$ by using (3.10), and define the sets $E^+$ and $E^-$. 
   2.2. (Quantify the error) For each particle $p$ calculate $u^n_{hp}(x)$ at the vertices of the element that contains such a particle by applying (3.11).
   2.3. (Error correction) Calculate the new $u^n_h(x)$ by applying the formulae (3.12a)–(3.12c).
   2.4. If necessary, do reseeding; see [16].
3.3. The QMSL method for the reinitialization equation. Based on section 2.1, the reinitialization process is carried out at each time $t_n$ by a mixed procedure consisting of calculating directly the distance to the zero level set $\Gamma_{\mathcal{h}0}(t_n)$ of the mesh points which are inside of a neighborhood $\Sigma_{\mathcal{h}0}(t_n)$ of it, and solving the PDE (2.5a) in the rest of $D$; here, $\Gamma_{\mathcal{h}0}(t_n)$ denotes the numerical approximation to the exact zero level set $\Gamma(t_n)$. Therefore, the steps of the reinitialization process are the following. First, we construct the band $\Sigma_{\mathcal{h}0}(t_n)$ defined by

$$\Sigma_{\mathcal{h}0}(t_n) = \left\{ \bigcup_k T_k, \ T_k \in D_h : T_k \cap \Gamma_{\mathcal{h}0}(t_n) \neq \emptyset \right\}.$$ 

Figure 3.2 shows graphically a piece of $\Sigma_{\mathcal{h}0}(t_n)$. The construction of $\Sigma_{\mathcal{h}0}(t_n)$ is easy and fast to do in a finite element context because, starting from $\Sigma_{\mathcal{h}0}(t_{n-1})$, we identify elements $T_k$ by inspection of the signs of $u^n_h$ at the vertices taking into account that the zero level set function intersects a triangle if the sign of $u^n_h$ at one vertex is different from the signs at the other two; also it may happen that the sign of $u^n_h$ at one vertex is zero and its signs at the other two are + and −, respectively, and then we calculate $\Gamma_{\mathcal{h}0}(t_n)$ by linear interpolation. Second, let $\{x_i\}$ be the set formed by the vertices of the elements $T_k$ of $\Sigma_{\mathcal{h}0}(t_n)$; we calculate the signed distance to $\Gamma_{\mathcal{h}0}(t_n)$ for each point $x_i$ either by geometric or optimization procedures and denote by $D_i$ such a distance. For orthogonal quadrilateral meshes, Chopp [9] proposes, in the framework of the fast marching method, a second order method by combining a bicubic interpolation procedure with a Newton method to calculate $\Gamma_{\mathcal{h}0}(t_n)$ and the distance $D_i$. Third, we solve numerically (2.5a) in $D \backslash \Sigma_{\mathcal{h}0}(t_n)$ using $D_i$ as prescribed values at the nodes $x_i$ of $\Sigma_{\mathcal{h}0}(t_n)$. To do so, we define $\Omega_1 := D \backslash (D_2 \cup \Sigma_{\mathcal{h}0}(t_n))$ and $\Omega_2 := D \backslash (D_1 \cup \Sigma_{\mathcal{h}0}(t_n))$, and for each pseudo time interval $[\tau_{m-1}, \tau_m]$, $1 \leq m \leq m_1$, recast (2.5a) as

$$d(x, \tau_m) = d(X_w(x, \tau_m; \tau_{m-1}), \tau_{m-1}) + \begin{cases} \Delta \tau & \text{if } x \in \Omega_1, \\ -\Delta \tau & \text{if } x \in \Omega_2 \end{cases}$$

with the initial condition $d(x, \tau_0) = u^n_h(x)$. Here, $X_w(x, \tau_m; \tau_{m-1})$ denotes the departure point of the trajectory described by a particle that moving with velocity $w$ will reach the point $(x, \tau_m)$. $X_w(x, \tau_m, \tau)$ is the solution of the initial value problem (2.6d) for $\tau_{m-1} \leq \tau < \tau_m$. To calculate the finite element solution we approximate $d(X_w(x, \tau_m; \tau_{m-1}), \tau_{m-1})$ and $d(x, \tau_m)$ by the functions $\overline{d}^{m-1}_h$ and $\overline{d}^{m}_h \in V_h$, respectively; hence,

$$\overline{d}^{m-1}_h = \sum_{i=1}^{NN} \overline{D}^{m-1}_i \psi_i$$
with the particularity that we calculate $\overline{D}_i^{m-1}$ as

$$\overline{D}_i^{m-1} = QMSLd_h^{m-1}(X_{hw}(x_i, \tau; \tau_{m-1})), \quad (3.14)$$

where $X_{hw}(x_i, \tau; \tau_{m-1})$ represents the numerical approximation to $X_w(x_i, \tau; \tau_{m-1})$ calculated by solving (2.6d) with the same numerical method used for the calculation of $X_h(x_i, t_n; t_{n-1})$ and $QMSLd_h^{m-1}(X_{hw}(x_i, \tau; \tau_{m-1}))$ denotes the value of $d_h^{m-1}(X_{hw}(x_i, \tau; \tau_{m-1}))$ calculated by the QMSL algorithm. The function $d_h^m(x)$ is of the form

$$d_h^m = \sum_{i=1}^{NN} D_i^m \psi_i, \quad (3.15)$$

with the conditions that at the nodes $x_i$ of $\Sigma_{G_h(t_n)}$, $D_i^m = D_i$, and (the initial condition) at $\tau = 0$, $d_h^m = u_h^n$. The finite element formulation of (3.13) is as follows.

Find $d_h^m \in V_h$, satisfying the above conditions, such that for all $v_h \in V_h$

$$\int_{\Omega_j} d_h^m v_h dx = \int_{\Omega_j} \overline{T}_h^m v_h dx + (-1)^{j+1} \Delta \tau \int_{\Omega_j} v_h dx, \quad j = 1, 2. \quad (3.15)$$

Writing this equation in algebraic form we have for each $j$ a linear system of equations

$$M D^m = M \overline{D}^{m-1} + \Delta \tau S,$$

where $M$ is the so-called mass matrix, which is symmetric and positive definite, and whose entries $m_{ik}$ are given by

$$m_{ik} = \int_{\Omega_j} \psi_i \psi_k dx.$$

$S$, $\overline{D}^{m-1}$, and $D^m$ are vectors with entries

$$S_i = (-1)^{j+1} \int_{\Omega_j} \psi_i dx,$$

and $\overline{D}_i^{m-1}$ are given by (3.14). Noting that $M^{-1}S = 1 := [1, \ldots, 1]^T$, it follows that for $m = 1, 2, \ldots, m_1$

$$\begin{cases} 
D_i^m = \overline{D}_i^{m-1} + \Delta \tau & \text{if } x_i \in \Omega_1, \\
D_i^m = \overline{D}_i^{m-1} - \Delta \tau & \text{if } x_i \in \Omega_2, \\
D_i^m = D_i & \text{if } x_i \in \Sigma_{G_h(t_n)}.
\end{cases} \quad (3.16)$$

When $m = m_1$, the new level set function at time $t_n$ is then

$$u_n^h = d_h^{m_1}. \quad (3.17)$$

Several remarks are now in order.

**Remark 3.1.** The band $\Sigma_{G_h(t_n)}$ will be well defined if the mesh is sufficiently fine, in particular in regions of high curvature, such that in 2D the intersection of the zero level set with the elements $T_k$ is either a vertex point or a straight segment, whereas in 3D such an intersection is either a vertex point or a triangle or a quadrilateral.
Remark 3.2. In the reinitialization procedure the velocity \( w \) given by (2.5b) depends, at each \( \tau_m \), upon the solution \( d \), and therefore, we cannot use the exact \( w \) to calculate the departure points \( X_w(x, \tau_m; \tau_{m-1}) \); instead, we shall use an approximate \( w_h^m \) calculated via the approximate solution \( d_h^m \in V_h \). The numerical solutions of (3.13) are calculated in \( \Omega_1 \) and \( \Omega_2 \), so then by virtue of (2.4a) and (2.5b) we can set

\[
(3.18) \quad w_h^m(x) = \begin{cases} 
   n_h^m(x) & \text{if } x \in \Omega_1, \\
   -n_h^m(x) & \text{if } x \in \Omega_2,
\end{cases}
\]

where \( n_h^m \) is an approximate normal vector to the level sets of \( d_h^m \) in \( D \). Noting that \( \nabla d_h^m \) is a piecewise constant function in \( D \), a direct application of the formula

\[
\frac{\nabla d_h^m}{|\nabla d_h^m|}
\]

yields a piecewise constant normal vector \( n_h^m \) in \( D \); consequently, the numerical solution of (2.6d) at each \( m \) would not be a unique solution in \( L^\infty((\Omega_j \times (0, T^*))^d) \), \( j = 1 \) and \( 2 \). To remedy this fact, we shall calculate a vector \( n_h^m \in V_h \) by using the orthogonal \( L^2 \) projection of \( \frac{\nabla d_h^m}{|\nabla d_h^m|} \) onto \( V_h \) restricted to the domains \( \Omega_1 \) and \( \Omega_2 \). Thus, setting for each \( m \)

\[
n_h^m = \sum_k N_h^m \psi_k \quad \text{in } \Omega_1 \text{ and } \Omega_2,
\]

the coefficients \( N_h^m = (N_h^m_1, N_h^m_2) \), which are the components of \( n_h^m \) at the mesh points \( x_k \) in \( \Omega_1 \) and \( \Omega_2 \), are such that for all \( k \)

\[
\int_{\Omega_j} \left( n_h^m - \frac{\nabla d_h^m}{|\nabla d_h^m|} \right) \psi_k dx = 0.
\]

For each \( j \), this equation yields the algebraic linear system of equations

\[
MN^m = R^m,
\]

where \( M \) is the mass matrix restricted to \( \Omega_j \) and \( R^m \) is a vector whose entries are the values of \( \int_{\Omega_j} \frac{\nabla d_h^m}{|\nabla d_h^m|} \psi_k dx \). Observing that in each element \( T_i \), \( \frac{\nabla d_h^m}{|\nabla d_h^m|} \mid_{T_i} \) is constant and \( \psi_k \) is piecewise linear, this integral can be calculated exactly by the same quadrature rule as the one used above; thus, applying such a rule gives for each \( k \)

\[
R_k^m = \frac{1}{3} \sum_i |T_i| \frac{\nabla d_h^m}{|\nabla d_h^m|} \mid_{T_i},
\]

where \( T_i \) are the elements that form the support of \( \psi_k \). Lumping the matrix \( M \) it follows that the components of the approximate normal vector at the mesh points of \( \Pi_1 \) and \( \Pi_2 \) are given by

\[
N_h^m = \sum_i |T_i| \sum_i |T_i| \frac{\nabla d_h^m}{|\nabla d_h^m|} \mid_{T_i}.
\]

To calculate \( n_h^m(x) \) at the mesh points \( \{x_i\} \) of \( \Sigma_{r_0}(t_n) \) we apply this formula with the elements \( T_i \) being the support of the nodal basis functions \( \psi_i(x) \mid_{\Sigma_{r_0}(t_n)} \). In [27] the following result is proved.

Lemma 3.3. Assuming that the interface \( \Gamma_0(t) \in C^3 \) and \( u(x, t) \in L^\infty(0, T; W^{3, \infty} (D)) \), we have that for all \( t_n \) there exists a constant \( C \) independent of \( h \) and \( \Delta t \) such that

\[
\|n^n - n_h^n\|_{L^\infty(\Gamma_0(t_n))} \leq Ch^2 \|u\|_{L^\infty(0, T; W^{3, \infty}(D))}.
\]
The algorithmic version of the QMSL-PLS method is as follows.

**QMSL-PLS algorithm with reinitialization.**

Choose the parameters $\beta$, $n_p$, $r_{\text{max}}$, $r_{\text{min}}$ and $m_1$. $u_h^0$ and $v_h^0$ are data

For $n = 1, 2, \ldots, N$

1.1 Apply the QMSL algorithm to calculate $u_h^n$ in $D$.

1.2 Apply the PLS algorithm to correct $u_h^n$.

1.3 Reinitialization stage

1.3.1 Find the zero level set $\Gamma_{h_0}(t_n)$ and calculate the band $\Sigma_{\Gamma_{h_0}(t_n)}$ and the domains $\Omega_1 = D - (D_2 \cup \Sigma_{\Gamma_{h_0}(t_n)})$ and $\Omega_2 = D - (D_1 \cup \Sigma_{\Gamma_{h_0}(t_n)})$.

1.3.2 Calculate the signed distances $\{D_i\}$ to $\Gamma_{h_0}(t_n)$ at the mesh points $\{x_i\}$ in the band $\Sigma_{\Gamma_{h_0}(t_n)}$, and set

$$
D_i^0 = \begin{cases} 
U^n_i & \text{for } x_i \in \Omega_1 \cup \Omega_2, \\
D_i & \text{for } x_i \in \Sigma_{\Gamma_{h_0}(t_n)}. 
\end{cases}
$$

1.3.3 Calculate $N_i^0$ by applying (3.19) for all the mesh points $x_i$ and calculate $u_h^n$ by the formula (3.18).

1.3.4 For $m = 1, 2, \ldots, m_1$

Calculate $D_i^{m-1}$ by applying the QMSL algorithm to $d_h^{m-1}$ in $\Omega_1$ and $\Omega_2$.

Calculate $D_i^m$ for the mesh points $x_i$ in $\Omega_1$ and $\Omega_2$ by applying (3.16).

Calculate $N_i^m$ by applying (3.19) and $w_h^m$ by the formula (3.18).

1.4 When $m = m_1$, set $w_h^m = d_h^{m_1}$

**4. Analysis.** For the analysis we shall employ the maximum mesh-dependent norm $\| \cdot \|_{h, \infty}$ which is defined by

$$
\|v\|_{h, \infty} = \max_i |v(x_i)|, \quad v \in C(D).
$$

We show that for all $v_h \in V_h$ the maximum mesh dependent norm is equivalent to the $L^\infty$-norm, that is,

$$
\|v_h\|_{h, \infty} \leq \|v_h\|_{L^\infty(D)} \leq \|v_h\|_{h, \infty}
$$

because

$$
\|v_h\|_{L^\infty(D)} = \left\| \sum_i V_i \psi_i(x) \right\|_{h, \infty} \leq \max_i |V_i| \sum_i |\psi_i(x)| = \max_i |V_i| = \|v_h\|_{h, \infty}.
$$

Here we have used the following properties of the basis functions $\psi_i(x)$: (1) for all $i$ and $x$, $1 \geq \psi_i(x) \geq 0$, and (2) $\sum_i \psi_i(x) = 1$; but since $x_i$ is in $D$,

$$
\max_i |V_i| \leq \|v_h\|_{L^\infty(D)}.
$$

**4.1. Stability in the $L^\infty$-norm.** We shall prove the following result.

**Lemma 4.1.** Let $\Delta t \in (0, h_0)$ and $h \in (0, h_0)$, $0 < \Delta t < 1$ and $0 < h < 1$. Then for any $t_n \in [0, T]$ the solution obtained by the quasi-monotone algorithm satisfies

$$
\|u_h^n\|_{L^\infty(D)} \leq \|u_h^0\|_{L^\infty(D)}.
$$
Proof. Let \( k \) be an index such that \( \| u_h^{n-1} \|_{h,\infty} = |U_k^{n-1}| \), and then by construction it follows that there is an index \( l \) such that
\[
\| u_h^l \|_{h,\infty} = |U_l^{n-1}|;
\]
hence for all \( n \),
\[
\| u_h^n \|_{h,\infty} \leq \| u_h^{n-1} \|_{h,\infty}.
\]
Since
\[
\| u_h^n \|_{L^\infty(D)} \leq \| u_h^n \|_{h,\infty} \leq \| u_h^{n-1} \|_{h,\infty} \leq \cdots \leq \| u_h^0 \|_{h,\infty} \leq \| u^0 \|_{L^\infty(D)},
\]
the result follows. \( \square \)

Note that \( \| u_h^n \|_{h,\infty} \leq \| u_h^{n-1} \|_{h,\infty} \) means that (see (3.4))
\[
(4.1) \quad \max_i |(1 - \beta_i^{n-1}) I_h u_h^{n-1}(X_h(x_i, t_n; t_{n-1})) + \beta_i^{n-1} I_H u_h^{n-1}(X_h(x_i, t_n; t_{n-1}))| \\
\leq \max_i |U_i^{n-1}|.
\]

4.2. Error analysis of the approximate level set solution. In this section we estimate the error in the mesh dependent norm \( \| \|_{h,\infty} \) for the QMSL solutions (3.3) and (3.17) considering that the departure points are the exact ones, i.e., \( X(x, t_n; t_{n-1}) \).

First, we state the following results assuming that (i) \( \text{div} v = 0 \text{ a.e.} \) and (ii) either \( v \mid_{\partial D} = 0 \) or \( n \cdot v \mid_{\partial D} = 0 \).

Lemma 4.2. Assume that \( v \in L^\infty(0; T; W^{1,\infty}(D)') \) and \( s - \tau \) is sufficiently small, then \( x \in D \rightarrow X(x, s; t) \) is a quasi-isometric homeomorphism of \( D \) onto \( D \) and its Jacobian determinant \( J = 1 \text{ a.e.} \) in \( D \). Moreover,
\[
(4.2) \quad K_v^{-1} |x - z| \leq |X(x, s; \tau) - X(z, s; \tau)| \leq K_v |x - z|,
\]
where \( K_v = \exp (s - \tau) |\nabla v|_{L^\infty(0; T; L^\infty(D)')} \) and \( |a - b| \) denotes the Euclidean distance between the points \( a \) and \( b \) in \( \mathbb{R}^d \).

For a proof of this lemma see [31]. In the following lemma we collect some facts concerning the solution of (3.2c) which are standard in the theory of ODE systems.

Lemma 4.3. Assume that \( v \in L^\infty(0, T; W^{r+1,\infty}(D)') \), \( r \geq 0 \). Then for any integer \( n \), \( 0 \leq n \leq N - 1 \), the unique solution \( t \rightarrow X(x, t_n; t) \) of (3.2c) is such that \( X(x, t_n; t) \in W^{r+1,\infty}(0, T; W^{r+1,\infty}(D)') \). Furthermore, let the multi-index \( \alpha \in \mathbb{N}^d \), then for all \( \alpha \) such that \( 1 \leq \alpha \leq k \), \( \partial_{x_j}^\alpha X(x, t_n; t) \in C([0, T]; L^\infty(D \times [0, T])) \), \( 1 \leq i, j \leq d \).

We denote by \( \overline{\eta}_h^n \) the numerical solution (3.3) calculated by application of the QMSL algorithm, and by \( u_h^n \) the numerical solution after the application of the QL and QMSL reinitialization algorithms to \( \overline{\eta}_h^n \). In this way, at time \( t_n \) we have two error functions, namely, \( e^n \) and \( \overline{\eta}^n \), defined as
\[
e^n = u^n - u_h^n \quad \text{and} \quad \overline{\eta}^n = u^n - \overline{\eta}_h^n.
\]
To estimate \( \| e^n \|_{h,\infty} \) we use the ansatz
\[
(4.3) \quad \| e^n \|_{h,\infty} \leq \gamma^n \| \overline{\eta}^n \|_{h,\infty},
\]
where \( \gamma^n \) are positive real numbers \( \leq 1 \). The ansatz makes sense because one observes experimentally that \( u_h^n \) approximates \( u^n \) better than \( \overline{\eta}_h^n \) does. Next, assuming that for all \( n \), \( u^n \in C(D) \), we can write that
\[
\| e^n \|_{h,\infty} = \| I_h u^n - u_h^n \|_{h,\infty} \quad \text{and} \quad \| \overline{\eta}^n \|_{h,\infty} = \| I_h u^n - \overline{\eta}_h^n \|_{h,\infty}
\]
because at the mesh points \( \{x_i\} \), \( u^n(x_i) = I_hu^n(x_i) = u^{n-1}(X(x_i, t_n; t_{n-1})) \). \( V^n(x_i) = u^{n-1}(X(x_i, t_n; t_{n-1})) \) is calculated by the formula (3.4), where the coefficients \( \beta_{i}^{n-1} \) are the largest possible values that minimize \( \|u^n - V^n\|_{h, \infty} \) while \( V^n \) satisfies the maximum principle locally; hence, if we take other limiting coefficients, say, \( \alpha_{i}^{n-1} \), such that for all \( i \) and \( n \), \( 0 \leq \alpha_{i}^{n-1} \leq \beta_{i}^{n-1} \leq 1 \), and denote by \( V^n_i \in V_h \) the function whose mesh point values \( U_i \) are calculated by the formula

\[
(4.4a) \quad \overline{U}^n_i = (1 - \alpha_{i}^{n-1})I_h u^{n-1}_h(X(x_i, t_n; t_{n-1})) + \alpha_{i}^{n-1}I_H u^{n-1}_h(X(x_i, t_n; t_{n-1})) ,
\]

we have that

\[
(4.4b) \quad \|u^n - \overline{U}^n\|_{h, \infty} \leq \|u^n - V^n\|_{h, \infty} .
\]

Let \( \tilde{\beta}_{i}^{n-1} \) and \( \tilde{\alpha}_{i}^{n-1} \) be the limiting coefficients for \( u^{n-1}(X(x, t_n; t_{n-1})) - u^{n-1}_h(X(x, t_n; t_{n-1})) \) and \( u^{n-1}(X(x, t_n; t_{n-1})) \), respectively, and define \( \alpha_{i}^{n-1} \) as

\[
\alpha_{i}^{n-1} = \min(\tilde{\beta}_{i}^{n-1}, \tilde{\alpha}_{i}^{n-1}, \beta_{i}^{n-1}),
\]

and then we obtain the following estimate.

**Theorem 4.4 (error estimate with exact departure points).** Assume that for all \( n \) the ansatz (4.3) holds and \( u^n \in W^{r+1, \infty}(D) \), \( r \geq 0 \). Then there exist positive constants \( C_4 \) and \( K \), \( C_4 \) being independent of \( \Delta t \) and \( h \), and \( 0 < K \leq 1 \), such that for \( p = \min(2, r + 1) \) and \( q = \min(3, r + 1) \)

\[
\|e^n\|_{h, \infty} \leq \frac{Kt_n}{\Delta t} \min\left(1, \frac{\Delta t \|v\|_{L^\infty((0, t_n) \times D)^d}}{h}\right) \times C_4 \left[ \max_{i, n}(1 - \alpha_{i}^{n-1})h^p + h^q \right] |v|_{L^\infty((0, t_n); W^{r+1, \infty}(D))} .
\]

**Proof.** From (4.4b) it follows that

\[
\|u^n - V^n_i\|_{h, \infty} \leq \max_i \left| u^n_i - \overline{U}^n_i \right| ,
\]

and the right-hand side of this inequality is bounded as follows. First, we note that

\[
(4.7a) \quad \begin{cases}
  u^n_i = u^{n-1}(X(x_i, t_n; t_{n-1})) = (1 - \alpha_{i}^{n-1})u^{n-1}_i(X(x_i, t_n; t_{n-1})) \\
  + \alpha_{i}^{n-1}u^{n-1}_h(X(x_i, t_n; t_{n-1})) \\
  - [(1 - \alpha_{i}^{n-1})I_h u^{n-1}_h(X(x_i, t_n; t_{n-1})) + \alpha_{i}^{n-1}I_H u^{n-1}_h(X(x_i, t_n; t_{n-1}))] \\
  + [(1 - \alpha_{i}^{n-1})I_h u^{n-1}_h(X(x_i, t_n; t_{n-1})) + \alpha_{i}^{n-1}I_H u^{n-1}_h(X(x_i, t_n; t_{n-1}))].
\end{cases}
\]

Second, let \( I \) be the identity operator, and then we can set

\[
(4.7b) \quad \begin{cases}
  u^n_i - \overline{U}^n_i = (1 - \alpha_{i}^{n-1})(I - I_h)u^{n-1}_h(X(x_i, t_n; t_{n-1})) \\
  + \alpha_{i}^{n-1}(I - I_H)u^{n-1}_h(X(x_i, t_n; t_{n-1})) \\
  + (1 - \alpha_{i}^{n-1})I_h(u^{n-1}_h(X(x_i, t_n; t_{n-1})) - u^{n-1}_h(X(x_i, t_n; t_{n-1}))) \\
  + \alpha_{i}^{n-1}I_H(u^{n-1}_h(X(x_i, t_n; t_{n-1})) - u^{n-1}_h(X(x_i, t_n; t_{n-1}))).
\end{cases}
\]

Next, we apply approximation theory to bound the first and second terms of (4.7b), and bound the third and fourth terms by the stability result, see (4.1), considering that
for all \( i, \alpha_i^{n-1} \leq \mu_i^{n-1} \). Then we have that for \( p = \min(2, r + 1) \) and \( q = \min(3, r + 1) \)

\[
\| \tilde{v}^n \|_{h, \infty} \leq \| u^n - \tilde{u}^n_h \|_{h, \infty} + C_4 \max_{i,n} (1 - \alpha_i^{n-1}) h^p |u_i^{n-1}|_{W^{r+1, \infty}(D)} + C_4 h^q |u_i^{n-1} - u_h^{n-1}|_{h, \infty},
\]

where \( C_4 \) is the approximation constant. Hence, for all \( n \) it follows that

\[
\| \tilde{r}^n \|_{h, \infty} \leq \max_{i,n} (1 - \alpha_i^{n-1}) h^p + h^q |u_i^{n-1}|_{W^{r+1, \infty}(D)},
\]

where \( \tilde{r}^n \) is the approximation constant. Hence, for all \( n \) it follows that

\[
\| e^n \|_{h, \infty} \leq \frac{K t_n}{\Delta t} C \left[ \max_{i,n} (1 - \alpha_i^{n-1}) h^p + h^q \right] |u|_{l^{\infty}(0,t_n;W^{r+1, \infty}(D))},
\]

because if all \( \gamma^i \) were equal to 1, then \( K^* = n = \frac{t_n}{\Delta t} \), and therefore, \( 0 < K^* \leq \frac{t_n}{\Delta t} \) and there exists a positive constant \( K, 0 < K \leq 1 \), such that \( K^* = \frac{K t_n}{\Delta t} \). This way of arguing is valid when \( \Delta t = O(h) \), but when \( \Delta t \) is much smaller, and eventually \( \Delta t \to 0 \), this procedure yields an estimate that would be unbounded. To estimate \( \| u^n - u_h^n \|_{h, \infty} \) when \( \Delta t \) is much smaller than \( h \) we note two items. First, setting \( \rho_1^n = u^n - I_h u^n \) and \( \rho_2^n = u^n - I_H u^n \), so that \( \rho_1^n(x_i) = \rho_2^n(x_i) = 0 \), we can write

\[
(1 - \alpha_i^{n-1})(I - I_h)u_i^{n-1}(X(x_i, t_n; t_{n-1})) + \alpha_i^{n-1}(I - I_H)u_i^{n-1}(X(x_i, t_n; t_{n-1})) = (1 - \alpha_i^{n-1}) \left( \rho_1^{n-1}(X(x_i, t_n; t_{n-1})) - \rho_1^{n-1}(x_i) \right) + \alpha_i^{n-1} \left( \rho_2^{n-1}(X(x_i, t_n; t_{n-1}) - \rho_2^{n-1}(x_i) \right),
\]

and hence,

\[
\left\{ \begin{array}{l}
\quad u_i^n - U_i^n = (1 - \alpha_i^{n-1}) \left( \rho_1^{n-1}(X(x_i, t_n; t_{n-1})) - \rho_1^{n-1}(x_i) \right) \\
\quad + \alpha_i^{n-1} \left( \rho_2^{n-1}(X(x_i, t_n; t_{n-1}) - \rho_2^{n-1}(x_i) \right) \\
\quad + (1 - \alpha_i^{n-1}) I_h(u_i^{n-1}(X(x_i, t_n; t_{n-1}) - u_h^{n-1}(X(x_i, t_n; t_{n-1})) \\
\quad + \alpha_i^{n-1} I_H(u_i^{n-1}(X(x_i, t_n; t_{n-1}) - u_h^{n-1}(X(x_i, t_n; t_{n-1}))).
\end{array} \right.
\]

Now, arguing as above it readily follows that

\[
\| u^n - u_h^n \|_{h, \infty} \leq \max_{i,n} (1 - \alpha_i^{n-1}) \left( \| \rho_1^{n-1}(X(x, t_n; t_{n-1})) - \rho_1^{n-1}(x) \|_{h, \infty} \\
\quad + \| \rho_2^{n-1}(X(x, t_n; t_{n-1}) - \rho_2^{n-1}(x) \|_{h, \infty} + \| u^{n-1} - u_h^{n-1} \|_{h, \infty}.
\]

To bound the first and second terms on the right-hand side of this inequality, we use the fact that for \( j = 1, 2 \)

\[
| \rho_j^{n-1}(X(x, t_n; t_{n-1}) - \rho_j^{n-1}(x) | = \left| \int_{t_{n-1}}^{t_n} \frac{d \rho_j^{n-1}(X(x, t_n; t))}{dt} dt \right|,
\]
and hence,
\[ |\rho_j^{n-1}(X(x, t_n; t_{n-1})) - \rho_j^{n-1}(x)| \leq \Delta t \|v\|_{L^\infty((0,T) \times D)} \|\nabla \rho_j\|_{L^\infty((t_{n-1}, t_n) \times D)} . \]

Then, noting that
\[ \|\rho_j^{n-1}(X(x, t_n; t_{n-1})) - \rho_j^{n-1}(x)\|_{h, \infty} \leq \|\rho_j^{n-1}(X(x, t_n; t_{n-1})) - \rho_j^{n-1}(x)\|_{L^\infty(D)} \]
and using approximation theory to bound the term \( \|\nabla \rho_j\|_{L^\infty((t_{n-1}, t_n) \times D)} \), it follows that for \( p = \min(2, r + 1) \) and \( q = \min(3, r + 1) \)
\[ \|\rho_1^{n-1}(X(x, t_n; t_{n-1})) - \rho_1^{n-1}(x)\|_{h, \infty} \leq C \Delta t \|v\|_{L^\infty((0,T) \times D)} h^{p-1} \|u^{n-1}\|_{W^{r+1, \infty}(D)} \]
and
\[ \|\rho_2^{n-1}(X(x, t_n; t_{n-1})) - \rho_2^{n-1}(x)\|_{h, \infty} \leq C \Delta t \|v\|_{L^\infty((0,T) \times D)} h^{q-1} \|u^{n-1}\|_{W^{r+1, \infty}(D)} . \]

Since \( \|u^n - u_h^n\|_{h, \infty} \leq \|u^n - u_h^n\|_{h, \infty} \), we now have that for all \( n \)
\[ \|e^n\|_{h, \infty} \leq \|e^{n-1}\|_{h, \infty} + C \max_i (1 - \alpha_i^{n-1}) h^p + h^q \]
\[ \times \frac{\Delta t \|v\|_{L^\infty((0,T) \times D)} h}{h} \|u^{n-1}\|_{W^{r+1, \infty}(D)} , \]
and arguing as above we obtain that
\[ \|e^n\|_{h, \infty} \leq C_4 \frac{Kt_n}{\Delta t} \left( \frac{\Delta t \|v\|_{L^\infty((0,T) \times D)}}{h} \right) \]
\[ \times \left[ \max_n \max_i (1 - \alpha_i^{n-1}) h^p + h^q \right] \|u\|_{L^\infty(0,T; W^{r+1, \infty}(D))} . \]

Since \( \|e^n\|_{h, \infty} \) satisfies both (4.8) and (4.10), it satisfies their minimum.  

5. Numerical tests.  We study the performance of the QMSL-PLS method in benchmark problems such as (1) Zalesak’s slotted disk, (2) the single vortex flow, and (3) the rising bubble.
5.1. Zalesak’s slotted disk. This is a test to measure the ability of the numerical schemes to deal with sharp discontinuities. The problem is defined by

\[
\begin{aligned}
\frac{\partial u}{\partial t} + v \cdot \nabla u &= 0 \quad \text{in } (0,1)^2 \times (0,T], \\
u(x,0) &= u^0(x) = \pm \min_{y \in \Gamma_0} |x-y|,
\end{aligned}
\]

where \(\Gamma_0(0)\) is the zero level set at time \(t=0\) and is represented by the boundary of a circle of radius 0.15 centered at \((0.5,0.75)\) with a slot of depth 0.25 and width 0.05. The stationary velocity field \(v\) is given by

\[
v_1(x_1,x_2) = 0.5 - x_2, \quad v_2(x_1,x_2) = x_1 - 0.5
\]

and represents a divergence-free rotating flow.

In this test we have used three meshes, specifically, mesh \(M_1\) with 50 square cells, mesh \(M_2\) with 100 square cells, and mesh \(M_3\) with 200 square cells. Each cell of these meshes is further divided into two \(P_2\) triangles by its lower left-upper right diagonal, and from each \(P_2\) triangle four \(P_1\)-iso \(P_2\) triangles are generated by joining the mid-side points of the edges of the \(P_2\) triangles. The sizes of the time steps employed are \(\Delta t_1 = 5.0 \times 10^{-2}\) with mesh \(M_1\), \(\Delta t_2 = 10^{-2}\) with mesh \(M_2\), and \(\Delta t_3 = 5 \times 10^{-3}\) with mesh \(M_3\). The number of time steps to complete a revolution is 126 in mesh \(M_1\), 628 in mesh \(M_2\), and 1256 in mesh \(M_3\). Other parameters of this test are the following: (for the PLS method) \(r_{\min} = 0.01h\), \(r_{\max} = 0.05h\), \(n_p = 1.5 \times 10^4\); and \(\beta = 1.5\). The reinitialization step is switched on at every time step, and the number of reinitialization steps is \(m_1 = 4\) with \(\Delta r = \frac{\Delta t}{10}\). We represent in Figure 5.1 the solutions after one revolution. We note that the graphs of the solutions obtained in meshes \(M_2\) and \(M_3\) are almost indistinguishable from each other and from the graph of the exact solution.

The area loss defined by the formula

\[
A_{\text{loss}} = \int_D (H(u) - H(u_h)) \, dx
\]
is a measure of the accuracy of the method (see [16], [17], and [26]). We calculate it by substituting the Heaviside graphs $H(u)$ and $H(u_h)$ by the mollified versions $H_\eta(u)$ and $H_\eta(u_h)$, respectively, shown in (5.4), with $\eta = h$, and approximating the integral $\int_D (H(u) - H_\eta(u_h)) dx$ by a Gaussian quadrature rule of seven points. Table 5.1 shows the area loss, the error in the $l^\infty$ norm, and the order of convergence after one revolution. To calculate the departure points as well as the forward motion of the particles we have used the second order scheme of [19]. However, to test the influence that the accuracy of the forward trajectories of the particles has on the accuracy of zero level set, we have carried out one experiment in mesh $M_2$ in which the trajectories of the particles are calculated with an explicit fourth order seven-stage Runge–Kutta scheme with minimal dispersion and dissipation, and we have obtained that for $\Delta t = 10^{-2}$ the area loss after one revolution is much smaller.

Following a suggestion of one of the referees, we shall compare the QMSL-PLS with quadratic interpolation with the conventional semi-Lagrangian-PLS (SL-PLS) method using linear interpolation [17]. From Theorems 4.4 and 4.5 it follows that if the solution is sufficiently smooth the results given by the QMSL-PLS method with quadratic interpolation will be more accurate than those produced by the conventional SL-PLS with linear interpolation; however, when the solution is not regular enough, as in this example, it is not clear which method is better when considering accuracy versus CPU time. Because of this it is illustrative to compare both methods in this example in which the solution is not smooth. Thus, in Table 5.2 we collect the errors $\int_D (H_\eta(u) - H_\eta(u_h)) dx$, after one revolution, for a different number of massless particles and four increasingly refined meshes $H = \{1/32, 1/64, 1/128, 1/256\}$, and in Table 5.3 we show the corresponding CPU times taking as reference the time for a run without particles and linear interpolation. As can be noticed in Table 5.2, SL-PLS with linear interpolation results in a larger error than the QMSL-PLS with quadratic

\begin{table}[h]
\centering
\caption{Numerical results for Zalesak’s slotted disk after one revolution.}
\begin{tabular}{|c|c|c|c|c|}
\hline
Meshes & Area loss & $l^\infty$ & $A_{loss}$-order & $l^\infty$-order \\
\hline
$M_1$ & $3.92 \cdot 10^{-3}$ & $7.62 \cdot 10^{-2}$ & NA & NA \\
$M_2$ & $1.88 \cdot 10^{-3}$ & $2.44 \cdot 10^{-2}$ & 1.06 & 1.55 \\
$M_3$ & $7.13 \cdot 10^{-4}$ & $2.40 \cdot 10^{-2}$ & 1.40 & 0.024 \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\caption{\(\int_D (H_\eta(u) - H_\eta(u_h)) dx\) errors of the SL-PLS method with linear interpolation (e1) and the QMSL-PLS method with quadratic interpolation (e2) for Zalesak’s slotted cylinder configuration after one revolution.}
\begin{tabular}{|c|c|c|c|c|}
\hline
Mesh & $32 \times 32$ & $64 \times 64$ & $128 \times 128$ & $256 \times 256$ \\
\hline
$e_1, n_p = 0$ & $8.68 \cdot 10^{-2}$ & $6.90 \cdot 10^{-2}$ & $5.58 \cdot 10^{-2}$ & $4.04 \cdot 10^{-2}$ \\
$e_2, n_p = 0$ & $6.42 \cdot 10^{-2}$ & $5.18 \cdot 10^{-2}$ & $3.83 \cdot 10^{-2}$ & $2.47 \cdot 10^{-2}$ \\
$e_1, n_p = 1.5 \cdot 10^3$ & $1.01 \cdot 10^{-2}$ & $9.10 \cdot 10^{-3}$ & $2.49 \cdot 10^{-3}$ & $1.31 \cdot 10^{-3}$ \\
$e_2, n_p = 1.5 \cdot 10^3$ & $2.89 \cdot 10^{-3}$ & $2.07 \cdot 10^{-3}$ & $1.61 \cdot 10^{-3}$ & $9.35 \cdot 10^{-4}$ \\
$e_1, n_p = 1.5 \cdot 10^4$ & $6.05 \cdot 10^{-3}$ & $2.28 \cdot 10^{-3}$ & $8.22 \cdot 10^{-4}$ & $2.85 \cdot 10^{-4}$ \\
$e_2, n_p = 1.5 \cdot 10^4$ & $1.76 \cdot 10^{-3}$ & $9.13 \cdot 10^{-4}$ & $5.52 \cdot 10^{-4}$ & $1.75 \cdot 10^{-4}$ \\
$e_1, n_p = 1.5 \cdot 10^5$ & $6.11 \cdot 10^{-3}$ & $1.69 \cdot 10^{-3}$ & $5.93 \cdot 10^{-4}$ & $2.38 \cdot 10^{-4}$ \\
$e_2, n_p = 1.5 \cdot 10^5$ & $1.73 \cdot 10^{-3}$ & $7.24 \cdot 10^{-4}$ & $4.26 \cdot 10^{-4}$ & $2.07 \cdot 10^{-4}$ \\
$e_1, n_p = 1.5 \cdot 10^6$ & $4.96 \cdot 10^{-3}$ & $1.55 \cdot 10^{-3}$ & $5.53 \cdot 10^{-4}$ & $2.47 \cdot 10^{-4}$ \\
$e_2, n_p = 1.5 \cdot 10^6$ & $1.73 \cdot 10^{-3}$ & $7.01 \cdot 10^{-4}$ & $4.11 \cdot 10^{-4}$ & $2.22 \cdot 10^{-4}$ \\
\hline
\end{tabular}
\end{table}
interpolation for all mesh levels and number of particles \( n_p \). Further, the behavior of the error with the number of particles shows that the gains of employing QMSL-PLS with quadratic interpolation are all the more important when dealing with “low” mesh resolutions (32 \( \times \) 32, 64 \( \times \) 64) and “medium” number of particles (1.5 \( \times \) 10^3 \( \div \) 1.5 \( \times \) 10^6) and decreases as the mesh resolution improves.

However, the addition of any number of massless particles in this problem proves to be even more beneficial as the mesh grows refined, as can be seen when comparing the rows from Table 5.2 without \( (n_p = 0) \) and with \( (n_p > 0) \) particles. There also seems to be an “optimum” number of particles for this configuration in the finest mesh (256 \( \times \) 256), \( n_{p_{\text{opt}}} \approx 1.5 \times 10^4 \), both for first and second order advection, beyond which an increase in \( n_{p_{\text{opt}}} \) does not correlate with a reduction in the error. As regards the computational time, Table 5.3 collects all data in dimensionless units, taking as reference the time for a run of the SL-PLS without particles in the coarsest mesh 32 \( \times \) 32. All these serial computations were carried out in a Pentium Dual Core @ 3.2 GHz using GNU-gcc 4.6 compiler and -O3 optimization level. Though SL-PLS linear advection is slightly faster than the QMSL-PLS method with quadratic interpolation for all mesh levels without particles, employing any number of them yields that the QMSL-PLS method with quadratic interpolation is faster than the LS-PLS method. A possible explanation for this is that the number of particles “struggling” to improve the solution in the QMSL-PLS method is lower than the number in the SL-PLS method because the QMSL-PLS error is lower than the SL-PLS error. In addition, using the largest number of particles \( (n_p = 1.5 \times 10^6) \) in a certain mesh is usually more expensive than resorting to a better mesh one level (from 64 \( \times \) 64 to 128 \( \times \) 128, for example) and taking a reduced number of particles \( (n_p \approx 1.5 \times 10^4) \); this latter option often provides a lower error as well. From all these considerations, we think that (at least with medium number of particles \( (n_p \approx 10^4 \div 10^5) \) and medium mesh resolutions (64 \( \times \) 64, 128 \( \times \) 128)) the QMSL-PLS method with quadratic interpolation is better than the conventional SL-PLS method with linear interpolation.

### 5.2. Single vortex flow

This is a test to illustrate the ability of our method to resolve thin filaments at scales of the mesh which occur in stretching and tearing flows. The problem is defined in (5.1) with the interface \( \Gamma _0 (0) \) being the boundary of a circle of radius 0.15 and center at \((0.5, 0.75)\). The velocity field is given by

\[
\begin{align*}
    v_1 (x_1, x_2) &= -\sin^2 (\pi x_1) \sin (2\pi x_2) \cos \left( \frac{\pi x_1}{T} \right), \\
    v_2 (x_1, x_2) &= -\sin (2\pi x_1) \sin^2 (\pi x_2) \cos \left( \frac{\pi x_1}{T} \right),
\end{align*}
\]

---

**Table 5.3**

\( \text{CPU time (dimensionless units) of the SL-PLS method with linear interpolation (t1) and the QMSL-PLS method with quadratic interpolation (t2) for Zalesak’s slotted cylinder configuration.} \)

<table>
<thead>
<tr>
<th>Mesh</th>
<th>32 ( \times ) 32</th>
<th>64 ( \times ) 64</th>
<th>128 ( \times ) 128</th>
<th>256 ( \times ) 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1, ( n_p = 0 )</td>
<td>1.00 ( \times ) 10^0</td>
<td>1.09 ( \times ) 10^1</td>
<td>7.74 ( \times ) 10^1</td>
<td>6.48 ( \times ) 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 0 )</td>
<td>1.06 ( \times ) 10^0</td>
<td>1.12 ( \times ) 10^1</td>
<td>7.80 ( \times ) 10^1</td>
<td>6.62 ( \times ) 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^3 )</td>
<td>1.39 ( \times ) 10^0</td>
<td>1.18 ( \times ) 10^1</td>
<td>8.43 ( \times ) 10^1</td>
<td>6.85 ( \times ) 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^3 )</td>
<td>1.24 ( \times ) 10^0</td>
<td>1.18 ( \times ) 10^1</td>
<td>8.50 ( \times ) 10^1</td>
<td>6.74 ( \times ) 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^4 )</td>
<td>3.23 ( \times ) 10^0</td>
<td>1.51 ( \times ) 10^1</td>
<td>9.38 ( \times ) 10^1</td>
<td>7.60 ( \times ) 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^4 )</td>
<td>2.99 ( \times ) 10^0</td>
<td>1.45 ( \times ) 10^1</td>
<td>9.34 ( \times ) 10^1</td>
<td>7.73 ( \times ) 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^5 )</td>
<td>1.93 ( \times ) 10^0</td>
<td>4.80 ( \times ) 10^1</td>
<td>1.60 ( \times ) 10^2</td>
<td>8.90 ( \times ) 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^5 )</td>
<td>1.82 ( \times ) 10^0</td>
<td>4.62 ( \times ) 10^1</td>
<td>1.58 ( \times ) 10^2</td>
<td>8.87 ( \times ) 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^6 )</td>
<td>1.78 ( \times ) 10^2</td>
<td>3.73 ( \times ) 10^2</td>
<td>8.20 ( \times ) 10^2</td>
<td>2.18 ( \times ) 10^3</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^6 )</td>
<td>1.65 ( \times ) 10^2</td>
<td>3.64 ( \times ) 10^2</td>
<td>8.07 ( \times ) 10^2</td>
<td>2.15 ( \times ) 10^3</td>
</tr>
</tbody>
</table>
Table 5.4

Numerical results of the single vortex flow at time $T = 8$.

<table>
<thead>
<tr>
<th>Meshes</th>
<th>$n_p$</th>
<th>Area loss</th>
<th>$l^\infty$</th>
<th>$A_{loss}$-order</th>
<th>$l^\infty$-order</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>$1.5 \cdot 10^4$</td>
<td>$3.53 \cdot 10^{-4}$</td>
<td>$1.46 \cdot 10^{-1}$</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$1.5 \cdot 10^5$</td>
<td>$1.70 \cdot 10^{-4}$</td>
<td>$2.87 \cdot 10^{-2}$</td>
<td>1.04</td>
<td>2.34</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$1.5 \cdot 10^6$</td>
<td>$2.58 \cdot 10^{-5}$</td>
<td>$1.42 \cdot 10^{-2}$</td>
<td>2.72</td>
<td>1.05</td>
</tr>
</tbody>
</table>

![Numerical solution for the single vortex flow in mesh $M_2$ at time instants $t = 0$ (upper left panel), $t = 1$ (upper right panel), $t = 3$ (lower left panel), and $t = 5$ (lower right panel).](image)

Fig. 5.2. The numerical solution for the single vortex flow in mesh $M_2$ at time instants $t = 0$ (upper left panel), $t = 1$ (upper right panel), $t = 3$ (lower left panel), and $t = 5$ (lower right panel).

where $T$ is the time at which the flow returns to its initial shape, in this case $T = 8$. As described in [17], the velocity field stretches out the circle into a very long thin fluid element which progressively wraps itself toward the center of the domain. In underresolved regions, the particles will not be close enough to accurately represent the interface, and thin filament structures will break apart.

The meshes, the corresponding sizes of the time steps, and the parameters $r_{\min}$, $r_{\max}$, and $\beta$ employed in this test are the same as those of Zalesak’s disk. Table 5.4 shows, for different meshes and numbers of particles $n_p$, the area loss, $A_{loss}$, and the $l^\infty$ error at $T = 8$.

The results of Table 5.4 as well as those represented in Figure 5.2 were obtained without reseeding of particles and reinitializing every time step with $\Delta \tau = \frac{\Delta t}{m_1}$ and $m_1 = 2$. Figure 5.2 displays the numerical solutions in mesh $M_2$ at time instants $t = 0, 1, 3, 5$. The results for this example compare very well with those reported in [17] and [16], where in the latter the PLS method was used in combination with a fifth order HJ-WENO scheme for both transport and reinitialization equations. In Table 5.5 we compare, for a mesh of $128 \times 128$ square cells at time $T = 8$, the $\int_D |(H_0(u) - H_0(u_i))| \, dx$ error of the QMSL-PLS method for a different number of particles $n_p$ and no reseeding at $T = 8$ with the cellwise volume fraction difference.
errors, $E_{cf}$, of the methods presented in references [1], [10], and [24]. The errors of these references are computed by the formula (14) of [1] (that corresponds to formula (14) of [10]), which is the analogous in the context of volume of fluid methods to our formula in a level set context.

It is clear that the solution of QMSL-PLS ($n_p = 1.5 \cdot 10^3$) is comparable to the solution of the adaptive mesh refinement-moment of fluid (AMR-MOF) method and is better than the solutions provided in [10] and [24]. Moreover, we note that the solution of the QMSL-PLS method improves by a factor larger than 2 when $n_p = 1.5 \cdot 10^4$ and $n_p = 1.5 \cdot 10^5$. Similar results (not shown here) hold when the comparison is carried out in meshes of $32 \times 32$ and $64 \times 64$ square cells. From these results, $n_p = 1.5 \cdot 10^4$ seems to be the right choice for this problem.

As we did in the previous example, we also compare now the results of the QMSL-PLS method with quadratic interpolation with those of the conventional SL-PLS method with linear interpolation. This comparison is done for $T = 0.5$ and $T = 8.0$ and the results are shown in Tables 5.6-5.7. The calculations were carried out with $\Delta t = h$.

The error for $T = 8$ and $T = 0.5$ mimics the pattern observed for Zalesak’s slotted cylinder so that the observations made there are valid here as well. We note that linear interpolation is unable to deal with the hard $T = 8$ version of the problem if the meshes are not sufficiently fine; however, the addition of massless particles does improve the resolution of the method and provides a solution (if somewhat gross) when the coarser meshes are employed. For the much milder version $T = 0.5$ linear interpolation is capable of obtaining a solution in all meshes.

The computational times collected in Tables 5.8 and 5.9 show that the savings obtained by using linear advection are not that important when using any number of particles due to the fact that, as already remarked in Zalesak’s problem, linear interpolation must correct the surface with more particles than quadratic interpolation.
Table 5.7

\[ \int_D \left( H_q(u) - H_q(u_0) \right) dx \] errors of the SL-PLS method with linear interpolation (e1) and the QMSL-PLS method with quadratic interpolation (e2) for the single vortex flow when \( T = 0.5 \).

<table>
<thead>
<tr>
<th>Mesh</th>
<th>32 × 32</th>
<th>64 × 64</th>
<th>128 × 128</th>
<th>256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1, ( n_p = 0 )</td>
<td>5.57 \times 10^{-3}</td>
<td>2.51 \times 10^{-3}</td>
<td>1.19 \times 10^{-3}</td>
<td>5.85 \times 10^{-4}</td>
</tr>
<tr>
<td>e2, ( n_p = 0 )</td>
<td>6.62 \times 10^{-4}</td>
<td>2.11 \times 10^{-4}</td>
<td>3.25 \times 10^{-5}</td>
<td>1.47 \times 10^{-5}</td>
</tr>
<tr>
<td>e1, ( n_p = 1.5 \times 10^3 )</td>
<td>1.35 \times 10^{-3}</td>
<td>6.00 \times 10^{-4}</td>
<td>2.54 \times 10^{-4}</td>
<td>1.34 \times 10^{-4}</td>
</tr>
<tr>
<td>e2, ( n_p = 1.5 \times 10^3 )</td>
<td>6.60 \times 10^{-4}</td>
<td>2.11 \times 10^{-4}</td>
<td>3.20 \times 10^{-5}</td>
<td>1.43 \times 10^{-5}</td>
</tr>
<tr>
<td>e1, ( n_p = 1.5 \times 10^4 )</td>
<td>8.96 \times 10^{-4}</td>
<td>3.43 \times 10^{-4}</td>
<td>8.58 \times 10^{-5}</td>
<td>4.15 \times 10^{-5}</td>
</tr>
<tr>
<td>e2, ( n_p = 1.5 \times 10^4 )</td>
<td>6.63 \times 10^{-4}</td>
<td>2.08 \times 10^{-4}</td>
<td>2.86 \times 10^{-5}</td>
<td>1.30 \times 10^{-5}</td>
</tr>
<tr>
<td>e1, ( n_p = 1.5 \times 10^5 )</td>
<td>8.26 \times 10^{-4}</td>
<td>2.87 \times 10^{-4}</td>
<td>4.54 \times 10^{-5}</td>
<td>1.55 \times 10^{-5}</td>
</tr>
<tr>
<td>e2, ( n_p = 1.5 \times 10^5 )</td>
<td>6.54 \times 10^{-4}</td>
<td>2.06 \times 10^{-4}</td>
<td>2.52 \times 10^{-5}</td>
<td>1.15 \times 10^{-5}</td>
</tr>
<tr>
<td>e1, ( n_p = 1.5 \times 10^6 )</td>
<td>8.10 \times 10^{-4}</td>
<td>2.79 \times 10^{-4}</td>
<td>3.86 \times 10^{-5}</td>
<td>9.42 \times 10^{-6}</td>
</tr>
<tr>
<td>e2, ( n_p = 1.5 \times 10^6 )</td>
<td>6.06 \times 10^{-4}</td>
<td>2.07 \times 10^{-4}</td>
<td>2.40 \times 10^{-5}</td>
<td>1.05 \times 10^{-5}</td>
</tr>
</tbody>
</table>

Table 5.8

CPU time (dimensionless units) of the the SL-PLS method with linear interpolation (t1) and the QMSL-PLS method with quadratic interpolation (t2) for the single vortex flow when \( T = 8 \).

<table>
<thead>
<tr>
<th>Mesh</th>
<th>32 × 32</th>
<th>64 × 64</th>
<th>128 × 128</th>
<th>256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1, ( n_p = 0 )</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>7.43 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 0 )</td>
<td>1.00 \times 10^0</td>
<td>8.14 \times 10^0</td>
<td>8.69 \times 10^1</td>
<td>8.30 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^3 )</td>
<td>1.00 \times 10^0</td>
<td>7.94 \times 10^0</td>
<td>7.93 \times 10^1</td>
<td>7.98 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^3 )</td>
<td>1.05 \times 10^0</td>
<td>8.45 \times 10^0</td>
<td>8.76 \times 10^1</td>
<td>8.30 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^4 )</td>
<td>1.88 \times 10^0</td>
<td>1.02 \times 10^1</td>
<td>8.73 \times 10^1</td>
<td>8.25 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^4 )</td>
<td>1.87 \times 10^0</td>
<td>9.95 \times 10^0</td>
<td>8.97 \times 10^1</td>
<td>8.38 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^5 )</td>
<td>1.01 \times 10^1</td>
<td>2.54 \times 10^1</td>
<td>1.16 \times 10^2</td>
<td>8.90 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^5 )</td>
<td>1.00 \times 10^1</td>
<td>2.46 \times 10^1</td>
<td>1.16 \times 10^2</td>
<td>8.89 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^6 )</td>
<td>9.10 \times 10^1</td>
<td>1.70 \times 10^2</td>
<td>3.76 \times 10^2</td>
<td>1.41 \times 10^3</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^6 )</td>
<td>9.05 \times 10^1</td>
<td>1.69 \times 10^2</td>
<td>3.74 \times 10^2</td>
<td>1.40 \times 10^3</td>
</tr>
</tbody>
</table>

Table 5.9

CPU time (dimensionless units) of the the SL-PLS method with linear interpolation (t1) and the QMSL-PLS method with quadratic interpolation (t2) for the single vortex flow when \( T = 0.5 \).

<table>
<thead>
<tr>
<th>Mesh</th>
<th>32 × 32</th>
<th>64 × 64</th>
<th>128 × 128</th>
<th>256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1, ( n_p = 0 )</td>
<td>1.00 \times 10^0</td>
<td>7.54 \times 10^0</td>
<td>7.47 \times 10^1</td>
<td>6.49 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 0 )</td>
<td>1.02 \times 10^0</td>
<td>7.61 \times 10^0</td>
<td>7.57 \times 10^1</td>
<td>6.53 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^3 )</td>
<td>1.01 \times 10^0</td>
<td>7.67 \times 10^0</td>
<td>7.50 \times 10^1</td>
<td>6.50 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^3 )</td>
<td>1.18 \times 10^0</td>
<td>7.80 \times 10^0</td>
<td>7.63 \times 10^1</td>
<td>6.56 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^4 )</td>
<td>1.82 \times 10^0</td>
<td>9.13 \times 10^0</td>
<td>7.72 \times 10^1</td>
<td>6.53 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^4 )</td>
<td>1.86 \times 10^0</td>
<td>9.18 \times 10^0</td>
<td>7.84 \times 10^1</td>
<td>6.60 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^5 )</td>
<td>9.43 \times 10^0</td>
<td>2.17 \times 10^1</td>
<td>9.90 \times 10^1</td>
<td>7.00 \times 10^2</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^5 )</td>
<td>9.31 \times 10^0</td>
<td>2.17 \times 10^1</td>
<td>1.00 \times 10^2</td>
<td>7.05 \times 10^2</td>
</tr>
<tr>
<td>t1, ( n_p = 1.5 \times 10^6 )</td>
<td>8.28 \times 10^0</td>
<td>1.44 \times 10^2</td>
<td>3.13 \times 10^2</td>
<td>1.11 \times 10^3</td>
</tr>
<tr>
<td>t2, ( n_p = 1.5 \times 10^6 )</td>
<td>8.28 \times 10^0</td>
<td>1.44 \times 10^2</td>
<td>3.12 \times 10^2</td>
<td>1.11 \times 10^3</td>
</tr>
</tbody>
</table>

to produce an error of similar magnitude; thus, the correction step takes longer in that case, and the a priori cheaper linear interpolation may give rise to an overall more expensive free-surface method.

In light of these results, we again recommend the use of the QMSL-PLS method with medium number of particles (\( n_p \approx 10^4 \div 10^5 \)) and medium to high mesh resolutions.
5.3. Two-phase interfacial flows. This test is more complex than the previous ones in several respects. We simulate the rising of a bubble in a Newtonian fluid with the geometry of the problem as shown in Figure 5.3. In the rectangular domain, \( D = (0, 1) \times (0, 2) \), there is initially a bubble of radius \( R = 0.25 \) and center at \((0.5, 0.5)\). The bubble represents geometrically the domain \( D_2 \) and is composed from a Newtonian fluid of constant density and kinematic viscosity \( \rho_2 \) and \( \mu_2 \), respectively; the boundary \( \Gamma_0(t) \) of \( D_2 \) is the interface which is considered as the zero level set of the signed distance function \( u(x, t) \). The domain \( D_1 = D \setminus (D_2 \cup \Gamma_0) \) is filled by a Newtonian fluid of constant density and kinematic viscosity \( \rho_1 \) and \( \mu_1 \), respectively. Assuming that \( \rho_2 < \rho_1 \), the bubble, starting from a rest state, will rise by the action of buoyancy forces. Capillary forces must be taken into account. The governing equations in \( D \times (0, T) \) are the nonstationary incompressible Navier–Stokes equations for velocity \( v(x, t) \) and pressure \( p(x, t) \), plus the transport equation for the level set function \( u(x, t) \). The dimensionless form of the equations is

\[
\begin{align*}
\rho(u) \left( \frac{\partial v}{\partial t} + v \cdot \nabla v \right) + \nabla p &= \frac{1}{Re} \nabla \cdot (\mu(u)(\nabla v + (\nabla v)^T)) \\
&+ \frac{\rho(u)(-e_2)}{Fr^2} + \frac{1}{We} \kappa(u) \delta(u) \mathbf{n}, \\
\nabla \cdot v &= 0, \\
\frac{\partial u}{\partial t} + v \cdot \nabla u &= 0,
\end{align*}
\]

(5.2a)

where \( \delta \) is the Dirac function, \( \kappa \) denotes the dimensionless curvature, and \( Re, Fr^2, \) and \( We \) are dimensionless numbers to be defined below. The boundary conditions for velocity are as follows:

1. On the lateral walls: free slip, that is,

\[
\begin{align*}
v \cdot \mathbf{n} &= 0 \quad \text{and} \quad \mathbf{n} \cdot \sigma \cdot \mathbf{t} = 0,
\end{align*}
\]

(5.3a)
where \( t \) is the unit tangent vector and \( \sigma \) denotes the viscous stress tensor, i.e.,

\[
(5.3b) \quad \sigma_{ij} = -p \delta_{ij} + \frac{1}{Re} \left( \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right), \quad 1 \leq i, j \leq 2.
\]

(2) On the upper and lower boundaries:

\[
(5.3c) \quad v = 0.
\]

As initial conditions, we impose that the velocity is zero everywhere in \( D \), and \( u(x, 0) \) is the signed distance function to \( \Gamma_0(0) \).

The dimensionless numbers \( Re, Fr^2 \), and \( We \) are giving by

\[
Re = \frac{\rho_1 UL}{\mu_1}, \quad We = \frac{\rho_1 U^2 L}{\sigma}, \quad Fr = \frac{U^2}{gL},
\]

where \( U = \sqrt{2gL} \) and \( L = 2R \) represent the characteristic velocity and length scales, \( g \) is the modulus of the gravity acceleration vector, and \( \sigma \) is the coefficient of the surface tension.

The coefficients \( \rho(u) \) and \( \mu(u) \) are given by the formulas

\[
\rho(u) = \frac{\rho_2}{\rho_1} + \left( 1 - \frac{\rho_2}{\rho_1} \right) H_\eta(u), \quad \mu(u) = \frac{\mu_2}{\mu_1} + \left( 1 - \frac{\mu_2}{\mu_1} \right) H_\eta(u);
\]

here, \( H_\eta(u) \) is a mollified Heaviside graph the expression of which is

\[
(5.4) \quad H_\eta(u) = \begin{cases} 0 & \text{if } u < -\eta, \\ \frac{1}{2} \left[ 1 + \frac{u}{\eta} + \frac{1}{\pi} \sin \left( \frac{\pi u}{\eta} \right) \right], & \text{if } u > \eta. \\
\end{cases}
\]

To calculate the numerical solution to \((5.2a)–(5.2c)\) we introduce the finite dimensional spaces

\[
\nabla_H = \{ v_H \in C^0(D)^d : v_H \mid_T \in P_2(T)^d \forall T \in D_H \}
\]

and

\[
\mathcal{Y}_H = \{ q_H \in C^0(D) : q_H \mid_T \in P_1(T) \forall T \in D_H \text{ and } \int_D q_H dx = 0 \}.
\]

Moreover, let \( \partial D = \partial D_1 \cup \partial D_2 \) with \( \partial D_1 \) and \( \partial D_2 \) being those parts of the boundary, respectively, where homogeneous Dirichlet and free slip boundary conditions are imposed. Following [3] we also need the subspace

\[
\nabla_{H_0} = \{ v_H \in \nabla_H : v_H \mid_{\partial D_1} = 0 \text{ and } v_H \cdot n \mid_{\partial D_2} = 0 \}.
\]

The time discretization of \((5.2a)–(5.2b)\) is a second order backward differentiation formula applied backward in time along the characteristic curves of the operator \( \frac{Dv}{Dt} \). This yields the second order in time modified Lagrange–Galerkin method of [5] and
[6]. Thus, for \( n = 1,2,\ldots,N \), we find \((v_H^{n+1}, p_H^{n+1}) \in V_{H0} \times Y_H \) as a solution of the system
\[
\begin{align*}
&\frac{3}{2\Delta t} (\rho(u_h^{n+1}) v_H^{n+1}, \bar{\psi}_H) + \frac{1}{Re} (\mu(u_h^{n+1}) \nabla v_H^{n+1}, \nabla \bar{\psi}_H) - (p_H^{n+1}, \text{div} \bar{\psi}_H) \\
&= \frac{2}{\Delta t} \left( \rho(u_h^{n+1}) v_H^n (X_h^{n,n+1}(x), \bar{\psi}_H) - \frac{1}{2\Delta t} \left( \rho(u_h^{n+1}) v_H^{n-1} (X_h^{n-1,n+1}(x), \bar{\psi}_H) \\
&+ \frac{1}{Fr^2} (\rho(u_h^{n+1})(-e_2), \bar{\psi}_H) + \frac{1}{We} (\kappa(u_h^{n+1}) n_h^{n+1}, \bar{\psi}_H)_{\Gamma_{h0}(t_{n+1})} \quad \forall \bar{\psi}_H \in V_{H0}, \\
&\text{div} v_H^{n+1}, q_H) = 0 \quad \forall q_H \in Y_H,
\end{align*}
\]
where \( X_h^{n,n+1}(x) \), \( t = 0,1 \), is an approximation to the numerical solution of
\[
\begin{align*}
&\frac{dX_h(x,t_{n+1},t)}{dt} = v_H(X_h(x,t_{n+1},t), t), \quad t_{n-1} \leq t < t_{n+1}, \\
&X_h(x,t_{n+1},t_{n+1}) = x \quad \forall x \in D,
\end{align*}
\]
and \( u_h^{n+1} \in V_h \) is the numerical solution at time instant \( t_{n+1} \) to (5.2c) calculated by the QMSL-PLS method. The following notation has been used in (5.5): \((a,b) = \int_D abdx \) and \((a,b)_{\Gamma_{h0}(t_{n+1})} = \int_{\Gamma_{h0}(t_{n+1})} abds \). The use of high order quadrature rules is recommended to calculate the integrals \((\rho(u_h^{n+1})a,b) \) and \((\mu(u_h^{n+1})a,b) \), in particular when any of the ratios \( \frac{\int D}{\int Q} \) or \( \frac{\int D}{\int T} \) is high; in this numerical test we have used a Gaussian quadrature rule of seven points. The integral \((\kappa(u_h^{n+1}) n_h^{n+1}, \bar{\psi}_H)_{\Gamma_{h0}(t_{n+1})} \) is calculated as
\[
\int_{\Gamma_{h0}(t_{n+1})} \kappa(u_h^{n+1}) n_h^{n+1} \cdot \bar{\psi}_H ds = \sum_j \int_{I_j} \kappa(u_h^{n+1}) n_h^{n+1} \cdot \bar{\psi}_H ds,
\]
where \( \{I_j\} \) are the line segments of \( \Gamma_{h0}(t_{n+1}) \), that is, \( \Gamma_{h0}(t_{n+1}) = \cup_j I_j \). Note that such segments, as well as \( n_h^{n+1} \), are calculated in the reinitialization stage of the QMSL-PLS algorithm. The integrals on the line segments are calculated by the Gaussian quadrature of three points. It remains to describe the calculation of the curvature \( \kappa(u_h^{n+1}) = -\nabla \cdot n_h^{n+1} \). As noticed in [27], it is more accurate to evaluate \( \kappa(u_h^{n+1}) \) as the \( L^2 \)-projection of \(-\nabla \cdot n_h^{n+1} \) onto \( V_h \) than calculating it as \(-\nabla \cdot n_h^{n+1} \). Writing
\[
\kappa(u_h^{n+1}) = \sum_{k=1}^{NN} \gamma_k^{n+1} \psi_k,
\]
The coefficients \( \gamma_k^{n+1} \) are calculated from
\[
\int_D (\kappa(u_h^{n+1}) + \nabla \cdot n_h^{n+1}) \psi_k dx = 0.
\]
Following the arguments of Remark 3.2 we have that for each \( k \) \[(5.6) \quad \gamma_k^{n+1} = \frac{-1}{\sum_i |T_i|} \sum_i |T_i| \nabla \cdot n_h^{n+1} |T_i|,
\]
where \( T_i \) are the elements forming the support of \( \psi_k \).
Assuming that $u_h^n, v_h^n,$ and $p_h^n$ are known, the procedure to calculate $u_h^{n+1}, v_h^{n+1},$ and $p_h^{n+1}$ consists of the following steps:

1. Calculate $u_h^{n+1}$ by the QMSL-PLS method with reinitialization algorithm.
2. Calculate $n_h^{n+1}$ and $p_h^{n+1}$ applying (3.19) and (5.6), respectively.
3. Calculate $v_h^{n+1}, p_h^{n+1}$ by solving (5.5).

We must note that in order to proceed with the calculation from $n = 1$ in (5.5), we need to know that $(v_1^H, p_1^H)$ as well as $p_h^0$. $p_h^0$ is calculated as a solution of

\[
(\nabla p_h^0, \nabla q_H) = \frac{1}{F \tau} (\rho(u_h^0)(-e_2), \nabla q_H) + \frac{1}{W \varepsilon} (\kappa(n_h^0)n_h^{n+1}, \nabla q_H)_{\Gamma_{h0}(t_0)} \quad \forall q_H \in Y_H,
\]

and then using a second order single step time discretization scheme applied in an iterative way we calculate $(v_1^H, p_1^H)$ and $u_1^H$.

To study the performance of the QMSL-PLS method in this problem, we solve the equations in meshes of $40 \times 80, 80 \times 160,$ and $160 \times 320$ square cells with the values of the physical parameters corresponding to those of test case 1 of [21], that is,

\[
\rho_1 = 1000, \quad \rho_2 = 100, \quad \mu_1 = 10; \quad \mu_2 = 1.0, \quad g = 0.98, \quad Fr = 1, \quad \sigma = 24.5, \quad Re = 35, \quad Wc = 10.
\]

Other parameters used in the calculations are the following: $\Delta t = H/8$, $n_p = 1.5 \cdot 10^4$, $\eta = h$, and $\beta$, $r_{max}$, and $r_{min}$ the same as in the Zalesak's slotted disk test. The reinitialization was applied every time step with $m_1 = 2$ and $\Delta \tau = \Delta t/10$; there was no reseeding of particles in the PLS step. Following the recommendations of [21] for benchmark computations of bubble dynamics to test the performance of numerical methods, we calculate the following time dependent quantities: position and rise velocity of the center of gravity, and the circularity of the bubble,

\[
x_{cog}(t) = (x_{1cog}(t), x_{2cog}(t)) = \frac{\int_{D_2} x(t) \, dt}{D_2}, \quad v_{cog}(t) = \frac{dx_{cog}}{dt}, \quad circ(t) = \frac{P_a}{P_h},
\]

where $P_a$ denotes the perimeter or circumference of a circle of diameter $d_a$ which has an area equal to that of the bubble with perimeter $P_h$. For a perfect circular bubble the circularity will be equal to one and will decrease as the bubble is deformed. We display in Figures 5.4 and 5.5 the results for circularity, rise velocity, center of gravity, and final shape provided by the QMSL-PLS method in these meshes together with the results of the TP2D method in the mesh $H = 1/320$. The numerical results for these magnitudes are collected in Table 5.10 for all three mesh levels, and a comparison among all algorithms (TP2D, FREELIFE, MooNMD, QMSL-PLS) is offered in Table 5.11. We notice a certain offset at the time when the minimum circularity occurs, similar in magnitude to that of FREELIFE; however, the minimum circularity as well as the rest of the magnitudes agree with the reference values within the error interval. From the data displayed in these figures and tables, we can conclude that the results obtained by the QMSL-PLS method seem to converge to values very close to those of the reference.

As noted in the literature, see, for instance [32], the error of the level set function $u_h$ has oscillating components that are more noticeable as the mesh is finer. These oscillating components will be magnified when the derivatives of $u_h$ are calculated numerically, yielding this way inaccurate point values of the curvature. For finer
grids \( (H \leq 1/160) \) we use the smoothing procedure proposed in [32] to filter out the oscillatory components of \( u_h \). Thus, at each time instant \( t_n \) we calculate \( \overline{u}_h \in V_h \) as a solution of

\[
(\overline{u}_h, v_h) + (\varepsilon \nabla \overline{u}_h, \nabla v_h) = (u_{h,n}^n, v_h) \quad \forall v_h \in V_h
\]
and then compute the mesh-point values of $\mathbf{n}_h = \frac{\nabla p}{|\nabla p|}$ and $\mathbf{n}_h = \nabla \cdot \mathbf{n}_h$ applying (3.19) and (5.6), respectively. Taking $\varepsilon = \frac{h^2}{4}$, this procedure effectively filters out the high frequency components of $u_h^p$. The function $\mathbf{n}_h$ is only used in this intermediate step to calculate the curvature and normal and is not used elsewhere.

6. Conclusions. In this paper, we have presented a QMSL-PLS method, capable of handling interface problems on unstructured meshes with accuracy and ease of implementation. The error analysis shows that this kind of scheme is unconditionally stable in the maximum discrete norm, and when the level set solution $u(t)$ is in the Sobolev space $W^{r+1,\infty}(D)$, $r \geq 0$, the convergence in the maximum norm is of the form $(KT/\Delta t)\min(1,\Delta t)\|v\|_{h,\infty}/h((1-\alpha)h^p+h^q)$, $p = \min(2,r+1)$, and $q = \min(3,r+1)$, where $v$ is a velocity. This means that at high CFL numbers, that is, when $\Delta t > h$, the error is $O((1-\alpha)h^{p+1})$, whereas at CFL numbers less than 1, the error is $O((1-\alpha)h^{p-1}+h^{q-1})$.

Several simulations were carried out to test the ability of the method to cope with different situations. Thus, the QMSL-PLS method in the Zalesak’s slotted cylinder configuration showed excellent results when compared to those of the literature; besides, the influence of the number of particles as well as a possible switch between first order and second order advection of the level set function was investigated in this problem: this study showed a preferred use of a higher order, quadratic option over a linear advection and highlighted the benefits of adding massless particles to correct the level set function. The highly stretched filaments developed in the single vortex flow problem were also nicely captured by the QMSL-PLS method; in this benchmark, our results also compared satisfactorily with those by AMR-MOF, GPCA (geometrical predictor-corrector advective), and Rider and Kothe, possibly evincing a certain optimum number of particles which can be added in a given mesh for maximum benefit. Finally, to check bidimensional two-fluid problems, a test case with high surface tension effects was also carried out so as to compare our results with those provided by the three different software tools: TP2D, FREELIFE, and MooNMD. The quantitative measurement of the position of center of gravity, maximum rise velocity, as well as circularity offered good agreement with the reference values as the mesh grew refined from the coarsest $H = 1/40$ to the finest $H = 1/160$ grid. In particular, a smoothing of the level set function for the finest grid, only used when computing normals and curvatures, proved to be useful in filtering out the high frequency values which could hamper the greater accuracy expected from such a refined mesh.

Acknowledgment. The second author thanks Dr. Carpio for valuable discussions.
REFERENCES