Mecánica Computacional Vol. XXIII, pp. 1659-1678
G.Buscaglia, E.Dari, O.Zamonsky (Eds.)
Bariloche, Argentina, November 2004

A NEW MASS-CONSERVING ALGORITHM FOR LEVEL SET REDISTANCING ON UNSTRUCTURED MESHES

Fernando Mut*, Gustavo C. Buscaglia*†, and Enzo A. Dari*†

*Instituto Balseiro, Universidad Nacional de Cuyo
8400, Bariloche, Argentina, e-mail: mutf@ib.cnea.gov.ar
†Centro Atómico Bariloche, CNEA
8400, Bariloche, Argentina (GCB and EAD also belong to CONICET)
e-mail: gustavo@cab.cnea.gov.ar, darie@cab.cnea.gov.ar

Key Words: Level-Set Method, Redistancing, Mass conservation

Abstract. *The Level Set Method is becoming increasingly popular for the simulation of several problems that involve interfaces. The level set function is advected by some velocity field, with the zero-level set of the function defining the position of the interface.The advection distorts the initial shape of the level set function, which needs to be re-initialized to a smooth function preserving the position of the zero-level set. Many algorithms re-initialize the level set function to (some approximation of) the signed distance from the interface. Efficient algorithms for level-set redistancing on cartesian meshes have become available over the last years, but unstructured meshes have received little attention.*

This presentation concerns algorithms for construction of a distance function from the zero-level set, in such a way that mass is conserved on arbitrary unstructured meshes. The algorithm is consistent with the hyperbolic character of the distance equation ($\| \nabla d \| = 1$*) and can be localized on a narrow band close to the interface, saving computing effort. The mass-rebalancing step is weighted according to local mass differences, an improvement over usual global rebalancing techniques.*

1 INTRODUCTION

The Level Set Method (LSM) has become a popular choice for numerically handling problems with interfaces. The basic idea is to represent the interface (which we will denote by $\mathcal{S}$ as the zero-level set of a level set (LS) function ϕ. The normal to the interface thus satisfies

$$\vec{n}(\vec{x}) = \frac{\nabla\phi}{\parallel \nabla\phi \parallel} \tag{1}$$

for all $\vec{x}$ such that $\phi(\vec{x}) = 0$. The curvature of $\mathcal{S}$ can also be obtained from ϕ, i.e.,

$$\kappa(\vec{x}) = \mathit{div}\, \vec{n} = \mathit{div}\, \left(\frac{\nabla\phi}{\parallel \nabla\phi \parallel} \right) \tag{2}$$

It is evident that, if $\vec{\beta}$ is any vector field such that $c_n = \vec{\beta} \cdot \vec{n}$ is the interface speed, then ϕ must satisfy the LS equation

$$\frac{\partial\phi}{\partial t} + \vec{\beta} \cdot \nabla\phi = 0 \tag{3}$$

at least locally at $\mathcal{S}$.

In multiphase flows, $\mathcal{S}$ represents the boundary between two immiscible fluids. Let us assume that just two fluids (A and B) are present inside a domain Ω, so that the region occupied by fluid A is

$$\Omega_A = \{\vec{x} \in \Omega,\, \phi(\vec{x}) > 0\} \tag{4}$$

If $\vec{\beta}$ (assumed smooth enough) is solenoidal inside Ω_A, then the volume of fluid A (denoted by $|\Omega_A|$) will be conserved since

$$\frac{d\,|\Omega_A|}{dt} = \int_{\mathcal{S}} c_n \, d\Gamma = \int_{\mathcal{S}} \vec{\beta} \cdot \vec{n} \, d\Gamma = \int_{\Omega_A} \mathit{div}\, \vec{\beta} \, d\Omega = 0 \tag{5}$$

It is important to remark that though (3) is a transport equation, it does not arise from a conservation law. Usual methods to deal with (3) have been designed to conserve mass when ϕ is a density; i.e., they conserve $\int_\omega \phi \, d\omega$ for some family of subdomains ω. This conservation property is in fact useless in LS formulations, since it does not imply that the zero-level set of ϕ will propagate at the correct speed, and thus in general mass of each fluid will be created or destroyed at the interface due to numerical error.

Much effort has lately been devoted to improving the LSM so as to minimize numerical inaccuracies in the zero-level set propagation.[1–6] The underlying idea is that if the initial data $\phi_0(\vec{x}) = \phi(\vec{x}, t = 0)$ is smooth in a neighborhood of $\mathcal{S}$, then any high-order numerical method for (3) will propagate the interface without significant error from $t = 0$ to some time $t = T$ provided that the mesh is fine enough. The time bound T arises because, in general, ϕ will not remain smooth indefinitely and will thus be more prone to numerical inaccuracies.

The most popular approach for initializing ϕ as a function that is smooth close to $\mathcal{S}$ is to choose ϕ as the signed distance d to the interface. Of course, after some simulation time (smaller

than T) the function ϕ is re-initialized (or "redistanced"), so that LS distortion is kept under control. This article presents a general method for computing d in unstructured meshes, which is related to fast-marching methods developed for cartesian grids.[7,8] The method is developed for simplices (triangles in 2D, tetrahedra in 3D), and is consistent with the hyperbolic character of the distance equation $\| \nabla d \| = 1$.

However, no matter how accurately d is computed at the mesh nodes, the function ϕ which coincides with d at the nodes will not preserve the exact location of the interface. This may result in an additional spurious local mass loss or gain, which is added to that coming from numerical errors in the solution of (3). Almost all re-distancing algorithms thus involve some sort of mass-rebalancing step.[5,9,10] Our method includes one such step that is local and involves no adjustable parameter.

We are leaving aside many other sources of error that play a role in LS formulations of multi-fluid flows. Among others, the numerical difficulty of correctly computing the transport velocity $\vec{\beta}$ close to the interface, where density and viscosity are discontinuous. Very comprehensive accounts of the LSM for fluid interfaces are available.[11,12]

2 REDISTANCING ALGORITHM

2.1 Preliminaries

We consider an arbitrary triangulation $\mathcal{T}_h$ of the domain Ω, where h is a characteristic mesh size, and the associated space V_h of continuous functions that are linear inside each simplex. Let $\phi_h \in V_h$ be a function, and let $\mathcal{S}$ be its zero-level set. Our aim is to find a function $\tilde{\phi}_h \in V_h$ which approximates the signed distance function d to $\mathcal{S}$, defined for any closed set $\mathcal{S}$ as

$$d(\vec{x}) = \text{sign}\,[\phi_h(\vec{x})] \min_{\vec{y} \in \mathcal{S}} \| \vec{x} - \vec{y} \| \tag{6}$$

This function satisfies $\| \nabla d \| = 1$ almost everywhere in Ω, but does *not*, in general, belong to V_h. In what follows we will assume Ω to be bounded, and all curves (in 2D) or surfaces (in 3D) to be compact (adding the adherence points if the curve or surface intersects the boundary of Ω.

The algorithm we consider is based on the following basic property of the distance function to a compact set $\mathcal{S}$:

Proposition 2.1 *Let $\mathcal{C}$ be a surface in $\mathbb{R}^n$ (of co-dimension 1) which divides $\mathbb{R}^n$ into two open sets, ω^+ and ω^-, such that $\mathcal{S} \subset \omega^-$. Then, for any $\vec{y} \in \omega^+$,*

$$|d(\vec{y})| = \min_{\vec{x} \in \mathcal{C}} [|\vec{y} - \vec{x}| + |d(\vec{x})|] \tag{7}$$

Proof: Let $\vec{s} \in \mathcal{S}$ satisfy $|d(\vec{y})| = |\vec{y} - \vec{s}|$, and let $\vec{a}$ be the intersection of the segment $\overline{ys}$ with $\mathcal{C}$ (which exists because $\mathcal{C}$ divides $\mathbb{R}^n$). Notice first that $|d(\vec{a})| = |\vec{a} - \vec{s}|$ because, if there existed a point $\vec{w}$ such that $|\vec{a} - \vec{w}| < |\vec{a} - \vec{s}|$, then

$$|\vec{y} - \vec{w}| \leq |\vec{y} - \vec{a}| + |\vec{a} - \vec{w}| < |\vec{y} - \vec{a}| + |\vec{a} - \vec{s}| = |\vec{y} - \vec{s}| = |d(\vec{y})|$$

in contradiction with the definition of $|d(\vec{y})|$ as the minimal distance from $\vec{y}$ to any point in $\mathcal{S}$. Defining now

$$\xi(\vec{y}) = \min_{\vec{x} \in \mathcal{C}} [|\vec{y} - \vec{x}| + |d(\vec{x})|] \tag{8}$$

it is easy to show that $\xi(\vec{y}) \leq |d(\vec{y})|$. In fact,

$$|d(\vec{y})| = |\vec{y} - \vec{s}| = |\vec{y} - \vec{a}| + |\vec{a} - \vec{s}| = |\vec{y} - \vec{a}| + |d(\vec{a})| \geq \min_{\vec{x} \in \mathcal{C}} [|\vec{y} - \vec{x}| + |d(\vec{x})|] = \xi(\vec{y})$$

It remains to show that $\xi(\vec{y}) \geq |d(\vec{y})|$. To see this, let $\vec{z} \in \mathcal{C}$ satisfy $\xi(\vec{y}) = |\vec{y} - \vec{z}| + |d(\vec{z})|$ and let $\vec{w} \in \mathcal{S}$ satisfy $|d(\vec{z})| = |\vec{z} - \vec{w}|$. Then

$$\xi(\vec{y}) = |\vec{y} - \vec{z}| + |\vec{z} - \vec{w}| \geq |\vec{y} - \vec{w}| \geq |d(\vec{y})|$$

and the proof is complete. □

The previous proposition shows that the distance function can be computed "layer after layer", using the values computed on some surface $\mathcal{C}$ to compute the values of d at points that lie "outside" $\mathcal{C}$. With these new values, one can redefine $\mathcal{C}$ as the new boundary of the subdomain in which d is already known, and in this way march outwards from $\mathcal{S}$ until d is known everywhere. The idea of a *marching method*[7,8] has been made evident in the previous argument.

But what happens when one considers *perturbations* of the original problem? We have already recalled that the exact function d does not belong to V_h. Let $\mathcal{P}$ be the set of nodal points that are adjacent to the zero-level set of ϕ_h, in the sense that they are vertices of simplices inside which ϕ_h changes sign. If one makes the simple assignment $\tilde{\phi}_h(\vec{X}) = d(\vec{X})$ for all $\vec{X} \in \mathcal{P}$, there is a volume loss (or gain) which could render the algorithm useless for physical simulations. To be precise, assigning $\tilde{\phi}_h(\vec{X}) = d(\vec{X})$ results in $\int_{\phi_h(\vec{x})<0} d\vec{x} \neq \int_{\tilde{\phi}_h(\vec{x})<0} d\vec{x}$.

The values of $\tilde{\phi}_h$ at the nodes adjacent to the zero-level set must thus be "adjusted" so as to preserve volume, and the function $\tilde{\phi}$ must be calculated at the remaining nodes using the adjusted values at $\mathcal{P}$. In general, this "adjusted" distance $\tilde{\phi}_h$, linearly interpolated from the nodal values, is *not* the distance to some "adjusted" set $\mathcal{S}'$. This is a perturbation of the problem considered in Prop. 2.1, and it is appropriate to adapt the result to consider this case. Simultaneously, we will rephrase the result in a way that motivates the algorithm we are using (which is not a marching method).

Proposition 2.2 *Let $\mathcal{S}$ be a closed set in $\mathbb{R}^n$ and let ψ be a continuous, positive function on $\mathcal{S}$. Let us define*

$$\eta(\vec{y}) = \min_{\vec{x} \in \mathcal{S}} [\psi(\vec{x}) + |\vec{y} - \vec{x}|] \tag{9}$$

(notice that $\eta(\vec{y}) = |d(\vec{y})|$ if $\psi = 0$), and let $\vec{y}$ be arbitrary in $\mathbb{R}^n \setminus \mathcal{S}$. Then, for any surface $\mathcal{C}$ (as in Prop. 2.1) such that $\mathcal{S} \subset \omega^-$ and $\vec{y} \in \omega^+$,

$$\eta(\vec{y}) = \min_{\vec{x} \in \mathcal{C}} [\eta(\vec{x}) + |\vec{y} - \vec{x}|] \tag{10}$$

Proof: Let

$$\xi(\vec{y}) = \min_{\vec{x} \in \mathcal{C}} [\eta(\vec{x}) + |\vec{y} - \vec{x}|] \tag{11}$$

We must prove that $\eta(\vec{y}) = \xi(\vec{y})$. Let $\vec{s}$ verify $\eta(\vec{y}) = \psi(\vec{s}) + |\vec{y} - \vec{s}|$, and let $\vec{a}$ be the intersection of the segment $\overline{ys}$ with $\mathcal{C}$. From (9),

$$\eta(\vec{a}) \leq \psi(\vec{s}) + |\vec{a} - \vec{s}| = \psi(\vec{s}) + |\vec{y} - \vec{s}| - |\vec{y} - \vec{a}| = \eta(\vec{y}) - |\vec{y} - \vec{a}|$$

implying that

$$\eta(\vec{y}) \geq \eta(\vec{a}) + |\vec{y} - \vec{a}| \geq \xi(\vec{y})$$

It remains to prove that $\eta(\vec{y}) \leq \xi(\vec{y})$. Let $\vec{z} \in \mathcal{C}$ satisfy $\xi(\vec{y}) = \eta(\vec{z}) + |\vec{y} - \vec{z}|$ and let $\vec{w} \in \mathcal{S}$ satisfy $\eta(\vec{z}) = \psi(\vec{w}) + |\vec{z} - \vec{w}|$. Then

$$\xi(\vec{y}) = \psi(\vec{w}) + |\vec{z} - \vec{w}| + |\vec{y} - \vec{z}| \geq \psi(\vec{w}) + |\vec{y} - \vec{w}| \geq \eta(\vec{y})$$

which completes the proof. □

Remark 2.3 *In the proof above it is not difficult to see that $\vec{y}$, $\vec{w}$ and $\vec{z}$ are aligned. Let $\vec{b}$ be the intersection of the segment $\overline{yw}$ with $\mathcal{C}$. If $\vec{b} \neq \vec{z}$, then*

$$\xi(\vec{y}) = \psi(\vec{w}) + |\vec{y} - \vec{z}| + |\vec{z} - \vec{w}| > \psi(\vec{w}) + |\vec{b} - \vec{w}| + |\vec{y} - \vec{b}| \geq \eta(\vec{b}) + |\vec{y} - \vec{b}|$$

in contradiction with (10). The alignment implies that, if $\vec{s}$ is as before such that $\eta(\vec{y}) = \psi(\vec{s}) + |\vec{y} - \vec{s}|$, then for any $\vec{z}$ in the straight segment $\overline{ys}$, $\eta(\vec{z}) = \psi(\vec{s}) + |\vec{z} - \vec{s}|$. The curves that join each point of the domain with the minimizing argument of the right hand side of (9) are straight lines ("rays", by analogy with optics).

Remark 2.4 *It is also possible to prove that, at points where η is differentiable, $|\nabla \eta| = 1$ in much the same way as $|\nabla d| = 1$, which is a particular case corresponding to $\psi = 0$. To see this, first notice that along the "rays" the directional derivative of η is equal to one. It remains to prove that along a direction perpendicular to the local ray direction the derivative vanishes, assuming that it exists. Let $\vec{y}$ and $\vec{s}$ be as before, and let $\vec{D}$ be a unit vector orthogonal to the segment $\overline{ys}$. Because of the orthogonality,*

$$|\vec{y} + \epsilon \vec{D} - \vec{s}| = |\vec{y} - \vec{s}| + \frac{1}{2} \frac{\epsilon^2}{|\vec{y} - \vec{s}|} + \mathcal{O}(\epsilon^4)$$

Now, using also (9),

$$\eta(\vec{y} + \epsilon \vec{D}) \leq \psi(\vec{s}) + |\vec{y} + \epsilon \vec{D} - \vec{s}| \leq \psi(\vec{s}) + |\vec{y} - \vec{s}| + \mathcal{O}(\epsilon^2) = \eta(\vec{y}) + \mathcal{O}(\epsilon^2)$$

From this it is immediate that, if η is differentiable at $\vec{y}$, then its derivative along $\vec{D}$ must vanish.

2.2 Computing the distance

It is clear that the sign of $\tilde{\phi}_h$ adds no difficulty, since it simply equals that of ϕ_h. We will thus describe the calculation of $\tilde{\phi}_h$ just on Ω_A, where ϕ_h is positive. Let $\mathcal{P}$ be the set of nodal points that are adjacent to the zero-level set of ϕ_h, in the sense that they are vertices of simplices inside which ϕ_h changes sign, and let $\mathcal{P}_A$ be the subset of $\mathcal{P}$ with positive values of ϕ_h (i.e., $\mathcal{P}_A = \mathcal{P} \cap \Omega_A$). We assume $\tilde{\phi}_h$ given on $\mathcal{P}_A$ (its calculation is described later). The rest of the nodes in Ω_A is denoted by $\mathcal{R}_A$.

Step 1 (Initialization): There exist several options for initializing $\tilde{\phi}_h$ over $\mathcal{R}_A$.
a) Let I be a node in $\mathcal{R}_A$, and let C_I be the set of nodes connected to I, I not included (notice that $C_I \subset (\mathcal{P}_A \cup \mathcal{R}_A)$. The initial guess we use for $\tilde{\phi}_h$ is a distance-along-edges approximation, i.e., the unique function satisfying

$$\tilde{\phi}_h(\vec{X}_I) = \min_{J \in C_I} \left[\tilde{\phi}_h(\vec{X}_J) + |\vec{X}_I - \vec{X}_J| \right]$$

In the process of initializing $\tilde{\phi}_h$ with this option, the elements can be ordered so as to render the algorithm more effective.
b) If one wants to calculate $\tilde{\phi}_h$ up to a distance δ from $\mathcal{S}$, one simply initializes $\tilde{\phi}_h$ as equal to δ over $\mathcal{R}_A$.

Step 2 (Evaluation): The simplices in the mesh are swept until $\tilde{\phi}_h$ no longer changes. For each simplex, and for each node I of the simplex (coordinates denoted by $\vec{X}_I$), $\tilde{\phi}_h$ is interpolated linearly on the opposite face F_I, using the current values at the nodes. Then, a tentative new value η_I of $\tilde{\phi}_h$ at node I is calculated as

$$\eta_I = \min_{\vec{x} \in F_I} \left[\tilde{\phi}_h(\vec{x}) + |\vec{X}_I - \vec{x}| \right]$$

Finally, $\tilde{\phi}_h(\vec{X}_I)$ is updated to the value η_I *if the current value is greater than* η_I.

2.3 Volume preservation

The key in preserving volume in the algorithm is the correct computation of $\tilde{\phi}_h$ on the set of nodes adjacent to the interface (denoted by $\mathcal{P}$). Les us define $\mathcal{K}(\phi_h)$ as the set of simplices in which ϕ_h changes sign, so that $\mathcal{S} \subset \mathcal{K}(\phi_h)$. The objective is thus to calculate $\tilde{\phi}_h$ such that it approximates the signed distance d while at the same time preserving the volume

$$V(\phi_h) = \int_{\mathcal{K}(\phi_h)} H(\phi_h(\vec{x}))\, d\vec{x} \tag{12}$$

where H is the Heaviside function ($H(s) = 1$ if $s > 0$, $H(s) = 0$ otherwise). The contribution to this volume of each simplex $K \in \mathcal{K}(\phi_h)$ is

$$V_K(\phi_h) = \int_K H(\phi_h(\vec{x}))\, d\vec{x} \tag{13}$$

The algorithm is again structured in a sequence of steps.

Step 1 (Initialization): The function $\tilde{\phi}_h$ is initialized, over the nodes in $\mathcal{P}$, to a first estimate $\tilde{\phi}_h^0$, calculated as the true signed distance to $\mathcal{S}$.

Step 2 (Evaluation of a simplex-wise correction): In general, the initialization ends up with a function $\tilde{\phi}_h^0$ for which $V_K(\tilde{\phi}_h^0) \neq V_K(\phi_h)$, though the difference is quite small. We then solve, on K, the nonlinear system

$$R_K(\Delta_K) = V_K(\tilde{\phi}_h^0 + \Delta_K) - V_K(\phi_h) = 0 \tag{14}$$

to determine which (constant over K) value should be added to $\tilde{\phi}_h^0$ to achieve local volume preservation. The values Δ_K are computed using a simple secant algorithm $\Delta_K^{(i+1)} = \Delta_K^{(i)} - R_K^{(i)}(\Delta_K^{(i)} - \Delta_K^{(i-1)})/(R_K^{(i)} - R_K^{(i-1)})$, which converges in very few steps, and stored.

Step 3 (Node-wise correction): From the previous step, simplex-wise values that should be added to $\tilde{\phi}_h^0$ to preserve volume for all $K \in \mathcal{K}(\phi_h)$ are available. We now compute a node-wise direction ψ_h by averaging over the simplices that share a node. Let I be a node in $\mathcal{P}$, and let N_I be the number of simplices in $\mathcal{K}(\phi_h)$ that contain I, then we define

$$\psi_h(\vec{X}_I) = \frac{1}{N_I} \sum_{\substack{K \in \mathcal{K}(\phi_h) \\ I \in K}} \Delta_K \tag{15}$$

The value of $\tilde{\phi}_h$ on $\mathcal{P}$ is finally calculated over $\mathcal{P}$ as

$$\tilde{\phi}_h = \tilde{\phi}_h^0 + C\, \psi_h \ , \tag{16}$$

where C is the value of C such that

$$\int_{\mathcal{K}(\phi_h)} H(\tilde{\phi}_h^0(\vec{x}) + C\, \psi_h(\vec{x}))\, d\vec{x} = \int_{\mathcal{K}(\phi_h)} H(\phi_h(\vec{x}))\, d\vec{x} \tag{17}$$

The nonlinear system for C is again solved by a simple secant method and converges in very few iterations.

Remark 2.5 *The mass-conserving correction (16) is of the same kind as those proposed in other papers. A common approach is simply to propose $\psi_h = 1$, so that a uniform value is added to $\tilde{\phi}_h^0$ to correct the mass.[3] This is however not optimal, since volume loss/gain is not uniform over the interface. In fact, the loss/gain in volume tends to concentrate in regions of higher curvature. In other approaches the function $\tilde{\phi}$ is obtained as the steady state of the heuristic equation*

$$\frac{\partial \tilde{\phi}}{\partial t} + S(\phi)(|\nabla \tilde{\phi}| - 1) = 0, \qquad \tilde{\phi}(\vec{x}, 0) = \phi(\vec{x})$$

in which S is a sign function or a suitably smeared approximation thereof. We refer to the book by Osher and Fedkiw,[12] page 65, for a thorough discussion on reinitialization procedures. Our approach is similar to that proposed by Sussman and Fatemi for structured grids,[9] in that it proposes a function $\tilde{\phi}_h$ that has no free parameters and concentrates the correction locally where the loss/gain in volume is higher, thus automatically accounting for curvature, mesh distortion, etc.

3 NUMERICAL EXAMPLES

In this section we present a few numerical examples that focus on different aspects of the algorithm. We start with an assessment of the distance computing algorithm on different, arbitrary grids. We then show the effect of the redistancing algorithm on the transport of the so-called Zalezak's disk, a usual benchmark for Level Set formulations.[5,13]

3.1 Assessment of the distance calculation

We first consider the unit squate $(0,1) \times (0,1)$ as the domain Ω, in which we randomly put 4 circles and 4 squares. The diameter of the circles and the edge-length of the squares are randomly chosen between 0.08 and 0.16. All other variables (center position, orientation) are also random. In this way we generate 9 zero-level sets that are shown in Fig. 1. Notice that since overlapping of the circles and squares is allowed the sets $\mathcal{S}$ so generated are quite general, though composed of simple, convex shapes. We test the distance calculation algorithm for these zero-level sets on the three grids shown in Fig. 2. Mesh A is quasi-uniform unstretched with 10201 nodes and 20000 elements, Mesh B is locally refined unstretched with 10233 nodes and 20200 elements and Mesh C is both locally refined and stretched with 10073 nodes and 19888 elements. The distance $\tilde{\phi}_h$ obtained on Mesh A can be seen in Fig. 3 for each of the nine cases, plotted as isovalue contours. In Fig. 4 we compare the true distance to the computed distance $\tilde{\phi}_h$, for each mesh. In these plots, the nodal values for all nine cases are incorporated. It can be seen that the computed distance follows the true distance closely, with maximum differences that are of the order of the mesh size (equal to 0.01 for Mesh A, variable for the others). A similar comparison on 3D meshes, including locally refined and stretched grids. Sample results are shown in Fig. 5, which we consider satisfactory.

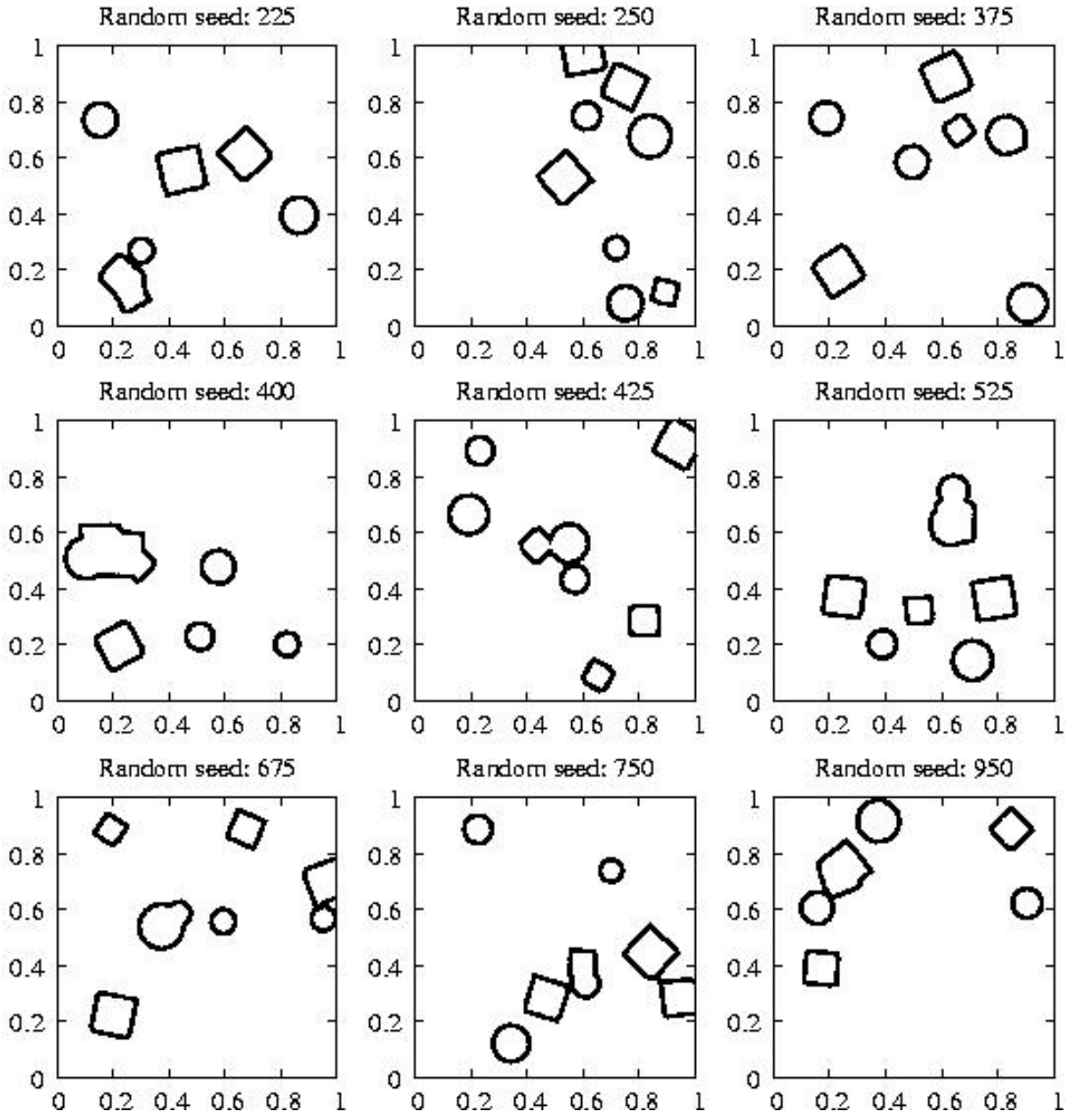

Figure 1: Nine zero-level sets used to the assessment of the distance calculation. Each ones contains 4 circles and 4 squares randomly put over domain.

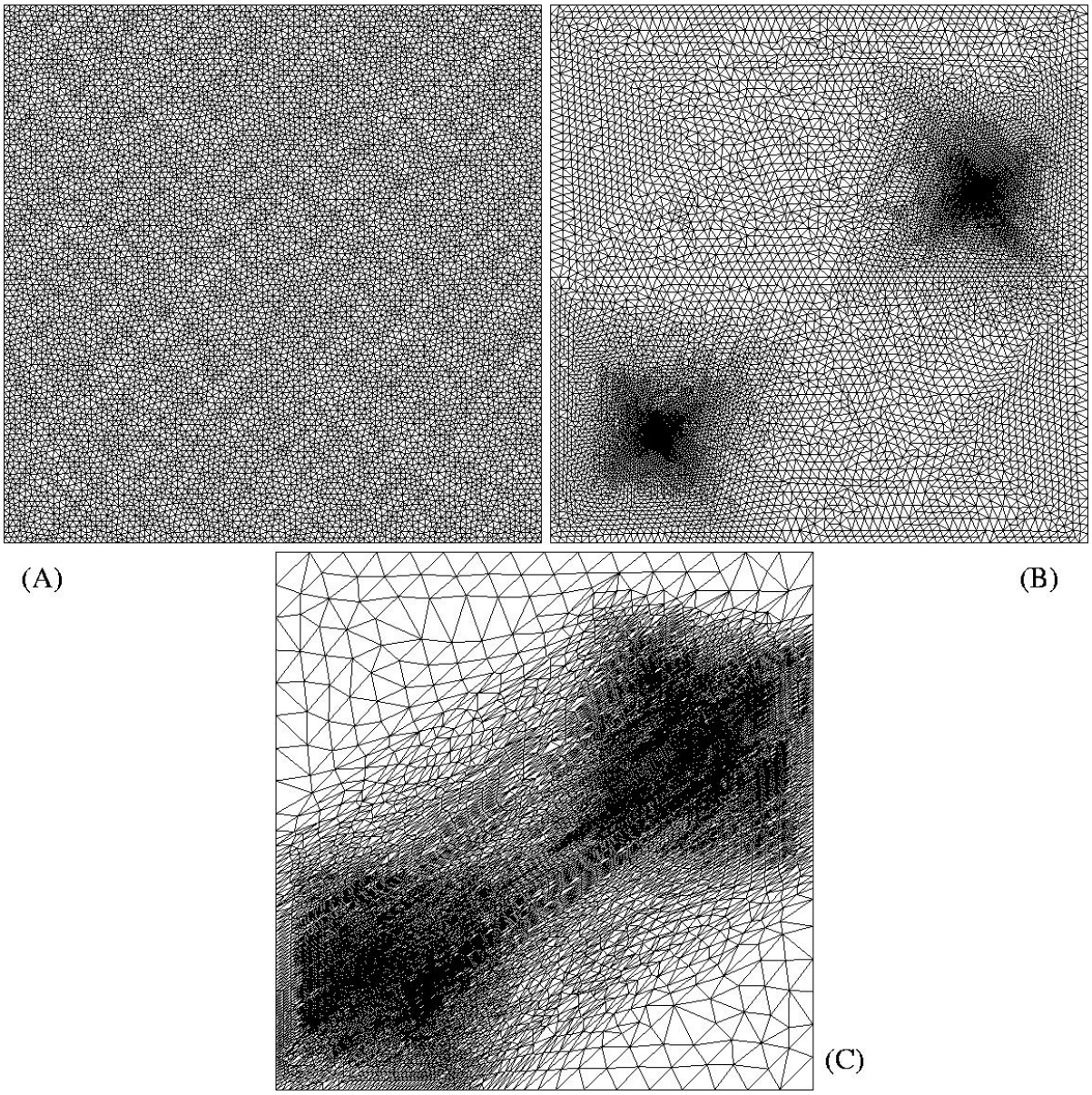

Figure 2: Three different grids used to "test" the distance calculation algorithm. Mesh A is quasi-uniform unstretched with 10201 nodes and 20000 elements, Mesh B is locally refined unstretched with 10233 nodes and 20200 elements and Mesh C is both locally refined and stretched with 10073 nodes and 19888 elements.

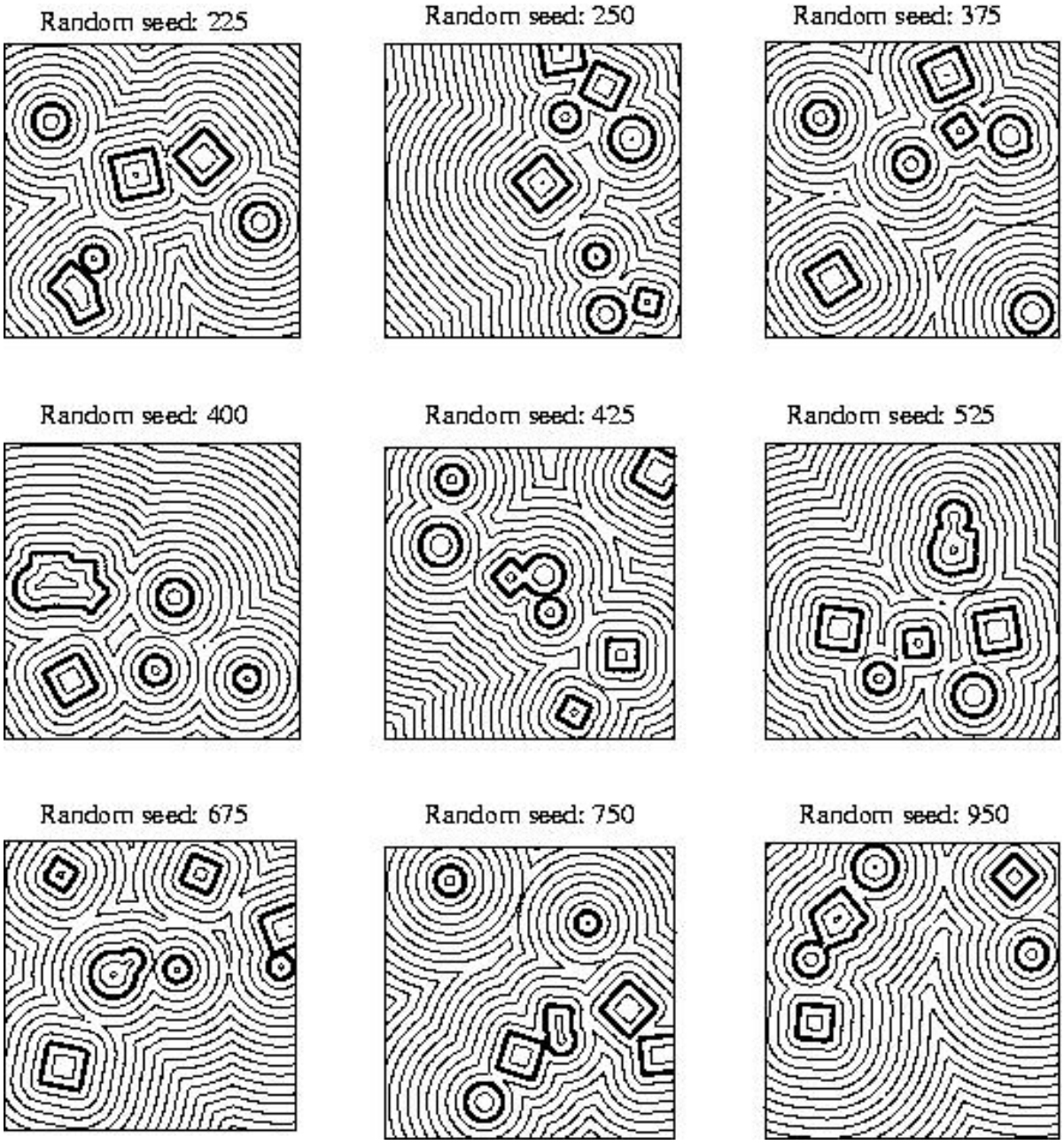

Figure 3: Isovalue contours of the distance function.

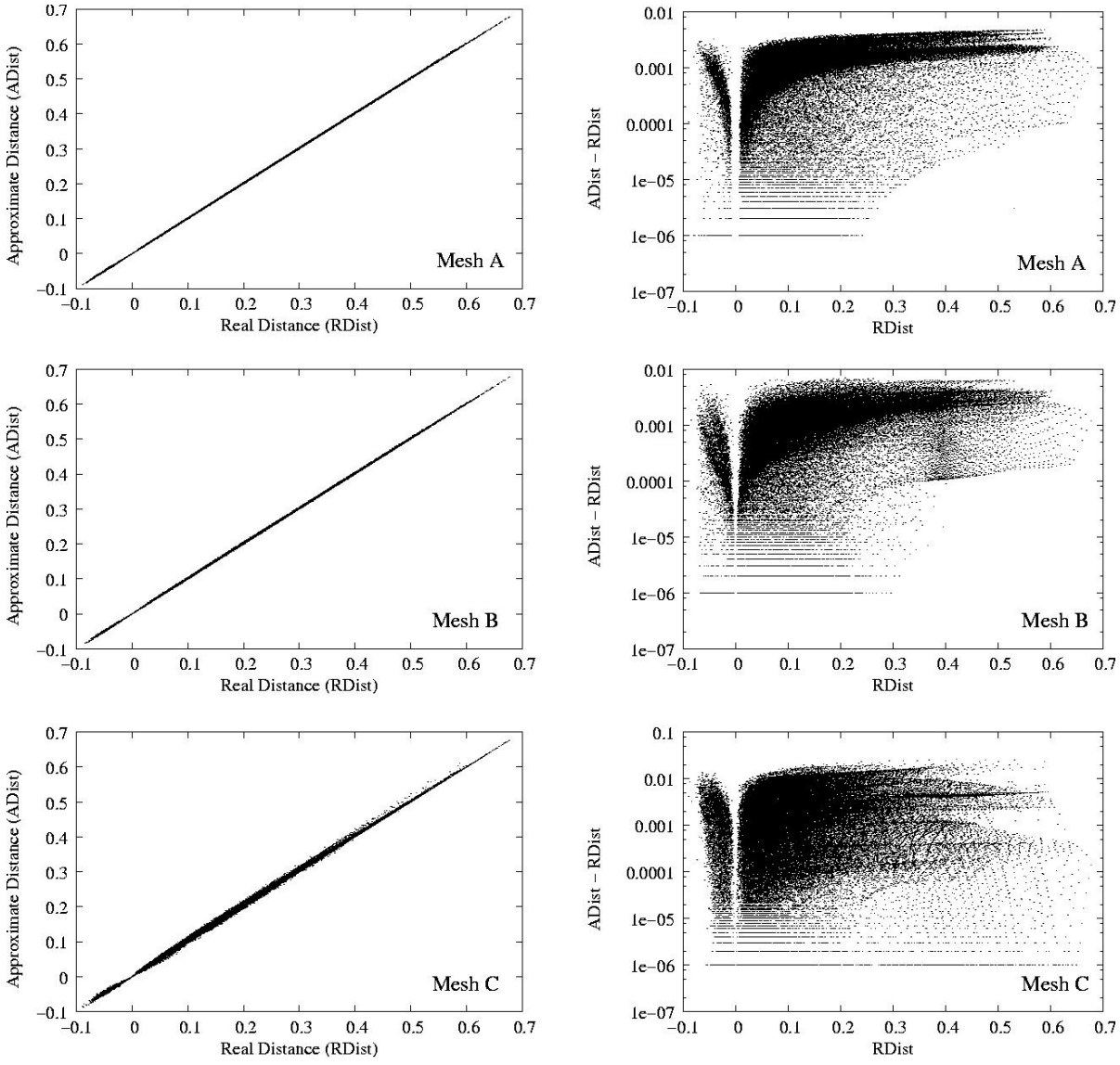

Figure 4: Comparison between true distance and computed distance in three different meshes.

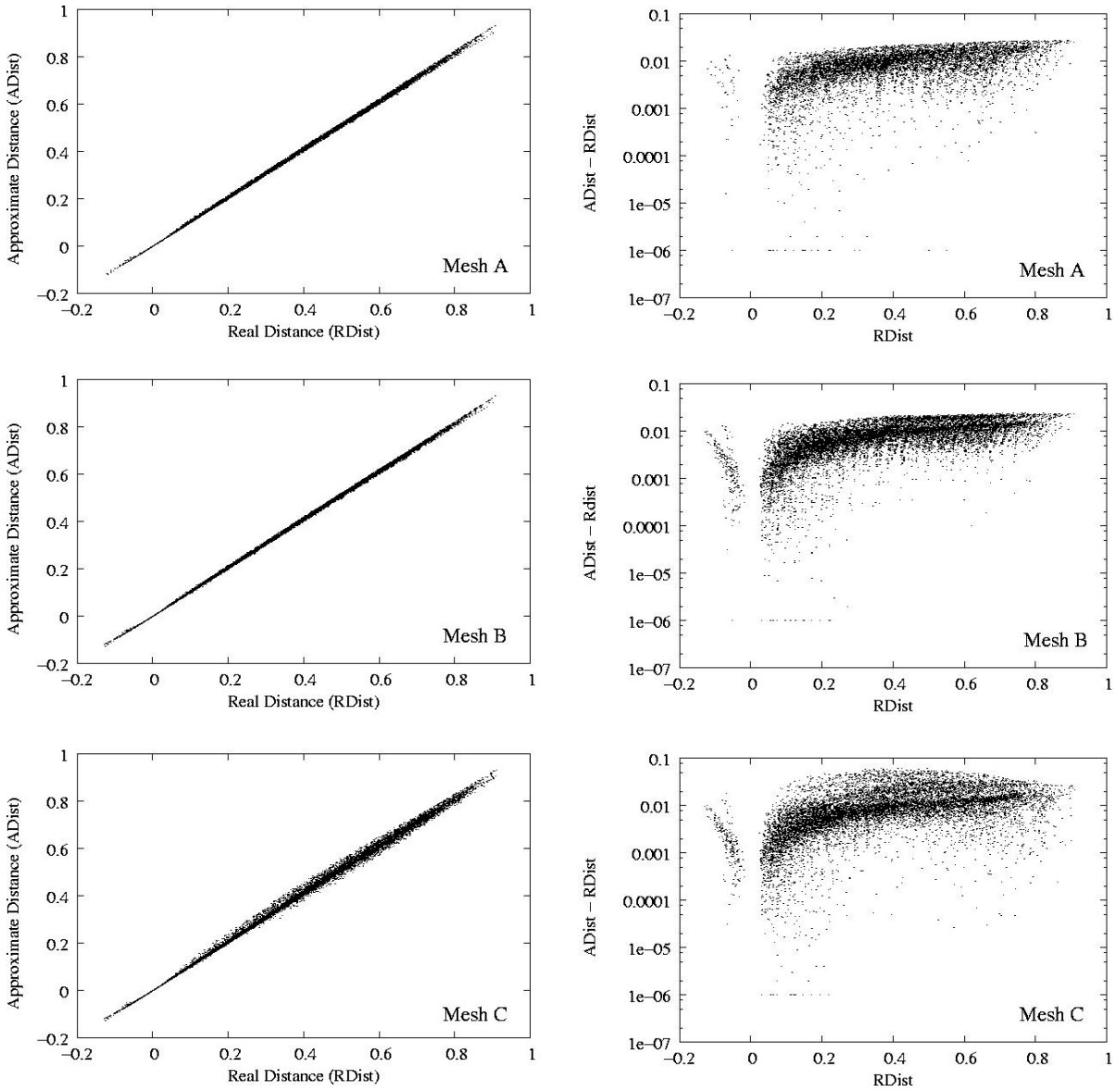

Figure 5: Comparison between true distance and computed distance in three different 3D meshes. Mesh A is quasi-uniform unstretched with 11145 nodes and 54245 elements, Mesh B is locally refined unstretched with 13998 nodes and 77626 elements and Mesh C is both locally refined and stretched with 13552 nodes and 72909 elements.

3.2 A pure transport calculation

In this paragraph we couple the redistancing algorithm with a numerical method for transport. The problem we consider is the rigid body rotation of the so-called Zalezak's disk. The domain is $\Omega = (0, 100) \times (0, 100)$. The initial data correspond to fluid A inside a slotted circle centered at $(50, 75)$ with a radius of 15. The slot length is 25 and its width 5. The velocity field is given by

$$\vec{\beta} = \frac{\pi}{314} (50 - x_2, x_1 - 50)$$

so that the disk completes one revolution every 628 time units.

The numerical method we use for (3) is a Streamline Upwind Petrov-Galerkin Finite Element method, with Crank-Nicolson treatment of the time derivative. Let V_h be the space of piecewise linear functions on the triangulation $\mathcal{T}_h$ of Ω. Let ϕ_h^0 be the initial data for the level set function, such that it is positive inside the disk and negative outside it.

To make explicit the numerical method, let ϕ_h^n denote the numerical solution at time $t = n\,\Delta t$, where Δt is the time step. Then, assuming ϕ_h^n given, ϕ_h^{n+1} is obtained as the unique solution in V_h of the discrete variational formulation

$$\int_\Omega \left\{ \left[\frac{\phi_h^{n+1} - \mathcal{G}(\phi_h^n)}{\Delta t} + \frac{1}{2}\,\vec{\beta} \cdot \nabla \left(\phi_h^{n+1} + \mathcal{G}(\phi_h^n) \right) \right] \left(v_h + \tau\,\vec{\beta} \cdot \nabla v_h \right) \right\} \, d\vec{x} = 0 \qquad (18)$$

for all v_h in V_h. In (18) we have introduced the mapping $\mathcal{G} : V_h \to V_h$, which allows us to incorporate the re-initialization step. We thus compare the plain algorithm ($\mathcal{G}(\phi_h) = \phi_h$) to the re-initialized one ($\mathcal{G}(\phi_h) = \tilde{\phi}_h$). Also in (18), τ is the SUPG characteristic time, which we calculate as $\tau = h/(4|\vec{\beta}|)$, with h the local mesh size.

In addition, we calculate the accuracy of the interface location using a first-order accurate error measure.[5]

$$\frac{1}{L} \int |H(\phi_{expected}) - H(\phi_{computed})| dxdy \qquad (19)$$

where L is the lenght of the expected interface. This integral is calculated exactly.

We used a uniform unstretched triangulated domain of 80000 elements for all tests. The distance algorithm was set to compute $\tilde{\phi}_h$ up to a distance of 4 from the zero-level set (Zalesak's disk). Fig. 6 shows the disk at 4 different instants: $t_1 = 0$, $t_2 = 157$, $t_3 = 314$ and $t_4 = 471$. In this example we can see an evident distortion of the iso-surface only due to the transport.

In Fig. 7 we show the evolution of the volume of fluid and the L1-distance in time for $\Delta t = 0.5$ and $\theta = 0.5$. This figure also contains a comparison between initial and final states. In the exact solution the volume is preserved, since *div* $\vec{\beta} = 0$.

Cases (a), where no-reinitialization is present, and (b), where reinitialization is done with the mass-correction step, show a similar behavior. This is mainly because the time step and mesh size are small enough for the transport of the level-set to be done without significant error. However, case (c) where the mass-correction step is not present, shows that the errors introduced in the reinitialization step accumulate destroying the global mass conservation. From the figure we

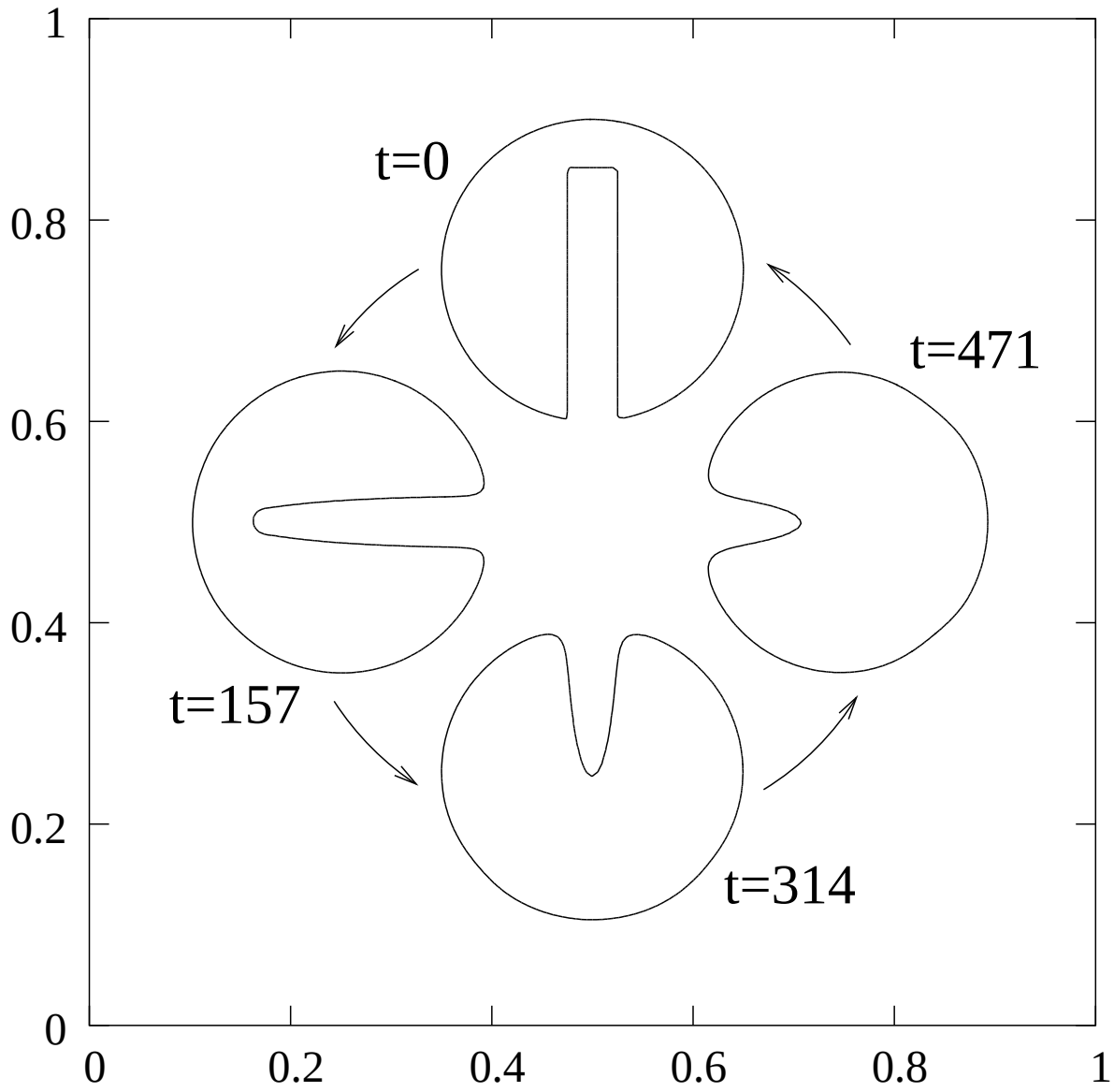

Figure 6: Zalesak's disk at 4 different instants: $t_1 = 0, t_2 = 157, t_3 = 314$ and $t_4 = 471$. The first one corresponds to the initial state. A uniform unstructured unstretched grid of 80000 elements was used. $\Delta t = 1$ and $\theta = 0.5$ was set in the transport algorithm.

can see that these errors are located in zones with greater curvature. Some authors recommend to restrict redistancing to a minimum because of this phenomenon. This example shows that our mass-conservation procedure allows redistancing at all time steps without problem. Moreover, it shows that this mass-correction step must be done in a *local sense*.

The second example was done with a fully-implicit scheme replacing the Crank-Nicolson presented before. The idea is to investigate the effects of more diffusive transport algorithms. Fig. 8 shows the evolution of the volume of fluid and of the L1-distance in time for $\Delta t = 0.5$.

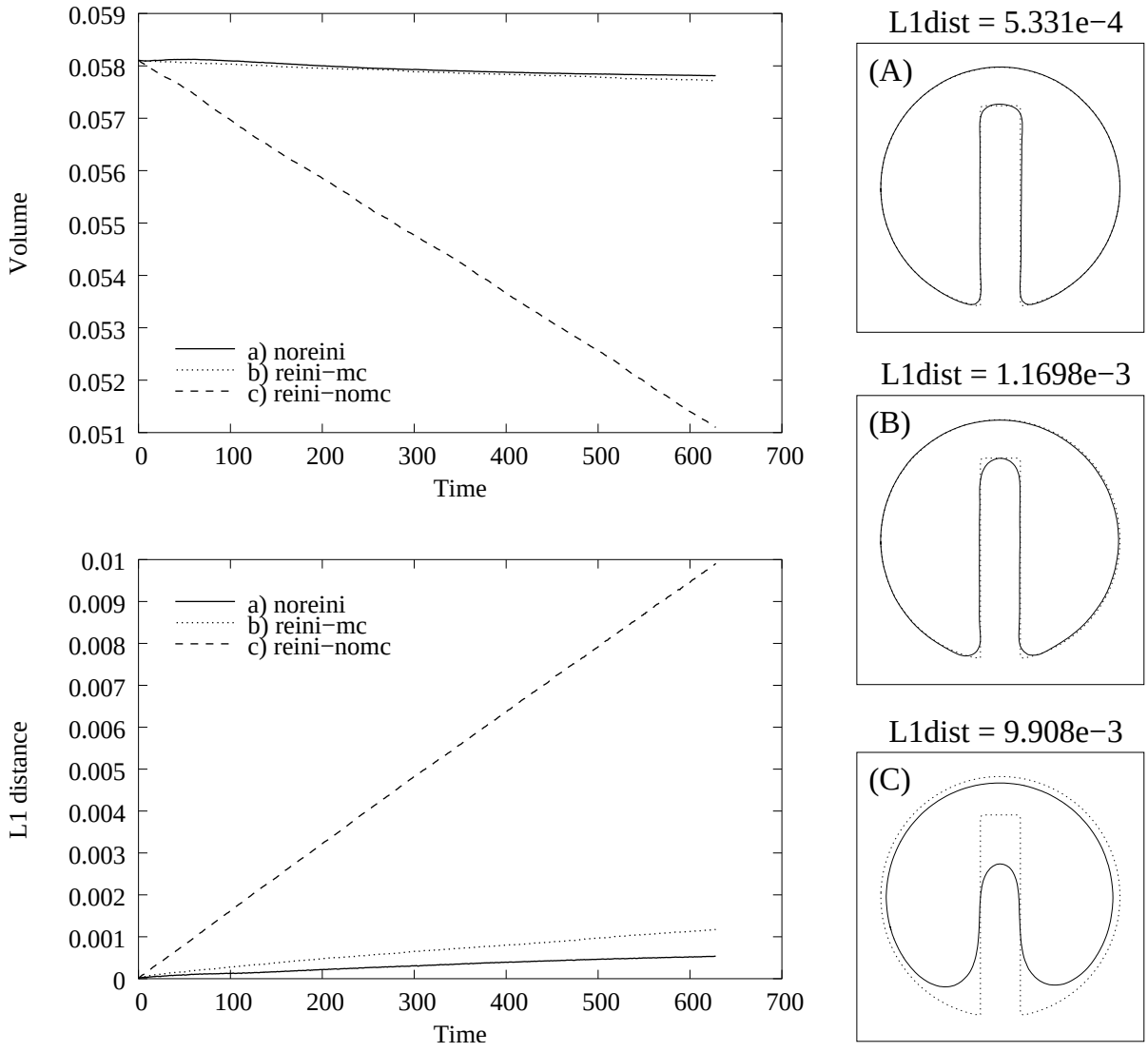

Figure 7: Zalesak's disk: Volume and L1-distance evolution for $\Delta t = 0.5$ and $\theta = 0.5$. Comparison between initial and final states (after one revolution). Domain is a uniform unstructured unstretched grid of 80000 elements.

From the comparison between initial and final states it is evident that the diffusive behavior is concentrated in zones with greater curvature. This is consistent with the increase of the volume of fluid in case (A), since the slot quickly disappears. In case (B), reinitialization of the LS reduces the damage to the slot reducing the global mass loss. The geometry is also preserved much more accurately.

Finally, the last test was thought to show the effect of reinitialization with large time steps. In Fig. 9 we show the evolution of the volume of fluid and the L1-distance in time for $\Delta t = 2$ (Crank-Nicolson scheme).

In case (A) the total mass loss is 1.35%, and in case (B) the amount of mass loss is 0.19%.

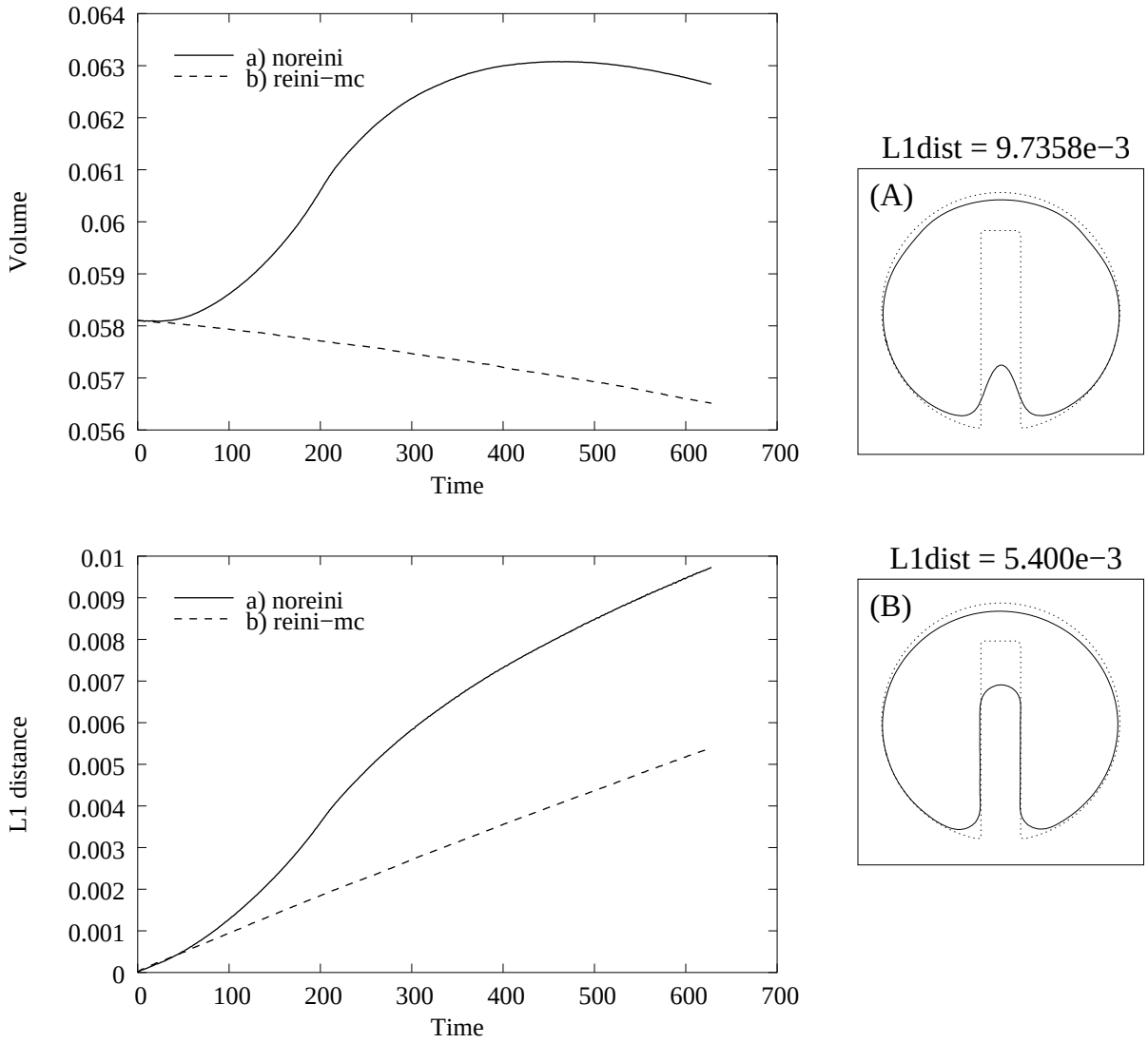

Figure 8: Zalesak's disk: Volume and L1-distance evolution for $\Delta t = 0.5$ and fully-implicit scheme. Comparison between initial and final states (after one revolution). Domain is a uniform unstructured unstretched grid of 80000 elements.

From the comparison between initial and final states we see again that the reinitialization step helps preserve the original geometry.

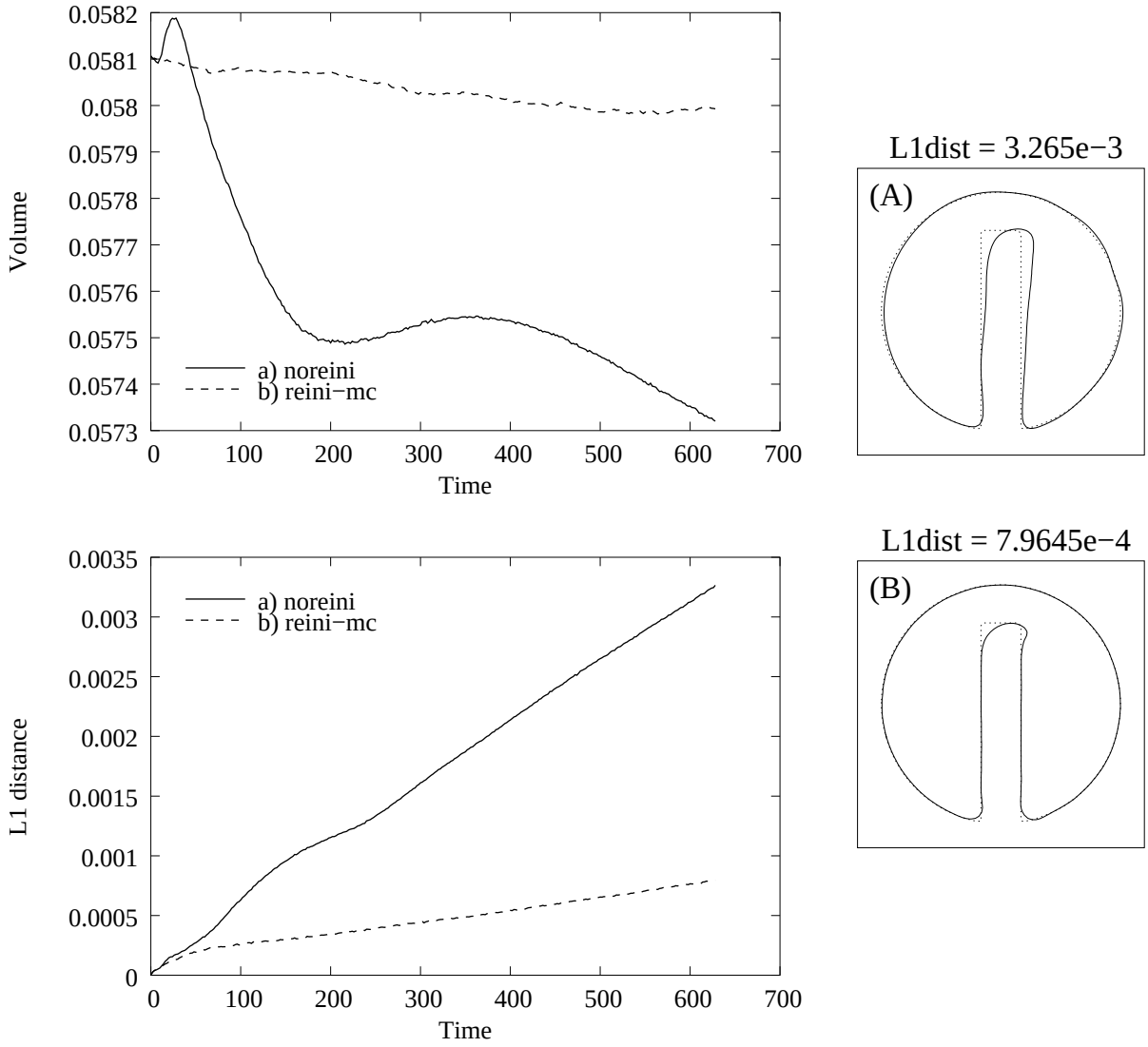

Figure 9: Zalesak's disk: Volume and L1-distance evolution for $\Delta t = 2$ and Crank-Nicolson scheme. Comparison between initial and final states (after one revolution). Domain meshed with a uniform unstructured unstretched grid of 80000 elements.

4 CONCLUSIONS

In this work we developed an algorithm to compute the signed distance function on general unstructured meshes in two and three dimensions. A mass-rebalancing step, which is weighted according to local mass differences, is proposed in the general algorithm in order to maintain the total mass during the reinitialization step.

A first set of tests was done to measure the errors in the estimation of distance. All tests show that the errors are bounded by the mesh size.

Then, we use the well-know Zalezak's disk test to focus on the general behavior of the algorithm in transport cases. We find that a mass-correction step during the reinitialization is needed, and moreover, that this step must be done in a local sense in order to preserve the original geometry. In addition, we observe that when numerical methods used to solve the transport equation have significant errors (e.g. excessive diffusion) the reinitialization step helps to both preserve the geometry and to reduce global mass loss. One should however point out that the techniques proposed in this article only work if the mesh is fine enough. For too-coarse meshes in which the error is governed by spatial discretization no improvement is found by applying the techniques proposed in this article.

ACKNOWLEDGMENTS: This work was partially supported by ANPCyT through grants PICT 12-6337 and 12-9848.

REFERENCES

[1] Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher. A level set formulation of eulerian interface capturing methods for incompressible fluid flows. *Journal of Computational Physics*, **124**, 449–464 (1996).

[2] D. Adalsteinsson and J.A. Sethian. The fast construction of extension velocities in level set methods. *Journal of Computational Physics*, **148**, 2–22 (1999).

[3] S. Aliabadi and T.E. Tezduyar. Stabilized-finite-element/interface-capturing technique for parallel computation of unsteady flows with interfaces. *Comput. Methods Appl. Mech. Engrg.*, **190**, 243–261 (2000).

[4] O. Soto and R. Codina. A numerical model for mould filling using a stabilized finite element method and the vof technique. *Int. J. Numer. Meth. Fluids*, (2000).

[5] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A hybrid particle level set method for improved interface capturing. *Journal of Computational Physics*, **183**, 83–116 (2002).

[6] D. Lakehal, M. Meier, and M. Fulgosi. Interface tracking towards the direct simulation of heat and mass transfer in multiphase flows. *Int. J. Heat and Fluid Flow*, **23**, 242–257 (2002).

[7] J. A. Sethian. Fast marching methods and level set methods for propagating interfaces. *von Karman Institute Lecture Series, Computational Fluid Mechanics*, (1998).

[8] Timothy J. Barth and J. A. Sethian. Numerical schemes for the hamilton-jacobi and level set equations on triangulated domains. *Journal of Computational Physics*, **145**, 1–40

(1998).

[9] M. Sussman and E. Fatemi. An efficient, interface-preserving level-set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM J. Sci. Comput.*, **20**(4), 1165–1191 (1999).

[10] D. L. Chopp. Some improvements of the fast marching method. *SIAM J. Sci. Comput.*, **23**(1), 230–244 (2001).

[11] J. A. Sethian and P. Smereka. Level set methods for fluid interfaces. *Annu. Rev. Fluid Mech.*, **35**, 341–72 (2003).

[12] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*, volume 153. Springer, (2003).

[13] M. Sussman and E. Fatemi. An efficient, interface-preserving level set redistancing algorithm and its aplication to interfacial incompressible fluid flow. *SIAM J. Sci Comput.*, **20**(4), 1165–1191 (1999).