Journal of Computational Physics 227 (2008) 9643–9669

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Front tracking with moving-least-squares surfaces

João Paulo Gois *, Anderson Nakano, Luis Gustavo Nonato, Gustavo C. Buscaglia

Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, Brazil

A R T I C L E   I N F O

Article history:
Received 20 December 2007
Received in revised form 3 July 2008
Accepted 23 July 2008
Available online 7 August 2008

Keywords:
Moving interface
Free surface
Front tracking
Lagrangian method
Clouds of points
Moving least squares

A B S T R A C T

The representation of interfaces by means of the algebraic moving-least-squares (AMLS) technique is addressed. This technique, in which the interface is represented by an unconnected set of points, is interesting for evolving fluid interfaces since there is no surface connectivity. The position of the surface points can thus be updated without concerns about the quality of any surface triangulation. We introduce a novel AMLS technique especially designed for evolving-interfaces applications that we denote RAMLS (for Robust AMLS). The main advantages with respect to previous AMLS techniques are: increased robustness, computational efficiency, and being free of user-tuned parameters.

Further, we propose a new front-tracking method based on the Lagrangian advection of the unconnected point set that defines the RAMLS surface. We assume that a background Eulerian grid is defined with some grid spacing h. The advection of the point set makes the surface evolve in time. The point cloud can be regenerated at any time (in particular, we regenerate it each time step) by intersecting the gridlines with the evolved surface, which guarantees that the density of points on the surface is always well balanced. The intersection algorithm is essentially a ray-tracing algorithm, well-studied in computer graphics, in which a line (ray) is traced so as to detect all intersections with a surface. Also, the tracing of each gridline is independent and can thus be performed in parallel.

Several tests are reported assessing first the accuracy of the proposed RAMLS technique, and then of the front-tracking method based on it. Comparison with previous Eulerian, Lagrangian and hybrid techniques encourage further development of the proposed method for fluid mechanics applications.

© 2008 Elsevier Inc. All rights reserved.

1. Introduction

We address in this article the problem of modeling the time evolution of an interface $\mathscr{S}(t)$ which separates two fluids (A and B), with the possibility of fluid B being the ambient air (the free-surface case). The problem takes place inside a finite domain Ω, with boundary $\partial\Omega$.

Purely Lagrangian methods for modeling moving interfaces consist of seeding the interface with marker particles and moving the particles as dictated by the velocity field. These methods have proved to be of high accuracy in many published studies [21,45,44,14,35], however with three drawbacks:

- It is difficult to simulate breakup and merging processes of the surface. Topology changes are hard to handle.
- It is difficult to keep the density of particles consistent with the desired level of discretization. They accumulate at some areas while other areas get depleted of particles. Effective particle creation and deletion strategies are needed to handle this issue.

* Corresponding author. Tel.: +55 1681557996.
E-mail addresses: jpgois@icmc.usp.br, jpgois@gmail.com (J.P. Gois), nakano@icmc.usp.br (A. Nakano), gnonato@icmc.usp.br (L.G. Nonato), gustavo.-buscaglia@icmc.usp.br (G.C. Buscaglia).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.07.013

- Most purely Lagrangian methods maintain the connectivity of the particles [45,44,35], needed to reconstruct the free surface from the scattered particles that move with it. This connectivity defines a mesh of the moving interface. If this mesh gets too distorted the reconstruction becomes unphysical, leading to collapse of the simulation.

The aforementioned drawbacks of Lagrangian methods have made Eulerian methods, such as the volume-of-fluid (VOF) method [20,32] or the level-set (LS) method [30,33,37,27,29,34] (or combinations thereof [36,39]) to be preferred in the modeling of complex interfaces undergoing topological changes such as bubble coalescence, wave breaking, etc. [40,38,47,8,9]. Eulerian methods are based on the advection of a scalar field ϕ defined on the whole flow domain. In the VOF method this scalar field represents the partial content of fluid A in each grid cell, whereas in the LS method ϕ implicitly defines the interface as its zero-level set. The improved flexibility of Eulerian methods, however, comes at the expense of a loss in accuracy due to interpolation errors, together with numerical errors in the transport of ϕ.

The previous considerations have motivated the search of numerical methods that combine the accuracy of Lagrangian methods with the flexibility of Eulerian ones. Some developments in this direction are related to the method proposed in this article.

Du et al. [11] combined a purely Lagrangian method with a grid-based reconstruction method which is applied locally in space and time where topological difficulties arise. Since the marker particles have an associated connectivity structure, the reconstruction of tangled regions of the interface is quite sophisticated, especially in three dimensions.

Torres and Brackbill [43] developed a point-set method in which front tracking was performed without a connectivity structure. Their method evolves marker particles according to the velocity field, and then builds a level-set-like function ϕ by solving Laplace's equation on Ω. By prescribing $\phi = 1$ at cells containing marker particles and $\phi = 0$ at $\partial\Omega$, the resulting function ϕ equals one inside $\mathscr{S}$, identifying the region occupied by fluid A. A smoothing procedure based on B-splines, followed by a correction step, are then applied to ϕ so that one of its level sets passes through the marker particles. They then regenerate interfacial points as projections of cell centers of an auxiliary finer grid onto the level set of ϕ.

Enright et al. [12], on the other hand, start from an Eulerian level-set method and improve it by incorporating the information about the location of the interface carried by a set of Lagrangian particles. More specifically, the information is incorporated by merging the interface that results from the Eulerian method with spheres centered at the Lagrangian particles which, at the beginning of the time step, are tangent to the interface. They adopt a surface merging technique that is well-established in computer graphics applications. Their method, known as particle level set (PLS) method, has proved quite successful in many applications [24,29], since topology changes are handled easily by the Eulerian part of the algorithm, which is a level-set method. Further, the PLS method does not require highly accurate Eulerian solvers for the level-set transport equation [13], since the Lagrangian part corrects the inaccuracies. Another method that uses Lagrangian particles to update the interface is the Lagrangian particle level-set method of Hieber and Koumoutsakos [16], in which a method based on smooth particle hydrodynamics (SPH) is implemented.

In this article, we explore the potential of moving-least-squares (MLS) [22] implicit representation of surfaces from point clouds, in the context of modeling interface motion. The proposed method can be viewed as a variant of the front-tracking method of Du et al. [11] without any connectivity of the marker particles. The maker particles follow a Lagrangian motion, but the interface is not assumed to exactly pass through them. Instead, the particles define the interface in an MLS sense. This methodology avoids the cost involved in the linear systems solved by Torres and Brackbill [43] to build ϕ. Further, from this implicit MLS representation of $\mathscr{S}(t)$ we simultaneously perform the re-generation of points and the construction of a level-set-like function ϕ that indicates the region occupied by each fluid. This is done by performing a ray-tracing-like [1,46] detection of $\mathscr{S}(t)$ along the gridlines of a fixed Cartesian mesh $\mathscr{T}_h$. The grid size h of $\mathscr{T}_h$ is the only parameter that governs the approximation of $\mathscr{S}(t)$. The distance between marker particles is automatically kept of order h, and features of dimension smaller than h are automatically ignored as in Eulerian methods, but the tracking of the interface remains Lagrangian and thus highly accurate. As compared to the PLS method, on the other hand, the cost of the Eulerian step is avoided and the particles are much less in number and more easily regenerated, since they actually lie on the surface instead of at the centers of tangent spheres.

The plan of this article is as follows: in Section 2 we remind the definitions of implicit algebraic MLS surfaces and perform some convergence tests to evaluate their accuracy. In Section 3 we introduce a parameter-free variant of algebraic MLS surfaces with improved robustness. Section 4 describes the proposed front-tracking algorithm based on MLS surfaces and provides details of its implementation. Extensive numerical testing is reported in Section 5, while Section 6 is devoted to the conclusions.

2. Implicit algebraic moving-least-squares surfaces

2.1. Basic definitions

Moving-least-squares (MLS) [22] is a method of producing continuous functions from a set of unorganized sampled point values based on the calculation of a weighted-least-squares approximation. In computer graphics, MLS is being used to produce smooth surfaces from point clouds, defining the *MLS surfaces* [4]. Firstly proposed by Alexa et al. [3] as the set of fixed points of the Levin's projection [23], MLS surfaces are becoming a well-established meshless method for modeling and rendering point clouds. Our scope is related to the implicit version of MLS surfaces, in which the resulting surface is given as the zero-level set of a so-called *implicit function* $F \in C^0(U)$, where $U \subset \mathbb{R}^n$ is an open neighborhood of the point cloud.

Let us consider a finite set of points $\mathscr{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_m\}$, and assume that the points are almost regularly spaced at distance h. We aim at defining a function F such that its zero-level set is a surface $\mathscr{S}$ that approximates the point cloud $\mathscr{P}$. The first step is to define, at any given point $\mathbf{x} \in U$,

$$F(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k(\mathbf{x}) q_k(\mathbf{x}), \tag{1}$$

where the basis $\{q_k\}_{k=1,N}$ consists of polynomials. We will concentrate on the specific choice considered by Guennebaud and Gross [15]:

$$N = 5, \quad q_1(\mathbf{x}) = 1, \quad q_2(\mathbf{x}) = x_1, \quad q_3(\mathbf{x}) = x_2, \quad q_4(\mathbf{x}) = x_3, \quad q_5(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2. \tag{2}$$

Remark 1. The specific basis given by (2) leads to an approximation based on *three-dimensional spheres*. Though the method will be detailed for this approximation, it is evident how to modify it for two-dimensional cases (*circles*), or for approximations based in *planes* (3D) or *straight lines* (2D). For example, in this latter case $N = 3$ and the basis is given by q_1, q_2 and q_3 above. We later on show examples of these alternative bases.

Denoting by $\underline{\alpha}$ the N-tuple of functions $(\alpha_1, \alpha_2, \ldots, \alpha_N)$, it is clear that F is completely defined by $\underline{\alpha}$ through Eq. (1). Whenever this dependence needs to be made explicit, we indicate $\underline{\alpha}$ as a subindex; e.g., $F_{\underline{\alpha}}$. Each $\underline{\alpha}$ (remember that it consists of functions of $\mathbf{x}$) also defines, implicitly, the surface $\mathscr{S}_{\underline{\alpha}}$ given by

$$\mathscr{S}_{\underline{\alpha}} = \{\mathbf{x} \in U | F_{\underline{\alpha}}(\mathbf{x}) = 0\}. \tag{3}$$

We now address the choice of $\underline{\alpha}(\mathbf{x})$. This choice should make the surface $\mathscr{S}_{\underline{\alpha}}$ to be close to the set of points $\mathscr{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_m\}$. For this purpose, we will adopt the algebraic MLS approach of Guennebaud and Gross [15].

2.2. Implicit algebraic least-squares spheres

Consider first the case in which each α_k is a constant. In this case, $\mathscr{S}_{\underline{\alpha}}$ is a sphere. Denoting by $d(\mathbf{p}_i, \mathscr{S}_{\underline{\alpha}})$ the (signed) geometric distance from $\mathbf{p}_i$ to this sphere, an optimal set of coefficients $\underline{\alpha}^*$ can be defined as

$$\underline{\alpha}^* = \arg\min_{\underline{\beta} \in \mathbb{R}^N} J(\underline{\beta}), \quad \text{with} \quad J(\underline{\beta}) = \sum_{i=1}^{m} w_i |d(\mathbf{p}_i, \mathscr{S}_{\underline{\beta}})|^2, \tag{4}$$

where we have introduced the set of non-negative weights $\{w_i\}_{i=1,\ldots,m}$ for later use. For the time being, the reader may think of all w_i's as being equal to one for simplicity.

Unfortunately, the minimization of J given by (4) is numerically inconvenient. The algebraic approach [31,42], instead, defines $\underline{\alpha}$ as

$$\underline{\alpha} = \arg\min_{\underline{\beta} \in Q} \widetilde{J}(\underline{\beta}), \quad \text{with} \quad \widetilde{J}(\underline{\beta}) = \sum_{i=1}^{m} w_i |F_{\underline{\beta}}(\mathbf{p}_i)|^2 \tag{5}$$

subject to

$$Q = \{\underline{\beta} \in \mathbb{R}^N | \beta_2^2 + \beta_3^2 + \beta_4^2 - 4\beta_1\beta_5 = 1\}. \tag{6}$$

The rationale behind this is that, on $\mathscr{S}_{\underline{\beta}}$,

$$\|\nabla F_{\underline{\beta}}\|^2 = \beta_2^2 + \beta_3^2 + \beta_4^2 - 4\beta_1\beta_5 \tag{7}$$

and thus $F_{\underline{\beta}}(\mathbf{p}_i) = d(\mathbf{p}_i, \mathscr{S}_{\underline{\beta}}) + \mathcal{O}(|d(\mathbf{p}_i, \mathscr{S}_{\underline{\beta}})|^2)$ for any $\underline{\beta} \in Q$. As a consequence, *if the set of points $\mathscr{P}$ is not very far from a sphere*, the minimization process defined by (5) and (6) will yield $\underline{\alpha}$ such that $\mathscr{S}_{\underline{\alpha}}$ is very close to the "optimal" sphere $\mathscr{S}_{\underline{\alpha}^*}$ defined by (4). Moreover, for any $\mathbf{x}$ sufficiently close to $\mathscr{S}_{\underline{\alpha}}$ the value $F_{\underline{\alpha}}(\mathbf{x})$ will approximate the signed distance $d(\mathbf{x}, \mathscr{S}_{\underline{\alpha}})$; i.e.,

$$F_{\underline{\alpha}}(\mathbf{x}) = d(\mathbf{x}, \mathscr{S}_{\underline{\alpha}}) + \mathcal{O}(|d(\mathbf{x}, \mathscr{S}_{\underline{\alpha}})|^2). \tag{8}$$

The advantage of the modified minimization problem (5) and (6) is that its solution is straightforward [41,42]. In fact, since

$$F_{\underline{\beta}}(\mathbf{p}_i) = \sum_{k=1}^{N} \beta_k q_k(\mathbf{p}_i), \tag{9}$$

it follows that

$$\begin{aligned}\widetilde{J}(\underline{\beta}) &= \sum_{i=1}^{m} w_i \left| \sum_{k=1}^{N} \beta_k q_k(\mathbf{p}_i) \right|^2 \\ &= \sum_{k,\ell=1}^{N} \beta_k \beta_\ell \left(\sum_{i=1}^{m} w_i q_k(\mathbf{p}_i) q_\ell(\mathbf{p}_i) \right) \\ &= \underline{\beta}^{\mathrm{T}} \underline{\underline{M}} \underline{\beta}, \end{aligned} \tag{10}$$

where we have introduced the matrix $\underline{\underline{M}}$,

$$M_{k\ell} = \sum_{i=1}^{m} w_i q_k(\mathbf{p}_i) q_\ell(\mathbf{p}_i), \tag{11}$$

independent of $\underline{\beta}$, to make evident that $\widetilde{J}$ is a quadratic form in $\mathbb{R}^N$.

Further, notice that the constraint in the definition of Q (Eq. (6)) is also quadratic [41,42,15]. In fact, defining a matrix $\underline{\underline{C}}$ in which the only non-zero elements are

$$C_{22} = C_{33} = C_{44} = 1, \quad C_{15} = C_{51} = -2$$

we can recast the minimization problem (5) and (6) as

$$\underline{\alpha} = \arg\min_{\substack{\underline{\beta} \in \mathbb{R}^N \\ \underline{\beta}^{\mathrm{T}} \underline{\underline{C}} \underline{\beta} = 1}} \underline{\beta}^{\mathrm{T}} \underline{\underline{M}} \underline{\beta} \tag{12}$$

Denoting by λ the Lagrange multiplier associated to the constraint $\underline{\beta}^{\mathrm{T}} \underline{\underline{C}} \underline{\beta} = 1$, the N-tuple $\underline{\alpha}$ is readily obtained as a generalized eigenvector solution of

$$\underline{\underline{M}} \underline{\alpha} = \lambda \underline{\underline{C}} \underline{\alpha}, \tag{13}$$

corresponding to the smallest eigenvalue λ (since $\widetilde{J}(\underline{\alpha}) = \lambda$), and normalized so that $\underline{\alpha}^{\mathrm{T}} \underline{\underline{C}} \underline{\alpha} = 1$. The implicit algebraic least-squares approximation of the set of points $\mathscr{P}$ with weights $\{w_i\}_{i=1,\dots,m}$ is thus defined as the zero-level set $\mathscr{S}_{\underline{\alpha}}$ with $\underline{\alpha}$ as defined above, and the (signed) algebraic distance from a point $\mathbf{x}$ to $\mathscr{S}_{\underline{\alpha}}$ is given by $F_{\underline{\alpha}}(\mathbf{x})$, which is a second-order approximation to $d(\mathbf{x}, \mathscr{S}_{\underline{\alpha}})$ if $\mathbf{x}$ is close to $\mathscr{S}_{\underline{\alpha}}$.

2.3. Algebraic moving-least-squares (AMLS) surfaces

Since the methodology described in the previous section generates nothing but spheres, we still have some way to go to describe a methodology that approximates point sets of arbitrary shape. The idea is quite simple, since the only added ingredient is to make the weights $\{w_i\}_{i=1,\dots,m}$ depend on $\mathbf{x}$.

We begin by defining a continuous non-negative localization function $\psi : \mathbb{R}^+ \to \mathbb{R}^+$ satisfying:

$$\psi(0) = 1, \quad \psi'(s) \leqslant 0, \quad \psi(s \to \infty) = 0. \tag{14}$$

Typically we adopt:

$$\psi(s) = \begin{cases} (1 - s^2)^4 & \text{if } s < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

An influence radius Δ is also defined, assumed constant for simplicity, which leads to a specific domain U in which the implicit function F will be computed, namely,

$$U = \{\mathbf{x} \in \mathbb{R}^n |\ \text{the number of points } \mathbf{p} \in \mathscr{P} \text{ with } \|\mathbf{x} - \mathbf{p}\| < \Delta \text{ is } \geqslant 4\}. \tag{16}$$

The need for at least 4 points at distance smaller than Δ is to ensure that the matrix $\underline{\underline{M}}$ be non-singular (minimum number of points required to determine a sphere). In fact, there exists the additional condition that these four points cannot belong to a common circle.

Now, for each $\mathbf{x} \in U$, define the set of weights $\{w_i(\mathbf{x})\}_{i=1,\dots,m}$ as

$$w_i(\mathbf{x}) = \psi\left(\frac{\|\mathbf{x} - \mathbf{p}_i\|}{\Delta} \right) \tag{17}$$

and with these weights compute the matrix $\underline{\underline{M}}$ defined by Eq. (11). Then solve the eigenproblem (13), yielding the coefficients $\underline{\alpha}(\mathbf{x})$, which depend on $\mathbf{x}$ because $\underline{\underline{M}}$ does. The implicit function at $\mathbf{x}$ is then given by Eq. (1), and, as discussed in Section 2.2, is the signed algebraic distance to the sphere that fits, in a least-squares sense and with weights $\{w_i(\mathbf{x})\}_{i=1,\dots,m}$, the set $\mathscr{P}$.

The continuity of F for arbitrary point sets is far from obvious. The zero-level-set of F,

$$\mathscr{S} = \{\mathbf{x} \in U | F(\mathbf{x}) = 0\}, \tag{18}$$

which we define as the surface approximating the point cloud $\mathscr{P}$, is thus not guaranteed to be an $(n-1)$th dimensional manifold. We will nevertheless call $\mathscr{S}$ the *AMLS approximating surface* (or *curve*, in 2D) of the point cloud.

Some examples of F and $\mathscr{S}$ are given in Fig. 1. In this figure, and throughout this article, all of the computations are made with $\Delta = 3h$, where h is the average distance between points. This choice was made after many numerical experiments with different values of Δ, some of which are shown in Fig. 2. The smoothing effect of increasing Δ is evident from that figure, so that Δ should be kept as small as possible. However, taking Δ too small may lead to spurious holes in the set U and thus in the curve/surface $\mathscr{S}$.

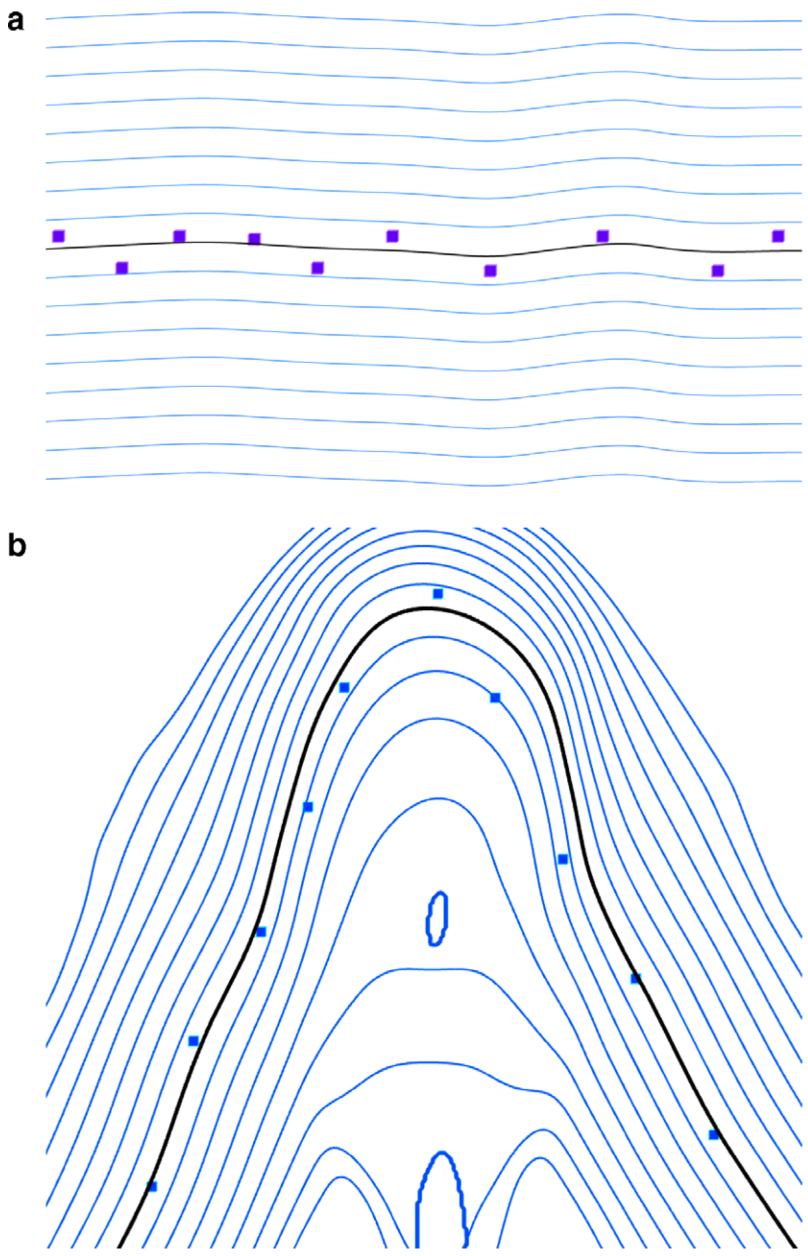

Fig. 1. Examples of AMLS implicit functions F (blue contours) and AMLS approximating curves $\mathscr{S}$ (black lines) for two point sets (black squares). (a) Randomly-perturbed samples of a straight segment (the vertical scale has been expanded by a factor of 7.5). (b) Randomly-perturbed samples of a corner. (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

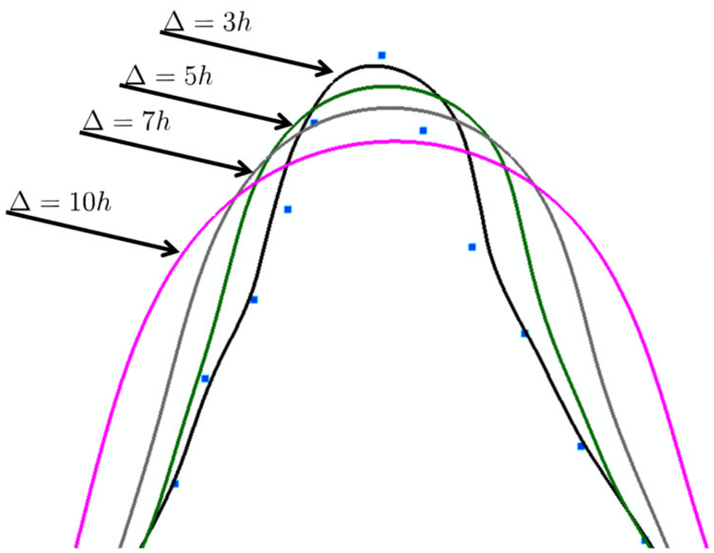

Fig. 2. Effect of increasing Δ on the shape of AMLS approximating curves. Shown are the following cases: $\Delta = 3h$ (black), $\Delta = 5h$ (green), $\Delta = 7h$ (gray) and $\Delta = 10h$ (pink). (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

2.4. The issue of orientation

A crucial issue in numerical implementations is the *orientation* of $\underline{\alpha}(\mathbf{x})$. Notice from Eq. (13) that the vector of coefficients $\underline{\alpha}(\mathbf{x})$ is defined up to a multiplication by -1, and that such a multiplication has no effect on $\mathscr{S}_{\underline{\alpha}}$ but *changes the sign of* $F(\mathbf{x})$. For any algorithm trying to detect the intersection of $\mathscr{S}_{\underline{\alpha}}$ with a given line (or *ray*), this indeterminacy in $\underline{\alpha}(\mathbf{x})$ leads to spurious intersections. In fact, this difficulty turns out to be equivalent to that of choosing the orientation of the unit normal, for which some techniques exist [15,17]. The technique we adopt here is especially suited for our application in front tracking.

We assume that a set of approximate normals $\{\tilde{\mathbf{n}}_i\}$ $(i = 1, \ldots, m)$ *is available at the points in* $\mathscr{P}$. This set of approximate normals is only used to define the orientation, so that they need not be accurate. For a given point $\mathbf{x}$, we define

$$\mathbf{g}_{\underline{\alpha}}(\mathbf{x}) = (\alpha_2(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_1, \alpha_3(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_2, \alpha_4(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_3). \tag{19}$$

Notice that $\mathbf{g}_{\underline{\alpha}}(\mathbf{x})$ is the gradient of the polynomial that generates the best-fit sphere with weights $\{w_i(\mathbf{x})\}_{i=1,\ldots,m}$. We also define, at $\mathbf{x}$, the weighted-average normal

$$\tilde{\mathbf{n}}(\mathbf{x}) = \sum_{i=1}^{m} w_i(\mathbf{x})\tilde{\mathbf{n}}_i. \tag{20}$$

Finally, *if the angle between* $\mathbf{g}_{\underline{\alpha}}(\mathbf{x})$ *and* $\tilde{\mathbf{n}}(\mathbf{x})$ *is greater than* $\pi/2$, *then* $\underline{\alpha}(\mathbf{x})$ *is replaced by* $-\underline{\alpha}(\mathbf{x})$; *otherwise it is left unchanged.*

We stress that for the AMLS method, especially in the presence of sharp features, the choice of orientation of $\underline{\alpha}(\mathbf{x})$ is critical. The technique above works quite well in most cases, but difficulties may appear near sharp features, as discussed in Section 2.6.

2.5. Convergence study of AMLS surfaces

The sense in which an AMLS surface would converge (or fail to do so) to a given exact surface depends on the application. If the point cloud originates from noisy measurements of the topography of an object, a statistical approach is needed to define convergence. In our case, as will be justified in later sections, we assume that the points in $\mathscr{P}$ lie exactly on some surface Σ. Our knowledge of Σ reduces to the set of points $\mathscr{P}$, from which we want to build a surface $\mathscr{S}$ which approximates Σ as good as possible.

Let us define the Hausdorff distance [26] between two surfaces as

$$D(\Sigma, \mathscr{S}) = \max\{\max_{\mathbf{x}\in\Sigma} \min_{\mathbf{y}\in\mathscr{S}} \|\mathbf{x}-\mathbf{y}\|, \max_{\mathbf{x}\in\mathscr{S}} \min_{\mathbf{y}\in\Sigma} \|\mathbf{x}-\mathbf{y}\|\}. \tag{21}$$

Let $\mathscr{P}_h$ be a family of regularly-spaced point clouds obtained by sampling the "exact" surface Σ, parameterized by the average spacing h. Also, let $\mathscr{S}_h$ be the associated family of AMLS approximating surfaces, obtained by the method described in Section 2.3. *The question that arises is, thus, whether* $D(\Sigma, \mathscr{S}_h)$ *goes to zero as* $h \to 0$ *and, if so, what the convergence rate r is in the sense*

$$D(\Sigma, \mathscr{S}_h) = \mathcal{O}(h^r). \tag{22}$$

2.5.1. Two-dimensional ellipse

We have examined the convergence for the case of the two-dimensional ellipse Σ given by

$$\frac{x_1^2}{9}+\frac{x_2^2}{4}=1.$$

For this purpose, we have generated point sets $\mathscr{P}_h$ on the ellipse with average distances $h = 1.5, 0.75, 0.375, \ldots, 2.34 \times 10^{-2}$, corresponding to N_h = 14, 27, 54, ..., 860 points. The point set corresponding to $h = 1.5$, together with the corresponding AMLS curve, is shown in Fig. 3a.

In Fig. 3b we show the geometric error $D(\Sigma, \mathscr{S}_h)$ as a function of h. The convergence is seen to be $\mathscr{O}(h^3)$. With just 54 sampling points, at an average distance of 0.375, the maximum distance between $\mathscr{S}_h$ and Σ is already as small as 7.3×10^{-4}.

In this 2D case the implicit function is given by

$$F(\mathbf{x}) = \alpha_1(\mathbf{x}) + \alpha_2(\mathbf{x})x_1 + \alpha_3(\mathbf{x})x_2 + \alpha_4(\mathbf{x})(x_1^2 + x_2^2). \tag{23}$$

At each point $\mathbf{x}$ of $\mathscr{S}_h$ we also compute the AMLS normal and curvature. The AMLS normal $\mathbf{n}_{\underline{\alpha}}(\mathbf{x})$ corresponding to the coefficients $\underline{\alpha}(\mathbf{x})$ is defined as the normal to the best approximating circle at $\mathbf{x}$, and is given by

$$\mathbf{n}_{\underline{\alpha}}(\mathbf{x}) = (\alpha_2(\mathbf{x}) + 2\alpha_4(\mathbf{x})x_1, \alpha_3(\mathbf{x}) + 2\alpha_4(\mathbf{x})x_2). \tag{24}$$

Denoting by $\mathbf{y}_\mathbf{x}$ the point in Σ that is closest to $\mathbf{x}$, the error in the normals is defined by

$$E_\mathrm{N}(\Sigma, \mathscr{S}_h) = \max_{\mathbf{x}\in\mathscr{S}_h} \|\mathbf{n}_{\underline{\alpha}}(\mathbf{x}) - \mathbf{n}(\mathbf{y}_\mathbf{x})\|, \tag{25}$$

where $\mathbf{n}(\mathbf{y}_\mathbf{x})$ is the exact normal to Σ at $\mathbf{y}_\mathbf{x}$. The AMLS curvature is the inverse of the radius of the best approximating circle, and is given by

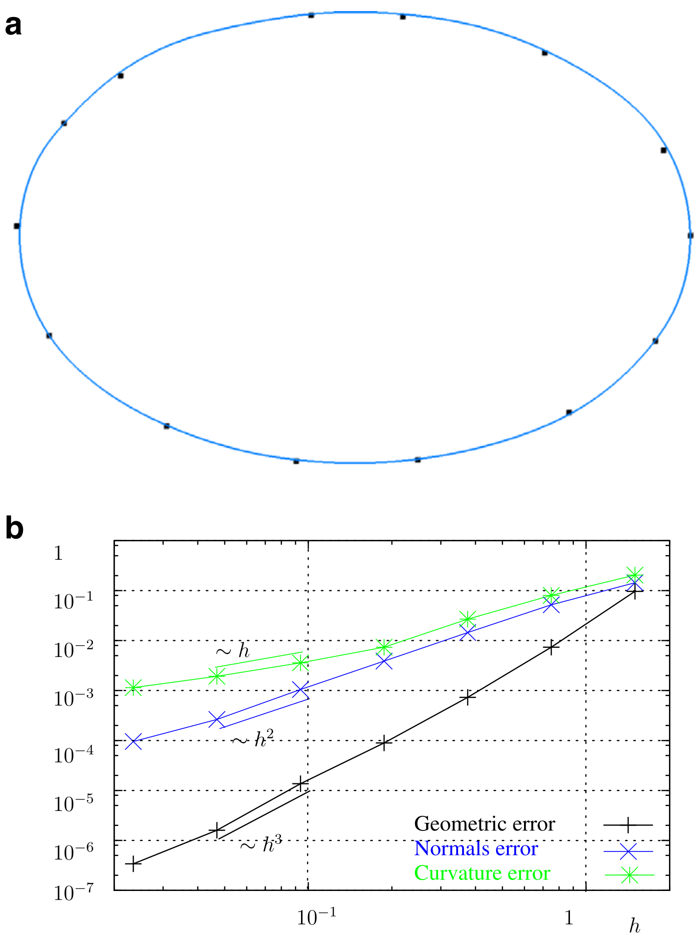

Fig. 3. (a) Point set $\mathscr{P}_h$ corresponding to $h = 1.5$ for the 2D ellipse, together with the resulting circles-based AMLS curve $\mathscr{S}_h$. (b) Convergence plots of $D(\Sigma, \mathscr{S}_h)$ (denoted by geometric error), of $E_\mathrm{N}(\Sigma, \mathscr{S}_h)$ (denoted by normals error), and of $E_\mathrm{C}(\Sigma, \mathscr{S}_h)$ (denoted by curvature error).

$$\kappa_{\underline{\alpha}}(\mathbf{x}) = \left(\frac{\alpha_2^2(\mathbf{x}) + \alpha_3^2(\mathbf{x})}{4\alpha_4^2(\mathbf{x})} - \frac{\alpha_1(\mathbf{x})}{\alpha_4(\mathbf{x})} \right)^{-\frac{1}{2}}. \tag{26}$$

The curvature error is defined as

$$E_C(\Sigma, \mathscr{S}_h) = \max_{\mathbf{x}\in\mathscr{S}_h} \|\kappa_{\underline{\alpha}}(\mathbf{x}) - \kappa(\mathbf{y_x})\|, \tag{27}$$

where $\kappa(\mathbf{y_x})$ is the exact curvature of Σ at $\mathbf{y_x}$.

Plots of E_N and E_C as functions of h are shown in Fig. 3b. Notice from their definitions that $\mathbf{n}_{\underline{\alpha}}(\mathbf{x})$ and $\kappa_{\underline{\alpha}}(\mathbf{x})$ are *not* the normal and curvature that correspond to $\mathscr{S}_h$, they are just those corresponding to the AMLS circle at $\mathbf{x}$. In fact, the true normal and curvature of $\mathscr{S}_h$ contain derivatives of $\underline{\alpha}(\mathbf{x})$, which would be very cumbersome to compute (further details can be found in [2,15]). Notwithstanding, E_N converges as h^2 and E_C as h, as one would expect from the true normal and curvature.

The previous example has illustrated the ability of 2D, circles-based AMLS curves to approximate smooth curves in two-dimensions. It is interesting to consider the behavior of the AMLS approximation based in straight lines in the same example. The calculation procedure is analogous to what has already been described, with just α_1, α_2 and α_3 as non-zero coefficients. Also, the AMLS curvature is not available since the local fitting is done with a straight line. Using the same point set as in the circles-based case, the resulting $\mathscr{S}_h$ is shown in Fig. 4a. Plots of $D(\Sigma, \mathscr{S}_h)$ and of $E_N(\Sigma, \mathscr{S}_h)$ as functions of h are shown in Fig. 4b. In this case the geometric error $D(\Sigma, \mathscr{S}_h)$ behaves as h^2, while the normals error $E_N(\Sigma, \mathscr{S}_h)$ behaves as h. One order of convergence is thus lost when passing from circles-based to straight-lines-based AMLS curves, which is not surprising.

2.5.2. Three-dimensional ellipsoid

We have performed an analogous convergence study for the 3D ellipsoid Σ given by

$$\frac{x_1^2}{16} + \frac{x_2^2}{9} + \frac{x_3^2}{4} = 1. \tag{28}$$

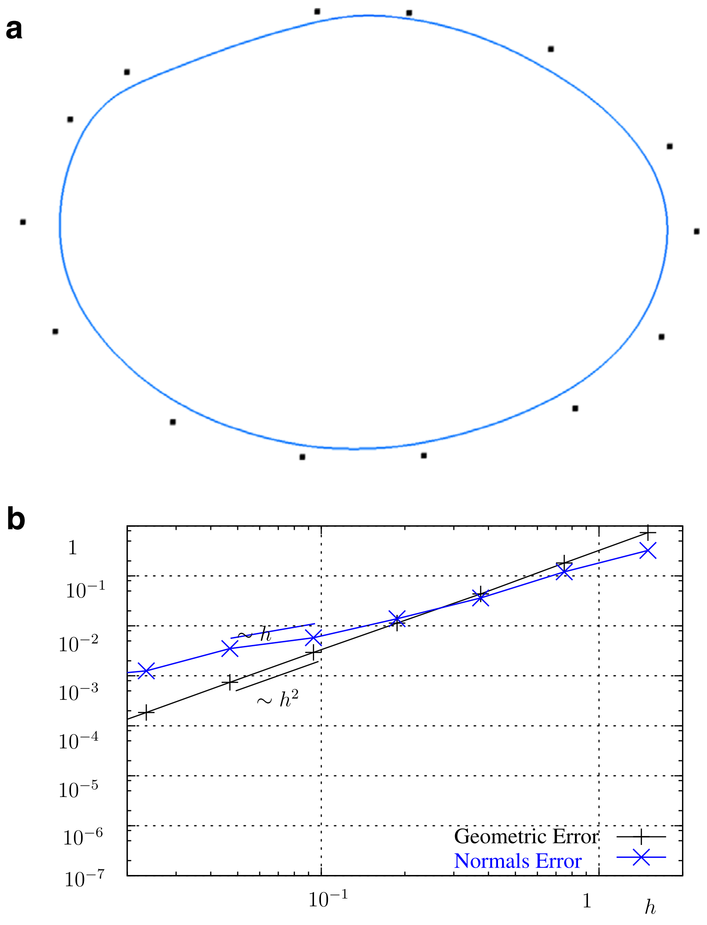

Fig. 4. (a) Point set $\mathscr{P}_h$ corresponding to $h = 1.5$ for the 2D ellipse, together with the resulting straight-line-based AMLS curve $\mathscr{S}_h$. (b) Convergence plots of $D(\Sigma, \mathscr{S}_h)$ (denoted by geometric error) and of $E_N(\Sigma, \mathscr{S}_h)$ (denoted by normals error).

The point sets $\mathscr{P}_h$, as before, had average spacing $h = 0.75, 0.375, \ldots, 2.34 \times 10^{-2}$, which corresponded to N_h = 265, 1085, 4369, ..., 276,414 points on Σ, respectively.

The convergence plots for the spheres-based AMLS method are shown in Fig. 5a. Notice that in three dimensions the expressions for the AMLS normals and curvature are

$$\mathbf{n}_{\underline{\alpha}}(\mathbf{x}) = (\alpha_2(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_1, \alpha_3(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_2, \alpha_4(\mathbf{x}) + 2\alpha_5(\mathbf{x})x_3) \tag{29}$$

and

$$\kappa_{\underline{\alpha}}(\mathbf{x}) = \left(\frac{\alpha_2^2(\mathbf{x}) + \alpha_3^2(\mathbf{x}) + \alpha_4^2(\mathbf{x})}{4\alpha_5^2(\mathbf{x})} - \frac{\alpha_1(\mathbf{x})}{\alpha_5(\mathbf{x})}\right)^{-\frac{1}{2}}. \tag{30}$$

The first conclusion from the error plots is that, in contrast with the 2D case, in three dimensions the convergence rate of the geometric error drops to quadratic. Clearly, the polynomial basis, with only one quadratic element, is too poor to achieve third-order convergence. The approximation is nevertheless quite good. With just 4369 points, at an average distance of $h = 0.375$, the distance between Σ and $\mathscr{S}_h$ is as small as 2.3×10^{-3}, two orders of magnitude smaller than the distance between samples. The normals error E_{N} behaves as $\mathscr{O}(h)$, and turns out to be smaller than 2% when $h = 0.375$. The AMLS curvature fails to converge to the exact one ($E_{\mathrm{C}} = \mathscr{O}(1)$). This is a drawback that needs to be fixed before applications involving capillary forces could be performed, for example. The study of a convergent approximation of the curvature based on the AMLS approach is however left outside the scope of this article. If needed, a simple least-squares approach on a rotated frame, as adopted by Du et al. [11], can be applied.

For the same point sets as before, the planes-based AMLS surfaces were also constructed. The convergence plots are shown in Fig. 5b. The rates of convergence are the same as for the spheres-based approximation, and the error in the normals, $E_{\mathrm{N}}(\Sigma, \mathscr{S}_h)$, is quite similar. However, the geometric error $D(\Sigma, \mathscr{S}_h)$ is larger by a factor of about 10, indicating that the quadratic term improves the approximation. Finally, in Figs. 6a and b we present the planes-based and spheres-based AMLS approximations corresponding to $h = 0.75$ (265 points), respectively. The convexity of the ellipsoid makes the planes-based AMLS to leave the point set outside it, while the sphere-based ones shows a remarkable fit for such a reduced number of points.

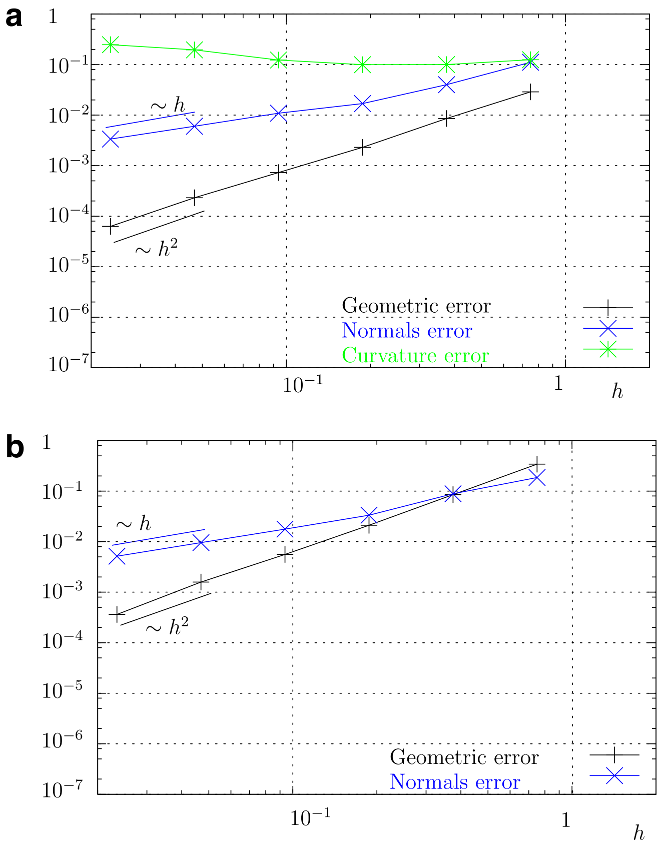

Fig. 5. Convergence plots of $D(\Sigma, \mathscr{S}_h)$ (denoted by geometric error), of $E_{\mathrm{N}}(\Sigma, \mathscr{S}_h)$ (denoted by normals error), and of $E_{\mathrm{C}}(\Sigma, \mathscr{S}_h)$ (denoted by curvature error) for the case of the 3D ellipsoid. (a) Spheres-based AMLS approximation. (b) Planes-based AMLS approximation (in this approximation the AMLS curvature is unavailable).

2.6. Sharp features

In fluid dynamics applications the interfaces evolve in time, possibly exhibiting sharp features or even topological changes along their evolution. The ability of a method to deal with non-smooth surfaces has a strong impact on the *robustness* of the overall simulation methodology. Less accurate methods, such as the level-set method, are often favored in technological applications on the basis of robustness considerations.

In this section we study the ability of the AMLS method presented above to deal with non-smooth geometries. Notice however that, for the time being, the surface Σ is fixed in space. We simply take samples of Σ at some average spatial sampling rate h and reconstruct the AMLS surface $\mathscr{S}_h$ that results from the sample-points set $\mathscr{P}_h$.

As case study we analyze a 3D surface Σ obtained by removing one quarter of a sphere of unit radius. We denote this case by "3/4-sphere case". We have built point sets $\mathscr{P}_h$ by sampling the 3/4-sphere with average spacing $h \simeq 0.12$ (1030 points), $h \simeq 0.08$ (2031 points) and $h \simeq 0.05$ (4748 points). In Fig. 7 we show the AMLS surfaces obtained from these point sets. It is clear that the neighborhoods of the sharp corners, where the two planar surfaces and the spherical sector meet, challenge the approximation capabilities of the AMLS method. In fact, an artifact can be observed at these corners, meaning a feature of the surface $\mathscr{S}_h$ that is not present in the original surface Σ. These artifacts constitute a lack of robustness of the method, since when coupled with a flow solver may lead to unphysical, possibly unstable, behavior.

3. AMLS surfaces with improved robustness

3.1. RAMLS surfaces

The last paragraph of the previous section showed that some improvement in the robustness of the AMLS methodology described so far is in order.

This issue has already been addressed by Guennebaud and Gross [15], who proposed the following two-step modification of the algebraic AMLS method described in Sections 2.1–2.3:

Step A *Compute the AMLS normals for all* $\mathbf{p}_i \in \mathscr{P}_h$: This is done by computing $\underline{\alpha}$ from (13), but just at the points $\mathbf{p}_i$ in the set $\mathscr{P}_h$. The normals $\mathbf{N}_i = \mathbf{n}_{\underline{\alpha}}(\mathbf{p}_i)$, $i = 1, \dots, m$ are then obtained from (29) in the 3D case, and from (24) in the 2D case.
Step B *Compute the implicit function* $F(\mathbf{x})$ for $\mathbf{x} \in U$: For this purpose, the coefficients $\underline{\alpha}(\mathbf{x})$ are not found by minimizing $\widetilde{J}$ as defined in (5) and (6). Instead, $\underline{\alpha}(\mathbf{x})$ is found as

$$\underline{\alpha}(\mathbf{x}) = \arg\min_{\underline{\beta}\in\mathbb{R}^N} G(\underline{\beta}) \tag{31}$$

with

$$G(\underline{\beta}) = \sum_{i=1}^{m} w_i |F_{\underline{\beta}}(\mathbf{p}_i)|^2 + K \sum_{i=1}^{m} w_i \|\mathbf{n}_{\underline{\beta}}(\mathbf{p}_i) - \mathbf{N}_i\|^2. \tag{32}$$

Compared to the Formulation (5) and (6), we observe that the minimization in (31) is unrestricted. The condition $\|\nabla F_{\underline{\beta}}\| \simeq 1$ at $\mathscr{S}_\beta$, which makes $F_{\underline{\beta}}$ to approximate the signed distance, is in fact enforced by the second term in $G(\underline{\beta})$, Eq.

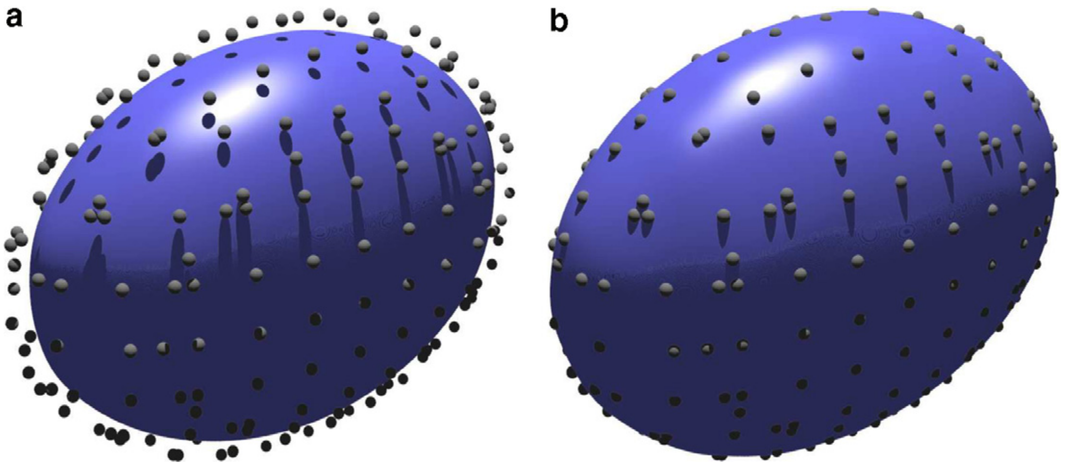

Fig. 6. Ellipsoids generated by (a) planes-based and (b) spheres-based AMLS technique. The point set consists of 265 points at an average distance $h = 0.75$.

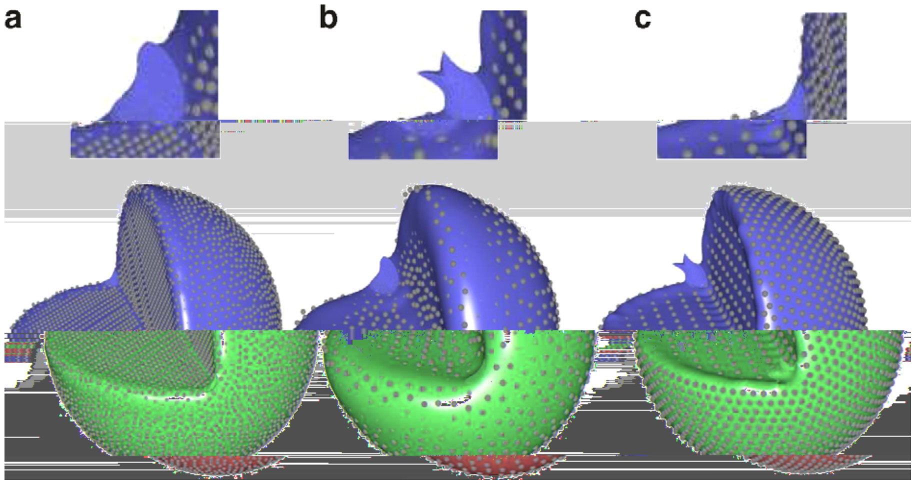

Fig. 7. Approximation of the 3/4-sphere case with AMLS surfaces based on point clouds of 1030 points (a), of 2031 points (b) and of 4748 points (c). The artifacts on the top show how the implicit function can fail.

(32). This term penalizes the deviations of the AMLS normals at the sample points calculated from $\underline{\alpha}(\mathbf{x})$ from the normals calculated in **Step A** of the algorithm. Since the second step consists of an unconstrained minimization problem, its algebraic burden is simply that of solving a small linear system, much smaller than that of solving the eigenvalue problem (13).

As in any penalization method, the choice of the constant K in (32) is not obvious and may affect both the accuracy and the numerical stability of the method. Guennebaud and Gross use $K = 10^6 h^2$ without further justification. We propose here a variant which contains no free parameter. In fact, it can be regarded as the limit of the formulation above when $K \to \infty$. A key observation is that the second term in G (Eq. (32)) does not depend on α_1, so that when K is very large the only role of the first term is to determine α_1. We will thus split the coefficient arrays into two parts separating the first coefficient, for example,

$$\underline{\beta} \in \mathbb{R}^N, \quad \underline{\beta} = (\beta_1, \hat{\underline{\beta}}),$$

where $\hat{\underline{\beta}} = (\beta_2, \ldots, \beta_N)$ belongs to $\mathbb{R}^{N-1}$. Since for any coefficient vector $\underline{\beta}$ the normal at any point $\mathbf{x}$ defined by (29) does not depend on β_1, we also denote it by $\mathbf{n}_{\hat{\underline{\beta}}}$.

We are now in a position to detail the proposed AMLS reconstruction by describing the algorithm that computes $\underline{\alpha}(\mathbf{x})$ at a generic $\mathbf{x} \in U$. We will refer to this approximation method as *RAMLS reconstruction* (for Robust Algebraic MLS). The RAMLS implicit function, which is still given by Eq. (1), will still be denoted by $F(\mathbf{x})$, since the only difference between the AMLS and RAMLS methods lies in the choice of the coefficients $\underline{\alpha}(\mathbf{x})$. Also, the RAMLS surface, still denoted by $\mathscr{S}$, remains the zero-level set of F. To avoid confusion, of course, each result will be given specifying if it corresponds to the AMLS or to the RAMLS method.

Proposed algorithm (RAMLS reconstruction):

Step A *Compute the AMLS normals* $\mathbf{N}_i = \mathbf{n}_{\underline{\alpha}}(\mathbf{p}_i)/\|\mathbf{n}_{\underline{\alpha}}(\mathbf{p}_i)\|$ *for all* $\mathbf{p}_i \in \mathscr{P}_h$. This is done by computing $\underline{\alpha}$ from (13), but just at the points $\mathbf{p}_i$. Then, the $\mathbf{N}_i$s are obtained from (29) in the 3D case, and from (24) in the 2D case. Since at $\mathbf{p}_i$ we have an approximate normal $\tilde{\mathbf{n}}_i$, the orientation of $\mathbf{N}_i$ is naturally taken such that the angle between $\mathbf{N}_i$ and $\tilde{\mathbf{n}}_i$ be smaller than $\pi/2$.

Step B *Compute the implicit function* $F(\mathbf{x})$ *for* $\mathbf{x} \in U$. The computation of the coefficients $\underline{\alpha}(\mathbf{x})$ is split into two substeps as follows:

Substep B.1 Determine $\hat{\underline{\alpha}}(\mathbf{x})$ as

$$\hat{\underline{\alpha}}(\mathbf{x}) = \arg \min_{\hat{\underline{\beta}} \in \mathbb{R}^{N-1}} \sum_{i=1}^{m} w_i \|\mathbf{n}_{\hat{\underline{\beta}}}(\mathbf{p}_i) - \mathbf{N}_i\|^2. \tag{33}$$

As a consequence, $\hat{\underline{\alpha}}(\mathbf{x})$ is the solution of the linear system

$$\underline{\underline{A}}\hat{\underline{\alpha}} = \underline{b} \tag{34}$$

where

$$A_{k\ell} = \sum_{i=1}^{m} w_i \nabla q_k(\mathbf{p}_i) \cdot \nabla q_\ell(\mathbf{p}_i) \tag{35}$$

$$b_k = \sum_{i=1}^{m} w_i \nabla q_k(\mathbf{p}_i) \cdot \mathbf{N}_i \tag{36}$$

and in particular, for the spheres-based method (basis defined in Eq. (2)) and denoting by (X_i, Y_i, Z_i) the coordinates of $\mathbf{p}_i$ and by (N_{i1}, N_{i2}, N_{i3}) the components of $\mathbf{N}_i$,

$$\underline{\underline{A}} = \begin{pmatrix} \sum_{i=1}^m w_i & 0 & 0 & 2\sum_{i=1}^m w_i X_i \\ 0 & \sum_{i=1}^m w_i & 0 & 2\sum_{i=1}^m w_i Y_i \\ 0 & 0 & \sum_{i=1}^m w_i & 2\sum_{i=1}^m w_i Z_i \\ 2\sum_{i=1}^m w_i X_i & 2\sum_{i=1}^m w_i Y_i & 2\sum_{i=1}^m w_i Z_i & 4\sum_{i=1}^m w_i \|\mathbf{p}_i\|^2 \end{pmatrix} \tag{37}$$

$$\underline{b} = \begin{pmatrix} \sum_{i=1}^m w_i N_{i1} \\ \sum_{i=1}^m w_i N_{i2} \\ \sum_{i=1}^m w_i N_{i3} \\ 2\sum_{i=1}^m w_i \mathbf{p}_i \cdot \mathbf{N}_i \end{pmatrix} \tag{38}$$

Notice the simplicity of the structure of matrix $\underline{\underline{A}}$, which allows for it to be solved by hand.

Substep B.2 Now that $\hat{\underline{\alpha}}(\mathbf{x}) = (\alpha_2(\mathbf{x}), \ldots, \alpha_N(\mathbf{x}))$ is known, we obtain $\alpha_1(\mathbf{x})$ by minimizing

$$\sum_{i=1}^m w_i |F_{\underline{\beta}}(\mathbf{p}_i)|^2 \tag{39}$$

over all the $\underline{\beta}$ such that $\hat{\underline{\beta}} = \hat{\underline{\alpha}}(\mathbf{x})$. The solution is, simply,

$$\alpha_1(\mathbf{x}) = -\frac{\sum_{i=1}^m w_i \left(\sum_{k=2}^N \alpha_k(\mathbf{x}) q_k(\mathbf{p}_i)\right)}{\sum_{i=1}^m w_i}. \tag{40}$$

Notice that the numerator in the right-hand side of (40) is $\sum_{i=1}^m w_i F_\beta(\mathbf{p}_i)$ taking $\beta = (0, \alpha_2(\mathbf{x}), \ldots, \alpha_N(\mathbf{x}))$. For the specific case of the spheres-based method the above equation reads

$$\alpha_1(\mathbf{x}) = -\frac{\sum_{i=1}^m w_i [\alpha_2(\mathbf{x}) X_i + \alpha_3(\mathbf{x}) Y_i + \alpha_4(\mathbf{x}) Z_i + \alpha_5(\mathbf{x}) \|\mathbf{p}_i\|^2]}{\sum_{i=1}^m w_i}. \tag{41}$$

The surface $\mathscr{S}$ defined by the RAMLS method (or RAMLS surfaces) is thus given by the zero-level set of $F(\mathbf{x})$, defined by Eq. (1) with the coefficients $\{\alpha_k(\mathbf{x})\}$ $(k = 1, \ldots, N)$ obtained from the algorithm proposed above. Once again, the continuity of F is far from obvious and it is only conjectured that $\mathscr{S}$ is a manifold. Moreover, notice from (33) that the determination of the coefficients $\alpha_2, \ldots, \alpha_N$ is based solely on the approximation of the normals and *not* on the minimization of the algebraic distance. Since this could affect the convergence rate of the method, several numerical experiments are reported in Section 3.2.

To conclude the presentation of RAMLS surfaces, let us mention two of their advantages:

- Their *computational cost* is much smaller than that of AMLS surfaces, as already discussed by Guennebaud and Gross [15]. In fact, to reconstruct the surface the number of points $\mathbf{x}$ at which F needs to be evaluated is much larger than the number of points $\mathbf{p}$ in $\mathscr{P}$. In the AMLS approach, an eigensystem needs to be solved for each $\mathbf{x}$, while in the RAMLS approach eigensystems are only solved for each $\mathbf{p}$ and then the computation of F at each $\mathbf{x}$ involves just the solution of a linear system of dimension $N - 1$.
- The issue of orientation discussed in Section 2.4 is much alleviated. In the AMLS approach the orientation must be decided for each $\mathbf{x}$ at which F needs to be evaluated. This decision is based on the weighted-average of approximate normals defined in (20), which is quite arbitrary and leads to difficulties near sharp features. In the RAMLS approach the orientation needs to be defined just at the points $\mathbf{p}$ in $\mathscr{P}$. Since the approximate normals are given in $\mathscr{P}$, no ambiguity exists.

3.2. *Accuracy of RAMLS surfaces*

We have conducted exactly the same tests of Section 3 with the RAMLS method: the 2D ellipse, the 3D ellipsoid, and the 3/4-sphere. The RAMLS approximations of normal vector and curvature are given by the same formulae as for the AMLS method (Eqs. (24), (26), (29) and (30)), though obviously with the corresponding coefficients $\underline{\alpha}(\mathbf{x})$. The convergence plots for the first two cases are shown in Figs. 8b and 9b, respectively. Clearly, the same behavior is observed as for the AMLS

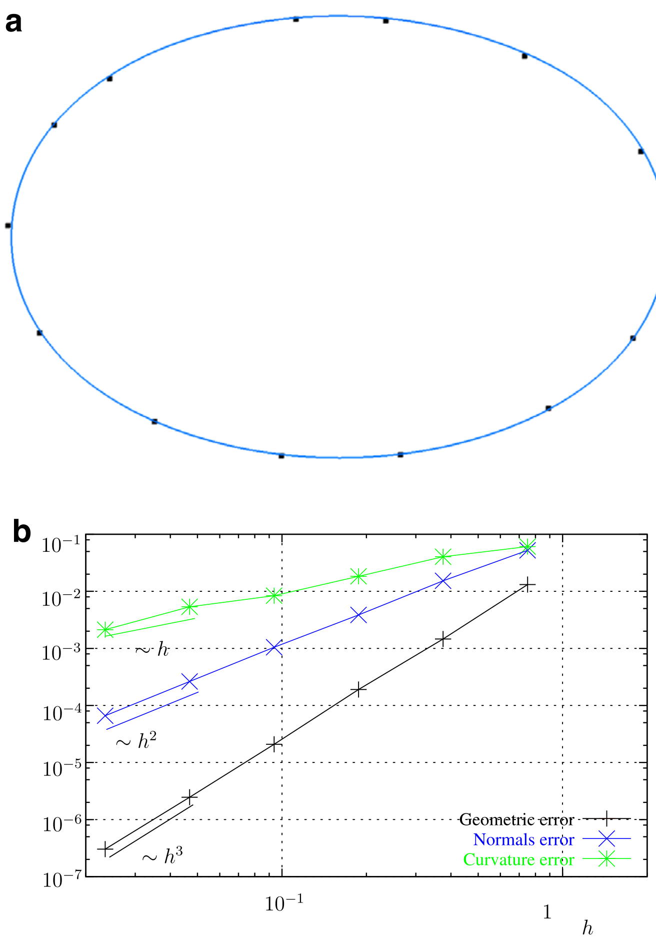

Fig. 8. Approximation of the 2D ellipse with RAMLS curve. (a) Point set $\mathscr{P}_h$ corresponding to $h = 1.5$ together with the resulting circles-based RAMLS curve. (b) Convergence plots of $D(\Sigma, \mathscr{S}_h)$ (denoted by geometric error), of $E_N(\Sigma, \mathscr{S}_h)$ (denoted by normals error), and of $E_C(\Sigma, \mathscr{S}_h)$ (denoted by curvature error).

method. The geometric error behaves as $\mathscr{O}(h^3)$ in 2D for the circles-based RAMLS method, while it behaves as $\mathscr{O}(h^2)$ in 3D for the spheres-based RAMLS method. In particular, and contrary to what is stated in [15], the RAMLS estimate of the curvature (30) does not converge to the exact one in 3D.

The improvement in robustness of RAMLS surfaces as compared to AMLS ones is illustrated in Fig. 10 (compare it to Fig. 7). The artifacts produced by the AMLS method near the corners are spurious zeros of F mainly due to errors in the choice of orientation of $\underline{\alpha}$. As discussed in the last section, the orientation issue is much alleviated in the RAMLS approach, and the artifacts do not appear at all. Finally, Fig. 11 presents the RAMLS approximations of the same data sets in Fig. 1. It can be noticed that RAMLS curves/surfaces are less sensitive to the perturbations in the point set.

4. Front tracking with RAMLS surfaces

4.1. The Algorithm

Let us now introduce a front-tracking method based on RAMLS surfaces. We consider the three-dimensional case, since its two-dimensional analog is easily inferred from it. Inside a bounded domain $\Omega \subset \mathbb{R}^3$, a closed initial surface $\mathscr{S}(0) \subset \Omega$ is given, together with a $\mathscr{C}^1$ velocity field $\mathbf{v}(\mathbf{x}, t)$ for all $\mathbf{x} \in \Omega$, $t \in [0, T]$, where T is the simulation time. The Lagrangian trajectory $\boldsymbol{\varphi}(\mathbf{x}, s, t)$ of a particle that is at $\mathbf{x}$ at time s and moves with the velocity field $\mathbf{v}$ is the solution of

$$\frac{\partial \boldsymbol{\varphi}}{\partial t}(\mathbf{x}, s, t) = \mathbf{v}(\boldsymbol{\varphi}(\mathbf{x}, s, t), t); \quad \boldsymbol{\varphi}(\mathbf{x}, s, s) = \mathbf{x} \tag{42}$$

and the addressed problem is that of finding $\mathscr{S}(t)$ defined by

$$\mathscr{S}(t) = \{\mathbf{y} \in \Omega | \mathbf{y} = \boldsymbol{\varphi}(\mathbf{x}, 0, t), \mathbf{x} \in \mathscr{S}(0)\}. \tag{43}$$

Notice that we are assuming that $\mathscr{S}(t) \subset \Omega$ for all $t \in [0, T]$.

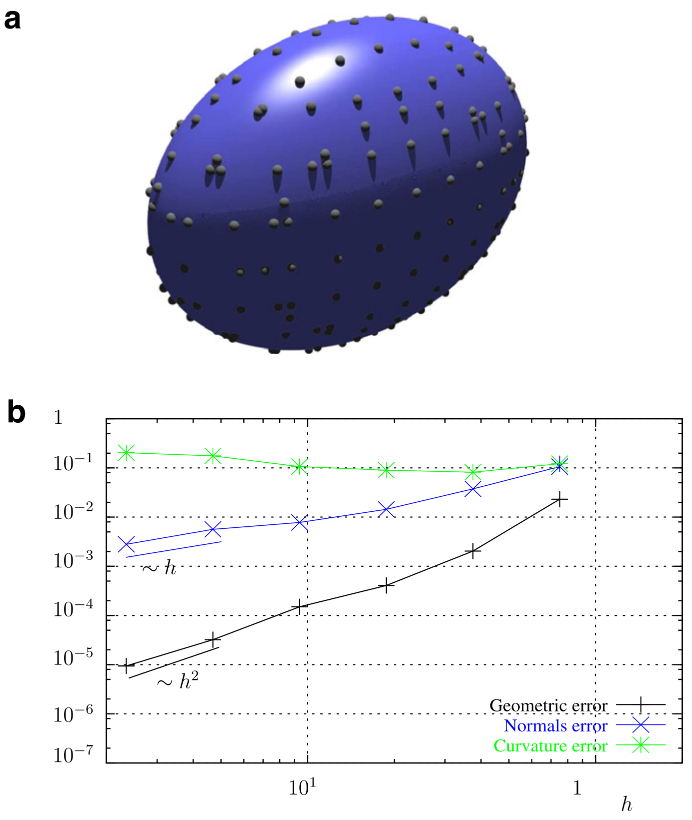

Fig. 9. Approximation of the 3D ellipsoid with RAMLS surfaces. (a) Point set $\mathscr{P}_h$ corresponding to $h = 0.75$ together with the resulting spheres-based RAMLS surface. (b) Convergence plots of $D(\Sigma, \mathscr{S}_h)$ (denoted by geometric error), of $E_N(\Sigma, \mathscr{S}_h)$ (denoted by normals error), and of $E_C(\Sigma, \mathscr{S}_h)$ (denoted by curvature error).

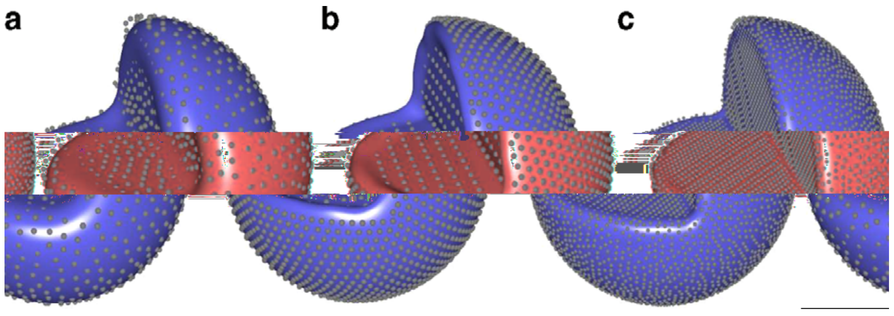

Fig. 10. Approximation of the 3/4-sphere case with RAMLS surfaces based on point clouds of 1030 points (a), of 2031 points (b) and of 4748 points (c).

The discretization starts with the definition of a structured mesh in Ω. We consider here a Cartesian mesh $\mathscr{T}_h$ with uniform spacing h, though extension to non-uniform, curvilinear grids is possible. This mesh, in fact, is only used to define a set R_h of lines (or "rays", as called in computer graphics):

$$R_h = R_{1h} \cup R_{2h} \cup R_{3h}, \tag{44}$$

where the sets R_{1h}, R_{2h} and R_{3h}, which are aligned with the x_1-, x_2-, and x_3-axes, respectively, are given by

$$R_{1h} = \bigcup_{j,k \in \mathbb{Z}} r_{1jk}; r_{1jk} = \{\mathbf{x} = (x_1, x_2, x_3) \in \Omega, x_2 = jh, x_3 = kh\} \tag{45}$$

$$R_{2h} = \bigcup_{i,k \in \mathbb{Z}} r_{2ik}; r_{2ik} = \{\mathbf{x} = (x_1, x_2, x_3) \in \Omega, x_1 = ih, x_3 = kh\} \tag{46}$$

$$R_{3h} = \bigcup_{i,j \in \mathbb{Z}} r_{3ij}; r_{3ij} = \{\mathbf{x} = (x_1, x_2, x_3) \in \Omega, x_1 = ih, x_2 = jh\}. \tag{47}$$

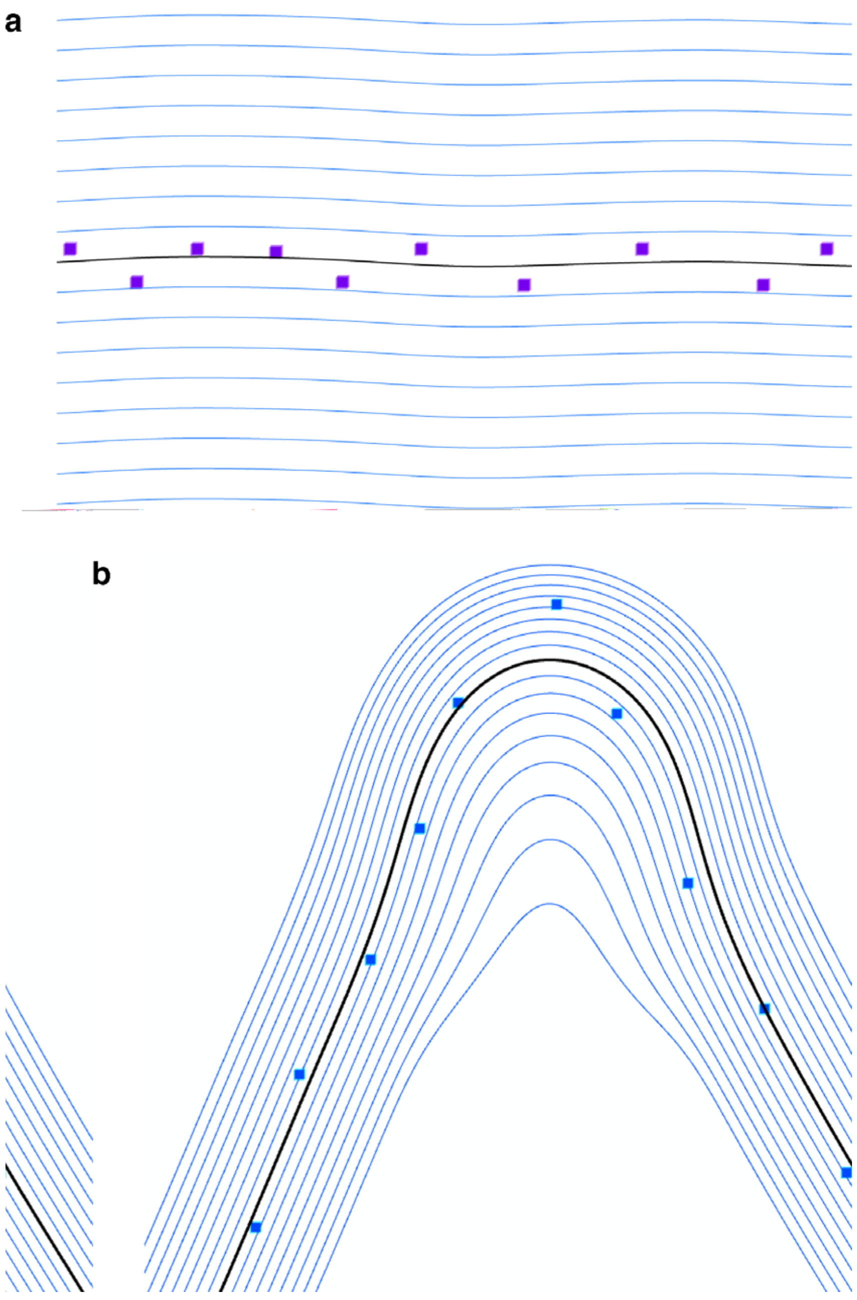

Fig. 11. RAMLS reconstructions of perturbed samples of (a) a straight line and (b) a corner. Compare to the analogous AMLS reconstructions shown in Fig. 1.

The set of rays R_h, in turn, defines a set of points $\mathcal{Q}_h(0)$ as the intersection of R_h with $\mathcal{S}(0)$; i.e.,

$$\mathcal{Q}_h(0) = R_h \cap \mathcal{S}(0). \tag{48}$$

The set of points $\mathcal{Q}_h(0)$ is then evolved in time *as Lagrangian particles* from $t = 0$ to $t = \Delta t$, where Δt is the time step, by solving (42), to yield $\mathcal{P}_h(\Delta t)$:

$$\mathcal{P}_h(\Delta t) = \boldsymbol{\varphi}(\mathcal{Q}_h(0), 0, \Delta t) = \{\mathbf{y} \in \Omega | \mathbf{y} = \boldsymbol{\varphi}(\mathbf{x}, 0, \Delta t), \mathbf{x} \in \mathcal{Q}_h(0)\}, \tag{49}$$

where we have introduced the notation $\boldsymbol{\varphi}(\mathcal{Q}_h(s), s, s + \Delta t)$ for the set of points obtained by moving $\mathcal{Q}_h(s)$ along the trajectories induced by the velocity field $\mathbf{v}$ from time s to $s + \Delta t$.

The set of points $\mathcal{P}_h(\Delta t)$, finally, defines the updated surface $\mathcal{S}_h(\Delta t)$ by means of the RAMLS reconstruction defined in Section 3.1. Denoting by $\mathbf{S}_{\text{RAMLS}}$ the operator that assigns to a set of points its RAMLS-reconstructed surface, this operation reads

$$\mathscr{S}_h(\Delta t) = \mathbf{S}_{\text{RAMLS}}(\mathscr{P}_h(\Delta t)). \tag{50}$$

An updated surface is then available, and the procedure can be repeated for another time step. The proposed method is thus defined as follows:

Initialization: Define the set of rays R_h, the time step Δt, and the initial surface $\mathscr{S}(0)$.
Step 0: Define $\mathscr{Q}_h(0) = R_h \cap \mathscr{S}(0)$.
Step 1: With $\mathscr{Q}_h(t)$ known, compute the trajectories of all points/particles in $\mathscr{Q}_h(t)$ from t to $t + \Delta t$. Define

$$\mathscr{P}_h(t + \Delta t) = \boldsymbol{\varphi}(\mathscr{Q}_h(t), t, t + \Delta t). \tag{51}$$

Step 2: Let $\mathscr{S}_h(t + \Delta t)$ be the RAMLS-generated surface associated to $\mathscr{P}_h(t + \Delta t)$ (i.e., $\mathscr{S}_h(t + \Delta t) = \mathbf{S}_{\text{RAMLS}}(\mathscr{P}_h(t + \Delta t))$): compute $\mathscr{Q}_h(t + \Delta t)$ as the intersection of R_h with $\mathscr{S}_h(t + \Delta t)$,

$$\mathscr{Q}_h(t + \Delta t) = R_h \cap \mathscr{S}_h(t + \Delta t). \tag{52}$$

Step 3: Set $t \leftarrow t + \Delta t$ and go back to **Step 1**.

Remark 2. To perform **Step 2** above, an approximate normal $\tilde{\mathbf{n}}_i$ is needed at each point $\mathbf{p}_i \in \mathscr{P}_h(t + \Delta t)$ to define the orientation in **Step A** of the RAMLS reconstruction algorithm. We simply choose as $\tilde{\mathbf{n}}_i$ the RAMLS normal computed at the same Lagrangian point/particle *before the transport*; i.e., at time t.

One time step of the method can thus be summarized as the *displacement of the point/particle set along trajectories* followed by the *intersection of the rays with the surface defined by the displaced points/particles* to generate a new point set (with which the next time step will be initiated). The first of these tasks (**Step 1** above) consists of solving the ordinary differential equation (42) in time, for which a standard Runge–Kutta scheme was adopted. The second task (**Step 2**), which is unfrequent in computational mechanics, is sketched in Fig. 12. Further details about its implementation are provided in Section 4.2.

4.2. Implementation of the intersection step

The intersection task (**Step 2**) is similar to the *ray-tracing* methodology [1,46] used to compute total illuminance in computer-graphics. In our case, as happens with illuminance computations with semi-transparent surfaces [1], not just the first but all intersections taking place along the ray need to be computed. In terms of the implicit function F, for each ray $r \in R_h$ we need to find all points $\mathbf{x} \in r$ such that $F(\mathbf{x}) = 0$.

The design of the intersection algorithm is based on two issues: first, that the implicit function F is only defined in a neighborhood U of $\mathscr{S}_h(t + \Delta t)$; and second, that the evaluation of F at a point is computationally expensive (about 1000 FLOPS in 3D).

Let $\mathscr{B}$ be the set of spheres of radius h around the points $\mathbf{p}_i$ in $\mathscr{P}_h(t + \Delta t)$; i.e.,

$$\mathscr{B} = \{\mathbf{b}_i : \mathbf{x} \in \mathbf{b}_i \Rightarrow \|\mathbf{x} - \mathbf{p}_i\| \leqslant h, \mathbf{p}_i \in \mathscr{P}_h(t + \Delta t), i \in 1, \dots, m\} \tag{53}$$

and let V be their union ($V = \bigcup_{i=1}^m \mathbf{b}_i$). Since by construction for any $\mathbf{q} \in \mathscr{Q}_h(t)$ there exists another point $\mathbf{p} \in \mathscr{Q}_h(t)$ such that $\|\mathbf{q} - \mathbf{p}\| < \sqrt{3}h$, then for Δt small enough it can be guaranteed that $\mathscr{S}_h(t + \Delta t) \subset V \subset U$. The idea is to first analytically compute the segment $\overline{\mathbf{p}_{\text{in}}\mathbf{p}_{\text{out}}}$, defined as the intersection of the ray r with a sphere $\mathbf{b}_i \in \mathscr{B}$, and then find the zero of F in this segment by the regula-falsi method.

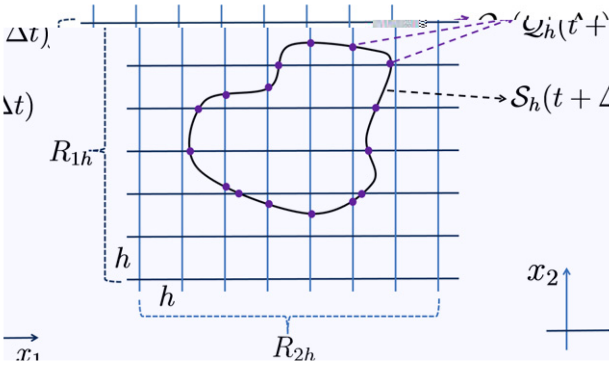

Fig. 12. Sketch of the sets of rays (R_{1h}, R_{2h}) which, intersecting the updated RAMLS-based curve $\mathscr{S}_h(t + \Delta t)$ define the new point set $\mathscr{Q}_h(t + \Delta t)$.

Algorithmically, **Step 2** in the proposed method is thus implemented as follows:

```
Step 2:
  2.0: Pre-processing
    2.0.1: Remove points from $\mathscr{P}_h(t+\Delta t)$ so that no two points are at a distance smaller than $\frac{h}{2}$. The resulting set will also be denoted by $\mathscr{P}_h(t+\Delta t)$, since no confusion can arise.
    2.0.2: For each $\mathbf{p}_i \in \mathscr{P}_h(t+\Delta t)$ compute the normal $\mathbf{N}_i$ as described in Step A of the RAMLS reconstruction algorithm.
  2.1: For each ray $r \in R_h$
    2.1.1: Find the subset of spheres $\mathscr{W} \subset \mathscr{B}$ satisfying $\mathbf{b}_i \cap r \neq \emptyset$
    2.1.2: For each $\mathbf{b}_i \in \mathscr{W}$,
      2.1.2.1: Compute the two point intersections $\mathbf{p}_{\text{in}}$ and $\mathbf{p}_{\text{out}}$ between $r$ and $\mathbf{b}_i$. Evaluate $F$ at $\mathbf{p}_{\text{in}}$ and $\mathbf{p}_{\text{out}}$, named $F(\mathbf{p}_{\text{in}})$ and $F(\mathbf{p}_{\text{out}})$ using Step B of the RAMLS reconstruction algorithm.
      2.1.2.2: If $F(\mathbf{p}_{\text{in}})F(\mathbf{p}_{\text{out}}) < 0$, find $s^* \in ]0,1[$ such that $F(\mathbf{p}_{\text{in}} + s^*(\mathbf{p}_{\text{out}} - \mathbf{p}_{\text{in}})) = 0$ by the regula-falsi method.
      2.1.2.3: Add $\mathbf{p} = \mathbf{p}_{\text{in}} + s^*(\mathbf{p}_{\text{out}} - \mathbf{p}_{\text{in}})$ to $\mathscr{Q}_h(t+\Delta t)$ if it has not been added previously (by another $\mathbf{b}_i$).
```

As stated in **Step 2.1.2.3**, it can happen that several spheres generate the same intersection point, as depicted in Fig. 13. This is avoided by checking, while processing some given sphere $\mathbf{b}_i$, whether there already exists an intersection of the ray r within $\mathbf{b}_i$, in which case the algorithm does not add a new intersection point. Notice that this procedure, together with the sign test in **Step 2.1.2.2**, implies that only one intersection is considered per sphere and per ray. If two leaves of the surface lie at a distance smaller than h, the intersection will not be computed and that part of the surface will be automatically removed.

We have observed that the approximation properties of RAMLS surfaces deteriorate when there exist points that are much closer to one another than the average spacing. This is the reason for **Step 2.0.1** above. It is worth to mention that all proximity queries in the method need to be performed with effective search algorithms based on tree structures (specifically, we make intensive use of kd-tree structures [10]). Finally, notice that **Step 2.1** can be made in parallel for each ray, since they are independent. The proposed method is thus scalable.

5. Numerical results

5.1. Convergence tests

The error of the proposed method can be analyzed in terms of the error of the RAMLS reconstruction studied in Section 3.2. In fact:

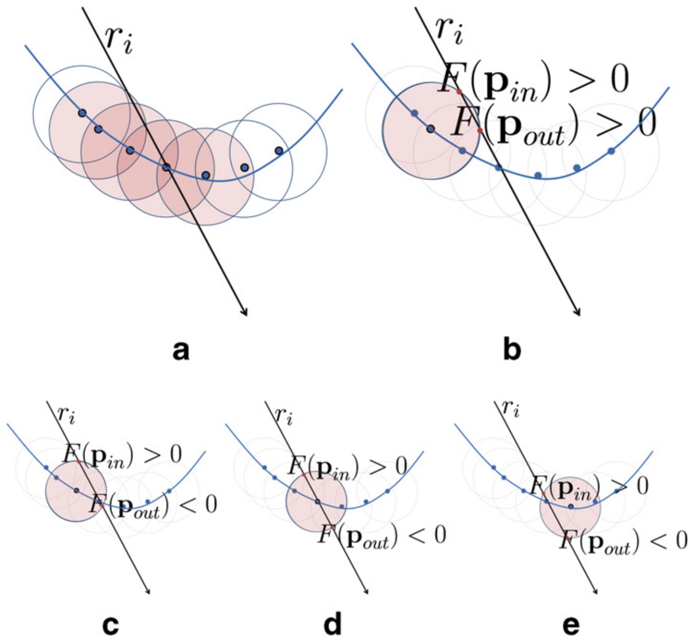

Fig. 13. Ray–spheres intersections: Three ray–sphere intersections produce the same zero on the surface. In this case we only perform the seek for the first sphere.

$$\begin{aligned}D(\mathscr{S}(t+\Delta t),\mathscr{S}_h(t+\Delta t)) &= D(\mathbf{S}_{\mathrm{RAMLS}}(\mathscr{P}_h(t+\Delta t)),\varphi(\mathscr{S}(t)))\\ &= D(\mathbf{S}_{\mathrm{RAMLS}}(\varphi(\mathscr{Q}_h(t))),\varphi(\mathscr{S}(t)))\\ &\leqslant D(\mathbf{S}_{\mathrm{RAMLS}}(\varphi(\mathscr{Q}_h(t))),\varphi(\mathscr{S}_h(t)))+D(\varphi(\mathscr{S}_h(t)),\varphi(\mathscr{S}(t))),\end{aligned}$$

where $D(\cdot,\cdot)$ is the Hausdorff distance introduced in (21). For brevity, we have omitted the last two arguments in φ, which must be taken equal to t and $t+\Delta t$, respectively (i.e.; $\varphi(\cdot)=\varphi(\cdot,t,t+\Delta t)$). Now notice that the first term of the last member is just the *reconstruction error* of the surface $\varphi(\mathscr{S}_h(t))$, because the points in $\mathscr{Q}_h(t)$ belong to $\mathscr{S}_h(t)$ (in fact, $\mathscr{Q}_h(t)=R_h\cap\mathscr{S}_h(t)$ and the transport operation φ is considered exact). If the time step is small enough, the continuity of φ implies that $\varphi(\mathscr{Q}_h(t))$ is a point cloud sampling $\varphi(\mathscr{S}_h(t))$ with average spacing $\simeq h$, so that

$$D(\mathbf{S}_{\mathrm{RAMLS}}(\varphi(\mathscr{Q}_h(t))),\varphi(\mathscr{S}_h(t)))\leqslant C_1h^r \tag{54}$$

with r the rate of convergence studied in previous sections. By the same token,

$$D(\varphi(\mathscr{S}_h(t)),\varphi(\mathscr{S}(t)))\leqslant(1+C_2\Delta t)D(\mathscr{S}_h(t),\mathscr{S}(t)) \tag{55}$$

with C_2 depending on the velocity field but not on h or Δt. We thus have

$$D(\mathscr{S}(t+\Delta t),\mathscr{S}_h(t+\Delta t))\leqslant C_1h^r+(1+C_2\Delta t)D(\mathscr{S}(t),\mathscr{S}_h(t))$$

and thus

$$D(\mathscr{S}(t),\mathscr{S}_h(t))\leqslant C_3\frac{h^r}{\Delta t}+e^{C_2t}D(\mathscr{S}(0),\mathscr{S}_h(0)), \tag{56}$$

so that the geometric error is $\mathscr{O}(h^r/\Delta t)$, typical of semi-Lagrangian methods. If in the previous analysis the time-discretization error in the computation of the trajectories is accounted for, the overall error becomes $\mathscr{O}(h^r/\Delta t)+\mathscr{O}(\Delta t^s)$, with s the order of the time integration method used for Eq. (42). In Table 1 we show the error of the proposed method with several meshes and time steps. The problem considered is a full turn (rigid 2π-rotation) of the ellipsoid of Section 2.5.2 around the x_3-axis with angular velocity equal to 2π (it takes one time unit to make the turn). The specific error measure tabulated is

$$D(\mathscr{S}(T),\mathscr{S}_h(T)). \tag{57}$$

These results are in good agreement with the error estimate (56).

5.2. Zalesak's disk and Zalesak's sphere

This well-documented tests of interface transport consist of the rigid body rotation of a slotted disk or sphere. In the 2D case the disk is centered at $(0.5, 0.75)$, has a radius of 0.15 and a slot of width 0.05 and length 0.25 in a squared domain (see Fig. 14a). The velocity field is given by $\mathbf{v}(x_1,x_2)=\frac{\pi}{314}(0.5-x_2,x_1-0.5)$, so that a complete revolution takes 628 time units. The time step is taken as $\Delta t=1$, and different values of h where taken. Notice that for any given mesh, for example the one with $h=1/512$, the numerical method *does not* compute $512^2=262,144$ nodal values at each time step. Instead, the proposed method computes the intersections of $2\times512=1024$ rays with the RAMLS surface. The accuracy of this computation (see Fig. 14e), which in the mean comprises 700 particles, looks equivalent to that of the PLS method [12] on a 100×100 Eulerian grid with 12,864 particles (estimated under the assumption of 16 particles per narrow-band cell, as in [16]).

Table 2 presents quantitative results of one and two complete turns of the Zalesak's disk. The accuracy of the RAMLS method is similar to that of other state-of-the-art techniques. For instance, for one turn of the disk on a 100×100 Eulerian grid, Enright et al. [12] report an area error of 0.31%, whereas Hieber and Koumoutsakos [16] report 0.30%. Notice that the order of the method goes asymptotically to 1, since the lack of smoothness of the disk makes this order the highest attainable.

The Zalesak's sphere test consists of a sphere of radius 0.15 initially centered at $(0.5, 0.75, 0.75)$ inside a unit cube, which has a slot of width 0.10 and length 0.20 and rotates according to the velocity field $\mathbf{v}(x_1,x_2,x_3)=\frac{\pi}{314}(0.5-x_2,x_1-0.5,0)$. The time step is again $\Delta t=1$ and it takes 628 steps to complete one revolution. The shape of the sphere for t = 0, 79, 157, 236, 314, 393, 471, 550 and 628 is shown in Fig. 15. The adopted mesh size was $h=1/256$, leading to a number of particles in this

Table 1
Error in rigid-body rotation of 3D ellipsoid around x_3-axis, one turn

	$h=\frac{12}{32}$	$h=\frac{12}{64}$	$h=\frac{12}{128}$
$\Delta t=\frac{1}{16}$	0.06261	0.00494	0.00065
$\Delta t=\frac{1}{64}$	0.14635	0.01007	0.00246
$\Delta t=\frac{1}{256}$	0.34056	0.02561	0.00756
$\Delta t=\frac{1}{1024}$	0.76286	0.07314	0.02352
$\Delta t=\frac{1}{4096}$	1.00345	0.16459	0.06323

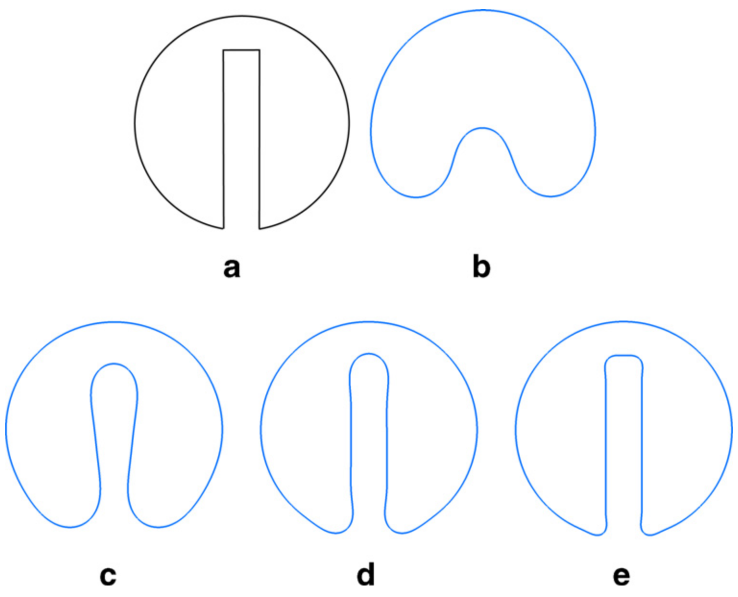

Fig. 14. Rotation of Zalesak's disk: results after one turn: (a) original; (b) $h = 1/64$; (c) $h = 1/128$; (d) $h = 1/256$; (e) $h = 1/512$.

Table 2
Zalesak's disk: RAMLS method (fixed time step, $\Delta t = 1$)

	h	Area	Area loss (%)	Geometric error[a]	Order
	Exact	0.05811580	–	–	–
One turn	1/64	0.05880127	−0.17501	0.03485	N/A
	1/128	0.05810201	0.02372	0.01409	1.30
	1/256	0.05812257	−0.01166	0.00628	1.16
	1/512	0.05811785	−0.00353	0.00335	0.90
Two turns	1/64	0.05859332	−0.82167	0.04470	N/A
	1/128	0.05813988	−0.04144	0.02052	1.12
	1/256	0.05812066	−0.00836	0.00833	1.29
	1/512	0.05812647	−0.01835	0.00415	1.00

[a] As defined in (57).

simulation of about 19,000. It can be noticed that the sharp edges of the solution get smoothed with time, but nevertheless the shape and volume are well preserved.

It is worth to mention that in this study the proposed RAMLS-based method generates a new particle set at each time step. This is certainly not necessary in the case of a rigid rotation, since the distance between the particles is preserved. If the particles were simply advected, as in the Lagrangian methods of Hieber and Koumoutsakos [16] and Du et al. [11], the final shape would practically coincide with the initial one.

5.3. Single vortex flow

We also test our method with the single vortex flow [6]. The initial geometry consists of a circle centered at (0.5, 0.75) with radius equals to 0.15 in a square unit domain. This problem is useful to assess the capability of the method in preserving mass, geometry and topology under severe deformations of the interface. The velocity field is given by

$$\mathbf{v}(x_1, x_2) = 2\left(\sin^2(\pi x_1)\sin(\pi x_2), \sin^2(\pi x_2)\sin(\pi x_1)\right).$$

We take $\Delta t = 0.01$ and discuss results for two grids ($h = 1/256$ and $h = 1/512$) considering t = 1, 3 and 5. The initial states comprise 233 and 469 points, respectively. This last number compares favorably with the 4000 particles used in the Lagrangian level-set method [16] and the 59,000 used in the PLS method [12]. It can be noticed in Fig. 16 that RAMLS results (blue curves) closely approximate the exact solution (black curves) even when the filament is very thin. The accuracy of these results is similar to that of the most effective methods available (compare to Figs. 20 and 21 of [12], Fig. 11 of [16] and Fig. 8 of [11]).

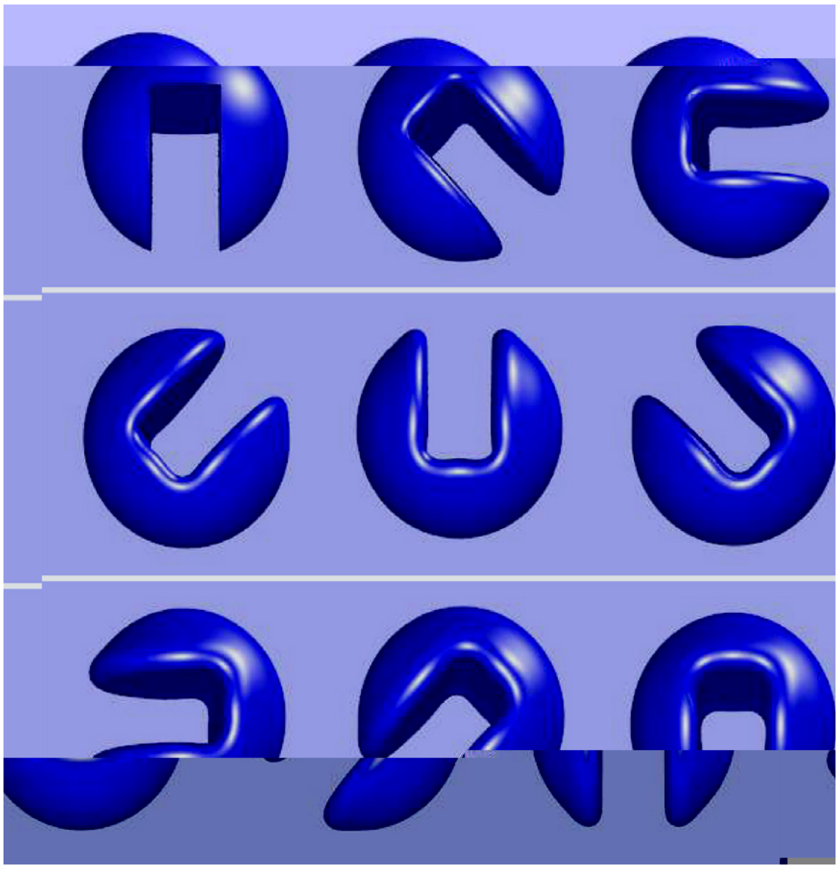

Fig. 15. Rotation of Zalesak's sphere with $h = 1/256$ (19,000 points/particles) and at times t = 0, 79, 157, 236, 314, 393, 471, 550 and 628 time units (from left to right and from top to bottom).

We also ran the single vortex flow test with the velocity field modulated in time by $\cos(\pi t/T)$, so that it is reversed after $t = T/2$ and the exact interface moves back to its original configuration at $t = T$. In our tests we consider T = 8 and 800 time steps. Fig. 17 presents the interface shape yielded by the proposed method at $t = T$ for grid resolutions of 64^2, 128^2, 256^2 and 512^2 (the exact shape coincides with the initial circle). In Table 3 we quantitatively assess these results in terms of area loss and distance between the computed interface and the exact one. The obtained area loss for $h = 1/256$, which turns out to be of 0.36%, is quite similar to that obtained with previous methods at the same resolution [16,12]. The obtained order of convergence is around 2, and not 3 as predicted by our error estimate (56). This is not surprising, since the tail filament is not well resolved in any of the simulations, not even in the one corresponding to $h = 1/512$. Finally, in Table 3, we also present the L_1 error:

$$\frac{1}{2\pi R}\int_0^{2\pi}\left|\frac{f(\theta)^2 - R^2}{2}\right| d\theta, \tag{58}$$

where $f(\theta)$ denotes the distance from the center of the circle to the RAMLS curve and R is the radius of the circle. In order to compute numerically this integral, we estimate $f(\theta)$ from a polygonalization of the RAMLS curve in a finer grid (5000×5000).

It can be seem that the convergence order obtained from L_1 norm is around 2.5 and the error is one order of magnitude smaller than the geometric error.

In Fig. 18 we plot the number of points as a function of time. It shows that the good accuracy obtained with $h = 1/256$ needs just about 1500 points at maximum roll-up of the shape. Further, notice that the curve is symmetric and quite continuous. The point sets are enriched and depleted seemlessly as the interface stretches and contracts. In the computation with $h = 1/128$, for example, the point set starts with 118 points and ends with 128. This is an improvement with respect to the Lagrangian level-set method [16], which starts with 1200 and ends with 2800.

To further assess the proposed method, we compare it to the front-tracking package FronTier of Du et al. [11], downloadable from the website http://frontier.ams.sunysb.edu/download. This package allows for adjustable time stepping, but to simplify the comparison it was run with a fixed time step $\Delta t = 0.002$. Since the interface is reconstructed every five time

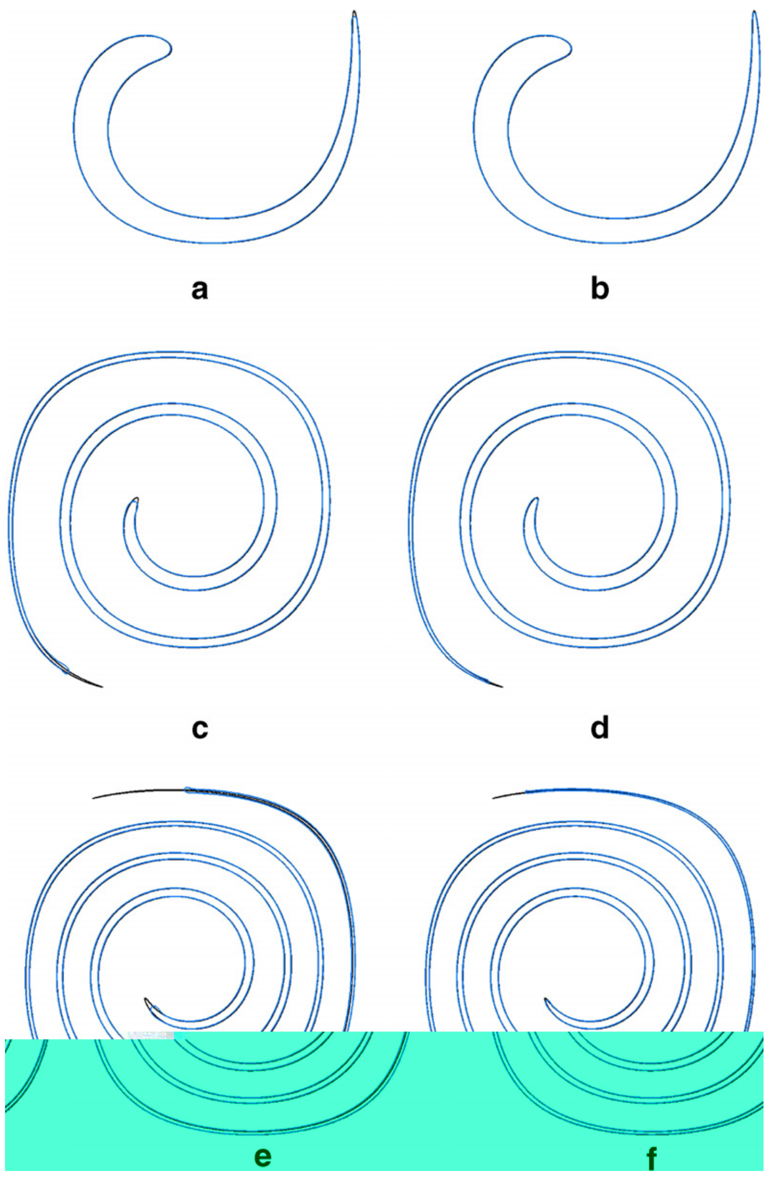

Fig. 16. Solutions obtained with the proposed method for the deformation of a circle under the single vortex flow (in blue), for different meshes and times: (a) $h = 1/256$, $t = 1$; (b) $h = 1/512$, $t = 1$; (c) $h = 1/256$, $t = 3$; (d) $h = 1/512$, $t = 3$; (e) $h = 1/256$, $t = 5$; (f) $h = 1/512$, $t = 5$. In black we plot the exact solution (obtained by Lagrangian tracking of about 100,000 points) for comparison. The time step is $\Delta t = 0.01$ for both meshes. (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

steps, this leads to the same regeneration frequency as that used for our method (i.e., regenerating every 0.01 time units). In Fig. 19 we compare the interface obtained with FronTier taking $h = 1/256$, to that shown in Fig. 16c. The h parameter in the FronTier package is roughly equivalent to that defined in our method, so that the comparison is meaningful. It can be observed that our method is slightly more accurate near the head and the tail of the filament.

Turning to the time-modulated case, in Fig. 20 we show the result of FronTier and compare it to that of our approach, taking $h = 1/256$ for both codes. It can again be observed that the method proposed in this article is slightly more accurate, which is not surprising since our representation of the surface is of higher order than that of FronTier. Regarding computational cost, FronTier was significantly faster than our code in this example, by a factor of about three. The lack of a connectivity structure and the rather large number of neighboring points used in the computation of the surface explain this observation. On the other hand, our RAMLS-based approach avoids the need of the sophisticated topology-reconstruction step used in FronTier, which is unexpensive in 2D but becomes an issue in 3D, as assessed in the next example.

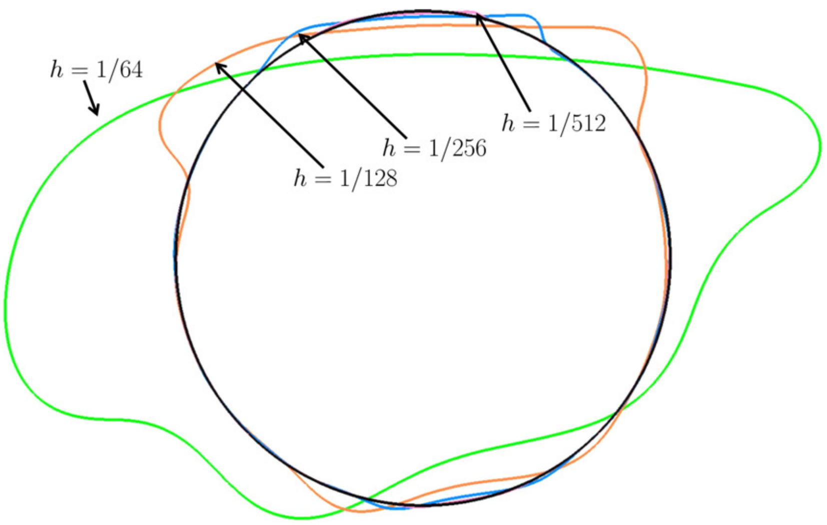

Fig. 17. Numerically obtained shapes at $t = T = 8$ for the single-vortex flow modulated in time, corresponding to grids with $h = 1/64$ (green), $h = 1/128$ (orange), $h = 1/256$, (blue) $h = 1/512$ (pink). The time step is $\Delta t = 0.01$ in all runs. The exact solution (circle) is shown in black for comparison. (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

Table 3
Single vortex flow: RAMLS method (modulated in time by $\cos(\pi t/T)$, $T = 8$ time units)

Grid cells	Area	Area loss (%)	Geom. error[a]	Order	L_1 error	Order
Exact	0.07069	–	–	–		
64	0.09724	−37.57	0.10649	N/A	0.04147	N/A
128	0.073988	−4.67	0.03247	1.71	0.00730	2.50
256	0.070943	−0.36	0.01044	1.64	0.00107	2.77
512	0.070676	0.013	0.00236	2.14	0.00018	2.59

[a] As defined in (57).

5.4. Three-dimensional sphere deformation

In this section, we consider a deformation field in three dimensions. The initial interface $\mathscr{S}(0)$ is a sphere of radius 0.15 centered at (0.35, 0.35, 0.35), the computational domain being the unit cube. The velocity field is given by

$$\mathbf{v}(x_1, x_2, x_3, t) = \begin{pmatrix} 2\sin^2(\pi x_1)\sin(2\pi x_2)\sin(2\pi x_3) \\ -\sin(2\pi x_1)\sin^2(\pi x_2)\sin(2\pi x_3) \\ -\sin(2\pi x_1)\sin(2\pi x_2)\sin^2(\pi x_3) \end{pmatrix} \cos\left(\frac{\pi t}{T}\right).$$

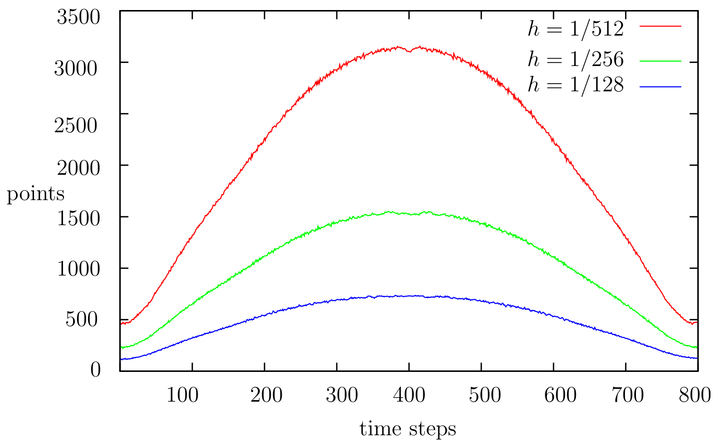

Fig. 18. Evolution of the number of points (or particles) in $\mathscr{Q}_h(t)$ along the 800 time steps in the single-vortex flow modulated in time, for different meshes.

After $t = T = 3$ time units the exact interface recovers the geometry of the original sphere. This benchmark test has also been considered by Enright et al. [12] and by Du et al. [11]. Fig. 21 shows numerical results at selected time steps corresponding to $h = 1/512$, $\Delta t = 0.02$ and 150 time steps. The proposed method preserves the topology well. The final geometry is very close to the original sphere, though with a small artifact (see Fig. 22).

If the mesh spacing is coarsened to $h = 1/256$, it is still possible to obtain acceptable results, as shown in Fig. 23. Though the shape at maximum distortion ($t = 1.5$) is quite accurately reproduced, the numerical artifacts in the final shape are much bigger than before. The FronTier code, which was run under equivalent conditions (fixed time step, 75 surface reconstruction steps along the simulation), recovers the initial sphere much better (see Fig. 24). This is a consequence of the connectivity-less definition of RAMLS surfaces. As the interface stretches, opposite sides of it come into close proximity and they interfere with one another. The FronTier code does not suffer from this because opposite sides of the interface are topologically disconnected. On the other hand, the update and maintenance of a topologically valid connectivity is not free of charge. In this example, the CPU time and memory needed by the FronTier code were 4 and 29 times greater, respectively, than those needed by our code.

5.5. A test with change in topology

As a last study case, we test our front-tracking method in a problem with change in topology. To this end, the initial interface $\mathscr{S}(0)$ is the surface of an "open" ring as shown in Fig. 25a. This interface moves with homogeneous and constant normal

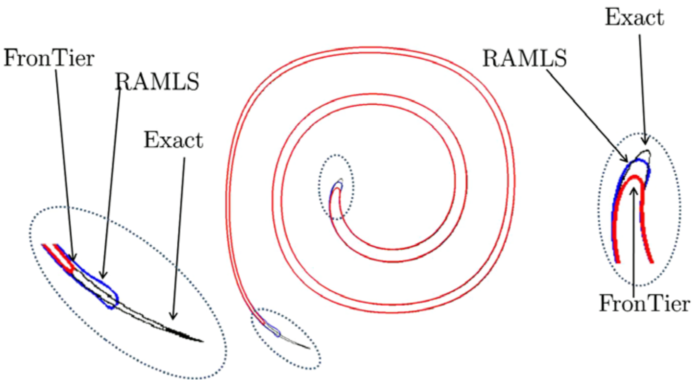

Fig. 19. Comparison of the proposed method (blue curve) with the front-tracking method FronTier (red curve) of Du et al. [11] for the deformation of a circle under the single vortex flow at time $t = 3$. For both codes the mesh size is $h = 1/256$ and the points are regenerated every 0.01 time units. In black we plot the exact solution (obtained by Lagrangian tracking of about 100,000 points for comparison). (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

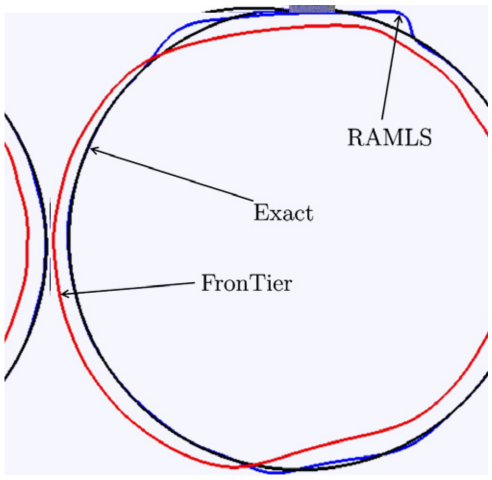

Fig. 20. Numerically obtained shapes at $t = T = 8$ for the single-vortex flow modulated in time. The mesh size is $h = 1/256$ and the points are regenerated every 0.01 time units. The blue curve refers to the RAMLS-based method, whereas the red refers to the FronTier solution. In black, we plot the exact solution. (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

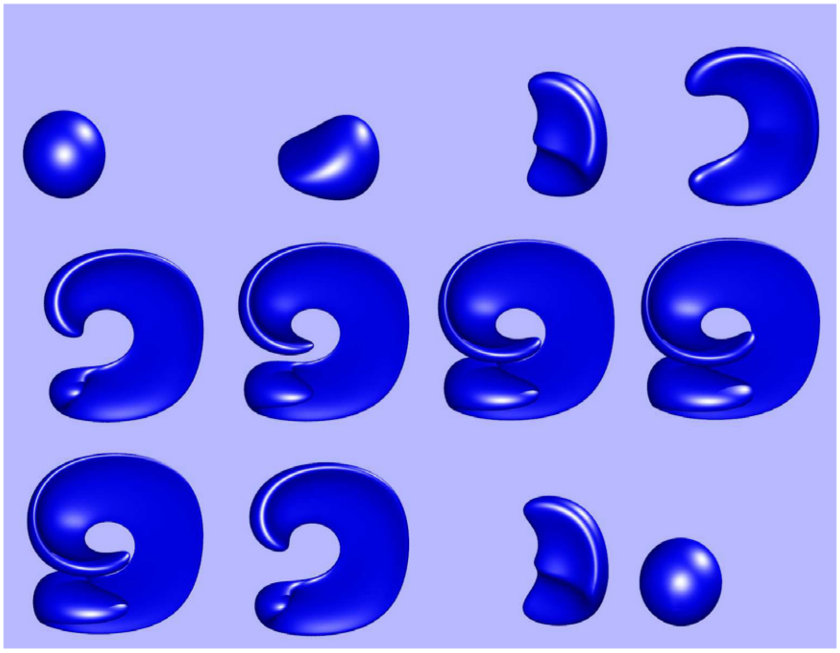

Fig. 21. Three-dimensional deformation of a sphere ($h = 1/512$). From left to right and from top to bottom: Numerically obtained surfaces at $t = 0, 10\Delta t$, $20\Delta t$, $30\Delta t$, $40\Delta t$, $50\Delta t$, $60\Delta t$, $70\Delta t$, $90\Delta t$, $110\Delta t$, $130\Delta t$, $150\Delta t$, where $\Delta t = 3/150$. The number of points (or particles) in the corresponding sets $\mathscr{P}_h$ are: 65,000, 73,790, 88,263, 136,299, 183,720, 228,456, 263,896, 280,663, 263,853, 183,932, 88,545 and 69,295.

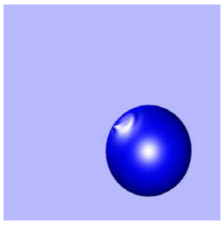

Fig. 22. The interface of Fig. 21 at $t = 3$ from another point of view. Notice the small spurious cusp.

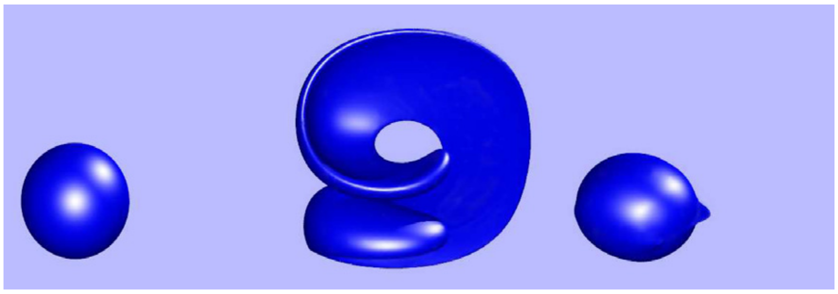

Fig. 23. Three-dimensional deformation of a sphere ($h = 1/256$). Results of the RAMLS-based method at $t = 0$, $t = 1.5$ and $t = 3$.

velocity equal to one, i.e., $\mathbf{v}(\mathbf{x}, t) = \mathbf{n}(\mathbf{x})$ for all $\mathbf{x} \in \mathscr{S}(t)$, for each t. The radius of the ring is 0.3, its initial thickness is 0.02, the grid size is $h = 1/64$ and the time step is $\Delta t = 0.01$.

Fig. 25 presents the results of this simulation, which was performed with the front-tracking algorithm as described in Sections 4.1 and 4.2, with no case-specific tuning of any kind. The change in topology (coalescence) is handled without any numerical disturbance, as is the case with level-set based Eulerian methods. For Lagrangian, surface-mesh-based front-tracking methods, the simulation would have collapsed (or at least needed major mesh surgery, as in the reconstruction step of the FronTier code [11]) at the stage of Fig. 25d.

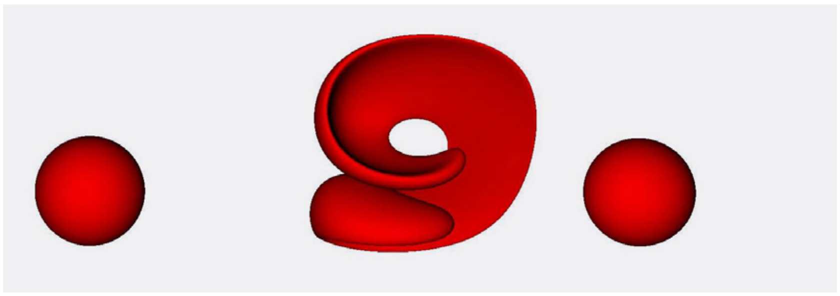

Fig. 24. Three-dimensional deformation of a sphere ($h = 1/256$). Results of the FronTier code at $t = 0$, $t = 1.5$ and $t = 3$.

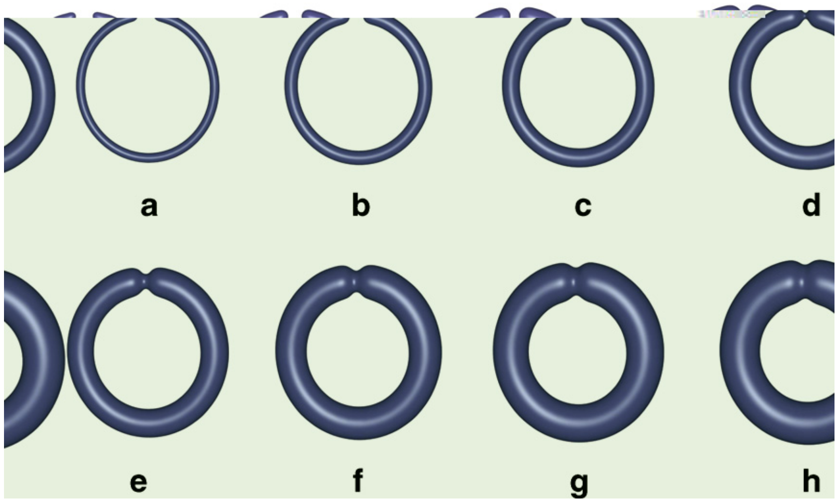

Fig. 25. Results of the test with change in topology at times (a) 0, (b) 0.2, (c) 0.4, (d) 0.6, (e) 0.8, (f) 1.0, (g) 1.2 and (h) 1.4. The change in topology takes place at $t = 0.6$ (part (d) of the figure).

The results of this test encourage the further development of the proposed method in such a way that a robust treatment of topology changes can be guaranteed in any situation. In its present form, if the same case is run with a twice larger time step, numerical artifacts pollute the solution.

The method also seems to suffer of some apparent "numerical surface tension" (see Figs. 25d–h). This should be properly accounted for in the simulation of multiphase flows with significant surface tension effects.

6. Conclusions

The simulation of a mechanical problem with evolving interfaces needs specific methods for two distinct, independent tasks: representing the interface along its evolution, and solving the corresponding equations in a time-varying domain.

Several unconnected point-set methods have already been proposed for the solution of fluid-mechanics problems, such as Smoothed Particle Hydrodynamics, meshless methods and particle methods (see, e.g., [5,28,7,18,25,19] and references there-in). These methods use particles over the whole domain, so that the two tasks mentioned above become coupled.

If, on the other hand, the task of interface representation is kept separate (to couple it with an Eulerian flow solver for example), only a few methods based on unconnected point sets exist [43,12,16]. In this article, we have described a powerful tool of representation of surfaces based on unconnected point sets, namely algebraic moving-least-squares (AMLS) surfaces. To our knowledge, this technique has not been hereto applied to evolving interface problems. It can accurately represent 3D shapes with a quite small number of points, requiring little regularity on their distribution (see Figs. 6b and 7). Further, we introduced in this article the technique of RAMLS surfaces, a parameter-free variant of a technique developed by Guenne-baud and Gross [15] to improve the robustness of the representation and reduce its computational cost, while maintaining its accuracy (see Figs. 9 and 10).

There exist many ways in which a front-tracking method can be devised on the basis of RAMLS surfaces. The most immediate one would be a purely Lagrangian motion of the points (or "particles") that define the RAMLS surface. This would obviously require of periodic reseeding/deletion of points to keep the population well distributed over the interface. In this article, however, we have devised a method in which the point set is re-generated at each time step, locating the new points at the intersections of a pre-defined collection R_h of lines (or "rays") with the interface. This automatically maintains a good distribution of points over the interface and provides a way to deal with topological changes. The idea behind this strategy is that R_h could in fact be taken as the gridlines of an underlying Eulerian grid that is used for the flow solver. In this case, the intersections of the gridlines with the interface are indeed needed by the flow solver and one would get the new point set at no extra cost. Notice that in this way the particle distribution is automatically made consistent with that of the flow-solver grid, and that extension to curvilinear Eulerian grids is immediate. Besides, in situations in which the intersections are not needed by the solver the re-generation of the point set could be performed less often (as, e.g., in [11,16]) thus alleviating the computational burden.

The numerous tests reported in the previous sections allow us to conclude that the proposed method of RAMLS representation of surfaces is competitive with previous methods in terms of accuracy and robustness. Further, the proposed front-tracking method based on RAMLS surfaces proved to perform well in benchmarks of interface transport/stretching, in general requiring significantly less particles than the Particle or Lagrangian level-set methods. It also compared favorably in terms of accuracy with the front-tracking method FronTier introduced by Du et al. [11], which maintains a connectivity structure on the set of marker particles.

Acknowledgments

Helpful comments made by the anonymous reviewers are gratefully acknowledged. The authors received financial support from Fundação de Amparo à Pesquisa do Estado de São Paulo – FAPESP (Brazil), from Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq (Brazil), and from Agencia Nacional de Promoción Científica y Tecnológica (Argentina). GCB is on leave from CNEA (Argentina) and CONICET (Argentina).

References

[1] A. Adamson, M. Alexa, Ray tracing point set surfaces, in: Proceedings of the Shape Modeling International, 2003, pp. 272–279.
[2] M. Alexa, A. Adamson, On normals and projection operators for surfaces defined by point sets, in: Proceeding of Eurographics Symposium on Point-Based Graphics, 2004, pp. 149–155.
[3] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva, Computing and rendering point set surfaces, IEEE Trans. Visual. Comput. Graph. 9 (1) (2003) 3–15.
[4] N. Amenta, Y. Joo Kil, Defining point-set surfaces, ACM Trans. Graph. 23 (3) (2004) 264–270.
[5] I. Babuška, U. Banerjee, J. Osborn, Survey of meshless and generalized finite element methods: a unified approach, Acta Numer. 12 (2003) 1–125.
[6] J.B. Bell, P. Colella, H.M. Glaz, A 2nd-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 2 (85) (1989) 257–283.
[7] T. Belytschko, Y. Guo, W.K. Liu, S. Xiao, A unified stability analysis of meshless particle methods, Int. J. Numer. Methods Eng. 48 (2000) 1359–1400.
[8] P. Carrica, R. Wilson, F. Stern, Unsteady RANS simulation for the forward speed diffraction problem, Comput. Fluids 35 (2006) 545–570.
[9] P. Carrica, R. Wilson, F. Stern, An unsteady single-phase level set method for viscous free surface flows, Int. J. Numer. Methods Fluids 53 (2007) 229–256.
[10] M. deBerg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Springer-Verlag, 1997.
[11] J. Du, B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, L. Wu, A simple package for front tracking, J. Comput. Phys. 213 (2) (2006) 613–628.
[12] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83–116.
[13] D. Enright, F. Losasso, R. Fedkiw, A fast and accurate semi-Lagrangian particle level set method, Comput. Struct. 83 (2005) 479–490.
[14] O. Gloth, D. Hänel, L. Tran, R. Vilsmeier, A front tracking method on unstructured grids, Comput. Fluids 32 (2003) 547–570.
[15] G. Guennebaud, M. Gross, Algebraic point set surfaces, in: SIGGRAPH'07: ACM SIGGRAPH 2007 Papers, Article 23, ACM, New York, NY, USA, 2007.
[16] S. Hieber, P. Koumoutsakos, A Lagrangian particle level set method, J. Comput. Phys. 210 (2005) 342–367.
[17] H. Hoppe, T. DeRose, T. Duchampy, J. McDonaldz, W. Stuetzlez, Surface reconstruction from unorganized points, Comput. Graph. 26 (2) (1992) 71–78.
[18] A. Huerta, T. Belytschko, S. Fernandez-Mendez, S. Rabczuk, Meshfree methods, in: E. Stein, R. de Borst, T.J.R. Hughes (Eds.), Encyclopedia of Computational Mechanics, vol. 1, Wiley, 2004, pp. 279–309.
[19] P. Koumoutsakos, Multiscale flow simulations using particles, Annu. Rev. Fluid Mech. 37 (2005) 457–487.
[20] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, J. Comput. Phys. 113 (1994) 134–147.
[21] D. Lakehal, M. Meier, M. Fulgosi, Interface tracking towards the direct simulation of heat and mass transfer in multiphase flows, Int. J. Heat Fluid Flow 23 (2002) 242–257.
[22] D. Levin, The approximation power of moving least-squares, Math. Comput. 67 (224) (1998) 1517–1531.
[23] D. Levin, Mesh-independent surface interpolation, Adv. Comput. Math. 2 (2001) 37–49.
[24] F. Losasso, R. Fedkiw, S. Osher, Spatially adaptive techniques for level set methods and incompressible flow, Comput. Fluids 35 (2006) 995–1010.
[25] J. Monaghan, Simulating free surface flows with SPH, J. Comput. Phys. 110 (1994) 399–406.
[26] J.R. Munkres, Topology, second ed., Prentice-Hall, 2000.
[27] F. Mut, G. Buscaglia, E. Dari, New mass-conserving algorithm for level set redistancing on unstructured meshes, ASME J. Appl. Mech. 73 (2006) 1011–1016.
[28] E. Oñate, S. Idelsohn, M. Celigueta, R. Rossi, Advances in the particle finite element method for the analysis of fluid-multibody interaction and bed erosion in free surface flows, Comput. Methods Appl. Mech. Eng. 197 (2008) 1777–1800.
[29] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Applied Mathematical Sciences, vol. 153, Springer, 2003.
[30] S. Osher, J.A. Sethian, Front propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[31] V. Pratt, Direct least-squares fitting of algebraic surfaces, SIGGRAPH Comput. Graph. 21 (4) (1987) 145–152.
[32] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annu. Rev. Fluid Mech. 31 (1999) 567–603.
[33] J.A. Sethian, Fast marching methods and level set methods for propagating interfaces, in: von Karman Institute Lecture Series, Computational Fluid Mechanics, 1998.
[34] J.A. Sethian, P. Smereka, Level set methods for fluids interfaces, Annu. Rev. Fluid Mech. 35 (2003) 341–372.
[35] F.S. Sousa, N. Mangiavacchi, L.G. Nonato, A. Castelo, M. Tomé, V. Ferreira, J. Cuminato, S. McKee, A front-tracking method for simulation of 3D multifluid flows with free surfaces, J. Comput. Phys. 198 (2004) 469–499.

[36] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, J. Comput. Phys. 187 (2003) 110–136.
[37] M. Sussman, E. Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, SIAM J. Sci. Comput. 20 (4) (1999) 1165–1191.
[38] M. Sussman, E. Fatemi, P. Smereka, S. Osher, An improved level set method for incompressible two-phase flows, Comput. Fluids 27 (5–6) (1998) 663–680.
[39] M. Sussman, E.G. Puckett, A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows, J. Comput. Phys. 162 (2000) 301–337.
[40] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[41] G. Taubin, Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 13 (11) (1991) 1115–1138.
[42] G. Taubin, An improved algorithm for algebraic curve and surface fitting, Comput. Vision (1993) 658–665.
[43] D. Torres, J. Brackbill, The point-set method: front-tracking without connectivity, J. Comput. Phys. 165 (2000) 620–644.
[44] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[45] S. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, J. Comput. Phys. 100 (1992) 25–37.
[46] I. Wald, H.P. Seidel, Interactive ray tracing of point-based models, in: Eurographics Symposium on Point-Based Graphics, 2005, pp. 1–8.
[47] R. Wilson, P. Carrica, F. Stern, Simulation of ship breaking bow waves and induced vortices and scars, Int. J. Numer. Methods Fluids 129 (2007) 1445–1459.