Journal of Computational Physics 305 (2016) 838–876

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A fourth-order accurate curvature computation in a level set framework for two-phase flows subjected to surface tension forces

Mathieu Coquerelle *, Stéphane Glockner

Université de Bordeaux, CNRS, UMR 5295, Bordeaux INP, 16, avenue Pey-Berland, 33607 PESSAC Cedex, France

ARTICLE INFO

Article history:
Received 28 January 2015
Received in revised form 16 September 2015
Accepted 8 November 2015
Available online 10 November 2015

Keywords:
Surface tension
Curvature computation
Level set method
Spurious currents
Two-phase flow
Continuum surface force
Balanced force algorithm
Closest point method

ABSTRACT

We propose an accurate and robust fourth-order curvature extension algorithm in a level set framework for the transport of the interface. The method is based on the Continuum Surface Force approach, and is shown to efficiently calculate surface tension forces for two-phase flows. In this framework, the accuracy of the algorithms mostly relies on the precise computation of the surface curvature which we propose to accomplish using a two-step algorithm: first by computing a reliable fourth-order curvature estimation from the level set function, and second by extending this curvature rigorously in the vicinity of the surface, following the Closest Point principle. The algorithm is easy to implement and to integrate into existing solvers, and can easily be extended to 3D. We propose a detailed analysis of the geometrical and numerical criteria responsible for the appearance of spurious currents, a well known phenomenon observed in various numerical frameworks. We study the effectiveness of this novel numerical method on state-of-the-art test cases showing that the resulting curvature estimate significantly reduces parasitic currents. In addition, the proposed approach converges to fourth-order regarding spatial discretization, which is two orders of magnitude better than algorithms currently available. We also show the necessity for high-order transport methods for the surface by studying the case of the 2D advection of a column at equilibrium thereby proving the robustness of the proposed approach. The algorithm is further validated on more complex test cases such as a rising bubble.

1. Introduction

Surface tension is a key mechanical force in multi-phase flows where it plays a particularly important role in hydrodynamics at small scales. Rising bubbles and airborne water drops are common examples of such flows (see for example [1–3]). The surface tension force lies at the interface between the two phases and is thus singular at that location. This singularity makes it a complicated continuum mechanics problem when coupled with the Navier–Stokes equations.

From a numerical point of view, this singular force is challenging as it requires precise localization of the surface, its associated normal vector and curvature. A variety of methods have been introduced to localize the surface advected by the flow, the front-tracking method [4] in a Lagrangian framework, the Volume Of Fluid (VOF) method [5] and the Level Set (LS) method [6] in a Eulerian framework. These three widely used approaches offer advantages and drawbacks such that these

* Corresponding author.
E-mail addresses: mathieu.coquerelle@bordeaux-inp.fr (M. Coquerelle), glockner@ipb.fr (S. Glockner).

http://dx.doi.org/10.1016/j.jcp.2015.11.014
0021-9991/

methods are actively researched today. We base our original work on the level set framework which permits easy capture of topological changes and offers the advantage of simpler algorithms and a natural extension from 2D to 3D. However, in comparison to the Lagrangian representation, precise localization of the surface is lost and the level set method usually suffers from more volume loss than, for example, VOF methods. A wide range of techniques has been proposed to alleviate these problems such as the hybrid particle level set [7], the CLSVOF method [8] and high-order semi-Lagrangian particles method [9]. As we demonstrate in this article, accurate transport of the surface is required to attain high-order computation of the dependent curvature. In a Eulerian framework, the surface is immersed in the fluid through the use of a Dirac mass and a Heaviside function to characterize the phases.

The surface tension force is commonly introduced in the Navier–Stokes equations as an external force located at the interface. The Continuum Surface Force (CSF) model introduced in [10] gives an elegant solution to regularize the singular term. The method is based on spreading the force over a small neighborhood around the interface. In the last decade, research has been focused on the reduction of spurious currents, arising in the vicinity of the surface [11–16]. These currents are numerical parasitic velocities introduced in the flow by errors in the estimation of the surface tension force. As demonstrated in [17], they are quadratically proportional to the curvature which leads to the need of an accurate curvature computation. Following the observations of François et al. [11], in the CSF model the creation of such currents is due to:

1. non-coherent numerical differentiation in solving the Navier–Stokes equations,
2. non-coherent computations of the surface position, normal vector and curvature.

The first criterion has been recently addressed by the introduction of a balanced-force algorithm [11] which is based on coherent numerical schemes for computing the surface tension force and the associated pressure jump. This technique reduces spurious currents down to machine precision when the curvature is prescribed exactly.

The second cause of these currents is still being investigated today through improving the accuracy of the interface normal and curvature computations, which are direct derivatives of the surface and thus also singularly located. In consequence, they need to be treated adequately in a Eulerian framework. For VOF methods, the Height Function (HF) [18] approach (along with some other improvements) is more precise and drastically reduces spurious currents as shown in [14]. However, their accuracy relies on the precision of the underlying phase function, itself dependent on the VOF transport algorithm (we refer the reader to [19] for a comparative study of the PLIC, WLIC, THINC and CLSVOF methods), which hardly reaches second-order in space accuracy as it is discussed in §2.4.1. More complex methods such as MOF [20] can improve the accuracy of the volume fraction while still not being precise enough to ensure spatial convergence of the surface curvature after its transport. That is why recent research focuses on coupled methods like CLSVOF [8] or CLSMOF [21] which use the underlying level set for more accuracy on the curvature computation.

For level sets, a curvature extension based on a closest point algorithm [22] has been efficiently used in [12], for unstructured grids, which reduces spurious currents much more than previous techniques. The implementation of this method is quite simple and can be easily extended to 3D.

In the CSF approach, the accuracy of the curvature calculation is essential to reduce the introduction of errors in the flow. In this paper we present a detailed study of the difficulties arising in the numerical computation of surface tension forces in a Eulerian framework, which relies on the extension of the curvature in the vicinity of the interface. In section 2.4 we develop three key criteria for evaluating the quality of this extension, and illustrate their respective effects on flow dynamics. These criteria are guidelines for the techniques described thereafter. In particular, we assert that a minimized variation of the curvature in the normal direction of the surface is crucial to reduce numerical errors on the surface tension term.

In §3.4, we present the principles of curvature extension for level sets and compare three different methods for their computation, the last being the novel method that we propose:

1. the osculatory circle approximation which consists in a linear extension of the curvature in the normal direction,
2. the Closest Point method based on the interpolation on the surface of a curvature approximation, as proposed by Herrmann in [12],
3. the proposed approach based on an improved Closest Point algorithm that ensures the colinearity criterion, and uses reinterpolation to reduce variation in the normal direction.

We discuss the respective advantages and drawbacks of each technique in order to propose a general understanding of curvature extension in a level set framework. The accuracy of the methods is then compared and analyzed on geometric cases in §4.2. All three methods give very good results with circular geometry. In order to discriminate between the methods, we studied the case of the ellipse which is more challenging because the variation of the curvature is less trivial. Consequently, the proposed Closest Point algorithm with colinearity properties and reinterpolation is finally retained for its high accuracy and robustness.

We first validate our approach for the classical static column, first with constant density §4.3 and then with variable density §4.4, and prove its ability to drastically reduce spurious currents at fourth-order convergence rate. Popinet [14] studied the spurious currents when a column is advected by a constant velocity. This case is a good challenge for surface tension methods because it induces more errors at the interface. In §4.5, we report very satisfactory results demonstrating

Table 1
Terminology. In the paper, all units are given in the International System of Units.

Symbol	Definition	Symbol	Definition
ρ	Fluid density	Γ	A surface
μ	Fluid dynamic viscosity	$\mathbf{n}$	Surface normal
$\mathbf{u}$	Fluid velocity	τ	Surface tangent
p	Pressure	κ	Surface curvature
σ	Surface tension coefficient	ϕ, ψ	Level set functions
H	The Heaviside function	$\widetilde{\kappa}$	Discrete representation for any κ
δ	The Dirac mass	κ_{ex}	Exact value for any κ
h	The spatial discretization step	κ_σ	The standard deviation of the curvature
Δt	The temporal discretization step	$\kappa_{\sigma,\mathbf{n}}$	The normal standard deviation of the curvature
$\mathbf{x}_{ij}$	The discrete value of $\mathbf{x}$ at i, j	RMS	Root Mean Square

that the algorithm proposed is robust to the transport of the surface. This robustness is due to high-order resolution of the advection equation coupled with accurate curvature extension. Finally, we validate our algorithm with a comparison of numerical results obtained from the simulation of the zero gravity drop oscillation §4.6 and the 2D bubble rise §4.7. This demonstrates that the approach proposed can be effectively used in Navier Stokes solvers based on level set methods.

2. Governing equations

2.1. Terminology

Before presenting the equations we present in Table 1 the terminology and notations that will be used through the whole document.

2.2. Two-phase fluid flow equations in the presence of surface tension forces

The incompressible Navier–Stokes equations with variable densities and viscosities following the continuum–surface–force (CSF) approach originally proposed by Brackbill et al. [10] can be written as:

$$\begin{aligned} \rho(\partial_t \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u}) &= -\nabla p + 2\nabla\cdot(\mu\mathbf{D}) + \sigma\kappa\delta_\Gamma\mathbf{n} \\ \partial_t\rho + (\mathbf{u}\cdot\nabla)\rho &= 0 \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned} \tag{1}$$

with $\mathbf{D} = \left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right)/2$ the deformation tensor and p the pressure field. The term $\sigma\kappa\delta_\Gamma\mathbf{n}$ is singularly located on the surface Γ separating the two phases by a Dirac delta function δ_Γ, oriented in the normal direction $\mathbf{n}$, with curvature κ and a physical surface tension coefficient σ. Equation (1) is valid under the assumption a constant σ along the surface between the two phases which is a common assumption in most fluid dynamics computations.

2.3. Level set representation for two-phase flows

In the level set approach, the phases of the fluid are spatially identified in n dimensions using a function $\phi : \mathbb{R}^n \to \mathbb{R}$ such that each phase stands on one side of a certain level set value α. Thus, they are separated by the surface:

$$\Gamma(t) = \{\mathbf{x} \,|\, \phi(\mathbf{x}, t) = \alpha\}. \tag{2}$$

For the sake of simplicity we choose $\alpha = 0$, implying that the first (resp. second) phase stands where ϕ is negative (resp. positive).

The local concentration c_i – or characteristic function – of the ith phase is written

$$c_i(\mathbf{x}, t) = H_i(\phi(\mathbf{x}, t)),$$

where $H_1(y) = H(y) = 1 - H_2(y)$ and $H(y)$ is the Heaviside function defined as $H'(y) = \delta(y)$.

The density ρ is expressed over the whole domain as a function of ϕ:

$$\rho(\mathbf{x}, t) = H(\phi(\mathbf{x}, t))\rho_1 + (1 - H(\phi(\mathbf{x}, t)))\rho_2, \tag{3}$$

where ρ_i is the constant density of the ith phase. A similar equation represents the viscosity $\mu(\mathbf{x}, t)$.

As in common level set methods, the function ϕ is passively transported by the fluid through the advection equation:

$$\partial_t\phi + \mathbf{u}\cdot\nabla\phi = 0 \tag{4}$$

which implicitly follows the position of the surface over time.

The normal and curvature fields can be computed for the whole domain as:

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}, \tag{5}$$

and

$$\kappa = \nabla\cdot\mathbf{n} = \nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) \tag{6}$$

Equations (5) and (6) coincide with the normal to and the curvature of the surface Γ for points $\mathbf{x}$ such as $\phi(\mathbf{x}) = \alpha$.

Due to the numerical discretization in the level set Eulerian framework we work with a smooth representation of those quantities where δ_Γ and H are not singular but spread in the vicinity of Γ. Thus we will use the term **extension** for any quantity that physically stands on the surface but is numerically and spatially extended around it.

2.4. Problems arising with surface tension forces

As stated in the introduction, a qualitative computation of surface tension forces to minimize the spurious currents is based on two criteria:

Criterion 1. *The consistent numerical discretization of the pressure and surface tension terms.*

Criterion 2. *The rational computation of the surface's curvature.*

By rational computation we mean a curvature computation that assures the three following aspects:

Criterion 2.1. *Fast convergence to the exact curvature.*

Criterion 2.2. *Minimized deviation along the surface.*

Criterion 2.3. *Minimized variation along the surface's normal.*

Several efficient and accurate numerical methods have been developed over the last decade focusing on the computation of the curvature and the surface tension term. The balanced-force algorithm used in different flow simulation frameworks and based on the Ghost Fluid Method [11], VOF [14], level set on unstructured grids [12] has greatly reduced the numerical errors by satisfying Criterion 1 with methods suitable for the curvature computation to satisfy Criterion 2.

Our flow solver presented in section 3 is based on the same balanced-force principle and an improved curvature computation that complies with Criteria 2.1, 2.2 and 2.3.

For the sake of clarity we explain the importance of those 3 criteria using a 2D case of the static column equilibrium which has been widely used in the literature as a minimal and representative case.

2.4.1. Fast convergence to the exact curvature

The first criterion is essential to computationally converge to the physical experiments, as shown in Fig. 2(a). Even in a stable simulation, a significant error in the calculation of the curvature can drastically change the hydrodynamics. This becomes particularly important when the computational grid is refined as high curvatures can arise and often dominate the dynamic processes at those scales.

Hence it is essential to have a precise and fast convergent computation of the curvature, at least at the same order as the flow solver. Whatever the choice Eulerian or Lagrangian for the surface representation, the curvature depends on the second derivatives of a given function ϕ. For example, the curvature of a parameterized curve $\Gamma = (x(t), y(t))$ is:

$$\kappa = \frac{x'y'' - y'x''}{\left(x'^2 + y'^2\right)^{3/2}}$$

or given by equation 6 for level sets.

In this section, we use the term "error" in a general context. In a numerical point of view, we use the most restrictive L_∞ norm to define the error E_∞ between an analytical solution ψ and its numerical approximation ϕ as:

$$E_\infty(\psi,\phi) = \max|\psi - \phi|\,.$$

Errors in second derivative computation.

Proposition 1. *The error in the second derivative is of order $m-2$ when the error on the surface representation is of order m.*

To prove this, we first illustrate the impact of numerical errors on the second derivatives of a function. This demonstration can be extended to the general case of surface and curvature computation, regardless of whether the surface is Eulerian or Lagrangian.

Let ψ be a 1D function of x, sufficiently differentiable. In addition, let ϕ be a perturbed version of ψ such that $\phi(x,h,m) = \psi(x) + e(x,h,m)$, where $e(x,h,m)$ represents a perturbation of amplitude of order h^m with spatial frequency $1/h$ (where, in a numerical computation, h is the spatial discretization, i.e. the smallest variation that can be captured). The associated numerical error is:

$$E_\infty(\psi,\phi) = \max|\psi(x) - \phi(x,h,m)| = \max|e(x,h,m)| = O(h^m).$$

The nth derivative of ϕ is a function of the nth derivatives of ψ and $e(x,h,m)$:

$$\frac{\partial^n \phi(x,h,m)}{\partial x^n} = \frac{\partial^n \psi(x)}{\partial x^n} + \frac{\partial^n e(x,h,m)}{\partial x^n}.$$

Then the perturbation on the nth derivative of ϕ will be of the order h^{m-n}, i.e. the associated numerical error is:

$$E_\infty\left(\frac{\partial^n \psi}{\partial x^n}, \frac{\partial^n \phi}{\partial x^n}\right) = \max\left|\frac{\partial^n \psi(x)}{\partial x^n} - \frac{\partial^n \psi(x,h,m)}{\partial x^n}\right| = \max\left|\frac{\partial^n e(x,h,m)}{\partial x^n}\right| = O(h^{m-n}).$$

Hence, if $m > n$ then the function $\frac{\partial^n \phi}{\partial x^n}$ will converge toward $\frac{\partial^n \psi}{\partial x^n}$ at a positive rate of $O(h^{m-n})$, otherwise it will not converge and lead to errors $> O(1)$ increasing as $h \to 0$. This is completely independent of the accuracy of the underlying numerical scheme used to discretize the quantities, especially the curvature.

A detailed example is given in Appendix A.

Errors in the initialization and transport of the surface. When a function ϕ represents the surface, whether in a Lagrangian or Eulerian fashion, and if the error in its initialization and/or transport is less than the order 2, then the errors in the second derivatives of ϕ – hence the curvature – will be at least $O(1)$, given the previous demonstration. This will lead to increasing error in the flow as the spatial discretization tends to 0.

Therefore, in order to have at least a second-order error in the computation of the curvature, it is mandatory to **initialize** and **transport** the surface with errors less than order h^4 compared to the exact solution. This is true regardless of the errors in the velocity field used to transport the surface as the exact solution also has to be transported by the perturbed flow.

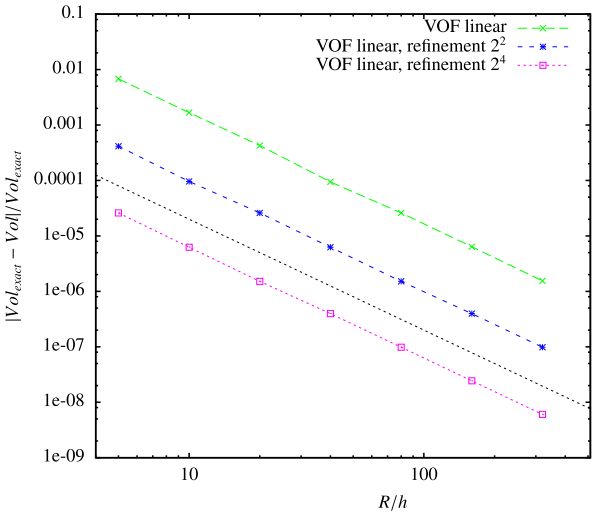
Examples for second-order precision

- In a Lagrangian representation, the discretized points have to stand closer than h^4 to the exact surface. The velocity interpolation at those points must be at least fourth-order.
- In a level set/signed distance approach, the distance to the surface has to be computed at least to fourth-order, which is easily attainable for common geometries. The transport equation must be solved at the same order, as well as reinitialization algorithms to ensure a similar order of precision. If fourth-order methods are used to transport and reinitialize ϕ then the curvature computation will never be better than second-order.
- In VOF methods, the fraction of volume occupied by the phase has to be computed at least to fourth-order, which is not trivially obtainable even for common geometries. In [18] the authors use a refinement technique to initialize the volume fraction more precisely, up to second-order and the curvature is computed with the HF method. In Fig. 9 and in the text of section 5.1, they show a second-order convergence of the curvature error. This statement is not correct as proven in the above paragraph. To demonstrate this we did the exercise with the same parameters and methods (with the HF algorithm implemented from [23]). As shown in Fig. 1, the curvature error clearly diverges when using finer meshes than the one used in [18]: what they have observed is a bias visible at coarse meshes (for $R/h < 40$) which is due to reduced numerical errors on the initial volume fraction, relatively to the exact value. However, these errors become dominant with finer discretization (for $R/h \geq 40$) and exhibit non-convergence, as proven in Proposition 1. More precise methods for the initialization have been proposed since, like for example [24], but as VOF is not the representation chosen for our method, we will not discuss the question further. Additionally, the transport equation also has to be at least fourth-order. Fig. 1 also shows that, even with the exact volume fraction computation, the use of a low order transport technique does not lead to spatial convergence for the curvature. Until recently, no VOF transport method was sufficiently precise to attain high-order dynamic computation of the curvature; recent methods such as DRAC [25] are fourth-order accurate in space.

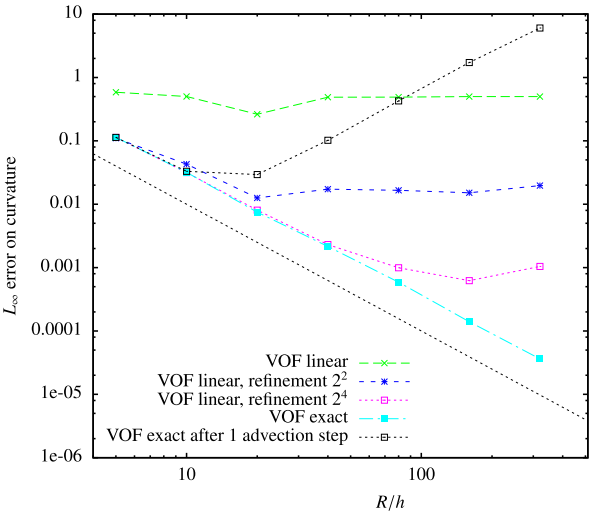
2.4.2. Minimized deviation along the surface

In the particular case of a static column in 2D (resp. a static sphere in 3D), the overall deviation of the curvature along the surface can be expressed through the standard deviation. It is described on the surface as:

$$\tilde{\kappa}_\sigma = \sqrt{\int_\Gamma (\tilde{\kappa} - \tilde{\kappa}_{mean})^2} \tag{7}$$



(a) Convergence of the L_1 error on the volume of the disk.



(b) Non convergence of the L_∞ curvature error for the low-order initialization and after transport of the exact volume fractions.

Fig. 1. L_1 volume and L_∞ curvature error with VOF representation, for a disk of radius $R = 0.25$ centered in a box of size 1, $h = 1/N$ is the spatial discretization step. The volume fractions are initialized with linear segments, with or without mesh refinement (like proposed in [18]) or with the exact solution obtained by integration of the circle equation. The curvature is computed with the second-order accurate Height Function algorithm implemented from [23].

where the tilde denotes discrete measures, the σ symbol is used here to mean standard deviation and $\tilde{\kappa}_{mean}$ is the numerically computed mean curvature:

$$\tilde{\kappa}_{mean} = \int_\Gamma \tilde{\kappa}. \tag{8}$$

As stated in [14], this second criterion is numerically relevant as, in the case of a balanced force algorithm, it reduces the spurious currents to the machine precision. This statement is true even in the case of a poor computation of the absolute physical curvature, i.e. even when $|\tilde{\kappa}_{mean} - \kappa_{exact}|$ is high.

For the static column equilibrium problem, the presence of oscillations on the curvature evaluation along the interface will result in the appearance of capillary waves, thus creating undesired spurious currents in the whole simulation. It is important to note that this criterion is hardly extendable to the general case where the curvature is not constant along the surface. However, one can picture those small variations in the curvature as a bias in the geometry between the position of the surface, its normal vector and curvature that will disrupt the flow dynamic through the surface tension errors, as it is depicted in Fig. 2(b).

In a more general context where the curvature is not constant, the deviation along the surface is less trivial. It can be expressed in the proximity of $\mathbf{x}$ a point on the surface Γ, as:

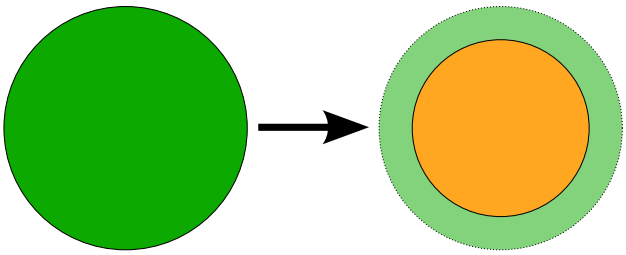
$$\tilde{\kappa}_\sigma(\mathbf{x}) = \sqrt{\int_{-\epsilon}^{\epsilon} \left(\tilde{\kappa}\left(\Gamma_{\mathbf{x}}(s)\right) - \tilde{\kappa}_{exact}\left(\Gamma_{\mathbf{x}}(s)\right)\right)^2 ds}$$

where $\Gamma_{\mathbf{x}}(s)$ is a parameterization of the surface around $\mathbf{x}$, with $\Gamma_{\mathbf{x}}(0) = \mathbf{x}$, and with $\epsilon \to 0$.

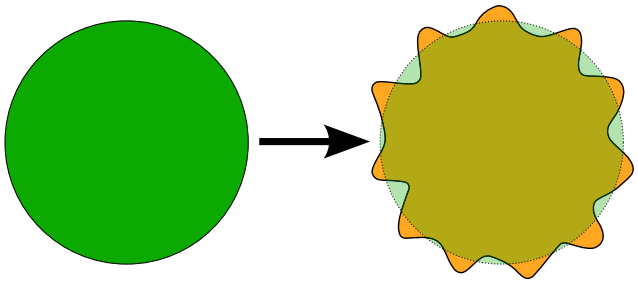
2.4.3. Minimized variation along the surface's normal

Following the description in §2.3 where the surface is implicitly represented in a Eulerian level set framework, a common strategy to solve PDEs on the phases or surfaces is to smoothly extend the physical properties in a region around the interface. Hence the third criterion, which can be geometrically understood as orthogonal to the previous one, consists in minimizing the errors in the calculation of the extension of the curvature, which will be treated in detail in §3.4.4. The precise extrapolation of quantities in the normal direction is necessary in other applications, for example for the mean curvature motion in a Eulerian framework (see [26] for a level set study). In a VOF framework, a smoothing or an extension of the curvature is computed in order to obtain a better estimation of κ at grid nodes, as presented in [18,27] for example.

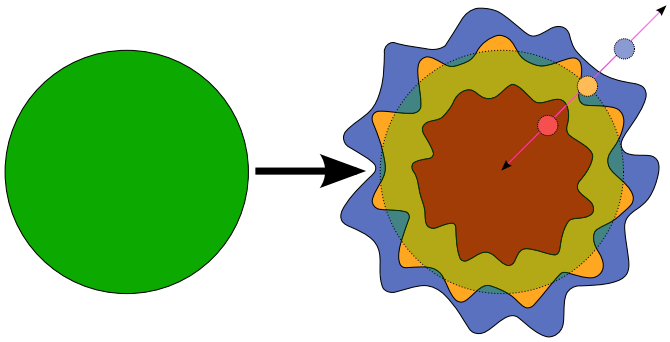
Errors in the normal direction will deviate the dynamics of the flow by introducing forces in the vicinity of the surface that will create undesired perturbations, as depicted in Fig. 2(c). Hence, it is necessary to compute an accurate curvature extension, i.e. a curvature that is constant in the normal direction, as pointed out by [17].



(a) Representation of the dynamic effect of curvature's absolute error. Here the curvature is overestimated hence the dynamic of the surface will be similar to the one of the smaller orange circle.



(b) Representation of the dynamic effect of curvature's variations along the surface. Even though the surface's position may be exact (the translucid green circle with dotted surface), the error on the curvature will mislead the dynamic as if the geometry was erroneous (the virtual orange perturbed surface).



(c) Representation of the dynamic effect of curvature's variations along the surface's normal. Even though the surface's position may be exact (the translucid green circle with dotted surface), the error on the extended curvature in the normal direction will mislead the dynamics as if the geometry was erroneous. The red, orange and blue perturbed forms correspond respectively to curvature computed on three different level sets, for example $\phi = \{-h, 0, +h\}$. The normal direction is noted with a purple segment on which three circles have been placed to represent spatial points at a distance of $\{-h, 0, +h\}$ from the exact surface.

Fig. 2. Visual representation of the effects of the different errors on curvature following Criteria 2.1, 2.2, 2.3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In the case of a level set representation of the surface, it can be easily seen that a simple computation of the curvature as $\kappa = \nabla \cdot \mathbf{n}$ will lead to a high deviation in the normal direction, as depicted in Fig. 3(a) representing the level sets of a circle. Moreover, we can see in this example that the curvature will be higher inside the circle and lower outside, compared to the curvature on the surface. This fact is true on any smooth geometry as the spatial discretization tends to zero.

In the presence of surface tension, this last phenomenon will dynamically stretch the level sets on one side of the surface while compacting it on the other side. This can lead to more numerical errors as the flow moves the level sets, particularly when using a signed distance function that will be highly distorted, requiring frequent reinitializations of the level set function.

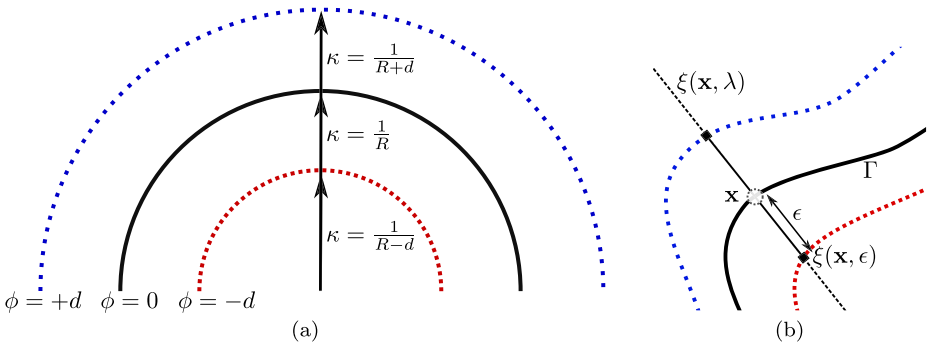


Fig. 3. (a) Local evaluation of the curvature based on a signed distance level set function. The curvature varies as $1/(R-\phi)$ on the ϕ's level set, R being the exact radius of the osculatory circle of the surface at $\phi = 0$. (b) Graphical representation of the line $\xi(\mathbf{x}, \lambda) = \mathbf{x} + \lambda \mathbf{n}(\mathbf{x})$ passing through $\mathbf{x}$ and following the normal $\mathbf{n}(\mathbf{x})$ of the surface Γ.

Given that the surface tension forces are spread in the vicinity of the surface, if the variation of the curvature along the normal is null then the surrounding level sets will move at the same speed hence reducing the numerical errors. In the case of the column equilibrium problem, this will reduce the spurious currents.

As a consequence, in order to reduce parasitic currents, one has to minimize the deviation $\kappa_{\sigma,\mathbf{n}}$ of the curvature along the normal. It can be expressed similarly to eq. (7) at a point $\mathbf{x}$ on the surface Γ, given any curvature κ that is extended in the whole domain:

$$\tilde{\kappa}_{\sigma,\mathbf{n}}(\mathbf{x}) = \sqrt{\int_{-\epsilon}^{\epsilon} \left(\tilde{\kappa}\left(\xi(\mathbf{x}, \lambda)\right) - \tilde{\kappa}(\mathbf{x})\right)^2 d\xi}$$

where $\xi(\mathbf{x}, \lambda) = \mathbf{x} + \lambda \mathbf{n}(\mathbf{x})$ is the line passing through $\mathbf{x}$ and following the normal $\mathbf{n}(\mathbf{x})$, as illustrated in Fig. 3(b). This measurement is meaningful in the vicinity of Γ and therefore on the segment $\{\xi(\mathbf{x}, \lambda),\ \lambda \in [-\epsilon, \epsilon]\}$, with $\epsilon \to 0$. In a Eulerian framework, it is evaluated for discrete points $\mathbf{x}$ close to the surface.

2.4.4. Conclusion

In the next section, we present a novel method based on:

1. the balanced force method as developed in [11] and detailed for a level set framework in §3.3,
2. an adaptation of the Closest Point method for curvature extension as first used in [12], that we improved as detailed in §3.4.

Our approach is focused on the accurate and efficient numerical computation of the curvature in the vicinity of the surface complying with fast convergence to the exact curvature (Criterion 2.1) and minimized deviation along the surface (Criterion 2.2) and along the normal direction (Criterion 2.3).

3. Numerical methods

In this section we first describe the general Navier–Stokes flow solver we used, then the level set transport equation algorithm. In the third subsection we present our implementation of the balanced-force surface tension method in a level set framework. We finally present the method that we developed for the accurate computation and extension of the surface curvature. The validation of the method with associated numerical results is presented in section 4 on pure geometric and dynamic cases.

3.1. Flow solver and discretization

For our numerical simulation, we used the Thetis flow solver [28,29]. Time discretization of the momentum equation is a first-order Eulerian scheme with an implicit formulation for the viscous term. The velocity/pressure coupling under the incompressible flow constraint is solved with the time splitting pressure correction method [30]. The equations are discretized on a staggered grid by means of the finite volume method. The space derivatives of the inertial and stress terms are discretized by a second-order centered scheme. Since the phase function is not defined on each grid point where viscosities and densities are needed for the Navier–Stokes discretization, the physical characteristics are interpolated on the staggered grid. We use a linear interpolation to calculate the density on the velocity nodes, whereas a harmonic interpolation is used for the viscosity. Linear systems of the prediction and correction steps are solved thanks to the direct MUMPS solver [31,32].

3.2. Level set transport

In order to obtain an accurate computation of the curvature, as stated in section 2.4.1, the level set function is advanced in time using a 5th order in space WENO method as first described in [33]. A second-order in time Runge Kutta scheme is used based on the advection of ϕ for half a time step:

$$\phi^{n+\frac{1}{2}} = \phi^n - \Delta t\,(\mathbf{u}^n \cdot \nabla)\phi^n$$
$$\phi^{n+1} = \phi^n - \Delta t\,(\mathbf{u}^n \cdot \nabla)\phi^{n+\frac{1}{2}},$$

with $\phi^n(\mathbf{x}) = \phi(\mathbf{x}, t_n)$ and $\phi^{n+1}(\mathbf{x}) = \phi(\mathbf{x}, t_n + \Delta t)$.

This method limits the spatial errors for ϕ around $O(h^5)$ which, with a precise curvature computation, theoretically yields a third order error in the curvature computation. We will see and explain in section 4 that we reach errors at even higher order in the presence of small velocities.

3.3. Balanced force surface tension

In order to satisfy the consistent numerical discretization scheme criterion (cf. §2.4, Criterion 1), the surface tension term is discretized on the MAC grid at the velocity nodes, i.e. on the faces of the control volume, with the same finite difference scheme used for the Poisson pressure equation. As in typical balanced force algorithms, we use the CSF approximation for the surface tension term rewritten relative to the phase concentration c (we dropped the index for clarity, posing $c = c_1$):

$$\sigma\kappa\delta_\Gamma\mathbf{n} \equiv \sigma\kappa\nabla c,$$

where ∇c is numerically computed at the faces of the cells, and points from phase 2 to phase 1.

As a consequence, recalling that the concentration c changes its value from 0 to 1 on either side of the interface, ∇c is located on the faces of the cells surrounding Γ depending on the stencil used to compute ∇c and on the smoothness of c.

Remark. Using a first order accurate function for the concentration, i.e. c equals 0 (resp. 1), if ϕ is negative (resp. positive) will lead to an aliased function following the cells' faces. Consequently the pressure profile will directly follow this aliasing as well as the surface. As a cell's level set value can vary with transport from a certain small value $\pm\epsilon$ around zero, the surface tension force (and thus the pressure jump) will undesirably oscillate from one cell to another. Moreover, derivating this function to compute ∇c for the surface tension force will lead to $O(1)$ errors.

Hence it is necessary to use sufficiently accurate representations of the physical quantities in order to avoid the aliasing problems.

A common choice for calculating the concentration is to use a regularized Heaviside function such as:

$$c(\mathbf{x}) = H_\epsilon(\phi(\mathbf{x}))$$

where $\epsilon = O(h)$. This smooth approximation can induce a smoother dynamic behavior. We did not find this to be a problem in the cases we studied in section 4. Common strategies for computing the Heaviside function in a level set framework do not show strict spatial convergence in 2D and 3D when $\epsilon = mh$, as proven in [34] for the associated δ_ϵ function. Moreover, as proven in [35] (and as studied in their references), in presence of high density ratios, the use of a smooth Heaviside function to define the physical quantities is crucial to increase numerical stability. From a physical point of view, the interface is thus perceived as a volume of fixed width assuring a continuous transition between the two phases. For the sake of simplicity, we used the same smoothed Heaviside function to define ρ, μ and c.

Hence, we did not find necessary to implement a more precise scheme, as for example proposed in [36]. The method proposed in this article is totally independent of this choice, however, a precise numerical approximation of the Heaviside function will benefit the overall numerical results.

In order to guarantee a precise enough Heaviside function for sufficiently fine spatial discretizations (as used in our numerical results of section 4), we chose a value of $\epsilon = 2h$ and:

$$H_\epsilon(y) = \begin{cases} 0 & \text{if } y \le -\epsilon \\ 1 & \text{if } y \ge \epsilon \\ \dfrac{1 + y/\epsilon + \sin\left(\frac{y\pi}{\epsilon}\right)/\pi}{2} & \text{otherwise} \end{cases} \tag{9}$$

with its associated smooth Dirac mass

$$\delta_\epsilon(y) = \begin{cases} 0 & \text{if } |y| \le \epsilon \\ \dfrac{1 + \cos\left(\frac{y\pi}{\epsilon}\right)}{2\epsilon} & \text{otherwise.} \end{cases} \tag{10}$$

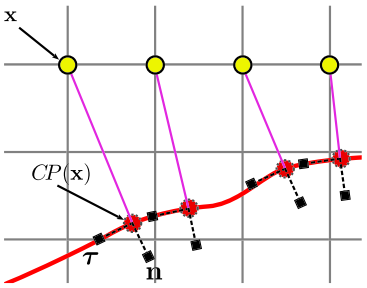


Fig. 4. Closest point method: a geometrical representation. Yellow dots are grid points with their corresponding closest point on the surface as red dots. Normals and tangents on the surface are drawn as dotted square ended segments. The purple lines show the colinearity between $\overrightarrow{\mathbf{x}CP(\mathbf{x})}$ and $\mathbf{n}(CP(\mathbf{x}))$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Moreover, so as to be sure to reduce the effect of interface spreading or shrinking when the level set is not a distance function, we used a renormalized representation of the level set as proposed in [37]:

$$c(\mathbf{x}) = H_\epsilon(\phi(\mathbf{x})/|\nabla\phi(\mathbf{x})|).$$

For the discretization of the ∇c, we used the same second-order scheme as for the pressure gradient: $\partial c_{i+\frac{1}{2},j}/\partial x = (c_{i+1,j} - c_{i,j})/\Delta x$. Hence, with $\epsilon = 2h$, the surface tension force is null everywhere but on the faces in the $3 \times 2 = 6$ cells band surrounding Γ.

Consequently, κ has to be computed accurately only at the center of cells around which ∇c is not null. In our flow solver, the curvature has to be known at the center of the faces of the staggered grid. We discuss this point in §3.4.5.

3.4. Curvature extension in a level set framework

In order to satisfy the rational computation of the surface's curvature criterion (cf. §2.4), one must use numerical methods that prove convergence in space discretization (Criterion 2.1) and minimal variation on the surface (Criterion 2.2) and along its normal (Criterion 2.3).

We present here a novel method to attain this goal in 2 steps:

1. The accurate initial estimation of the curvature,
2. The precise extension of the curvature in the vicinity of the interface.

First, we describe the general principle of the Closest Point method on which our approach is based. Then, we introduce error criteria that will help to analyze the numerical behavior of the method. Finally, we detail the novelties of our method which calculates an accurate curvature extension in a level set framework.

3.4.1. The Closest Point method

The Closest Point (CP) method, first introduced in [38], has been successfully used in [22] to solve PDEs defined on a surface by extending the physical quantities in the vicinity of the surface and then solving the equations in a Eulerian domain. The CSF principle from [10] is based on a parallel point of view for integrating the singular force residing on the surface.

Herrmann [12] first used the CP principle to extend the curvature in a level set framework. It consists in defining the curvature at a point $\mathbf{x}$ as the curvature of its closest point on the interface:

$$\kappa(\mathbf{x}) = \kappa(CP(\mathbf{x})) \tag{11}$$

In a Eulerian framework, it requires the use of interpolation functions as $CP(\mathbf{x})$ will generally be localized between grid points. That principle can be applied to any physical quantity defined on the surface for which an extension is needed. Our approach follows Herrmann's work by improving the CP research algorithm and enhancing the curvature's extension smoothness.

$CP(\mathbf{x})$ is the closest point from $\mathbf{x}$ to the surface Γ and is defined as:

$$CP(\mathbf{x}) = \min_{\mathbf{y}\in\Gamma}(\|\mathbf{x} - \mathbf{y}\|). \tag{12}$$

By construction the vector $\overrightarrow{\mathbf{x}CP(\mathbf{x})}$ is orthogonal to the local tangent at $CP(\mathbf{x})$ and hence is collinear with the normal of the surface at $CP(\mathbf{x})$:

$$\overrightarrow{\mathbf{x}CP(\mathbf{x})} \cdot \boldsymbol{\tau}(CP(\mathbf{x})) = 0 \tag{13}$$

where $\boldsymbol{\tau}$ is the unit tangent orthogonal to $\mathbf{n}$ as shown in Fig. 4.

3.4.2. Error measures

In order to analyze the accuracy of the numerical methods used to compute the curvature, we define four error measurements.

Exact curvature. In order to bound the errors of the numerically computed curvature, we need to define an exact curvature as reference. The curvature is only defined on the surface, in a level set framework, as we are searching for an extended representation of the curvature of a surface. Following equation (11), we define the exact curvature at a point $\mathbf{x}$ as:

$$\kappa_{ex}(\mathbf{x}) = \kappa_{ex}(CP(\mathbf{x})). \tag{14}$$

In the case of a circle of radius R, the exact curvature is constant over the whole domain: $\kappa_{ex}(\mathbf{x}) = 1/R$.

To understand how the closest point method extends physical quantities in the whole domain, we recall from §2.4.3 the definition of the line $\xi(\mathbf{y}, \lambda)$ that passes by the point $\mathbf{y}$ on the surface and follows the normal at $\mathbf{y}$:

$$\xi(\mathbf{y}, \lambda) = \mathbf{y} + \lambda \mathbf{n}(\mathbf{y}) \tag{15}$$

with $\lambda \in \mathbb{R}$. Based on equation (14), we know that $\mathbf{x}$ stands on the line $\xi(CP(\mathbf{x}), \lambda)$.

We can then define the dual extended curvature lines for every point $\mathbf{y}$ of the surface, along their associated normal by:

$$\kappa_{ex,\mathbf{n}}(\xi(\mathbf{y}, \lambda)) = \kappa_{ex}(\mathbf{y}). \tag{16}$$

The collection of segments of extended curvature lines $\kappa_{ex,\mathbf{n}}$ for which $|\lambda|$ is minimal is the extended curvature field κ_{ex} as defined by (14).

It is important to note that ambiguities arise when two or more lines $\xi(\mathbf{y}, \lambda)$ intersect for a same parameter length $|\lambda|$, i.e. when a point $\mathbf{x}$ has two or more closest points on the surface. These cases arise for points relatively far from the surface or when the surface curvature is large. In a numerical framework, it is statistically highly unlikely to encounter the case of two equidistant closest points. However, the closest point to $\mathbf{x}$ will vary from one point on the surface to another when the surface is transported, phenomenon that could lead to temporal jumps in the curvature field. This problem is not assessed in this paper, however we did not encounter any numerical instabilities as the extended curvature field is computed close enough to the surface and the curvature is not too high, i.e. $\kappa \ll h^{-1}$.

In practice, we only require κ_{ex} in the vicinity of a few cells (here N cells) around the interface as required by the CSF approximation (cf. §3.3). For that purpose we introduce the function $\overline{\delta_\Gamma^N}$ defined by:

$$\overline{\delta_\Gamma^N}(\mathbf{x}_{ij}) = \begin{cases} 1 & \text{if } \overline{\delta_\Gamma^{N-1}}(\mathbf{x}_{i'j'}) = 1 \text{ or is next} \\ & \quad \text{to a cell } i'j' \text{ such that } \overline{\delta_\Gamma^{N-1}}(\mathbf{x}_{i'j'}) = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

This band around the interface is constructed iteratively based on the cells containing the interface $\overline{\delta_\Gamma^0}$:

$$\overline{\delta_\Gamma^0}(\mathbf{x}_{ij}) = \begin{cases} 1 & \text{if } 0 < c(\mathbf{x}_{ij}) < 1, \\ 0 & \text{otherwise.} \end{cases}$$

In the next sections, for the sake of clarity, we drop the exponent N in $\overline{\delta_\Gamma^N}$.

L_2 and L_∞ norm errors. We define the L_2 and L_∞ normalized errors for the curvature on the overall domain as:

$$L_2 = \frac{1}{\kappa_{ex}} \sqrt{\frac{\sum (\tilde{\kappa} - \kappa_{ex})^2 \overline{\delta_\Gamma}}{\sum \overline{\delta_\Gamma}}}, \tag{18}$$

$$L_\infty = \frac{1}{\kappa_{ex}} \max |(\tilde{\kappa} - \kappa_{ex}) \overline{\delta_\Gamma}|. \tag{19}$$

Standard deviation on the surface. The (normalized) standard deviation on the surface is only significant for a circle/column where the exact curvature is constant along the surface, hence it can only be studied in that case:

$$\tilde{\kappa}_\sigma = \frac{1}{\tilde{\kappa}_{mean}} \sqrt{\sum (\tilde{\kappa} - \tilde{\kappa}_{mean})^2 \overline{\delta_\Gamma}} \tag{20}$$

where

$$\tilde{\kappa}_{mean} = \frac{\sum \tilde{\kappa} \overline{\delta_\Gamma}}{\sum \overline{\delta_\Gamma}}. \tag{21}$$

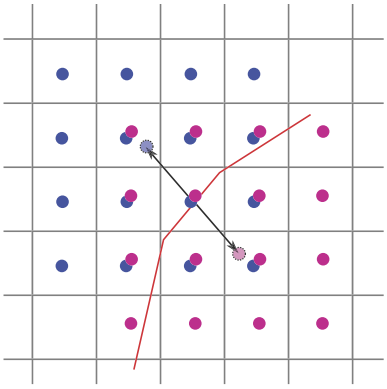


Fig. 5. Stencil used for computing the standard deviation along the normal with fourth-order Lagrange interpolation functions. The surface is shown in red, blue (resp. purple) dots are used for $k = 1$ (resp. $k = -1$) in eq. (23) and are spatially shifted for clarity. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Standard deviation along the normal. The (normalized) standard deviation along the line $\xi(\mathbf{y}, \lambda)$ defined at point $\mathbf{y}$ is computed as:

$$\kappa_{\sigma,\mathbf{n}}(\mathbf{y}) = \frac{1}{\kappa(\mathbf{y})}\sqrt{\int (\kappa(\xi(\mathbf{y},\lambda)) - \kappa(\mathbf{y}))^2\, d\lambda} \tag{22}$$

which is discretized in the M cells neighborhood of the cell $\mathbf{y}_{i,j}$ as:

$$\tilde{\kappa}_{\sigma,\mathbf{n}}(\mathbf{y}_{i,j}) = \frac{1}{\tilde{\kappa}(\mathbf{y}_{i,j})}\sqrt{\frac{1}{2M+1}\sum_{k=-M}^{+M} (\tilde{\kappa}(\xi(\mathbf{y}_{i,j}, kh)) - \tilde{\kappa}(\mathbf{y}_{i,j}))^2}. \tag{23}$$

In practice, as we are looking at the variation of κ in a small area around the interface, we fix $M = 1$. It should be noted that fourth-order Lagrange interpolation functions are used to compute $\tilde{\kappa}(\xi(\mathbf{y}_{i,j}, kh))$ because $\xi(\mathbf{y}_{i,j}, kh)$ will generally not coincide with the mesh points. The stencil used to compute $\tilde{\kappa}_{\sigma,\mathbf{n}}$ around one cell is presented in Fig. 5.

The overall L_2 and L_∞ errors for standard deviation along the normal are defined as:

$$L_{2,\tilde{\kappa}_{\sigma,\mathbf{n}}} = \sqrt{\frac{\sum \overline{\delta_\Gamma}\tilde{\kappa}^2_{\sigma,\mathbf{n}}}{\sum \overline{\delta_\Gamma}}}, \tag{24}$$

$$L_{\infty,\tilde{\kappa}_{\sigma,\mathbf{n}}} = \max \overline{\delta_\Gamma}\tilde{\kappa}_{\sigma,\mathbf{n}}. \tag{25}$$

3.4.3. Accurate estimation of the curvature

Before applying the Closest Point algorithm, we first need to get a precise initial estimation of the curvature at the surface points. Recalling eq. (6), the curvature can be computed in the Eulerian level set framework on the overall domain by numerically computing:

$$\kappa_{LS} = \nabla \cdot \mathbf{n} = \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right) \tag{26}$$

where we use the index *LS* to further differentiate this scalar field with the other methods. In order to reduce the numerical errors due to the computation of higher-order derivatives, we use a formula for the curvature where the second derivative errors are limited to only a part of the calculation:

$$\kappa_{LS} = \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right) = \frac{1}{|\nabla\phi|}\left(\Delta\phi - \left(\left[D^2\phi\right]\frac{\nabla\phi}{|\nabla\phi|}\right)\cdot\frac{\nabla\phi}{|\nabla\phi|}\right) \tag{27}$$

where $D^2\phi$ is the Hessian matrix of ϕ: $\left[D^2\phi\right]_{i,j} = \frac{\partial^2\phi}{\partial x_i \partial x_j}$.

Understanding the curvature field in a level set framework.

Fact 2. *For any smooth level set function ϕ, as $h \to 0$, the curvature computed by equation (26) on the surface, i.e. $\mathbf{x} \in \Gamma$, is the exact curvature of the surface at $\mathbf{x}$.*

We propose to extend this property to all the surfaces captured by ϕ, i.e. all iso-values:

Claim 3. *For any function ϕ, as $h \to 0$, the curvature computed by equation (26) at a point $\mathbf{x}$ is the exact curvature at $\mathbf{x}$ of the surface defined by the level set $\phi(\mathbf{x}) = \alpha$.*

Those two properties will be used below to express a first approximation of the extension of the curvature in the whole domain. To illustrate this point, under the assumption $\Gamma = \{\mathbf{x} \,|\, \phi(\mathbf{x}) = 0\}$, we describe three cases:

1. If ϕ is a signed distance function, i.e. $\phi(\mathbf{x}) = sd(\mathbf{x})$, then $\kappa_{LS}(\phi, \mathbf{x})$ is the curvature of the surface represented by the level set $\Gamma_{\phi(\mathbf{x})} = \{\mathbf{y} \,|\, \phi(\mathbf{y}) = \phi(\mathbf{x})\}$. The curvature in the whole domain can thus be seen as a stretched/shrunk extension of the curvature at the surface, growing as a function of $sd(\mathbf{x})$, as depicted in Fig. 3.
2. If ϕ can be defined as the composition of another level set ψ with a scalar function f, i.e. $\phi = f(\psi)$, then we can express the curvature by the relationship:
$$\kappa_{LS}(\phi, \cdot) = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) = \nabla \cdot \left(\frac{\nabla f(\psi)}{|\nabla f(\psi)|} \right) = sign\left(f'(\psi)\right) \left(\frac{\nabla \psi}{|\nabla \psi|} \right) = sign\left(f'(\psi)\right) \kappa_{LS}(\psi, \cdot)$$
which gives $\kappa_{LS}(\phi, \cdot) = \kappa_{LS}(\psi, \cdot)$ when f' is strictly positive. Hence any strictly increasing function f applied to a signed distance brings us back to the first case.
3. Otherwise, outside of Γ where ϕ cannot be interpreted as a function of a signed distance function, κ_{LS} may not be directly compared to the surface's curvature. However, if ϕ is sufficiently smooth, a re-normalized form (see [37]) of the level set function $\phi/|\nabla\phi|$ gives a first order approximation of a signed distance function near the interface which brings us back to the first case.

3.4.4. Extension of the surface's curvature

Here we present two different methods for extending the surface curvature and discuss the advantages and drawbacks of each method. The first one is based on Claim 3 and is presented to give a better geometrical understanding of the curvature in a level set framework. The second is the method we propose based on the Closest Point method presented in §3.4.1 and following Herrmann's works [12].

Osculatory circle approximation. This approximation is based on the fact that, under the assumption that locally, as the discretization tends to zero, the curvature of a surface can be defined as the inverse of the radius of the osculatory circle tangent to Γ at $\mathbf{y}$:

$$\kappa(\mathbf{y}) = 1/R(\mathbf{y}).$$

This assertion is also true in a level set framework as shown in Fig. 3(a). Let $d(\mathbf{x})$ be the signed physical distance from the point $\mathbf{x}$ to the surface at level set $\phi_0 = 0$. As $\mathbf{x}$ stands on the line $\mathbf{x} = \xi(\mathbf{y}, \lambda)$ (where ξ is defined by eq. (15)), or conversely $\mathbf{y} = CP(\mathbf{x}) \in \Gamma$, we can write $d(\mathbf{x}) = sign(\lambda).\|\mathbf{x} - \mathbf{y}\|$. Then the radius of the osculatory circle at $\mathbf{x}$ is $R(\mathbf{x}) = R(\mathbf{y}) + d(\mathbf{x}) = \kappa^{-1}(\mathbf{y}) + d(\mathbf{x}) = \kappa^{-1}(\mathbf{x})$.

Consequently we can define an extended surface curvature as:

$$\kappa_{osc}(\mathbf{x}) = \frac{1}{R(\mathbf{y})} = \frac{1}{\kappa^{-1}(\mathbf{x}) - d(\mathbf{x})}$$

which does not require any knowledge of $\mathbf{y}$. In a general level set framework the equation is:

$$\kappa_{osc}(\mathbf{x}) = \frac{1}{\kappa_{LS}^{-1}(\mathbf{x}) - d_\phi(\mathbf{x})} \tag{28}$$

with d_ϕ being the physical signed distance function computed from ϕ which, near the surface, as the discretization step tends to zero, can be approximated by $d_\phi(\mathbf{x}) = \frac{\phi(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}$. In the particular case of a signed distance function $d_\phi(\mathbf{x}) = \phi(\mathbf{x})$.

Limits The limits of this approximation are when:

- the local curvature is null, $\kappa(\mathbf{x}) = 0$, then the osculatory circle has an infinite radius which can raise numerical problems. A threshold is thus used and we set $\kappa_{osc}(\mathbf{x}) = 0$ in those cases,
- the radius $R(\mathbf{x}) = R(\mathbf{y}) + d(\mathbf{x}) = 0$, i.e. $\mathbf{x}$ is the singular center of the osculatory circle, where we do not extend the curvature. As we only look in the vicinity of the surface, this case will only arise for very high curvatures (resp. small radii), $\kappa = O(1/h)$ for which we propose a threshold for the maximum curvature value $\kappa_{max} = 1/h$, which is coherent with the physical maximum curvature that can be captured at the scale h.

This osculatory circle extension is of low order as it does not take into account the variation of a curvature which is essential in the general case. We present in 4.2 spatial convergence results that show this approximation to be fourth-order in the circular case but only first order in the ellipsoidal case.

Closest point on surface interpolation. As introduced in §3.4.1, another approach to extend the curvature away from the surface consists in finding the closest point to $\mathbf{x}$ on Γ and interpolating the curvature there. We present a detailed version of the Closest Point search algorithm that will serve as a base for the novel method proposed hereafter.

Without an analytical representation of the surface, finding $\mathbf{y} = CP(\mathbf{x})$ is not as straightforward as finding on which line $\mathbf{x} = \xi(\mathbf{y}, \lambda)$ stands. However, we can get an approximation of the closest point to $\mathbf{x}$ by following the back trajectory $\mathbf{y} = \xi^{-1}(\mathbf{x}, \lambda)$ starting at $\mathbf{x}$ and leading to $\mathbf{y}$.

In a level set framework, $\xi^{-1}(\mathbf{x}, \lambda)$ follows the gradient field of ϕ and, in general, is a curve and not a line. Thus, one needs to descend along that curve to find $\hat{\mathbf{y}} = \widetilde{CP}(\mathbf{x})$, a numerical approximation of the exact solution $\mathbf{y} = CP(\mathbf{x})$.

We introduce Algorithm 1, noted $CP_{\odot}$, to compute an approximate closest point. $CP_{\odot}$ is based on a first-order Newtonian method with a convergence criterion based on a threshold distance $\epsilon \ll h$ and a with virtual time step Δ_λ. We have used here the local approximation of the distance from the point $\hat{\mathbf{y}}^n$ to the surface: $d_\phi(\hat{\mathbf{y}}^n) = \frac{\phi(\hat{\mathbf{y}}^n)}{|\nabla\phi(\hat{\mathbf{y}}^n)|}$ and of the normal: $\mathbf{n}(\hat{\mathbf{y}}^n) = \frac{\nabla\phi(\hat{\mathbf{y}}^n)}{|\nabla\phi(\hat{\mathbf{y}}^n)|}$. Interpolation is used in order to compute those values from the discrete grid points. Fig. 6 shows a schematic view of the algorithm.

Algorithm 1 $CP_{\odot}$: first order Newton method for the closest point search in a level set framework.

1. Initialize $\hat{\mathbf{y}}^0 = \mathbf{x}$, $i = 0$
2. While $|\phi(\hat{\mathbf{y}}^i)| > \epsilon$ do
   (a) $\hat{\mathbf{y}}^{i+1} = \hat{\mathbf{y}}^i - \Delta_\lambda d(\hat{\mathbf{y}}^i)\mathbf{n}(\hat{\mathbf{y}}^i)$
   (b) $i \leftarrow i + 1$
3. $\widetilde{CP}_{\odot}(\mathbf{x}) = \hat{\mathbf{y}}^i$

In practice, the threshold distance ϵ is set to h^4 as we expect fourth-order convergence on the CP algorithms. The virtual time step is set to $\Delta_\lambda = 0.9$ in order to stay on the same "side" of the level set as much as possible, i.e. not crossing the zero level set in the presence of numerical errors. Setting a lower value for Δ_λ will help to converge more precisely for points far from the surface. However as we are mostly interested by points in the proximity of Γ, we have found this value to be sufficient.

Once the closest point $\widetilde{CP}(\mathbf{x})$ has been computed we interpolate the curvature at that position leading to the following formula for the extended curvature at the grid point $\mathbf{x}_{i,j}$:

$$\tilde{\kappa}_{CP_\cdot}(\mathbf{x}_{i,j}) = Interpolate(\widetilde{\kappa_{LS}}, \widetilde{CP}_\cdot(\mathbf{x}_{i,j})), \tag{29}$$

where we have used the symbol $\cdot$ replacing $\odot$ because any closest point algorithm can be used.

This method drastically reduces the error of curvature along the normal, depending on the κ_{LS} approximation and the interpolation errors, as shown in section 4.2.

Colinearity criterion improvement Recalling equation (13), for any point $\mathbf{x}$, its closest point $CP(\mathbf{x})$ on the surface Γ is collinear to the normal $\mathbf{n}_\Gamma(CP(\mathbf{x}))$ of that surface, or equivalently orthogonal to the tangent:

$$\overrightarrow{\mathbf{x}\,CP(\mathbf{x})} \cdot \boldsymbol{\tau}_\Gamma(CP(\mathbf{x})) = 0$$

Here we used the notations $\mathbf{n}_\Gamma$ and $\boldsymbol{\tau}_\Gamma$ to distinguish between the normal and tangent defined on the surface Γ and in the whole Eulerian domain. In a level set framework, we can compute the cosine angle between a vector $\overrightarrow{\mathbf{xy}}$ and the tangent at $\mathbf{y}$ with:

$$\omega(\mathbf{x}, \mathbf{y}) = \frac{\overrightarrow{\mathbf{xy}}}{|\overrightarrow{\mathbf{xy}}|} \cdot \boldsymbol{\tau}_\Gamma(\mathbf{y}).$$

Algorithm 1 does not guarantee the colinearity property. We propose Algorithm 2, noted $CP_\perp$, that converges to colinearity while still remaining on the surface. This algorithm can be understood as finding a candidate closest point to $\mathbf{x}$ on the interface and then correcting it toward the direction of colinearity. As this new corrected candidate may not be on the interface, we need to project it onto the interface. We apply this procedure iteratively until both the distance to the interface and the angle are smaller than a given threshold. A schematic view is proposed in Fig. 6.

Algorithm 2 $CP_\perp$: first order Newton method for the closest point search in a level set framework with colinearity property.

1. Initialize $i = 0$,
2. Find a first approximation of the closest point $\hat{\mathbf{y}}^0 = \widetilde{CP_{\odot}}(\mathbf{x})$
3. While $|\omega(\mathbf{x}, \hat{\mathbf{y}}^i)| > \epsilon$ do
   (a) $\hat{\mathbf{y}}^{i+1} = \widetilde{CP_{\odot}}(\hat{\mathbf{y}}^i + \omega(\mathbf{x}, \hat{\mathbf{y}}^i)\,\boldsymbol{\tau}(\hat{\mathbf{y}}^i))$
   (b) $i \leftarrow i + 1$
4. $\widetilde{CP}_\perp(\mathbf{x}) = \hat{\mathbf{y}}^i$

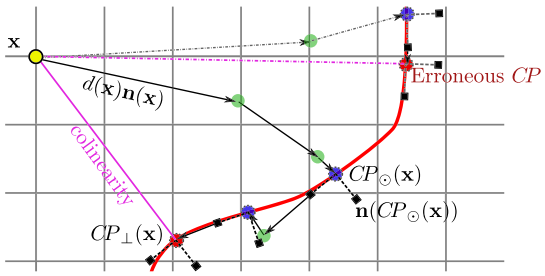


Fig. 6. Schematic view of the Closest Point algorithms. Errors on gradient vectors have been exaggerated and redundant notations omitted for the sake of clarity. Green dots stand for intermediate points, blue dots for $CP_{\odot}$ results and the red dot for the final $CP_{\perp}$ result. Normals on the surface are drawn as dotted square ended segments. The top gray dashed paths show an erroneous closest point due to poor approximations of the direction $\nabla\phi$ toward the surface. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Remarks on the convergence of the algorithm and its computational cost This algorithm works under the assumption that ϕ is a smooth enough function where the initial closest point computed with $CP_{\odot}$ will not bring us close to a local minimum ensuring colinearity while not being the closest point on the surface, as shown by the dashed paths in Fig. 6. This will be the case particularly when ϕ is not close to a distance function ($\nabla\phi$ might pointing the wrong direction) and/or for points far from the surface. More efficient and precise numerical methods can be used to calculate the closest point. However, as presented in the numerical results, the proposed algorithm is adequate for computing accurate surface tension driven simulations.

In comparison to the previous algorithm, this one will compute more iterations. Ensuring the colinearity property greatly improves the accuracy of the Closest Point search and the curvature interpolation. We refer the reader to section 4.2 for numerical results and efficiency comparison.

Reinterpolating The extended field obtained by interpolation at the Closest Point can suffer from important variations of the interpolated field in the few cells around the surface. In order to reduce this perturbation and further homogenize the extended field, we perform a second interpolation process on the first field's extension:

$$\begin{aligned}\tilde{\kappa}_{CP^2_{\perp}}(\mathbf{x}_{i,j}) &= Interpolate(\widetilde{\kappa_{CP_{\perp}}}, \widetilde{CP_{\perp}}(\mathbf{x}_{i,j})),\\ \tilde{\kappa}_{CP^2_{\perp}}(\mathbf{x}_{i,j}) &= Interpolate(Interpolate(\widetilde{\kappa_{LS}}, \widetilde{CP_{\perp}}(\mathbf{x}_{i,j})), \widetilde{CP_{\perp}}(\mathbf{x}_{i,j})),\end{aligned} \tag{30}$$

which is the algorithm that we used in our numerical simulations.

This process only costs one more interpolation per cell as we have already computed the closest point $\widetilde{CP_{\perp}}(\mathbf{x}_{i,j})$.

We report in sections 4.2 and 4.3 the numerical results obtained with this method for geometrical and dynamic cases.

Remarks and discussion

- In the remainder of the article, if not stated explicitly, when noting CP, we assume the reinterpolation algorithm has been used, dropping the exponent 2 and the index $\perp$ for simplicity.
- The errors appearing in the numerical closest point computation compared to the exact closest point are due to:
  1. The Newton's descent first order discretization. It has to be noted that this first order descent is optimal in the case of a signed distance function where $|\nabla\phi| = 1$ over the whole domain.
  2. The interpolation functions used.
- In the case where ϕ is a signed distance function, the $CP_{\odot}$ and $CP_{\perp}$ will theoretically lead to exactly the same point on the surface as $\nabla\phi$ points to the exact closest point. However, due to the discretization of the Newton's descent and numerical and interpolation errors, the Closest Point computed will not be the same and $CP_{\perp}$ will guarantee colinearity.
- The calculation of the Closest Point in the whole domain – or a band around the interface – gives a straightforward result for reinitialization of the level set. The order of precision of the signed distance function reconstructed in this way depends on the precision of the CP algorithm used. To attain 4th order precision on the curvature one would need a precision of at least order 6 in the signed distance function. However, as reinitialization is not the topic of the current article, we will not detail the subject further.

3.4.5. Curvature computation on the staggered grid

On a staggered grid, the level set values and the curvature are computed at the center of the cells while the velocities and forces are computed at the center of the faces. Hence, in order to compute the surface tension term $\sigma\kappa\nabla c$, it is necessary to interpolate κ at these positions.

Given the Closest Point algorithm, one can easily interpolate the curvature field at any position of the computation domain, for example for the face between $\mathbf{x}_{i,j}$ and $\mathbf{x}_{i+1,j}$ noted $\mathbf{x}^f_{i+\frac{1}{2},j}$:

$$\kappa^f_{i+\frac{1}{2},j} = \widetilde{\kappa_{CP.}}(\mathbf{x}_{i+\frac{1}{2},j}) = Interpolate\left(\widetilde{\kappa_{LS}}, \widetilde{CP.}(x_{i+\frac{1}{2},j})\right)$$

or use the reinterpolation scheme as described above. However, this operation is at the cost of 2 (resp. 3) times more closest point computation in 2D (resp. 3D) which will affect the overall computational time. Alternatively, one can use interpolation to compute the curvature from the center of cells to the faces with adequate precision:

$$\kappa^{f}_{i+\frac{1}{2},j} = \frac{\kappa_{CP.\,i,j} + \kappa_{CP.\,i+1,j}}{2} \tag{31}$$

for a second-order accurate interpolation and

$$\kappa^{f}_{i+\frac{1}{2},j} = \frac{-\kappa_{CP.\,i-1,j} + 9\kappa_{CP.\,i,j} + 9\kappa_{CP.\,i+1,j} - \kappa_{CP.\,i+2,j}}{16} \tag{32}$$

for fourth-order accuracy. The later formula has a wider stencil of 4 points where the curvature is needed. It thus requires a larger band around the interface inside which the closest points has to be computed, consequently increasing the overall computational cost.

In our numerical simulations, we have found that using equation (31) is a good balance between accuracy and efficiency.

4. Numerical results

4.1. Foreword

In all our test cases, the order of convergence for the L_2 and L_∞ norms are the same, thus we will only show and discuss the latter norm which is the more restrictive. For this study, we used smooth representation of the surface Γ and the characteristic phase function via δ_ϵ and H_ϵ from equations (9) and (10) with $\epsilon = 2h$ where h is the spatial discretization step. This regularization contains approximately 81% of the Dirac mass in the 3 cells around the interface, i.e. for $\phi \in [-h, +h]$. Thus, all the error measures are numerically computed around the interface in a 1 cell neighborhood, fixing $N = 1$ in eq. (17). The thresholds used for the CP algorithms are set to h^4 for the distance and the angle criterion in order to attain fourth-order errors, as shown in the results below.

4.2. Geometrical convergence analysis of the method

In the following paragraphs, we study the Eulerian discretization errors for a fixed geometry by varying the methods used: the level set curvature estimation, the osculatory circle approximation and the Closest Point extension. The numerical convergence analysis for the CP methods are detailed for the ellipse in §4.2.2 which is the most discriminating.

4.2.1. Circle case

A circle of radius $R = 0.2$ is centered at the origin of a unit square. The level set is initialized as the exact signed distance function:

$$\phi(\mathbf{x}) = |\mathbf{x}| - R.$$

Here the relevant parameter is the ratio R/h representing the number of grid cells for a radius.

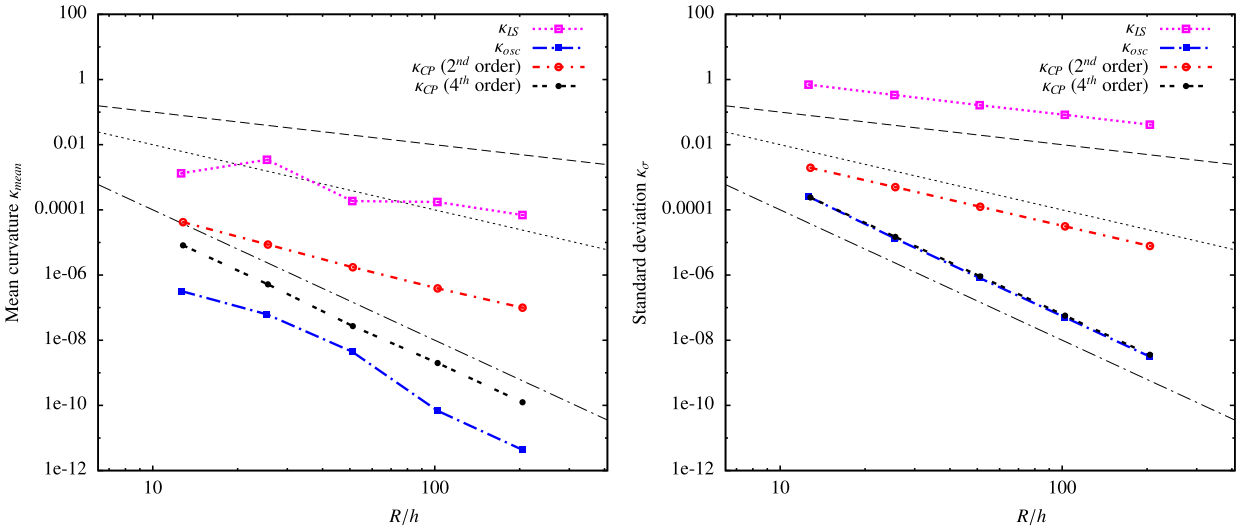
In this particular case the $CP_\odot$ and $CP_\perp$ methods give the same results so we will not distinguish them. This is due to the fact that for any point in the domain, by construction, the level set's gradient is always collinear to the normal at its closest point on the surface.

Order 2 and order 4 schemes. First, we show the importance of a high-order numerical computation of κ_{LS} from eq. (27). For that purpose, we have used classical central finite difference schemes of order 2 and 4 for gradients and Laplacian operators. Fig. 7 depicts the convergence of the different error measures relative to spatial discretization, taking $\kappa_{ex} = 1/R = 5$.

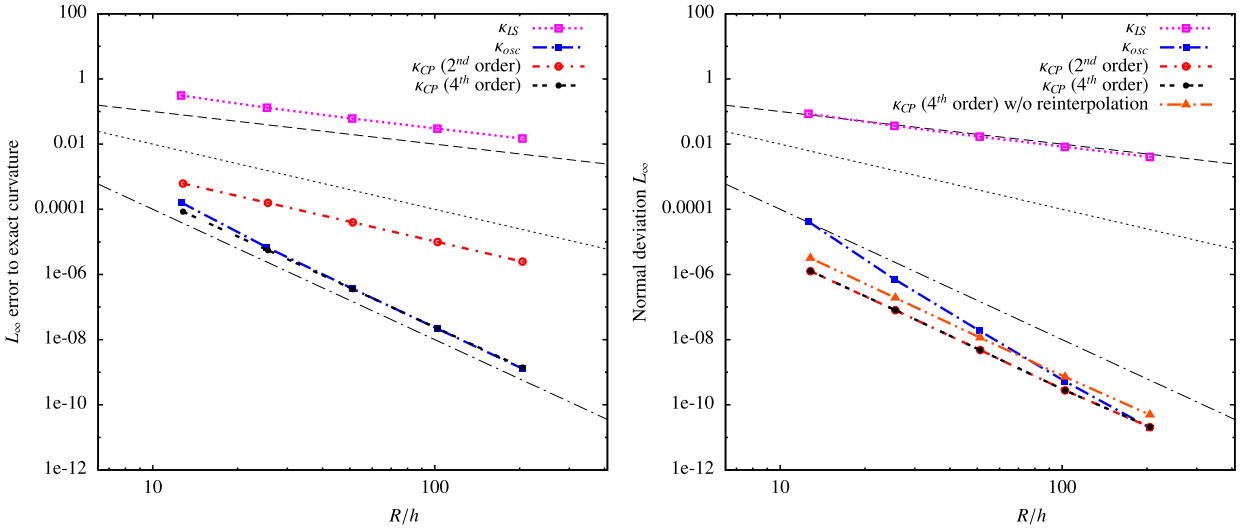
The results show a roughly first order convergence for the error measurements for κ_{LS}; in that case, the second and fourth-order schemes give qualitatively similar results. This is expected as the errors in the curvature come from the stretch/shrink effect as $\kappa_{osc} = \frac{1}{\kappa_{ex}^{-1} - d}$ away from the level set at $\phi = 0$, which dominates the error measures.

The CP method acts as expected, leading to the same order of convergence as the scheme used for the κ_{LS}. The normal deviation leads to fourth-order error convergence. This error is due to the interpolation functions used to compute the normal error deviation as discussed below. Concerning the CP reinterpolation method from eq. (30), the error measures are only different from the standard $CP_\perp$ method for the normal deviation: in that case we clearly see that reinterpolation of the curvature from a first curvature extension reduces the normal error by a factor of 3. As we will see in the ellipse test case below, this gain only holds for smooth enough curvatures without deteriorating the curvature field in the general case. However it reduces the spurious currents studied in §4.3.

The osculatory approximation based on a fourth-order scheme for κ_{LS} gives very good results: it converges at the same rate as the CP method for all the error measures except the normal deviation for which it converges more rapidly (around order 5). However, the absolute value of the normal deviation is approximately 100 times higher for $R/h \approx 12$. Despite this



(a) Mean curvature error and standard deviation.



(b) L_∞ error to exact curvature and L_∞ normal deviation.

Fig. 7. Circle geometrical test case: spatial convergence as a function of R/h for the 4th order κ_{LS}, the osculatory circle correction and the CP method with order 2 and order 4 κ_{LS}. κ_{LS} errors are only shown for $O(h^4)$ as the results with the $O(h^2)$ scheme are almost identical.

Table 2
Circle geometrical test case: approximate order of convergence of error measures for the different methods.

Method	L_∞	$\frac{\lvert\kappa_{mean}-\kappa_{ex}\rvert}{\kappa_{ex}}$	Std. dev. $\tilde{\kappa}_\sigma$	Normal $L_{\infty,\tilde{\kappa}_{\sigma,\mathbf{n}}}$
κ_{LS} 4th order	1	1	1	1
κ_{osc} (w/ κ_{LS} 4th order)	4	4	4	5
κ_{CP} (w/ κ_{LS} 2nd order)	2	2	2	4
κ_{CP} (w/ κ_{LS} 4th order)	4	4	4	4

good convergence rate, it is important to recall that the useful properties of this method rely on the first order approximation of the local curvature to the osculatory circle which will be deteriorated in the general case (i.e. contrary to this ideal circle case) as it will be shown in the next sections for the ellipse. The convergence rates are shown in Fig. 7 and summarized in Table 2.

The aliasing that can be observed in the mean curvature (Fig. 7(a)) is due to the fact that we use a sharp stencil $\overline{\delta_\Gamma}$ for the error measurements around the interface, affecting any measurement contrary to a smooth δ_Γ.

Even though using order 4 schemes for κ_{LS} yields much better results, it is important to note that it has to be avoided for fast varying ϕ, i.e. when the local curvature is of the order of h, where it would raise higher errors than a more compact

scheme because of high-frequency amplified errors. We chose a sufficiently large R/h ratio to avoid such perturbations. However, in the general case, one could think of mixing different schemes based on a first estimation of the curvature.

Errors in the curvature estimation in the presence of fictitious perturbations. In order to see how the method responds to numerical errors that will arise with the transport of the level set, we introduced small perturbations to the initial level set such as:

$$\hat{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + e(h^M) \tag{33}$$

where $\phi(\mathbf{x})$ is the signed distance function to the circle and $e(h^M)$ is a random function that takes its values in the interval $[-h^M, +h^M]$. One has to note that if $e(h^M) = h^M$ is constant then $\hat{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + h^M$ and the relative error for the level set function is

$$\frac{\hat{\phi}(\mathbf{x}) - \phi(\mathbf{x})}{\phi(\mathbf{x})} = \frac{h^M}{\phi(\mathbf{x})} = O(h^{M-1})$$

near the interface. Thus if we add a perturbation of order 3 then the expected perturbation on the curvature computation should be of the order $3 - 1 - 2 = 0$. However, in the following results, as $e(h^M) \in [-h^M, +h^M]$ is a random function, we expect a smaller deterioration on the order of convergence.

Here we used a fourth-order scheme for κ_{LS}. Fig. 8 shows that the introduction of a $O(h^3)$ (resp. $O(h^4)$) perturbation approximately reduces the order of convergence from 4 to 1 (resp. from 4 to 2), as expected.

4.2.2. Ellipse case

In order to test the robustness of the method in a more general case, we applied the same spatial convergence study to an ellipse. Particularly, we will show the effect of the collinear closest point method compared to the standard method. The surface is implicitly defined by the level set:

$$\phi(\mathbf{x}) = \sqrt{\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2} - R$$

with $a = 1.2$, $b = 0.8$ and $R = 0.2$, or in a parametric form:

$$\Gamma(\theta) = (aR\cos(\theta), bR\sin(\theta)).$$

It should be noted that the level set function ϕ is not a signed distance function which is not trivial to compute. The exact curvature extension that was constant for the circle case is here less trivial. Recall that the curvature extrapolation is:

$$\kappa_{ex}(\mathbf{x}) = \kappa_{ex}(CP_{ex}(\mathbf{x})) = \kappa_{\Gamma,ex}\left(\theta_{CP_{ex}}(\mathbf{x})\right),$$

where $\theta_{CP}(\mathbf{x})$ is the angular parameter for the closest point to $\mathbf{x}$ on the surface. We need to compute the exact closest point $CP_{ex}(\mathbf{x})$ on the ellipse which is obtained by solving the equation

$$\overrightarrow{\mathbf{x}\,CP_{ex}(\mathbf{x})} \cdot \Gamma(\theta) = 0$$

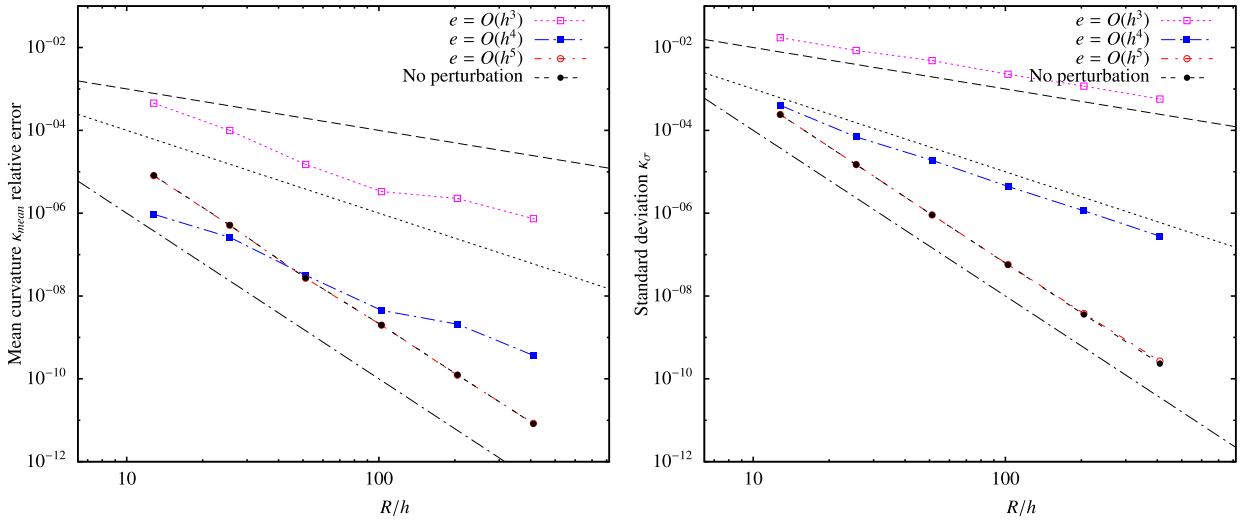
for θ, as the line from $\mathbf{x}$ to its closest point on Γ is collinear to the normal at that point. This non-linear equation has in the general case 2 solutions but only one in the nearest quadrant (known by the position of $\mathbf{x}$ compared to the ellipse's parameters). To solve it we used Newton's method with the search reduced to the nearest quadrant. The curvature at that point $\Gamma(\theta)$ is then computed with:

$$\kappa_{\Gamma,ex}(\theta) = \frac{ab}{\left(b^2\cos^2(\theta) + a^2\sin^2(\theta)\right)^{3/2}}.$$

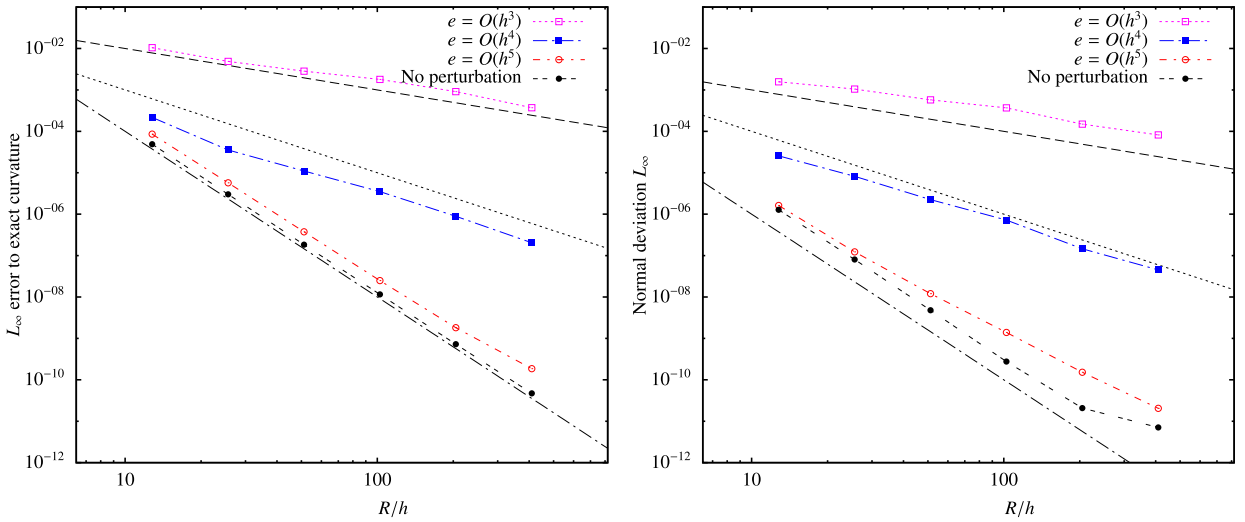
Note that this "exact" extended curvature is subject to discretization errors due to Newton's descent. In consequence the errors that we present below can be biased by this additional perturbation. However we expect this to be negligible.

Also, as stated previously in §2.4.2 comparing the mean curvature and the standard deviation is meaningless since the curvature varies along the surface, hence we removed those two error measures for this ellipse test case.

Closest point spatial convergence. In this paragraph, we study the spatial convergence of the two different CP algorithms. For this purpose, we look at the error in the distance map defined as the distance from $\mathbf{x}$ to its computed closest point on the surface $CP(\mathbf{x})$, and the distance between $CP(\mathbf{x})$ and the exact closest point $CP_{ex}(\mathbf{x})$. Those two errors have different meanings. While the error in the distance to the surface is meaningful, as we aim to interpolate values on the interface, the error in the closest point search is, in our case, more relevant.



(a) Mean curvature relative error and standard deviation.



(b) L_∞ error to exact curvature and L_∞ normal deviation. The last normal deviation measure for the calculus without perturbation is decreasing less rapidly because of real number double precision limits at such small scales ($h = 1/2048$ and errors of the order 10^{-11}).

Fig. 8. Circle geometrical test case: convergence of the different error measures as a function of R/h in presence of small perturbations for the fourth-order *CP* method.

The L_2 and L_∞ closest point error measures are defined as

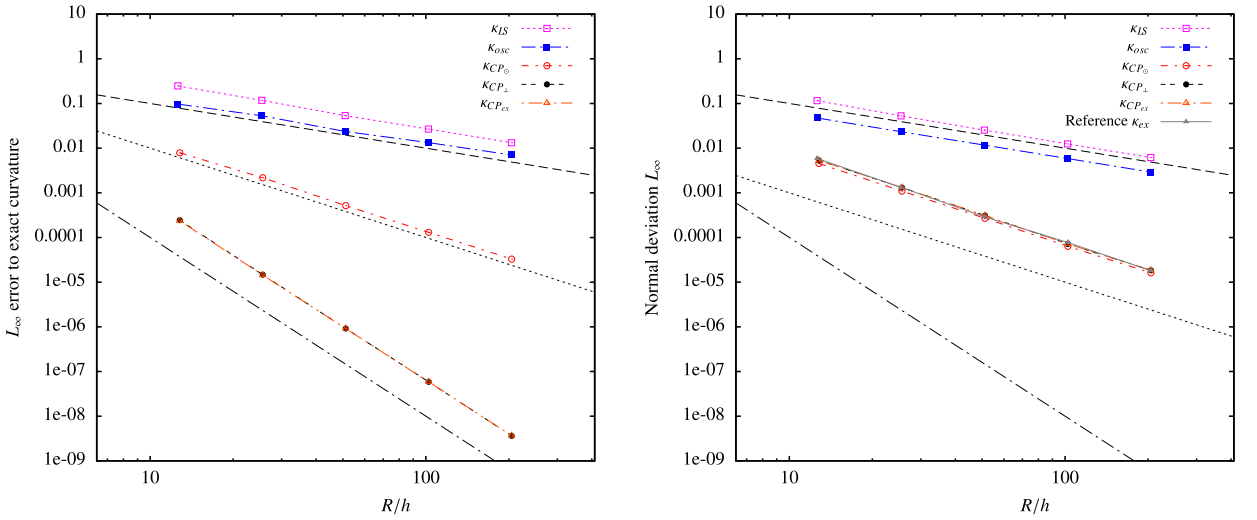
$$L_{2,CP.} = \sqrt{\int |CP.(\mathbf{x}) - CP_{ex}(\mathbf{x})|^2}$$

and

$$L_{\infty,CP.} = \max |CP.(\mathbf{x}) - CP_{ex}(\mathbf{x})|$$

where the dot index is either $\odot$ or $\perp$ standing for the two different CP algorithms. We also define the distance map error measures as

$$L_{2,d.} = \sqrt{\int (d(\mathbf{x}, CP.(\mathbf{x})) - d(\mathbf{x}, CP_{ex}(\mathbf{x})))^2}$$



(a) L_∞ errors to exact curvature. The last three curves stand for the three closest point estimates: $CP_\odot$, $CP_\perp$ and CP_{ex}. The use of $CP_\perp$ lead to a gain of 2 orders in the convergence and gives results similar to those of CP_{ex} for reference.

(b) Normal deviation with the lower bound reference curve using κ_{ex} which equals all the CP methods.

Fig. 9. Ellipse geometrical test case: spatial convergence as a function of R/h for κ_{LS} with an order 4 scheme, the osculatory circle correction and the $CP_\odot$ and $CP_\perp$ methods.

Table 3
Ellipse geometrical test case: approximate order of convergence of error measures for the different methods, a 4th order κ_{LS} computation is used.

Method	L_∞	Normal $L_{\infty,\tilde{\kappa}_{\sigma,\mathbf{n}}}$
κ_{LS}	1	1
κ_{osc}	1	1
$CP_\odot$	2	2
$CP_\perp$	4	2
CP_{ex}	4	2

and

$$L_{\infty,d_\cdot} = \max |d(\mathbf{x}, CP_\cdot(\mathbf{x})) - d(\mathbf{x}, CP_{ex}(\mathbf{x}))|$$

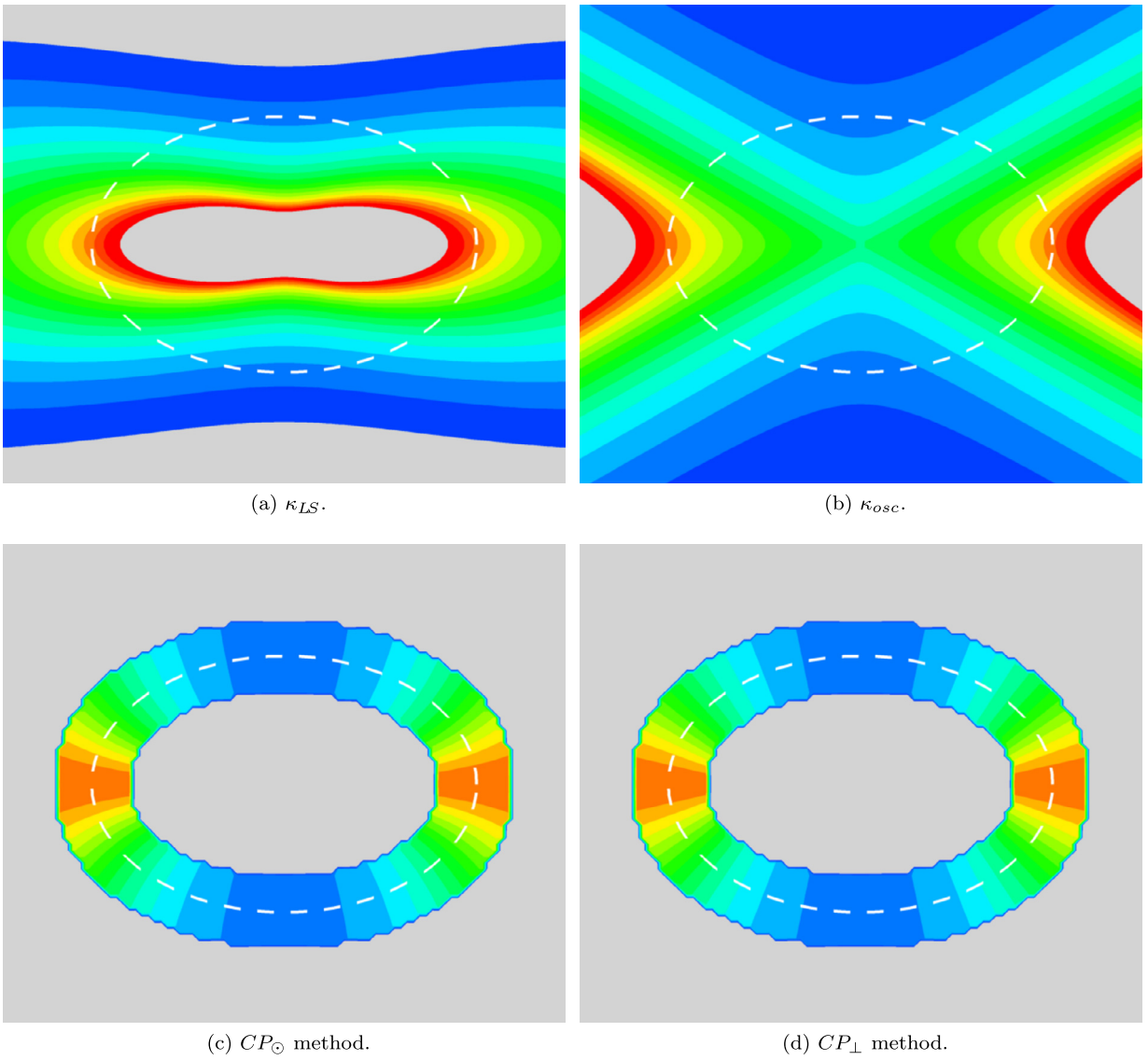
where $d(\mathbf{x}, \mathbf{y})$ is here the Euclidean distance from $\mathbf{x}$ to $\mathbf{y}$. In a similar fashion to §3.4.2 error measures are numerically evaluated in a band around the surface.

Curvature spatial convergence. Fig. 9 shows the spatial convergence for the computed curvature as a function of R/h for the L_∞ error measures (L_2 errors are qualitatively equivalent). Table 3 presents an overview. As expected, the osculatory circle method is decreased to first order while the CP method without colinearity converges at second-order, and the use of $CP_\perp$ converges at fourth-order showing the clear advantage of using an accurate Closest Point search. Moreover, this last method approximately equals the reference results when using CP_{ex}. In the ellipse case, the error measures obtained with and without reinterpolation of the level set (eq. (30)) are approximately equal and hence not shown. We can conclude that this process does not deteriorate the solution in the general case, while it has a clear benefit for smooth surfaces locally homogeneous to a circle.

In this test case the L_∞ normal errors are decreased to second-order for both CP methods. This loss of precision compared to the circle case is due to the variation of curvature created by the interpolations from equation (25). This can be observed in Fig. 9(b) where we have also plotted the normal deviation error based on the interpolation of the exact curvature computation $\kappa_{ex}(\mathbf{x})$ at $CP_{ex}(\mathbf{x})$, which is the numerical lower bound that can be attained by any method. All the results using any of the CP methods approximately equal this reference curve for $L_{\infty,\tilde{\kappa}_{\sigma,\mathbf{n}}}$.

Fig. 10 shows a spatial representation of the curvature in the domain for the different methods for $R/h = 25.4$. We can clearly see the effect of the CP algorithm on the extension of the curvature along the normal. Looking closely, the simple $CP_\odot$ method induces non-rectilinear iso-values for the curvature while the $CP_\perp$ method corrects this problem. The exact curvature profile obtained with CP_{ex} is not shown as it is indistinguishable from the $CP_\perp$ method.

Numerical convergence in the presence of perturbations. Fig. 11 shows the spatial convergence of the closest point error and the distance map as a function of R/h with varying perturbations applied to the level set as presented in eq. (33). Summarized



(a) κ_{LS}. (b) κ_{osc}.

(c) $CP_{\odot}$ method. (d) $CP_{\perp}$ method.

Fig. 10. Ellipse geometrical test case: curvature profiles for the different methods. The position of the ellipse is drawn with white dashed lines. The color variation goes from blue ($\kappa = 2.5$) to red ($\kappa = 10$), gray zones represent curvature values outside that interval or outside the computation band for the *CP* methods. The $CP_{\perp}$ method and the exact curvature profiles are indistinguishable hence the latter is not shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in Table 4, we observe a maximum order 2 for the closest point error measure in the case of the $CP_{\odot}$ algorithm while it increases to fourth-order for the $CP_{\perp}$ method. The $CP_{\odot}$ method's distance map has a maximum third-order convergence, while the $CP_{\perp}$ one is up to fourth-order. We believe that the convergence rate is limited because of the errors on the computation of $\nabla\phi$, on the numerical thresholds used, but more importantly because of the fourth-order interpolation functions used. However this fourth-order convergence rate is sufficient in our case and for the applications targeted.

The $CP_{\perp}$ algorithm we propose in this paper shows a clear benefit compared to the $CP_{\odot}$ used in the literature that will have an important impact on the order of convergence of the κ_{CP} methods studied in the next paragraph.

Computational cost of the closest point algorithms. Fig. 12 shows the mean number of iterations per grid point for the $CP_{\odot}$ and $CP_{\perp}$ methods. The use of a fourth-order accurate algorithm requires less than 3 times more iterations than using a second-order accurate. This difference is mostly due to the reduced threshold ϵ from h^2 to h^4 that is more hardly reached. With fourth-order precision, for coarse grids with more errors on the computation of the normal and interpolations, the $CP_{\perp}$ algorithm requires significantly more iterations than the simple $CP_{\odot}$ algorithm (around three times for $R/h = 12.8$). This ratio reduces to around 1.6 for a finer mesh. This behavior is expected as the higher the curvature to spatial step ratio, the farther away the first $CP_{\odot}$ iteration will bring us from the exact closest point. Fig. 13 reports the computation costs in

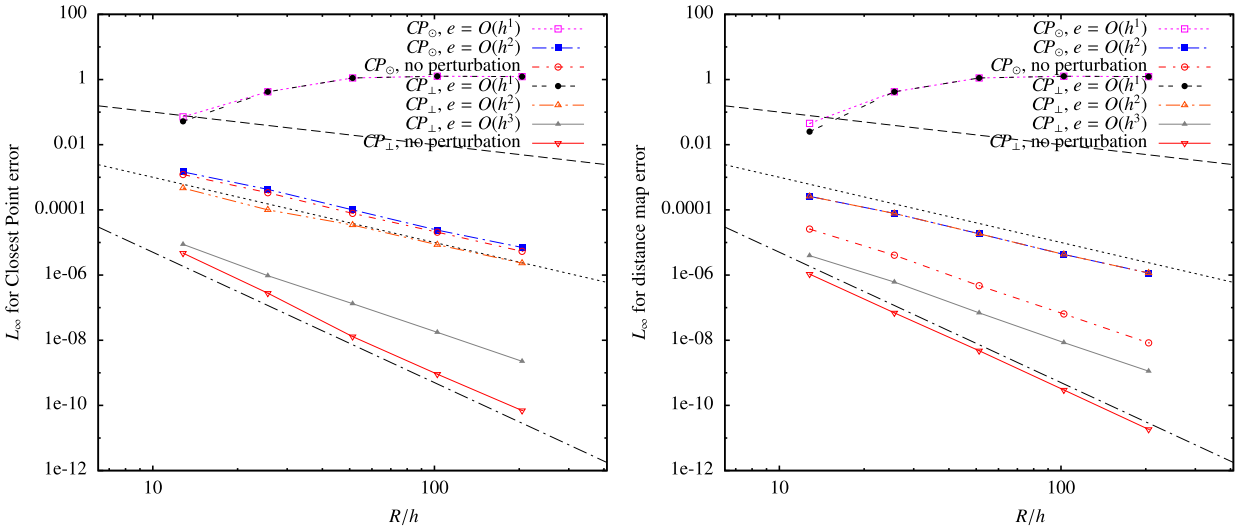
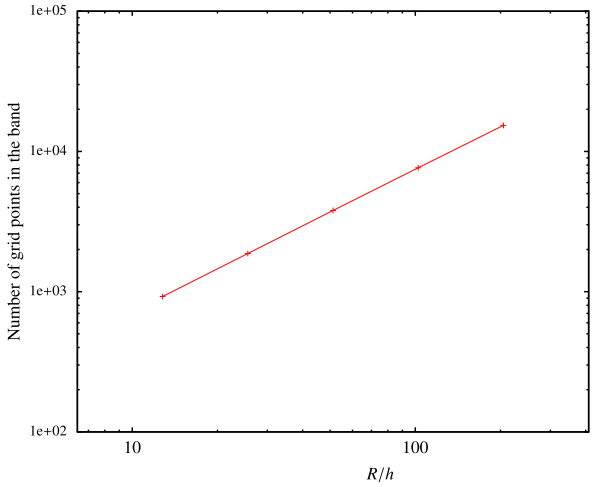


Fig. 11. Ellipse geometrical test case: convergence of the $L_{\infty,CP}$ closest point error (left) and $L_{\infty,d}$ distance map error (right) as a function of R/h in the presence of small perturbations for $CP_\odot$ and the $CP_\perp$ methods. Results for $CP_\odot$ with $O(h^3)$ and $O(h^4)$ errors are almost equal to the ones with no perturbation thus not shown. Similarly, results for $CP_\perp$ with $O(h^4)$ perturbations are not shown.

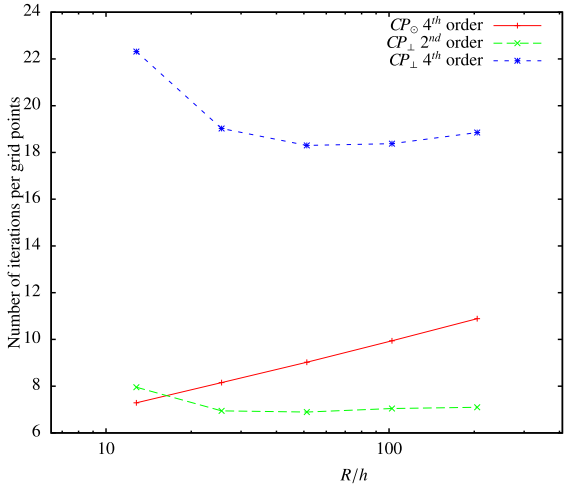
Table 4
Ellipse geometrical case: approximate orders of convergence of error measures for the two different CP methods and ϕ perturbations.

Method	Perturb.	$L_{\infty,CP}$	$L_{\infty,d}$
$CP_\odot$	$O(h^1)$	0	0
$CP_\odot$	$O(h^2)$	2	2
$CP_\odot$	$O(h^3)$	2	3
$CP_\odot$	$O(h^4)$	2	3

Method	Perturb.	$L_{\infty,CP}$	$L_{\infty,d}$
$CP_\perp$	$O(h^1)$	0	0
$CP_\perp$	$O(h^2)$	2	2
$CP_\perp$	$O(h^3)$	3	3
$CP_\perp$	$O(h^4)$	4	4



(a) Linear increase of the number of points in the $6h$ band with respect to R/h.

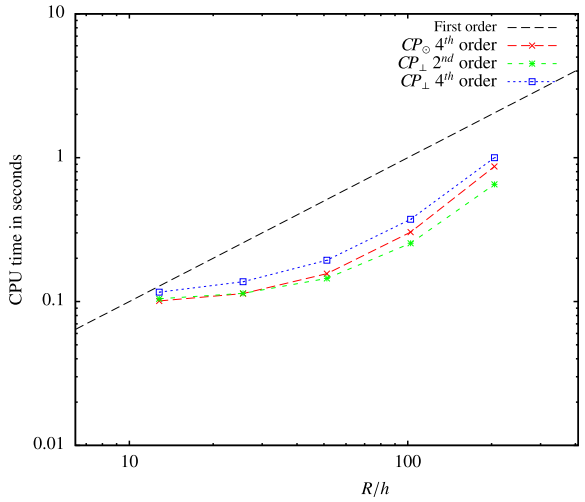


(b) Mean number of iterations per grid points in the $6h$ band for the closest point algorithms $CP_\odot$ with fourth-order precision and $CP_\perp$ with second and fourth-order precision.

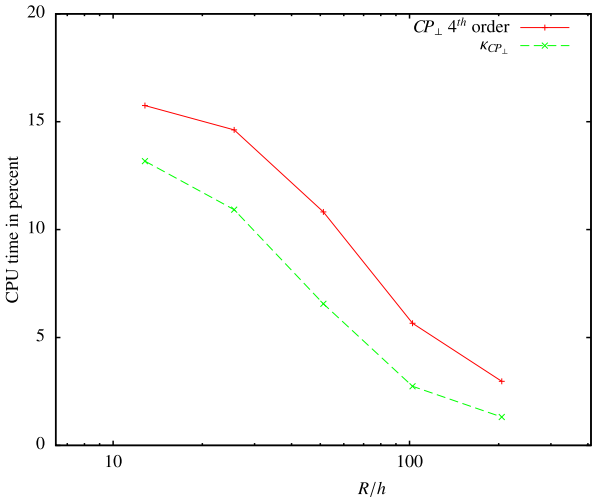
Fig. 12. Efficiency of the closest point algorithms.

terms of CPU time for the different methods. The use of the fourth-order accurate $CP_\perp$ algorithm against the second-order one (the quicker) requires only 1.5 times more computational time for the finer mesh. Moreover, in our fluid solver, the most accurate algorithm represents less than 3% of the overall computational time for one iteration.

As the number of grid points grows linearly with the spatial discretization, the use of the $CP_\perp$ with fourth-order precision is highly recommended and has a low impact on the computational time.



(a) Fourth-order $CP_\odot$, $CP_\perp$ with second and fourth-order precision.



(b) Percentage of the computational time for one fluid solver iteration for the closest point computation and the curvature computation.

Fig. 13. Computational cost of the CP algorithms as a function of the spatial discretization.

4.2.3. Conclusion of the geometrical study

Through this geometrical study on the circle and ellipse cases, we have shown that the CP method attains high order (up to 4) spatial convergence on the different error measures guaranteeing numerical accuracy for Criteria 2.1, 2.2 and 2.3 of §2.4 and thus for the surface tension driven simulations. The use of a fourth-order κ_{LS} computation is also a key in attaining this precision and is very important in the reduction of errors in dynamic numerical simulations.

Compared to the $CP_\odot$ method, the $CP_\perp$ method gains 2 orders of convergence for the curvature's L_2 and L_∞ error in the general (ellipse) case while only requiring double the number of interpolations. The reinterpolation $CP_\perp^2$ method improves the error in the normal deviation for the circle, which will help stabilize the static column equilibrium case studied below.

4.3. Static viscous column equilibrium with constant density

The problem of the equilibrium of a 2D column subjected to surface tension forces has been widely used as a reference in the literature, for example [39,16,40] propose numerical results and analysis with various methods in VOF or level set frameworks. From the Laplace law, without discretization errors, the surface tension forces should not create any current – the fluid should be at rest – as they are at equilibrium with the pressure jump between the two phases:

$$\nabla p = \frac{1}{We}\kappa \mathbf{n}\delta_\Gamma \tag{34}$$

giving a pressure inside the drop of $p_{drop} = \sigma\kappa$ when the reference pressure outside is 0.

The Weber number is given as

$$We = Re \cdot Ca = \frac{\rho U L}{\mu} \cdot \frac{\mu U}{\sigma} = \frac{\rho U^2 L}{\sigma}, \tag{35}$$

with U (resp. L) a reference velocity (resp. length) characterizing the problem. In our problem, $L = D$, the diameter of the column.

Two other relevant non-dimensional numbers are the Laplace and the Ohnesorge number relating the surface tension to the viscous forces:

$$La = \frac{\sigma\rho L}{\mu^2} = Oh^{-2} = \frac{Re^2}{We}. \tag{36}$$

The Laplace number characterizes the numerical behavior of the flow computed due to numerical errors on the discretization of the surface tension forces, i.e. the errors in the localization δ_Γ of the column's interface, its normal $\mathbf{n}$ and its curvature κ will create capillary waves and thus currents that will propagate in the fluids through the viscous term. The more viscous the fluid is then the more smoothly the problem will reach equilibrium. At high Laplace numbers, we expect the interface between the two phases to oscillate/vibrate around its equilibrium state more rapidly and for a longer time.

In the following sections, for clarity, we call "velocity" the root mean square of the fluid velocity as:

$$v_{rms} = \sqrt{\int_{\Omega} |\mathbf{v}(\mathbf{x})|^2 d\mathbf{x}} \tag{37}$$

and the capillary number is defined in absolute value as:

$$Ca = \frac{\mu |\mathbf{v}|_{max}}{\sigma}. \tag{38}$$

If not stated differently, v_{rms} is given in the T_σ and U_σ reference frames as defined below.

As shown in [11], applying a constant curvature κ_{exact} for the surface tension force in a balanced force CSF framework leads to numerical equilibrium instantly, i.e. v_{rms} is of the order of the machine precision. We study below the effect of errors on the computation of the curvature.

Origins of the parasitic currents. In [17], the authors analytically demonstrate that, for the static column at equilibrium, the parasitic currents are quadratically proportional to the curvature. This is particularly crucial as the numerical errors will increase rapidly at small scales and finally dominate the flow. We push their study further by introducing the errors in the computation of the volume fraction gradient $\nabla c = \overline{\nabla c} + \beta$, where $\overline{\nabla c}$ is the exact solution and β represents the numerical errors that is not necessarily the gradient of a function. The curvature κ is decomposed in a similar way: $\kappa = \overline{\kappa} + \kappa'$, where κ' is the error part. Neglecting the viscous stress tensor and posing $\rho = 1$ for clarity, the Navier–Stokes equation holds:

$$u_t + u \cdot \nabla u = -\nabla p + \sigma \kappa \nabla c. \tag{39}$$

When the disk is at equilibrium, the pressure gradient and the surface tension force are in balance, i.e. $\nabla p = \sigma \overline{\kappa} \overline{\nabla c}$, equation (39) becomes:

$$u_t + u \cdot \nabla u = \sigma (\overline{\kappa} \beta + \kappa' \overline{\nabla c} + \kappa' \beta).$$

The error term $\kappa' \beta$ is of higher-order and, numerically, the error in the gradient of c is of higher-order than the error on κ because they are usually derived from the same surface representation. Hence, if we drop those two terms, we find back the solution proposed in [17]. This demonstrates the origins of parasitic currents for the disk at equilibrium, dominated by the errors in the curvature computation.

Non-dimensional reference frames. All the velocities are given in the velocity scale for the inviscid problem:

$$U_\sigma = \sqrt{\frac{\sigma}{\rho D}}$$

as well as the corresponding time scale:

$$T_\sigma = \sqrt{\frac{\rho D^3}{\sigma}}$$

which is proportional to the period of the capillary waves. We also define the viscous time scale as:

$$T_\mu = \frac{D^2}{\mu}.$$

Time step restriction. As was clearly demonstrated in [41], the stability of flow computations subjected to surface tension is limited not by its explicit treatment but by constraints due to the physical propagation of waves. It yields to the following condition on the time step:

$$\Delta t_\sigma^{stat} \leq \frac{h}{\hat{c}_\sigma} \sqrt{\frac{(\rho_1 + \rho_2) h^3}{2\pi\sigma}}$$

for the static case and

$$\Delta t_\sigma^{dyn} \leq \frac{h}{\hat{c}_\sigma + u_\Sigma^k}$$

for the dynamic case, where u_Σ^k is the velocity of the flow at the interface parallel to the capillary wave of number k.

In most of the simulations studied below, we have respected the static time step constraint, however we have observed, like detailed in [41], that for problems that are sufficiently smooth, larger time steps can be used without introducing instabilities in the computations.

Table 5
Static column case: numerical results for the finest grid ($R/h = 102.4$) and approximate global order of convergence for the $CP^2_\perp$ method, order 2 and order 4 κ_{LS} calculation, $La = 120$. Detailed numerical results are given for varying R/h in Table 11.

(a) Velocity error measures and pressure error.

κ_{LS}	R/h	Ca at $t = \Delta t$	O	Ca at $t_\sigma = 30$	O	max Ca	O	$E(\Delta p_{mean})$ at $t_\sigma = 30$	O
Order 2 scheme	102.4	1.19×10^{-8}	1.98	1.34×10^{-14}	4.93	2.27×10^{-7}	1.99	9.91×10^{-7}	2.00
Order 4 scheme	102.4	3.74×10^{-11}	4.05	2.08×10^{-14}	4.30	4.13×10^{-10}	3.97	4.24×10^{-10}	3.94

(b) Curvature error measures at $t_\sigma = 30$.

κ_{LS}	R/h	$\lvert\frac{\kappa_{mean}-\kappa_{ex}}{\kappa_{ex}}\rvert$	O	L_∞ curvature error	O	κ_σ	O	L_∞ normal dev.	O
Order 2 scheme	102.4	9.91×10^{-7}	2.00	9.91×10^{-7}	2.00	6.31×10^{-11}	5.28	1.68×10^{-11}	5.09
Order 4 scheme	102.4	4.24×10^{-10}	3.94	4.47×10^{-10}	3.95	2.30×10^{-11}	4.93	1.27×10^{-11}	4.57

4.3.1. General configuration

We consider the physical problem of a column of diameter $D = 2R = 0.4$ centered at the origin of a unit square. The density and the viscosity are set to unity: $\rho = \mu = 1$. The Laplace number is thus varied by changing the surface tension coefficient σ.

The level set is initialized exactly as

$$\phi(\mathbf{x}) = |\mathbf{x}| - R.$$

No symmetry property is used; we compute solutions in the whole domain. No-slip conditions are applied to all boundaries.

The pressure error, considering the reference pressure outside the drop to be zero, is computed as:

$$E(\Delta p_{mean}) = \frac{|p_{2,mean} - p_{1,mean} - p_{drop}|}{p_{exact}}$$

where $p_{i,mean}$ is the mean pressure calculated in the ith fluid by integrating numerically in the whole domain, with $i = 2$ inside the column:

$$p_{i,mean} = \frac{\sum p \cdot H_{\epsilon,i}(\phi/|\nabla\phi|)}{\sum H_{\epsilon,i}(\phi/|\nabla\phi|)}$$

where $H_{\epsilon,i}$ is the regularized characteristic function associated with the ith phase.

4.3.2. Order 2 and order 4 κ_{LS} calculation

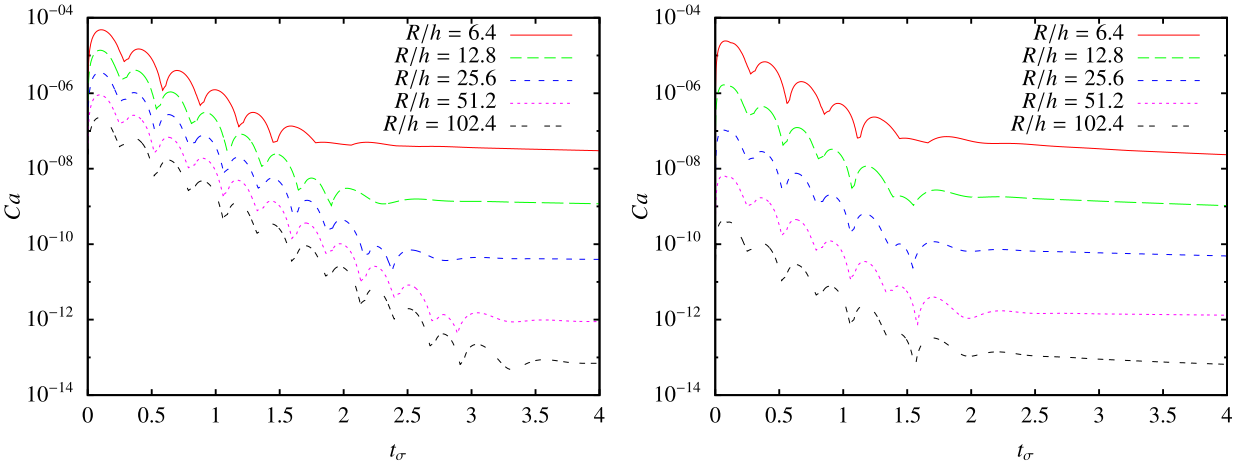
We study the dynamic effect of spatial errors due to the κ_{LS} computation precision on the $CP_\perp$ method by varying the scheme used and the number of grid points. The Laplace number is 120, corresponding to $\sigma = 300$, and the time step is set to $\Delta t = 3 \times 10^{-5}$ s. The curvature errors and the impact on the fluid velocity are shown in Fig. 14; as time evolution for the pressure, the mean curvature and its normal deviation are small, they are not shown. The numerical results are reported in details in Appendix A.2, Table 11, and summarized for the finest grid resolution in Table 5.

The results clearly show that, when compared to the second-order scheme, the fourth-order computation of κ_{LS} greatly reduces the capillary number in the first time steps to order 4 as well as decreases the maximum spurious currents. When the approximate numerical equilibrium is reached for all simulations, around $t_\sigma = 30$, the pressure error also converges at fourth-order, as expected from eq. (34) and from the balanced force CSF principle. After $t_\sigma = 30$, the velocity still declines but much less rapidly.

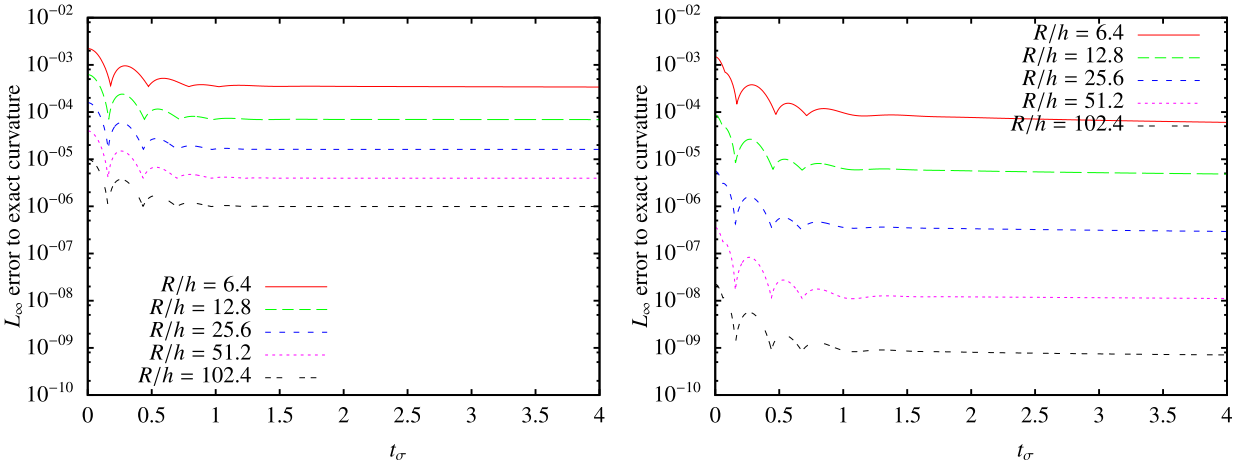
The capillary number converges toward machine precision at an even higher order when spatial discretization tends to zero. The RMS velocity follows qualitatively the same behavior thus we do not show it here. It is noticeable that the second and fourth-order schemes converge toward the same approximate equilibrium Ca and RMS velocity residual. This is due to the physical dynamics of the test case that searches for a numerical equilibrium state, which geometrically depends on h, through the level set function displacement by minimizing the standard deviation on the curvature. We see in Fig. 14(c) that for the same spatial step h, the approximate minimal κ_σ is qualitatively independent of the κ_{LS} accuracy. However, this minimum is reached at around $t_\sigma = 1.5$ for any discretization of the 4th order scheme, whereas that time is stretching with decreasing h for the second-order scheme. As with the latter scheme, more numerical errors are introduced in the early stages, and the numerical equilibrium is reached later implying more oscillations of the interface.

Surprisingly, we observe a faster convergence rate on the capillary number at $t_\sigma = 30$ for the second-order κ_{LS} computation. We believe that at very low resolution such small velocities are close to machine precision and thus very sensitive to numerical perturbations, which are non-negligible at these scales. For instance, the maximum velocities for the $R/h = 102.4$ discretization approximately equals 5×10^{-12} which is of the same order of magnitude as the maximum velocity introduced in the flow when applying the constant curvature $\kappa_{ex} = 1/R$.

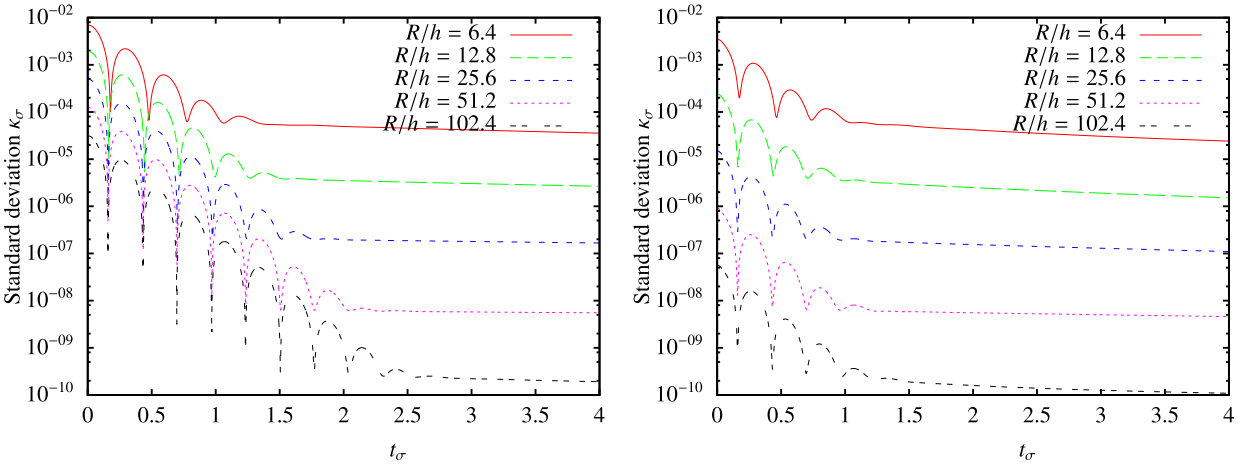
However, as the curvature and the surface are in better geometrical balance for high-order schemes for κ_{LS} (refer to 4.2 for the geometrical convergence analysis), the errors induced in the early time steps and on the maximum capillary number



(a) Capillary number.



(b) L_∞ error to exact curvature.



(c) Standard deviation to the mean curvature.

Fig. 14. Static column case: convergence study for the RMS velocity, L_∞ curvature error and standard deviation using the $CP^2_\perp$ method and order 2 (left) and order 4 (right) κ_{LS} calculation, $La = 120$.

Table 6
Static column case: convergence of capillary number in function of the Laplace number and R/h with the proposed method compared to results from [12,13,15,42].

(a) Varying Laplace number at $R/h = 6.4$.

R/h	Ca		La			
			120	1200	12 000	120 000
6.4	At $t_{cap} = 250$	Proposed method	6.74×10^{-9}	6.82×10^{-9}	5.42×10^{-7}	2.68×10^{-7}
		[42]	4.59×10^{-7}	7.80×10^{-7}	2.18×10^{-6}	3.00×10^{-6}
		[12] Cartesian	1.10×10^{-7}	1.20×10^{-7}	1.44×10^{-6}	3.09×10^{-6}
		[15]	2.18×10^{-6}	2.18×10^{-6}	2.22×10^{-6}	–
		[13]	5.71×10^{-6}	5.99×10^{-6}	8.76×10^{-6}	–
	At $t_\sigma = 80$	Proposed method	2.04×10^{-9}	1.37×10^{-9}	5.02×10^{-10}	4.68×10^{-10}
12.8	At $t_\sigma = 80$	Proposed method	9.16×10^{-11}	3.58×10^{-11}	1.78×10^{-11}	1.03×10^{-11}

(b) Spatial convergence for $La = 12\,000$ compared to results from the literature.

Ca		R/h			
		6.4	12.8	25.6	51.2
At $t_{cap} = 250$	Proposed method	5.42×10^{-7}	9.21×10^{-9}	1.13×10^{-9}	6.53×10^{-11}
	[42]	2.18×10^{-6}	2.32×10^{-7}	5.91×10^{-8}	–
	[12] Cartesian	1.44×10^{-6}	3.40×10^{-7}	5.00×10^{-8}	–
	[13]	6.68×10^{-6}	1.07×10^{-6}	1.20×10^{-7}	–
At $t_\sigma = 20$	Proposed method	1.52×10^{-9}	4.49×10^{-11}	1.72×10^{-12}	1.55×10^{-13}

are much smaller which will lead to reduced errors in the general case. We see in Figs. 14(a) and 14(b) the proportional link between the curvature's standard deviation and the spurious currents that arise mostly because of κ_σ. The normal deviation for this column case is, as expected, equal for κ_{LS} at orders 2 and 4 because of the regularity of the particular circular surface as it was shown in §4.2. When the standard deviation becomes stable after 4 half-oscillations, the surface oscillates one more time until parasitic currents are damped by viscosity and numerical equilibrium is reached.

The use of a fifth order WENO scheme for the advection of the level set does not change this convergence rate as expected (down to order $5 - 2 = 3$). We believe that this is due to the smoothness of the level set function and to the very small velocities, which introduce errors in ϕ much less than $O(h^5)$, added to the dynamics of the case that searches to minimize the geometrical errors. We show in §4.5 how the advection of the level set impacts the appearance of spurious currents.

4.3.3. Varying the Laplace number

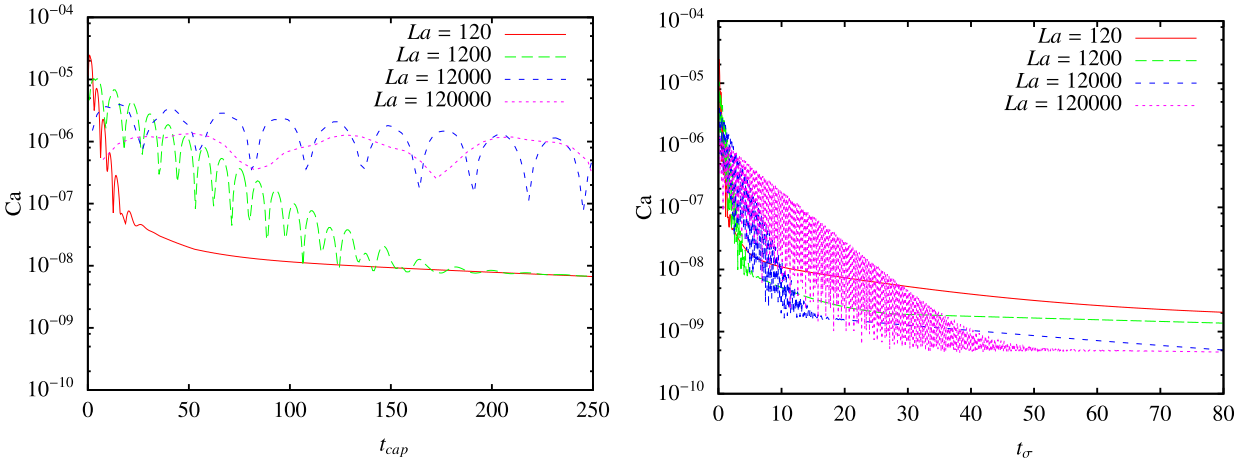
We study the effect of a varying the Laplace number by changing the surface tension coefficient σ at fixed density and viscosity. Augmenting La will make the interface response to numerical errors "stiffer" thus reaching a stable state after a longer time in the T_σ reference frame.

Here we used the $CP^2_\perp$ method with a 4th order κ_{LS} and the spatial discretization is $R/h = 6.4$. Because of the varying surface tension coefficient, the time step is adapted as $\Delta t = \{3, 1, 0.3, 0.1\} \times 10^{-4}$ respectively for $La = \{120, 1200, 12\,000, 120\,000\}$.

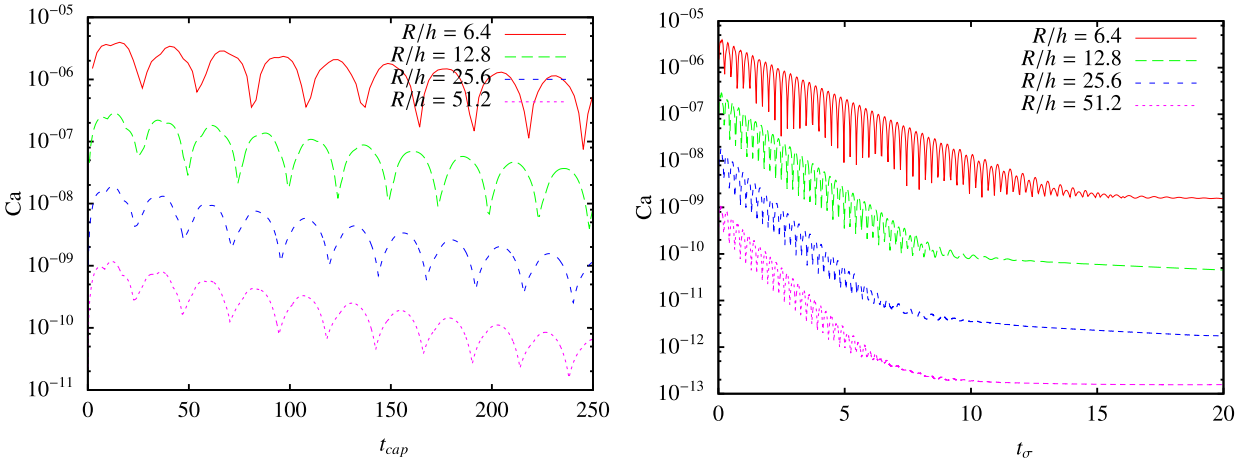
We have summarized and compared the numerical results to the ones from the literature in Table 6 where we can see that the spurious currents obtained are much smaller for the ratio $R/h = 6.4$. Moreover, the convergence rate with respect to h is also much higher. Note that in these references Ca is computed at $t_{cap} = t\sigma/(D\mu) = 250$, a time when the simulation has not converged to equilibrium velocity, which as we observed in our simulations (see Fig. 15(a)), is around $t_{cap} = 25\,000$ for $La = 120\,000$. Thus the velocity at that time is still oscillating significantly toward equilibrium and is not minimum as shown in Fig. 15(b). Consequently, we have given a new table of Ca number for different $t_\sigma = t/T_\sigma$ (when the velocity is stabilized), which we believe is more representative.

The results obtained for $R/h = 6.4$ suffer from excessively coarse spatial discretization as the regularization width is $\epsilon = 2h \simeq 0.3\,R$. Thus we propose to study the case where $R/h = 12.8$ for which we show more complete numerical results in Fig. 15, where Δt has been divided by 3. The results are qualitatively equivalent those obtained by [14] in a VOF framework for the same discretization. However, the velocity RMS is approximately 20 times smaller in our case thanks to the higher order of convergence of the curvature error of the $CP_\perp$ method. This difference will increase rapidly with finer discretization. As expected, increasing the Laplace number makes the interface stiffer and takes more time to reach equilibrium, where the oscillations observed converge toward an approximate period of $0.44\,T_\sigma$.

We observe in Figs. 16(c) and 16(d) a reduction of the curvature error measures when augmenting La. The Ca curve follows the standard and normal deviations showing the importance of those two criteria in the reduction of spurious currents. This is consistent with the idea that a higher surface tension coefficient makes the surface response stiffer, thus impacting the level set's movement in the vicinity of Γ making it numerically more homogeneous.



(a) Varying the Laplace number for $R/h = 6.4$.



(b) Varying the spatial discretization for $La = 12,000$.

Fig. 15. Static column case: convergence study using the $CP^2_\perp$ method and order 4 κ_{LS} calculation in the T_{cap} (left) and T_σ (right) reference frames.

4.4. Static viscous column equilibrium with variable density

In order to study how the method performs in general fluid simulations, we study the effect of surface tension on the column problem with a variable density.

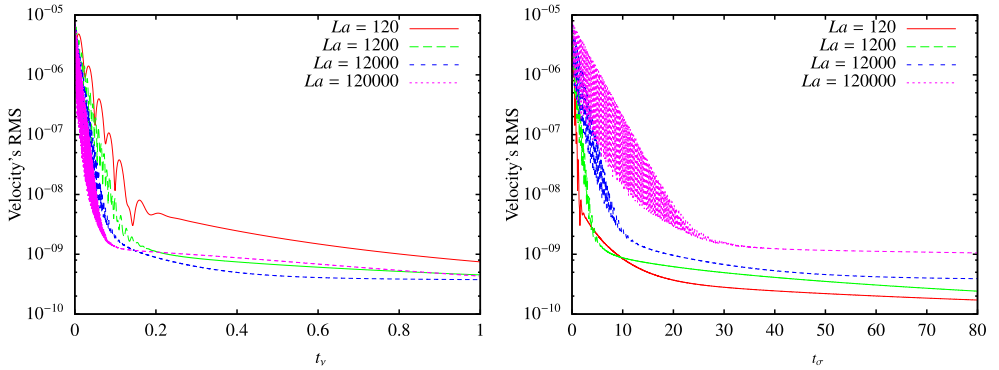
4.4.1. General configuration

We take exactly the same configuration as in §4.3 but change the density ρ_1 of the fluid inside the column while keeping the density outside ρ_2 fixed and σ constant. This has the effect of increasing the Laplace number. As ρ_2 does not change, the time step is restricted by this lower bound and we fixed $\Delta t = 1 \times 10^{-4}$. However, for increasing density, we observed in the first time steps a large error in the velocity field that we believe is due to the quick diffusion of the initial errors on the surface tension term by the reduced kinematic viscosity $\nu = \mu/\rho$. To counter this phenomenon, we reduced Δt to 1×10^{-6} for the first hundred computational steps, leaving time for the simulation to stabilize, and then smoothly increased it to 1×10^{-4}.

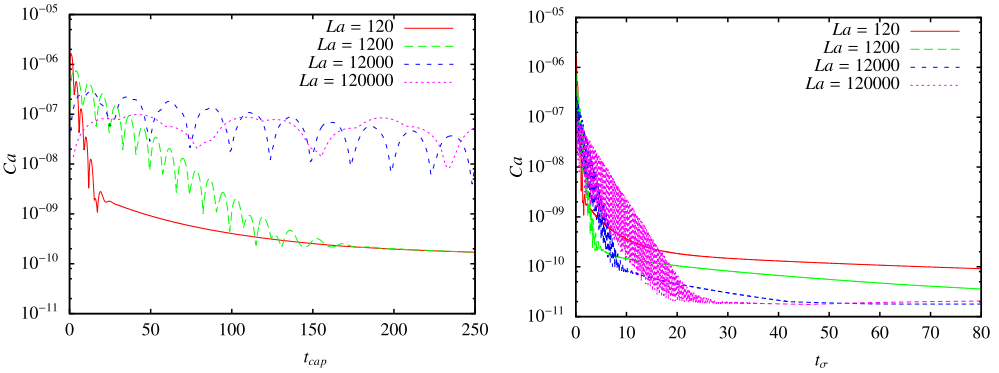
4.4.2. Varying the density

The surface tension coefficient is set to 300 while the density is varied as $\rho_1 = \{1, 10, 100, 1000\}$ with corresponding Laplace numbers $La = \{120, 660, 6060, 60\,060\}$, calculated as:

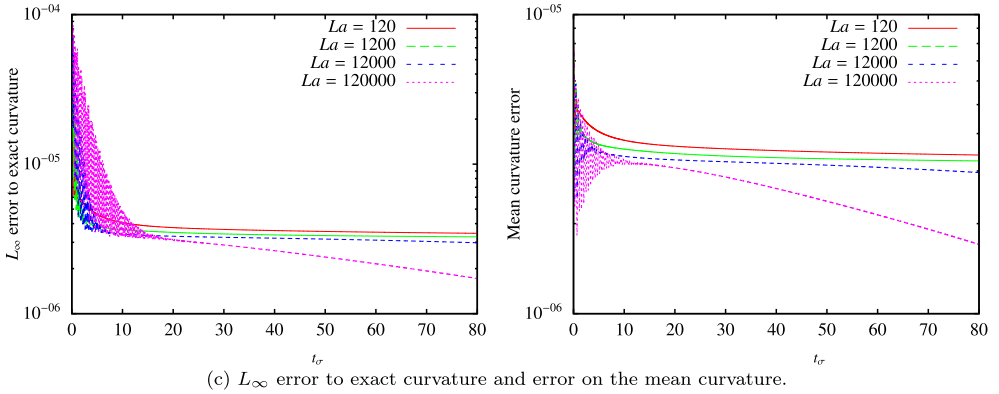
$$La_{\rho_1,\rho_2} = \frac{\sigma \overline{\rho} L}{\mu^2} \tag{40}$$



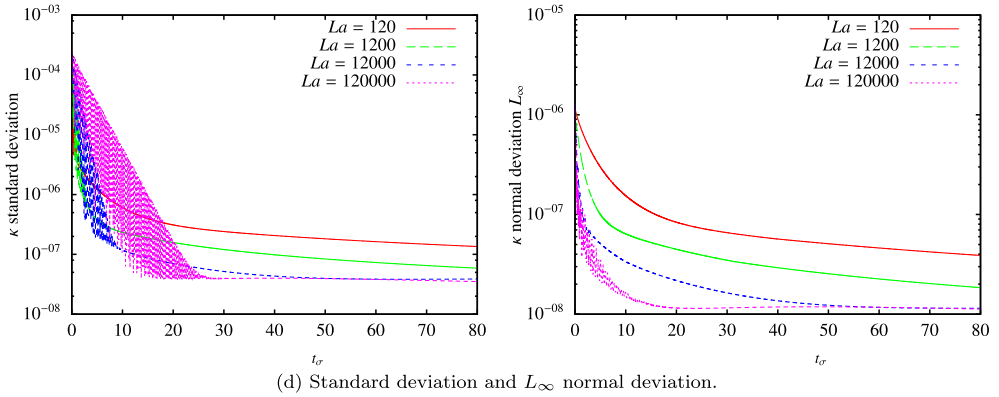
(a) Velocity's RMS for different time scales: T_ν and T_σ.



(b) Capillary number for different time scales: T_{cap} and T_σ.



(c) L_∞ error to exact curvature and error on the mean curvature.



(d) Standard deviation and L_∞ normal deviation.

Fig. 16. Static column case: convergence study by varying the Laplace number using the $CP^2_\perp$ method and order 4 κ_{LS} calculation with $R/h = 12.8$.

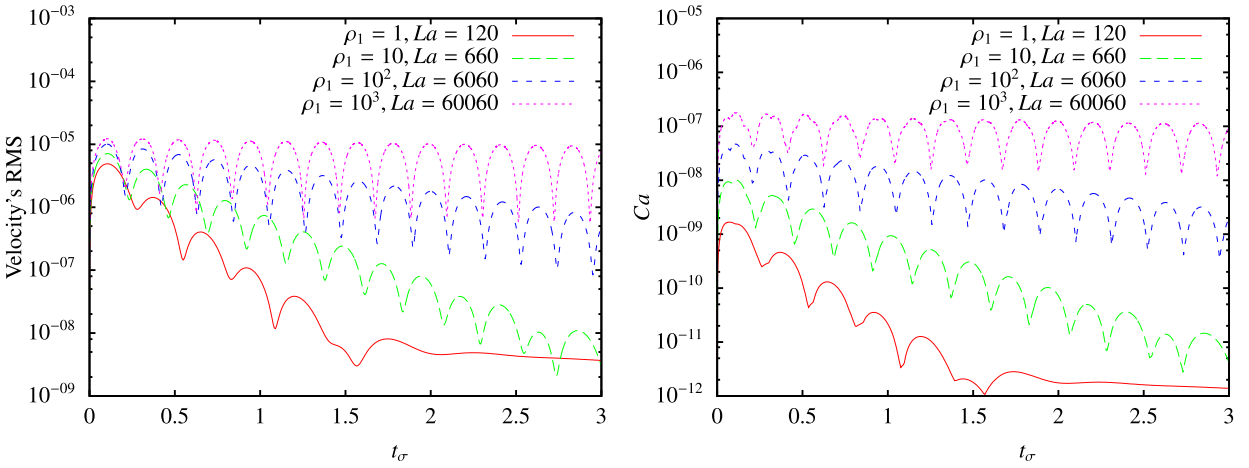


Fig. 17. Static column case with variable density: convergence study for the RMS of the velocity and *Ca* number by varying the density inside the column using the $CP^2_\perp$ method and order 4 κ_{LS} with $R/h = 12.8$.

with

$$\overline{\rho} = \frac{\rho_1 + \rho_2}{2}.$$

The reference scales T_σ and U_σ are defined with that same mean density.

Fig. 17 shows coherent results for the constant density test in §4.3 that demonstrates the stability and accuracy of the method in the presence of variable density.

4.5. Advected viscous column equilibrium

In these test cases, we study the impact of the errors due to the advection of the surface through the level set advection by an initial constant velocity field. Theoretically, in the reference frame of the transported column, the spurious currents should tend to zero. Due to the grid aliasing effect and advection errors we expect the interface to reach equilibrium with more difficulty, while still creating minimized spurious currents.

4.5.1. General configuration

The fluid parameters used are the same as in §4.3: $\rho = \mu = 1$ in the two phases. The domain is a rectangle of size $[1, 2]$ with symmetry conditions for the top and bottom boundaries, and periodic conditions in the horizontal direction. The column has a radius of $R = 0.2$. We initialized the x-velocity to $U_x = 2$ in the whole domain so that, ideally, the velocity field should remain $\mathbf{U} = (U_x, 0)$ through time. The spurious currents are measured in the column reference frame and calculated as:

$$v_{rms} = \sqrt{\int_\Omega |\mathbf{v}(\mathbf{x}) - \mathbf{U}|^2 d\mathbf{x}}$$

and

$$Ca = \frac{\mu |\mathbf{v} - \mathbf{U}|_{max}}{\sigma}.$$

All the velocities are given in the velocity scale for the advection problem:

$$U_{adv} = \sqrt{U_\sigma \cdot U_x}$$

as well as the corresponding time scale:

$$T_{adv} = D/U_x.$$

The CFL involved here will be much larger than the time step restriction due to surface tension: $\Delta t_{CFL} \gg \Delta t_\sigma$.

Table 7
Advected column case: numerical results for varying R/h and approximate order of convergence for the $CP^2_\perp$ method and order 4 κ_{LS} calculation. Mean measures are taken in the intervals $t_{adv} \in [0.06, 0.2]$ for $La = 1200$ and $t_{adv} \in [0.03, 0.1]$ for $La = 12\,000$, v_{rms} is given in the U_{adv} reference frame.

La	R/h	Mean v_{rms}	O	Mean Ca	O	max Ca	O	$\frac{\max Ca}{mean\ Ca}$
1200	6.4	7.83×10^{-5}	–	1.23×10^{-6}	–	1.05×10^{-5}	–	8.53
	12.8	2.02×10^{-6}	5.28	3.73×10^{-8}	5.04	7.57×10^{-7}	3.79	20.27
	25.6	6.94×10^{-8}	4.86	1.56×10^{-9}	4.58	4.87×10^{-8}	3.96	31.20
	51.2	3.65×10^{-9}	4.25	1.10×10^{-10}	3.83	3.07×10^{-9}	3.99	28.02
	102.4	1.17×10^{-10}	4.96	5.00×10^{-12}	4.46	1.91×10^{-10}	4.01	38.27
12 000	6.4	3.86×10^{-4}	–	1.10×10^{-6}	–	3.93×10^{-6}	–	3.56
	12.8	5.38×10^{-6}	6.16	1.55×10^{-8}	6.16	2.89×10^{-7}	3.77	18.66
	25.6	3.11×10^{-7}	4.11	9.90×10^{-10}	3.97	1.90×10^{-8}	3.92	19.23
	51.2	2.81×10^{-8}	3.47	1.02×10^{-10}	3.27	1.19×10^{-9}	4.00	11.60
	102.4	7.35×10^{-10}	5.26	3.02×10^{-12}	5.08	7.57×10^{-11}	3.97	25.08

4.5.2. Spatial convergence

We study the impact of the advection error of the level set coupled with the errors due to the surface representation and curvature calculation by varying the spatial discretization in the same manner as in §4.3. This case was first introduced by Popinet et al. in [14] in a VOF method with Height Function curvature framework. We set $\Delta t = 3 \times 10^{-6}$ for the $La = 1200$ simulations and $\Delta t = 1 \times 10^{-6}$ for $La = 12\,000$, except for the $R/h = 102.4$ discretization where we divided Δt by three to avoid instabilities.

As shown in Fig. 18, for early times, we observe the same numerical results as in §4.3, as the errors dominating here are due to the fact that the interface is not at equilibrium. After time $t_{adv} = 0.04$ (resp. $t_{adv} = 0.02$) for $La = 1200$ (resp. $La = 12\,000$) we clearly observe the effects of the errors on the advection perturbing the interface and the curvature in small oscillations around the equilibrium state. The period of these perturbations is of scale $1/h$ in the T_{adv} reference frame. This result is due to aliasing where the interface (i.e. the level set) suffers from approximately the same numerical discretization aliasing every time it has been translated by one cell. This observation coincides with the remarks given in [14]. Yet our method for advection and curvature calculation is more precise since the errors in the spurious currents are much smaller (by a factor around 3×10^{-4} for the $R/h = 12.8$ discretization), as shown in Fig. 18 and Table 7.

The simulations never reach a non-oscillating equilibrium, so we computed mean values for the velocity's RMS and Ca in the interval $t_{adv} = [0.06, 0.2]$ for $La = 1200$ and $t_{adv} = [0.03, 0.1]$ for $La = 12\,000$. The mean capillary number computed at equilibrium is much higher than the one for the static case. The ratio $\frac{\max Ca}{mean\ Ca}$ gives a good indication of how the method efficiently manages to reduce the spurious currents counteracting the errors constantly introduced into the level set. The curvature error measures presented in Fig. 18(c) were also computed as a mean in the interval $t_{adv} = [0.06, 0.2]$ for $La = 1200$. The orders of convergence are approximately equal to 4 for the L_∞ and standard deviation, and between 3.5 and 4 for the normal deviation, a reduction for the later that is due to the 5th order WENO advection scheme as expected by Proposition 1.

We observe in Fig. 18(a) two different regimes: the first one where the non-equilibrated geometric circle tends to find its equilibrium, followed by a second one where the errors in solving the ϕ transport equation become dominant and work against reaching the converged equilibrium as studied in §4.3. The peak capillary number converges at order 4 as observed previously in §14 while the mean v_{rms} and the mean Ca converge between the orders 4 and 5, which is notably higher than the results obtained for the static case in §4.3. We observe in Fig. 18(a) higher amplitudes for Ca in the transport error regime (compared to the initial geometry equilibrium search regime) for coarser resolutions where the grid aliasing is bigger and the order 2 Navier–Stokes errors are more present. Hence the errors on the velocities reduce even more as $h \to 0$, and converge faster toward the "optimum" reference values fixed by the static case results thus attaining an order of convergence higher than 4.

As shown in Fig. 18(c), the spatial convergences for the capillary number (similar to the RMS velocity) is at least order 4 which is 3 orders of magnitude better than the L_2 norm presented in [14]; their L_∞ errors do not show convergence while ours, through the Ca number, is better than fourth-order. This clearly demonstrates that the use of a high-order advection scheme coupled with high-order curvature extension is necessary to reduce the spurious currents in general flow simulations.

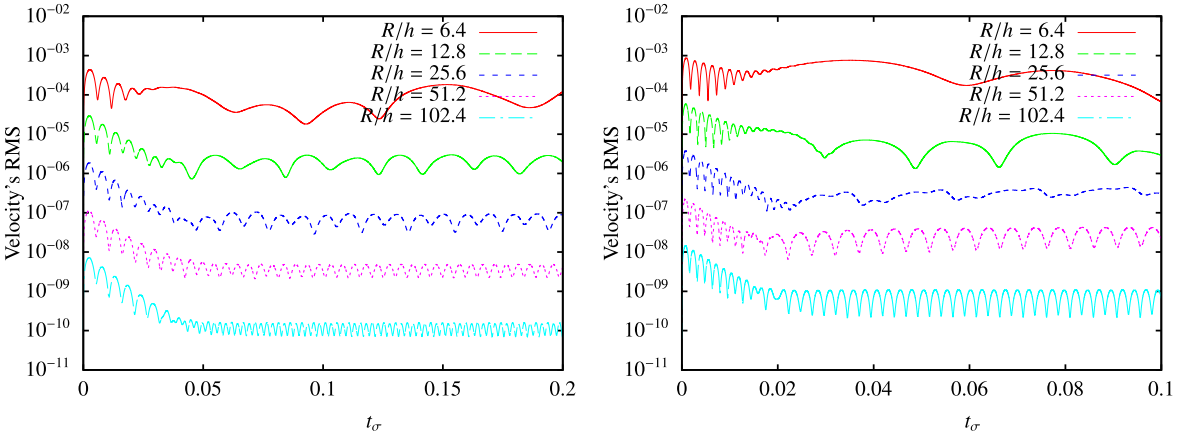
4.6. Zero gravity drop oscillation

In order to study the dynamic behavior of our method in the presence of non-constant curvature, we measured the oscillation period of a non-circular drop. The analytical results proposed in [43] were studied numerically in [12,4,44] and [45] in various frameworks.

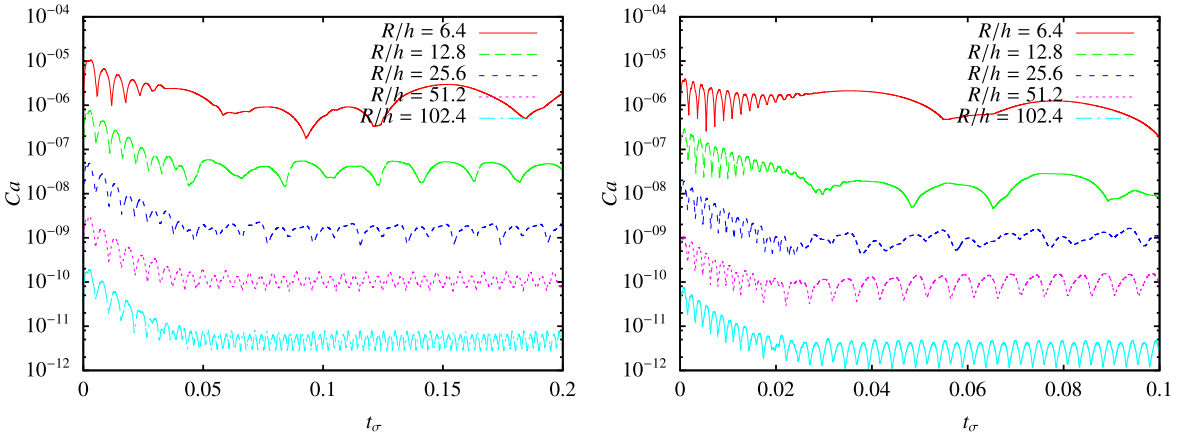
A 2D circular droplet is perturbed by a cosine function of mode n and amplitude A_0. The theoretical oscillation period ω as given by [43] is:

$$\omega^2 = \frac{n(n^2-1)\sigma}{(\rho_1+\rho_2)R_0^3}$$

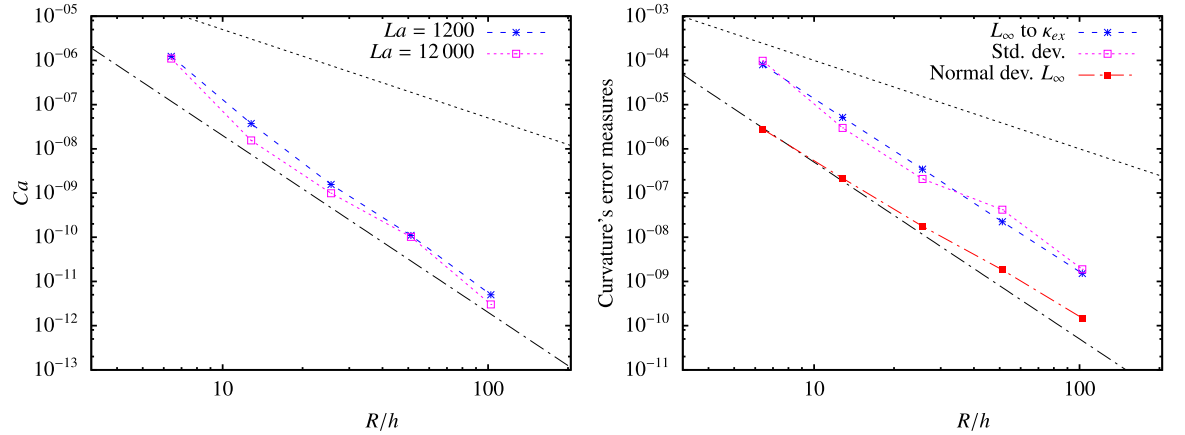
where ρ_1 (resp. ρ_2) is the density of the fluid inside the droplet (resp. outside).



(a) Velocity RMS for varying R/h, $La = 1200$ (left) and $La = 12\,000$ (right).



(b) Capillary number for varying R/h, $La = 1200$ (left) and $La = 12\,000$ (right).



(c) Left: mean Ca for $La = 1200$ and $La = 12\,000$. Right: mean curvature error measures for $La = 12\,000$. Spatial convergence with respect to R/h for mean values computed in the intevals $t_{adv} \in [0.06, 0.2]$ for $La = 1200$ and $t_{adv} \in [0.03, 0.1]$ for $La = 12\,000$.

Fig. 18. Advected column case: convergence study for the RMS velocity, Ca and curvature error measures for the $CP^2_\perp$ method and order 4 κ_{LS} computation, in the U_{adv} and T_{adv} reference frames.

Table 8
Spatial convergence of the oscillation error for the zero gravity drop oscillation, compared to the analytical solution from [43]. The value for T_{calc} at $R/h = \infty$ with our method has been computed with a Richardson's extrapolation.

E_ω	R/h					
	6.4	O	12.8	O	25.6	O
[12] Cartesian	4.04×10^{-2}	–	1.05×10^{-2}	1.94	0.37×10^{-2}	1.5
[45]	13.2×10^{-2}	–	6.1×10^{-2}	1.11	1.5×10^{-2}	2.02
Proposed method, 2nd order, eq. (31)	26.6×10^{-2}	–	8.30×10^{-2}	1.68	2.99×10^{-2}	1.47
Proposed method, 4th order, eq. (31)	6.57×10^{-2}	–	2.74×10^{-2}	1.26	0.54×10^{-2}	2.34
Proposed method, 4th order, eq. (32)	6.29×10^{-2}	–	2.68×10^{-2}	1.23	0.53×10^{-2}	2.33

T_{calc}	R/h				
	6.4	12.8	25.6	O	∞
Proposed method, 2nd order, eq. (31)	9.233	7.896	7.510	1.95	7.376
Proposed method, 4th order, eq. (31)	7.770	7.491	7.331	1.84	7.269
Proposed method, 4th order, eq. (32)	7.750	7.487	7.330	1.85	7.270
Analytical [43]	–	–	–	–	7.29138

4.6.1. General configuration

An initial column of radius $R_0 = 2$ is placed in the center of a $[-10, 10]$ square box with slip boundary conditions. The surface tension coefficient is fixed to $\sigma = 1$ and the physical parameters $\rho_1 = 1$, $\rho_2 = 0.01$, $\mu_1 = 0.01$ and $\mu_2 = 1 \cdot 10^{-4}$ which from eq. (40) corresponds to $La \approx 80\,000$. The column is perturbed by a cosine function which we translated in the level set representation as:

$$\phi(x) = |\mathbf{x}| - (R_0 + A_0 \cos(n\theta))$$

where we set $A_0 = 0.01\, R_0$ and $n = 2$. The time step was chosen as $\Delta_t = 3 \cdot 10^{-3}$.

4.6.2. Spatial convergence

Table 8 shows a comparison of the results in the literature with the results obtained by our method. The oscillation error is calculated as $E_\omega = \left| \frac{T_{calc}\omega}{2\pi} - 1 \right|$ with T_{calc} measured between the picks in the surface minimum x position, averaged over the 3 first periods. We show the impact of the precision of the Closest Point computation (with a corresponding threshold of $\epsilon = h^2$), the interpolation functions and κ_{LS} on the accuracy of the results which, for the most precise method, show an improvement by a factor of approximately 3 compared to [45] while the error is around 1.5 times higher compared to [12]. This difference can be explained by the sensitivity of these numerical results to the fluid solver and the different parameters, like for example the regularization parameter for the Heaviside function which is not given in [12] or the number of periods used for the measurements. We see an approximate second-order of convergence for the oscillation error.

As the oscillation error measures the dynamic behavior of the drop, we do not expect orders of convergence as high as the error on spurious currents. Indeed, in this more complex simulation, the errors are dominated by the flow solver which is restricted to second-order. However, we clearly observe on the error reduction the interest of computing the curvature extension up to a fourth-order accuracy. Particularly, the second-order method has higher error (3 to 5 times) compared to the fourth-order. Fig. 19 shows a better dynamical behavior when using the high order scheme, particularly for the low resolution $R/h = 6.4$ where the less accurate method leads to non-smooth oscillations with increasing amplitude. As expected, this demonstrates the clear advantage of using high order numerical methods, particularly when the curvature is large relatively to the spatial discretization.

Moreover, the theory is based on a linear regime study in an infinite fluid and so complex numerical conditions for the simulation may never converge to the analytical result but more toward a numerical equilibrium. The oscillation period extrapolated with Richardson's method on our measures is 7.270 which yields close to second-order spatial convergence, as expected by the fluid solver limitation.

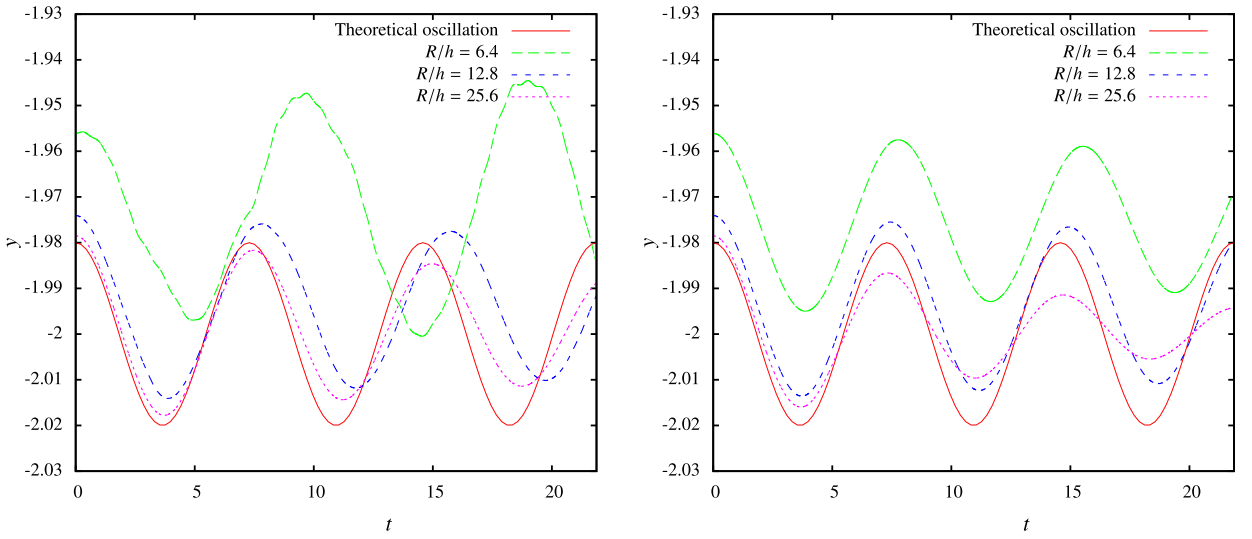
4.7. 2D bubble rise

4.7.1. General configuration

For this study we used the $CP^2_\perp$ algorithm and order 4 κ_{LS} calculation.

We consider the rise of a 2D bubble under the effect of buoyancy following [46,16] numerical studies. A disc of diameter $D = 0.5$ lies at position $[0.5, 0.5]$ at $t = 0$, inside a domain of size $[0, 0] \times [1, 2]$ with no-slip conditions on top and bottom walls and free-slip on left and right boundaries. The densities and viscosities inside (resp. outside) the bubble are ρ_2 and μ_2 (resp. ρ_1 and μ_1) and the gravity is set to 0.98.

The reference length is $L_{rise} = D = 0.5$ with an associated time scale of $T_{rise} = L_{rise}/U_{rise}$ and $U_{rise} = \sqrt{gD}$.



(a) Proposed method, 2^{nd} order, eq. 31 (b) Proposed method, 4^{th} order, eq. 32

Fig. 19. Spatial convergence of the oscillation for the zero gravity, compared to the analytical solution from [43], minimum height of the interface over time. The theoretical oscillation is plotted as reference, without the damping of the amplitude.

Table 9
2D bubble rise case: physical parameters.

Case	ρ_1	ρ_2	μ_1	μ_2	σ
A	1000	100	10	1	24.5
B	1000	1	10	0.1	1.96

We study the evolution of the center of mass of the bubble as numerically computed in the whole domain:

$$(x_c, y_c) = \frac{\sum \mathbf{x} H_\epsilon(\phi/|\nabla\phi|)}{\sum H_\epsilon(\phi/|\nabla\phi|)}.$$

The rising bubble velocity is computed in a similar fashion as:

$$(u_c, v_c) = \frac{\sum \mathbf{u} H_\epsilon(\phi/|\nabla\phi|)}{\sum H_\epsilon(\phi/|\nabla\phi|)}.$$

The circularity shows information on the deformation of the bubble surface and is written:

$$c = \frac{\pi D}{perimeter} \tag{41}$$

where the perimeter is numerically calculated with:

$$perimeter = h^2 \sum \delta_\epsilon(\phi/|\nabla\phi|). \tag{42}$$

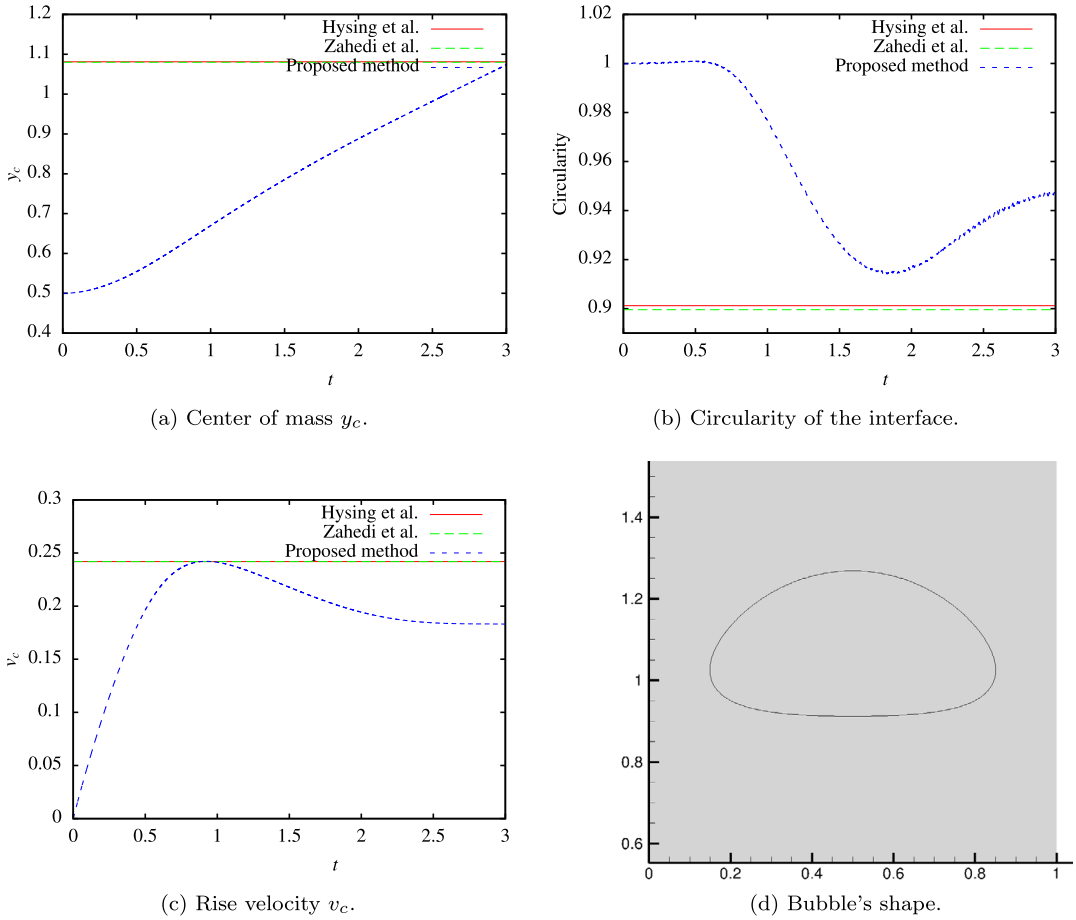
The time step used in all simulations is $\Delta t = 3 \times 10^{-4}$.

4.7.2. Spatial convergence

As seen in [16] for a finite element and level set framework, we propose to study the accuracy of the method in comparison to the numerical benchmark presented in [46] with various numerical methods. The numerical results compared in this test case have to be taken with care: they were obtained with multiple flow solvers that have different spatial and temporal discretization schemes. Moreover, the results in [46] presented here are obtained from extrapolation of numerical data, not theoretical nor experimental reference values.

The physical parameters of the two reference cases are presented in Table 9.

The numerical results for test case A are illustrated in Fig. 20 and summarized and compared to the literature in Table 10. They are in good agreement with the reference benchmark permitting us to validate our method for this rising bubble case. The maximum velocity of the bubble and the time it arises both fit the reference values and converge at the order 2. However, we observe a more important difference in the bubble circularity and center of mass between our results and



(a) Center of mass y_c.

(b) Circularity of the interface.

(c) Rise velocity v_c.

(d) Bubble's shape.

Fig. 20. 2D bubble rise case A: center of mass, circularity, rise velocity and bubble's shape at $t = 3$ for $R/h = 80$. Extremum results from [46,16] are drawn as lines.

Table 10
2D bubble rise case: numerical results compared to [46,16].

(a) Comparative results with $CP_{\perp}$ and order 4 κ_{LS}, for $R/h = 80$.

Measure	Proposed method	[16] coupled method	Ref. [46]
c_{min}	0.914	0.89955	0.9012 ± 0.0001
$t\|c = c_{min}$	1.817	1.909	1.9
$v_{c,max}$	0.2421	0.24190	0.2419 ± 0.0002
$t\|v = v_{c,max}$	0.9222	0.929	$0.921 \leq t \leq 0.932$
$y_c\|t = 3$	1.0761	1.07989	1.081 ± 0.001

(b) Results with the proposed method and the simple κ_{LS} method with no curvature extension, varying R/h. The order of convergence is computed by comparing the numerical results for the two finest discretizations to Richardson's extrapolation.

Measure	R/h				Richardson's extrapolation	O
	10	20	40	80		
Proposed method						
c_{min}	0.976	0.931	0.924	0.914	0.910	1.8
$t\|c = c_{min}$	1.769	1.934	1.806	1.817	1.826	1.2
$v_{c,max}$	0.2385	0.2411	0.2419	0.2421	0.2422	2.0
$t\|v = v_{c,max}$	0.9435	0.9282	0.9237	0.9222	0.9217	1.9
$y_c\|t = 3$	1.0587	1.0635	1.0708	1.0746	1.0761	1.8

the benchmark than in [16] for the same discretization. First, the circularity is computed in a level set framework from equations (41) and (42) which is as challenging as in a Eulerian framework it involves the regularized singular Dirac mass thus bringing oscillations because of the aliasing due to the grid. Therefore the computation of the value c_{min} and the measure of the time it takes to reach it is sensitive. We also observe a lower center of mass at $t = 3$.

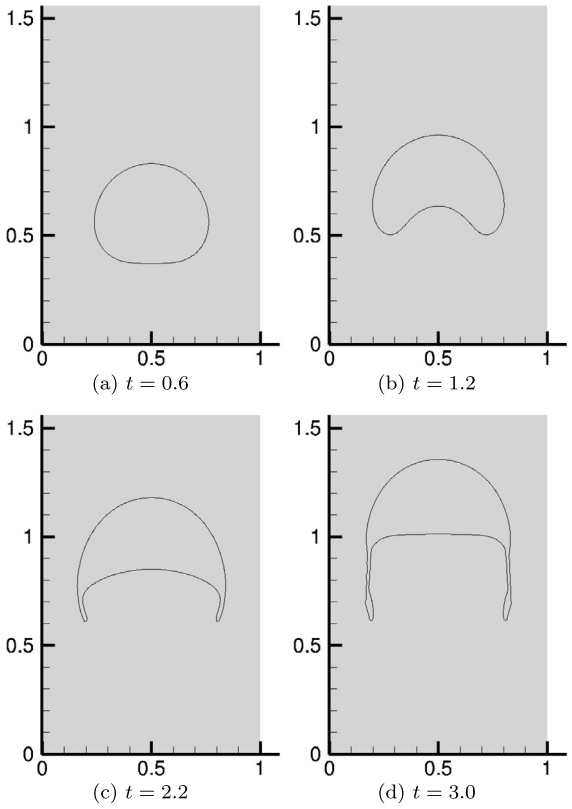


Fig. 21. 2D bubble rise case B: bubble shape for different times and for $R/h = 80$.

We believe that the numerical deviations come from the multiple differences in the numerical schemes and approximations used. We also believe that a better precision in the curvature computation at small scales with our method, as shown in §4.6, particularly on the bottom left and right corners where the curvature becomes high, helps to keep the bubble more circular showing more resistance to the vertical motion.

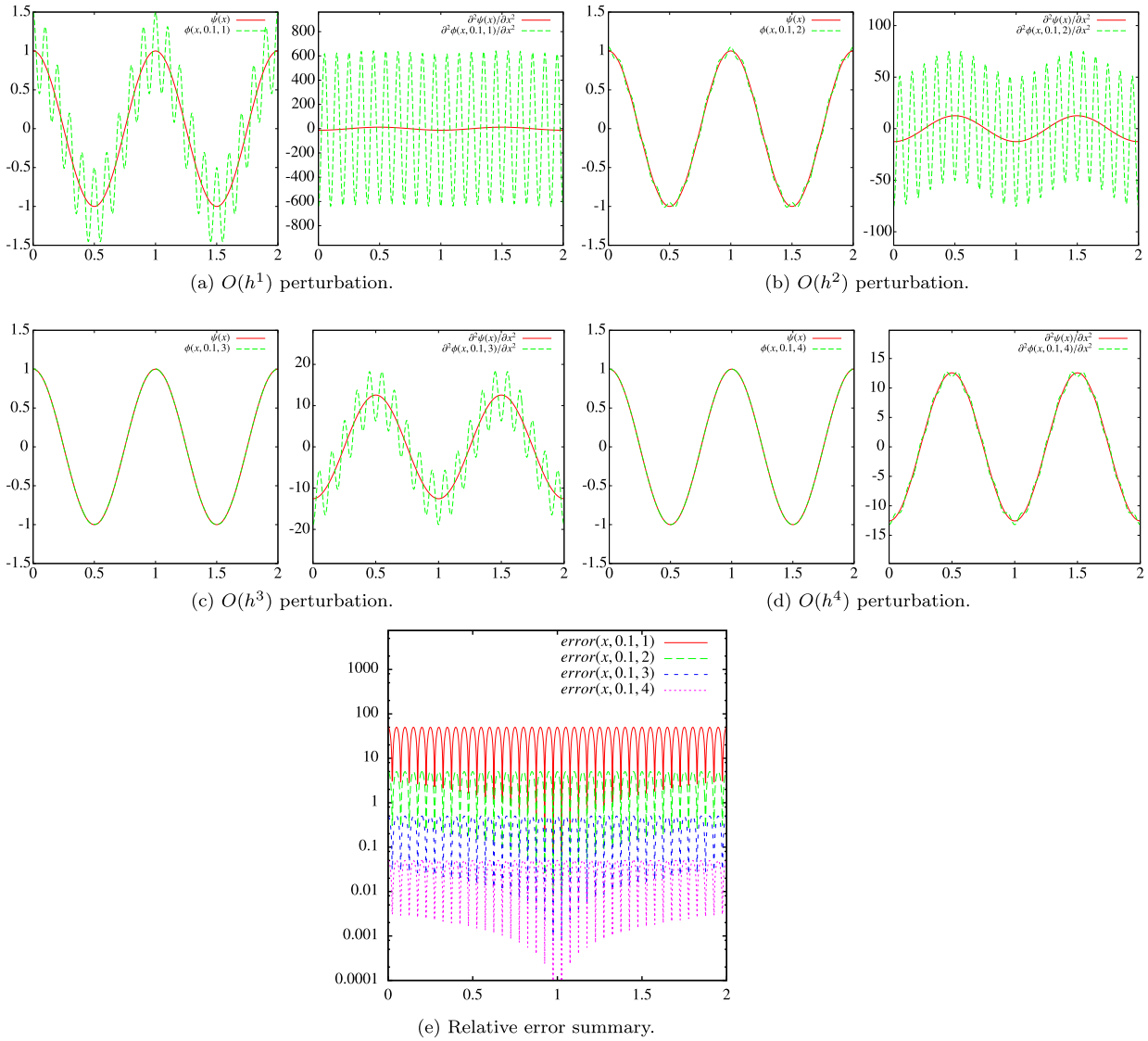
Fig. 21 shows qualitative results for test case B through the plots of the bubble's surface, simulated with a $R/h = 80$ discretization. In the early time-steps, $t < 2.2$, we see a similar behavior of this bubble to the one presented in [46]. After $t = 2.2$ we observe wider dragged filaments that resist reduction whereas in [46] the filaments disappear in the flow as residual small bubbles with the TP2D (finite elements and level set with reinitialization) method and are reduced to very thin filaments with MoonNMD (finite elements with a Lagrangian surface representation) and Fluent. We believe that the wider filaments we observe are due to:

1. The use of a regularized surface of width $\epsilon = 2h$ that smooths the blending zone and thus the dynamics of the bubble. This zone gets finer as $h \to 0$, though it does not disappear.
2. The more accurate computation of κ in high curvature areas, i.e. where the filaments arise, there is a more precise computation of the surface tension force preventing early decomposition of the surface.

5. Conclusion

In this article we present an improved method for the accurate computation of curvature in a level set framework. A complementary predicate (Criterion 2.3) concerning the normal deviation of the curvature in the CSF approach has been introduced and analyzed. We have studied the effect of perturbations on the surface's representation showing that it is necessary to use high-order transport techniques to calculate accurate and steady surface tension forces in a dynamic flow.

We illustrate the curvature calculation in a level set representation to demonstrate the necessity of an adequate curvature extension. The method proposed is based on a Closest Point algorithm improved with the colinearity criterion and reinterpolation. It yields fourth-order precision closest point approximation, enough to obtain fourth-order precision for the curvature error and standard deviation, and second-order precision for the normal error in general geometry. The algorithm is easy to implement, compared to Height Function methods in a VOF framework [18] and later works. It is also straightforward to extend to three dimensions while not being too computationally costly.



(a) $O(h^1)$ perturbation.

(b) $O(h^2)$ perturbation.

(c) $O(h^3)$ perturbation.

(d) $O(h^4)$ perturbation.

(e) Relative error summary.

Fig. 22. Second derivative errors in the presence of perturbations. Note that the y axis for the second derivatives plots have different scales.

We validate our approach on geometric, static, translating and complex dynamical cases in comparison to the literature. In particular we show that fourth-order accurate computation of curvature and surface tension forces can greatly reduce spurious currents even when the surface is transported, although the flow solver remains second-order accurate. We also observe good agreement of our numerical results with state-of-the-art complex simulations like the rise of a bubble.

Acknowledgements

This study was carried out with financial support from the French National Research Agency (ANR) in the framework of the “Investments for the future” Programme IdEx Bordeaux (ANR-10-IDEX-03-02), Cluster of excellence CPU.

The authors wish to thank the Aquitaine Regional Council for the financial support towards a 432-processor cluster investment, located in the I2M laboratory. Computer time for this study was also provided by the computing facilities MCIA (Mesocentre de Calcul Intensif Aquitain) of the Université de Bordeaux and of the Université de Pau et des Pays de l’Adour.

Appendix A

A.1. Second derivative error amplitude

In this paragraph we study the effect of numerical errors on derivatives through the introduction of perturbations in an analytical smooth function ψ. Let ψ, e and ϕ be three $\mathbb{R} \to \mathbb{R}$ functions defined as:

$$\psi(x) = \cos(2\pi x),$$
$$e(x,h,m) = h^m \cos(2\pi x/h),$$
$$\phi(x,h,m) = \cos(2\pi x) + h^m \cos(2\pi x/h),$$

as represented in Fig. 22. We can easily calculate the second derivative of ϕ:

$$\frac{\partial^2\phi}{\partial x^2} = -4\pi^2(\sin(2\pi x) + h^{m-2}\sin(2\pi x/h))$$

which is equal to $\frac{\partial^2\psi}{\partial x^2}$ perturbed at the scale h with an amplitude of the order h^{m-2}.

The error in the second derivative is measured by:

$$error(x,h,m) = \left|\frac{\partial^2\phi/\partial x^2 - \partial^2\psi/\partial x^2}{\partial^2\psi/\partial x^2}\right|$$

when $\partial^2\psi/\partial x^2 \neq 0$ and is plotted in Fig. 22 for different perturbations scales. We clearly see that perturbations bigger than h^2 lead to errors superior to 1 on the second derivative which is unpropitious in numerical computations.

A.2. Detailed numerical results

Table 11
Static column case: numerical results in function of R/h and order of convergence for the $CP^2_\perp$ method, order 2 and order 4 κ_{LS} calculation, $La = 120$.

(a) Velocity errors measures and pressure error.

κ_{LS}	R/h	Ca at $t = \Delta t$	O	Ca at $t_\sigma = 30$	O	max Ca	O	$E(\Delta p_{mean})$ at $t_\sigma = 30$	O
Order 2 scheme	6.4	2.02×10^{-6}	–	6.59×10^{-9}	–	4.83×10^{-5}	–	3.05×10^{-4}	–
	12.8	6.60×10^{-7}	1.62	1.96×10^{-10}	5.07	1.38×10^{-5}	1.81	6.71×10^{-5}	2.18
	25.6	1.82×10^{-7}	1.86	9.73×10^{-12}	4.33	3.60×10^{-6}	1.94	1.60×10^{-5}	2.07
	51.2	4.72×10^{-8}	1.95	4.09×10^{-13}	4.57	9.03×10^{-7}	1.99	3.96×10^{-6}	2.01
	102.4	1.19×10^{-8}	1.98	1.34×10^{-14}	4.93	2.27×10^{-7}	1.99	9.91×10^{-7}	2.00
Order 4 scheme	6.4	1.50×10^{-6}	–	5.26×10^{-9}	–	2.43×10^{-5}	–	2.63×10^{-5}	–
	12.8	1.30×10^{-7}	3.53	1.47×10^{-10}	5.16	1.67×10^{-6}	3.87	3.55×10^{-6}	2.89
	25.6	9.29×10^{-9}	3.81	7.58×10^{-12}	4.28	1.05×10^{-7}	3.98	1.75×10^{-7}	4.35
	51.2	6.20×10^{-10}	3.90	4.11×10^{-13}	4.21	6.47×10^{-9}	4.03	6.52×10^{-9}	4.74
	102.4	3.74×10^{-11}	4.05	2.08×10^{-14}	4.30	4.13×10^{-10}	3.97	4.24×10^{-10}	3.94

(b) Curvature error measures at $t_\sigma = 30$.

κ_{LS}	R/h	$\lvert\frac{\kappa_{mean}-\kappa_{ex}}{\kappa_{ex}}\rvert$	O	L_∞ curvature error	O	κ_σ	O	L_∞ normal dev.	O
Order 2 scheme	6.4	3.05×10^{-4}	–	3.10×10^{-4}	–	1.01×10^{-5}	–	2.65×10^{-6}	–
	12.8	6.71×10^{-5}	2.19	6.73×10^{-5}	2.20	4.60×10^{-7}	4.45	1.18×10^{-7}	4.49
	25.6	1.60×10^{-5}	2.07	1.60×10^{-5}	2.07	4.54×10^{-8}	3.34	1.14×10^{-8}	3.36
	51.2	3.96×10^{-6}	2.01	3.96×10^{-6}	2.01	2.45×10^{-9}	4.21	5.71×10^{-10}	4.32
	102.4	9.91×10^{-7}	2.00	9.91×10^{-7}	2.00	6.31×10^{-11}	5.28	1.68×10^{-11}	5.09
Order 4 scheme	6.4	2.66×10^{-5}	–	2.97×10^{-5}	–	6.92×10^{-6}	–	1.85×10^{-6}	–
	12.8	3.56×10^{-6}	2.90	3.66×10^{-6}	3.02	2.39×10^{-7}	4.86	6.52×10^{-8}	4.82
	25.6	1.75×10^{-7}	4.35	1.83×10^{-7}	4.32	1.80×10^{-8}	3.73	6.96×10^{-9}	3.23
	51.2	6.51×10^{-9}	4.74	6.90×10^{-9}	4.73	7.02×10^{-10}	4.68	3.02×10^{-10}	4.53
	102.4	4.24×10^{-10}	3.94	4.47×10^{-10}	3.95	2.30×10^{-11}	4.93	1.27×10^{-11}	4.57

References

[1] A. Yarin, D. Weiss, Impact of drops on solid surfaces: self-similar capillary waves, and splashing as a new type of kinematic discontinuity, J. Fluid Mech. 283 (1995) 141–173.
[2] D. Bhaga, M. Weber, Bubbles in viscous liquids: shapes, wakes and velocities, J. Fluid Mech. 105 (1981) 61–85.
[3] R. Clip, J. Grace, M. Weber, Bubbles, Drops, and Particles, Academic Press, 1978.
[4] G. Tryggvason, B. Bunner, O. Ebrat, W. Tauber, Computations of multiphase flows by a finite difference/front tracking method. I. Multi-fluid flows, in: Lecture Series – von Karman Institute For Fluid Dynamics, 1998, 7-7.
[5] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, S. Zaleski, Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows, J. Comput. Phys. 152 (2) (1999) 423–456.
[6] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159.
[7] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (1) (2002) 83–116.

[8] 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 (2) (2000) 301–337.
[9] G.-H. Cottet, J.-M. Etancelin, F. Pérignon, C. Picard, High order semi-Lagrangian particle methods for transport equations: numerical analysis and implementation issues, ESAIM: Math. Model. Numer. Anal. 48 (4) (2014) 1029–1060, http://dx.doi.org/10.1051/m2an/2014009, http://www.esaim-m2an.org/article_S0764583X14000090.
[10] J. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (2) (1992) 335–354.
[11] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, M.W. Williams, A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, J. Comput. Phys. 213 (1) (2006) 141–173.
[12] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, J. Comput. Phys. 227 (4) (2008) 2674–2706.
[13] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, Int. J. Numer. Methods Fluids 30 (6) (1999) 775–793.
[14] S. Popinet, An accurate adaptive solver for surface–tension–driven interfacial flows, J. Comput. Phys. 228 (16) (2009) 5838–5866.
[15] S. Shin, S. Abdel-Khalik, V. Daru, D. Juric, Accurate representation of surface tension using the level contour reconstruction method, J. Comput. Phys. 203 (2) (2005) 493–516.
[16] S. Zahedi, M. Kronbichler, G. Kreiss, Spurious currents in finite element based level set methods for two-phase flow, Int. J. Numer. Methods Fluids 69 (9) (2012) 1433–1456.
[17] F. Denner, B.G.M. van Wachem, Fully-coupled balanced-force VoF framework for arbitrary meshes with least-squares curvature evaluation from volume fractions, Numer. Heat Transf., Part B, Fundam. 65 (3) (2014) 218–255, http://dx.doi.org/10.1080/10407790.2013.849996.
[18] S.J. Cummins, M.M. Francois, D.B. Kothe, Estimating curvature from volume fractions, Comput. Struct. 83 (6) (2005) 425–434.
[19] W. Aniszewski, T. Ménard, M. Marek, Volume of fluid (VoF) type advection methods in two-phase flow: a comparative study, Comput. Fluids 97 (2014) 52–73.
[20] V. Dyadechko, M. Shashkov, Reconstruction of multi-material interfaces from moment data, J. Comput. Phys. 227 (11) (2008) 5361–5384, http://dx.doi.org/10.1016/j.jcp.2007.12.029.
[21] M. Jemison, E. Loch, M. Sussman, M. Shashkov, M. Arienti, M. Ohta, Y. Wang, A coupled level set-moment of fluid method for incompressible two-phase flows, J. Sci. Comput. 54 (2–3) (2013) 454–491, http://dx.doi.org/10.1007/s10915-012-9614-7.
[22] C.B. Macdonald, S.J. Ruuth, Level set equations on surfaces via the closest point method, J. Sci. Comput. 35 (2–3) (2008) 219–240.
[23] G. Bornia, A. Cervone, S. Manservisi, R. Scardovelli, S. Zaleski, On the properties and limitations of the height function method in two-dimensional Cartesian geometry, J. Comput. Phys. 230 (4) (2011) 851–862.
[24] S. Bnà, S. Manservisi, R. Scardovelli, P. Yecko, S. Zaleski, Numerical integration of implicit functions for the initialization of the VoF function, Comput. Fluids 113 (2015) 42–52.
[25] Q. Zhang, On a family of unsplit advection algorithms for volume-of-fluid methods, SIAM J. Numer. Anal. 51 (5) (2013) 2822–2850.
[26] L.C. Evans, J. Spruck, et al., Motion of level sets by mean curvature I, J. Differ. Geom. 33 (3) (1991) 635–681.
[27] P. Liovic, M. Francois, M. Rudman, R. Manasseh, Efficient simulation of surface tension-dominated flows through enhanced interface geometry interrogation, J. Comput. Phys. 229 (19) (2010) 7520–7544.
[28] A. Poux, S. Glockner, M. Azaïez, Improvements on open and traction boundary conditions for Navier–Stokes time-splitting methods, J. Comput. Phys. 230 (10) (2011) 4011–4027.
[29] P. Lubin, S. Glockner, Numerical simulations of three-dimensional plunging breaking waves: generation and evolution of aerated vortex filaments, J. Fluid Mech. 767 (2015) 364–393, http://dx.doi.org/10.1017/jfm.2015.62, http://journals.cambridge.org/article_S0022112015000622.
[30] K. Goda, A multistep technique with implicit difference schemes for calculating two-or three-dimensional cavity flows, J. Comput. Phys. 30 (1) (1979) 76–95.
[31] P.R. Amestoy, I.S. Duff, J.-Y. L'Excellent, J. Koster, A fully asynchronous multifrontal solver using distributed dynamic scheduling, SIAM J. Matrix Anal. Appl. 23 (1) (2001) 15–41.
[32] P.R. Amestoy, A. Guermouche, J.-Y. L'Excellent, S. Pralet, Hybrid scheduling for the parallel solution of linear systems, Parallel Comput. 32 (2) (2006) 136–156.
[33] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1) (1996) 202–228.
[34] B. Engquist, A.-K. Tornberg, R. Tsai, Discretization of Dirac delta functions in level set methods, J. Comput. Phys. 207 (1) (2005) 28–51.
[35] F. Denner, B. van Wachem, On the convolution of fluid properties and surface force for interface capturing methods, Int. J. Multiph. Flow 54 (2013) 61–64, http://dx.doi.org/10.1016/j.ijmultiphaseflow.2013.03.004, http://www.scopus.com/inward/record.url?eid=2-s2.0-84876485278&partnerID=40&md5=3826186e24b1a52c1d56c3ed4210fe65.
[36] C. Min, F. Gibou, Robust second-order accurate discretizations of the multi-dimensional Heaviside and Dirac delta functions, J. Comput. Phys. 227 (22) (2008) 9686–9695.
[37] G.-H. Cottet, E. Maitre, A level set method for fluid–structure interactions with immersed surfaces, Math. Models Methods Appl. Sci. 16 (3) (2006) 415–438.
[38] S.J. Ruuth, B. Merriman, A simple embedding method for solving partial differential equations on surfaces, J. Comput. Phys. 227 (3) (2008) 1943–1961.
[39] F. Denner, D.R. van der Heul, G.T. Oud, M.M. Villar, A. da Silveira Neto, B.G. van Wachem, Comparative study of mass-conserving interface capturing frameworks for two-phase flows with surface tension, Int. J. Multiph. Flow 61 (2014) 37–47.
[40] D. Fuster, G. Agbaglah, C. Josserand, S. Popinet, S. Zaleski, Numerical simulation of droplets, bubbles and waves: state of the art, Fluid Dyn. Res. 41 (6) (2009) 065001.
[41] F. Denner, B. van Wachem, Numerical time-step restrictions as a result of capillary waves, J. Comput. Phys. 285 (2015) 24–40, http://dx.doi.org/10.1016/j.jcp.2015.01.021, http://www.scopus.com/inward/record.url?eid=2-s2.0-84925115207&partnerID=40&md5=0e733960a4f87c2deb45cbfffc0142d9.
[42] M. Owkes, O. Desjardins, A mesh-decoupled height function method for computing interface curvature, J. Comput. Phys. 281 (2015) 285–300, http://dx.doi.org/10.1016/j.jcp.2014.10.036, http://www.sciencedirect.com/science/article/pii/S0021999114007189.
[43] S.H. Lamb, Hydrodynamics, Dover Books, 1945.
[44] S.V. Shepel, B.L. Smith, On surface tension modelling using the level set method, Int. J. Numer. Methods Fluids 59 (2) (2009) 147–171.
[45] D. Torres, J. Brackbill, The point-set method: front-tracking without connectivity, J. Comput. Phys. 165 (2) (2000) 620–644.
[46] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, L. Tobiska, Quantitative benchmark computations of two-dimensional bubble dynamics, Int. J. Numer. Methods Fluids 60 (11) (2009) 1259–1288.