A robust 3D crack growth method based on the eXtended Finite Element Method and the Fast Marching Method

M. Le Cren, A. Martin, P. Massin, Nicolas Moes

Int J of Fract manuscript No.
(will be inserted by the editor)

A robust 3D crack growth method based on the eXtended Finite Element Method and the Fast Marching Method

M. Le Cren · A. Martin · P. Massin · N. Moës

Received: date / Accepted: date

Abstract In the context of the eXtended Finite Element Method (X-FEM), the use of two level set functions allows the representation of the crack to be achieved regardless of the mesh. The initial crack geometry is represented by two distinct level set functions, and the crack propagation is simulated by an update of these two level set functions. In this paper, we propose a new approach, based on the Fast Marching Method (FMM), to update the level set functions. We also propose a new implementation of the FMM, designed for tetrahedral volume meshes. We then extend this method to all types of volume elements (tetrahedra, hexahedra, pentahedra, pyramids) available in a standard finite element library. The proposed approach allows one to use the same mesh to solve the mechanical problem and to update the level set functions. Non-planar quasi-static crack growth simulations are presented to demonstrate the robustness of the approach, compared to existing methods based on the integration of Hamilton-Jacobi equations or geometric approaches.

Keywords Level sets · Signed distance functions · Fast Marching Method · Crack propagation

M. Le Cren · A. Martin · P. Massin
IMSIA, UMR EDF/CNRS/CEA/ENSTA 9219, 828 Boulevard des Maréchaux, 91762 Palaiseau Cedex, France

A. Martin (✉)
Université Clermont Auvergne, CNRS, LMBP, F-63000 Clermont-Ferrand, France
E-mail: alexandre.martin@uca.fr

N. Moës
École Centrale de Nantes, GéM, UMR CNRS/ECN/UN 6183, 1 Rue de La Noë, 44321 Nantes, France

N. Moës
Institut Universitaire de France, 1 rue Descartes, 75005 Paris, France

1 Introduction

The level set method coupled with X-FEM (Belytschko and Black 1999; Moes et al. 1999) is very effective to simulate 2D (Stolarska et al. 2001) and 3D (Moes et al. 2002; Gravouil et al. 2002) crack growth. The initial crack is represented by two level set functions: the faces of the crack belong to the zero level set of a first level set function while the second level set function is defined in such a way the intersection of the zero level set of the two level set functions describes the crack front. The two level set functions are signed distance functions. The two level set functions are orthogonal in the sense that their gradients are orthogonal. The growth of the crack is discretized in increments. Linear elastic fracture mechanics is used to describe the displacement of a point of the crack front from its position at the beginning of the growth to its position at the end of the growth increment. At the end of each growth increment, the level set functions are updated to describe the new crack position.

The udpate of both level set level set functions is based on the level set method introduced by Osher and Sethian (1988). The procedure proposed by Gravouil et al. (2002) is as follows. In a first step, called the extension step, the displacement field is first extended to a neighborhood of the zero level set. In a second step, called the update step, the Hamilton-Jacobi equation describing the evolution of each level set function is integrated. The intersection of the zero level set of the two functions obtained after the update step describes the new position of the crack front. These functions are not expected to be true signed functions. The last step consists of two interlocked steps, called the reinitialization step and the orthogonalization step. The reinitialization step builds two new level set functions, from the

functions obtained at the end of the update step. The orthogonalization step ensures the two updated level set functions are orthogonal.

Diffrent approaches to perform each of the four steps of the update of the two level set functions have been explored. Gravouil et al. (2002) proposed to integrate Hamilton-Jacobi equations to perform these four steps. Sukumar et al. (2003, 2008) and Shi et al. (2010) formulated the extension and the reinitialization steps in terms of eikonal equations and used the Fast Marching Method, introduced by Sethian (1996). Prabel et al. (2007) performed the four steps of the update of the two level set functions on a dedicated regular grid, in order to use an upwind scheme to integrate Hamilton-Jacobi equations. Colombo and Massin (2011) introduced a geometric approach to perform the extension and update steps. Colombo (2012) extended this geometric approach to perform also the reinitialization and orthogonalization steps, in order to obtain a more robust method.

The developments are implemented in code_aster finite element package, developped by EDF (1989–2021). Three methods are available to perform the reinitialization and orthogonalization steps. Nevertheless, none of these methods fit our desire of a robust tool, able to perform the mixed mode propagation of a crack in an industrial structure and, for example, the Brokenshire test (see Barr and Brokenshire (1996)) is still a challenge.

In this paper, we propose a new approach in which the geometric approach proposed by Colombo (2012) is used for the orthogonalizaton step and the Fast Marching Method for the reinitialization step. Our implementation of this approach is designed for triangulated meshes. We then extend this method to all types of (linear) volume elements available in a standard finite element library.

2 Level set update to model the growth of a crack

The growth of the crack is discretized in growth increments. At the begining of a growth increment k, the crack Γ^k is described by two level set functions : the normal level set function Φ_n^k and the tangential level set function Φ_t^k. The crack faces are described by the set of points:

$$\Gamma^k = \{M, \Phi_n^k(M) = 0\} \cap \{M, \Phi_t^k(M) < 0\}. \tag{1}$$

The crack front Γ_0^k is discretized by linear segments (*cf.* Fig. 1). Let P^i be a point on the crack front. The

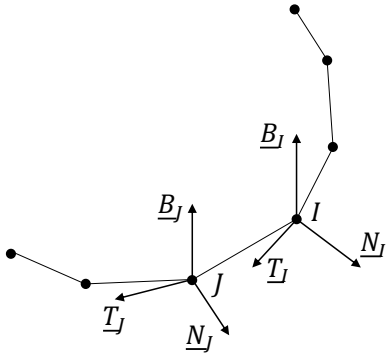


Fig. 1 Discretized crack front and local frame associated with each point on the crack front.

local frame $(\mathbf{T}^i, \mathbf{N}^i, \mathbf{B}^i)$ is computed from the level set functions Φ_n^k and Φ_t^k:

$$\begin{aligned}
\mathbf{B}^i &= \frac{\boldsymbol{\nabla}\Phi_n^k(P^i)}{\|\boldsymbol{\nabla}\Phi_n^k(P^i)\|}, \\
\mathbf{T}^i &= \frac{\boldsymbol{\nabla}\Phi_t^k(P^i) \times \boldsymbol{\nabla}\Phi_n^k(P^i)}{\|\boldsymbol{\nabla}\Phi_t^k(P^i) \times \boldsymbol{\nabla}\Phi_n^k(P^i)\|}, \\
\mathbf{N}^i &= \mathbf{B}^i \times \mathbf{T}^i.
\end{aligned} \tag{2}$$

The standard finite element interpolation of the level set functions is used to compute $\boldsymbol{\nabla}\Phi_n^k(P^i)$ and $\boldsymbol{\nabla}\Phi_t^k(P^i)$.

The mechanical fields are computed by means of X-FEM. We typically use an energetic approach to compute the energy release rate G^i and stress intensity factors $\mathrm{K}_{\mathrm{I}}^i$, $\mathrm{K}_{\mathrm{II}}^i$ and $\mathrm{K}_{\mathrm{III}}^i$. G^i is then computed using the domain integral method (Destuynder and Djaoua 1981; Li et al. 1985) while the stress intensity factors are computed using interaction integrals (Gosz and Moran 2002). We can also use the displacement jump extrapolation technique (Chan et al. 1970) to compute $\mathrm{K}_{\mathrm{I}}^i$, $\mathrm{K}_{\mathrm{II}}^i$ and $\mathrm{K}_{\mathrm{III}}^i$ and apply Irwin's formula (Irwin 1957) to obtain G^i. These quantities are used to determine the crack growth size and crack growth direction.

Following Gravouil et al. (2002) and Sukumar et al. (2008), we assume a fatigue crack growth law and we use a modified Paris' law (Paris and Erdogan 1963), in which G plays the role of the stress intensity range $\Delta\mathrm{K}$. The maximal crack growth size during a growth increment $\Delta a^{\max}$ is an input parameter of the simulation. The value of $\Delta a^{\max}$ corresponds to the typical element size in the neighborhood of the crack. The crack growth size Δa^i associated to the point P^i on the crack front is:

$$\Delta a^i = \left(\frac{G^i}{G^{\max}}\right)^m \Delta a^{\max}, \tag{3}$$

where $G^{\max}$ is the maximum of the energy release rate G, for all the points of the crack front, and C and m are the parameters of Paris' law.

The maximum hoop stress criterion is used to find the crack growth direction. The angle β of the crack growth direction with respect to the plane tangent to the crack is obtained by (Erdogan and Sih 1963):

$$\beta = 2\tan^{-1}\left[\frac{1}{4}\left(\frac{\mathrm{K_I}}{\mathrm{K_{II}}} - \frac{\mathrm{K_{II}}}{|\mathrm{K_{II}}|}\sqrt{\left(\frac{\mathrm{K_I}}{\mathrm{K_{II}}}\right)^2 + 8}\right)\right]. \quad (4)$$

A planar crack growth will correspond to assume $\beta = 0$. Finally, the displacement of the point P^i on the crack front is given by the vector:

$$\mathbf{\Delta a}^i = \Delta a^i(\cos\beta^i \mathbf{N}^i + \sin\beta^i \mathbf{B}^i), \quad (5)$$

where β^i is the angle β evaluated at point P^i.

At the end of the growth increment, the point Q^i, defined by:

$$\mathbf{OQ}^i = \mathbf{OP}^i + \mathbf{\Delta a}^i, \quad (6)$$

lies on the new crack front Γ_0^{k+1}. The problem is now to build new level set functions Φ_n^{k+1} and Φ_t^{k+1} so that the new crack front Γ_0^{k+1} can be described as the intersection of the zero level set of Φ_n^{k+1} and the zero level set of Φ_t^{k+1}.

2.1 PDE-based approaches: Simplex and Upwind Methods

We now expose how the extension, update, reinitialization and orthogonalization steps are implemented for Simplex and Upwind Methods. The general scheme follows the approach proposed by Colombo and Massin (2011).

2.1.1 Extension step

The extension step extends the displacement defined on the crack front to a neighbourhood of the crack front. This neighbourhood is the domain in which the level set functions will be updated during the update step. The simplest approach consists in updating the level set functions in the whole domain. One can also restrict the updated domain, in order to save computational time.

The first step of the extension consists in mapping each point of the domain to a point on the crack front, by projecting the point of the domain onto the crack-front. This first step is always performed for all the nodes in the domain, because it enables the computation of the distance from each node of the mesh to the crack front. This distance can then be used to restrict the domain in which the level set functions will be udpdated to a torus surrounding the crack front, following the procedure described in (Colombo and Massin 2011).

Let M be a node in the domain. The projection algorithm finds point P on the crack front Γ_0^k such that point M belongs to the plane $(P, \mathbf{N}, \mathbf{B})$. The algorithm computes point P and segment $[P^i P^j]$ to which P belongs. The displacement of point P reads:

$$\mathbf{\Delta a}^P = \Delta a(\cos\beta \mathbf{N} + \sin\beta \mathbf{B}). \quad (7)$$

The computation of crack growth size Δa, angle β and local frame $(\mathbf{T}, \mathbf{N}, \mathbf{B})$ at point P, from the data known at points P^i and P^j, is discussed in detail in (Colombo and Massin 2011).

Displacement at point M is computed from displacement at point P. Displacement Δa_n^P is associated to the displacement of the zero level set of Φ_n and displacement Δa_t^P is associated to the displacement of the zero level set of Φ_t. Displacement at point M is decomposed as:

$$\mathbf{\Delta a}^M = \mathbf{\Delta a}_n^M + \mathbf{\Delta a}_t^M, \quad (8)$$

where $\mathbf{\Delta a}_n^M$ is associated to the displacement of the zero level set of Φ_n and $\mathbf{\Delta a}_t^M$ is associated to the displacement of the zero level set of Φ_t. Displacement $\mathbf{\Delta a}_n^M$ is zero where Φ_t^k is negative to ensure $\Gamma^k \subset \Gamma^{k+1}$ at the end of the growth increment. Displacement $\mathbf{\Delta a}_n^M$ where Φ_t^k is positive is computed assuming $\|\mathbf{\Delta a}_n\|$ varies linearly from $\|\mathbf{\Delta a}_n\| = 0$ where $\Phi_t^k = 0$ (*i.e.* on the crack front Γ_0^k) to $\|\mathbf{\Delta a}_n\| = \Delta a \sin\beta$ where $\Phi_t^k = \Delta a \cos\beta$.

2.1.2 Update step

We use the geometrical approach proposed in (Colombo and Massin 2011). The updated level set functions at point M read:

$$\begin{aligned} \widetilde{\Phi}_n(M) &= \Phi_n^k(M) - \mathbf{\Delta a}_n^M \cdot \boldsymbol{\nabla}\Phi_n^k(M), \\ \widetilde{\Phi}_t(M) &= \Phi_n^k(M) - \mathbf{\Delta a}_t^M \cdot \boldsymbol{\nabla}\Phi_t^k(M). \end{aligned} \quad (9)$$

Updated level set functions $\widetilde{\Phi}_n$ and $\widetilde{\Phi}_t$ are neither expected to be signed distance functions nor expected to be orthogonal.

2.1.3 Reinitialization and orthogonalization steps

The reinitialization and orthogonalization steps are interlocked. Actually, in the case of a non-planar crack, it is not possible to fulfill the orthogonality and unit gradient conditions at the same time, except in the neighborhood of the crack front. We use the following procedure:

	Simplex Method	Upwind Method	Geometric Method	Fast Marching Method
Extension step	Projection of the nodes onto the crack front.	Projection of the nodes onto the crack front.	Projection of the nodes onto the crack front.	Projection of the nodes onto the crack front.
Update step	Geometrical approach.	Geometrical approach.	Implicit in the fully geometric method.	Geometrical approach.
Reinitialization step (Φ_n)	Integration of a time-dependent PDE; solver dedicated to triangulated meshes; boundary condition: $\Phi_n = \Phi_n^{k+1}$ at nodes of $\mathcal{I}_n$.	Integration of a time-dependent PDE; solver dedicated to regular meshes; boundary condition: $\Phi_n = \Phi_n^{k+1}$ at nodes of $\mathcal{I}_n$.	Implicit in the fully geometric method.	Integration of a stationnary PDE; solver dedicated to triangulated meshes; boundary condition: $\Phi_n = \Phi_n^{k+1}$ at nodes of $\mathcal{I}_n$.
Orthogonalization step	Integration of a time-dependent PDE; solver dedicated to triangulated meshes; boundary condition: $\Phi_t = \widehat{\Phi}_t$ at nodes of $\mathcal{I}_n$.	Integration of a time-dependent PDE; solver dedicated to regular meshes; boundary condition: $\Phi_t = \widehat{\Phi}_t$ at nodes of $\mathcal{I}_n$.	Implicit in the fully geometric method.	Replaced by the computation of Φ_t^{k+1} at nodes of $\mathcal{I}_t$, using the fully geometric method.
Reinitialization step (Φ_t)	Integration of a time-dependent PDE; solver dedicated to triangulated meshes; boundary condition: $\Phi_t = \Phi_t^{k+1}$ at nodes of $\mathcal{I}_t$.	Integration of a time-dependent PDE; solver dedicated to regular meshes; boundary condition: $\Phi_t = \Phi_t^{k+1}$ at nodes of $\mathcal{I}_t$.	Implicit in the fully geometric method.	Integration of a stationnary PDE; solver dedicated to triangulated meshes; boundary condition: $\Phi_t = \Phi_t^{k+1}$ at nodes of $\mathcal{I}_t$.

Table 1 Comparison of the methods to update the level set functions studied in this article: Simplex Method, Upwind Method, Geometric Method and the new approach based on the Fast Marching Method.

1. *Reinitialization of the normal level set function.* The normal level set function at the end of growth increment Φ_n^{k+1} is computed as the steady-state solution of the following evolution problem, with respect to a virtual time τ:

$$\frac{\partial \Phi_n}{\partial \tau} = -\frac{\Phi_n}{|\Phi_n|}(\|\boldsymbol{\nabla}\Phi_n\| - 1), \tag{10}$$

with initial data:

$$\Phi_n(\tau = 0) = \widetilde{\Phi}_n. \tag{11}$$

Let $\mathcal{I}_n$ be the set of nodes belonging to cells which contain the zero level set of $\widetilde{\Phi}_n$. Level set function Φ_n^{k+1} is built so that the zero level set of Φ_n^{k+1} coincides with the zero level set of $\widetilde{\Phi}_n$. Thus, level set function Φ_n^{k+1} at a given node of set $\mathcal{I}_n$ is the distance from this node to the zero level set of $\widetilde{\Phi}_n$, where $\widetilde{\Phi}_n$ is positive, and the opposite of the distance from this node to the zero level set of $\widetilde{\Phi}_n$, where $\widetilde{\Phi}_n$ is negative. The computation of the distance from each node of $\mathcal{I}_n$ to the zero level set of $\widetilde{\Phi}_n$ is based on a projection algorithm proposed in (Colombo and Massin 2011).

2. *Orthogonalization.* A level set function $\widehat{\Phi}_t$ orthognal to Φ_n^{k+1} is computed as the steady-state solution of the following evolution problem:

$$\frac{\partial \Phi_t}{\partial \tau} = -\frac{\Phi_n^{k+1}}{|\Phi_n^{k+1}|}\frac{\boldsymbol{\nabla}\Phi_n^{k+1}}{\|\boldsymbol{\nabla}\Phi_n^{k+1}\|} \cdot \boldsymbol{\nabla}\Phi_t, \tag{12}$$

with initial data:

$$\Phi_t(\tau = 0) = \widetilde{\Phi}_t. \tag{13}$$

We use the result of the projection algorithm presented above to compute the value of the level set function $\widehat{\Phi}_t$ at nodes of $\mathcal{I}_n$. The tangential level set function $\widetilde{\Phi}_t$ is first interpolated at the vertices of the triangulation of the zero level set of $\widetilde{\Phi}_n$. Let M be a node in $\mathcal{I}_n$. The projection algorithm associates to M a triangle $\mathcal{T}$ and a point $P^{\mathcal{T}}$ in $\mathcal{T}$. The value of the level set function $\widehat{\Phi}_t$ at M is computed as the linear interpolation of the tangential level set function values computed at the vertices of the triangle $\mathcal{T}$ evaluated at point $P^{\mathcal{T}}$.

3. *Reinitialization of the tangential level set function.* The tangential level set function at the end of the

growth increment Φ_t^{k+1} is computed as the steady-state solution of the following evolution problem:

$$\frac{\partial \Phi_t}{\partial \tau} = -\frac{\Phi_t}{|\Phi_t|}(\|\boldsymbol{\nabla}\Phi_t\| - 1), \tag{14}$$

with initial data:

$$\Phi_t(\tau = 0) = \widehat{\Phi}_t. \tag{15}$$

Let $\mathcal{I}_t$ be the set of nodes belonging to cells which contain the zero level set of $\widehat{\Phi}_t$. The level set function Φ_t^{k+1} is built so that the zero level set of Φ_t^{k+1} coincides with the zero level set of $\widehat{\Phi}_t$. Thus, the level set function Φ_t^{k+1} at a given node of the set $\mathcal{I}_t$ is the distance from this node to the zero level set of $\widehat{\Phi}_t$, where $\widehat{\Phi}_t$ is positive, and the opposite of the distance from this node to the zero level set of $\widehat{\Phi}_t$, where $\widehat{\Phi}_t$ is negative. The computation of the distance from each node of $\mathcal{I}_t$ to the zero level set of $\widehat{\Phi}_t$ is based on the projection algorithm presented in the paragraph dedicated to the reinitialization of the normal level set function.

This procedure ensures the level set functions Φ_n^{k+1} and Φ_t^{k+1} are signed distance functions.

The difference between the Simplex Method and the Upwind Method is in the way the Hamilton-Jacobi equations (10), (12) and (14) are integrated. Simplex Method is dedicated to unstructured meshes (triangles or tetrahedra) and implements the method proposed in (Barth and Sethian 1998). Upwind Method is dedicated to regular grids (quadrangles or hexahedra) and implements the upwind scheme proposed in (Prabel et al. 2007). Both methods use a one step explicit time integration scheme (explicit Euler) to integrate the Hamilton-Jacobi equations. Therefore the time step must be less than a critical value in order to satisfy the CFL condition.

2.2 Geometric Method

We now expose the fully geometric method proposed by Colombo (2012). The extension step discussed in section 2.1.1 is first performed. Thus, point P on the crack front Γ_0^k is such that point M belongs to the plane $(P, \mathbf{N}, \mathbf{B})$ which is known. Displacement $\boldsymbol{\Delta}\mathbf{a}^P$ and angle β at point P are also known.

Let Q be the point on the crack front Γ_0^{k+1} corresponding to point P on the crack front Γ_0^k. Since $\boldsymbol{\Delta}\mathbf{a}^P$ is the displacement at point P, we have:

$$\mathbf{PQ} = \boldsymbol{\Delta}\mathbf{a}^P = \Delta a(\cos\beta\mathbf{N} + \sin\beta\mathbf{B}). \tag{16}$$

Since β is the angle of the crack growth direction to the plane tangent to the crack, local frame $(\mathbf{N}^Q, \mathbf{B}^Q)$ at point Q is obtained by rotating local frame $(\mathbf{N}, \mathbf{B})$ by an angle β with respect to $\mathbf{T}$. The level set values $\Phi_n^{k+1}(M)$ and $\Phi_t^{k+1}(M)$ form a cartesian coordinate system in plane $(Q, \mathbf{N}^Q, \mathbf{B}^Q)$ and we have:

$$\mathbf{QM} = \Phi_n^{k+1}(M)\mathbf{B}^Q + \Phi_t^{k+1}(M)\mathbf{N}^Q. \tag{17}$$

The level set functions at the end of the growth increment are then computed as:

$$\begin{aligned} \Phi_n^{k+1}(M) &= \mathbf{QM} \cdot \mathbf{B}^Q, \\ \Phi_t^{k+1}(M) &= \mathbf{QM} \cdot \mathbf{N}^Q. \end{aligned} \tag{18}$$

This method condenses the update, reinitialization and orthogonalization steps in one step. No property of the underlying mesh is exploited so that this method can handle all types of elements.

Table 1 gives a concise comparison of the methods to update the level set functions. How each method performs extension, update, reinitialization and orthogonalization steps is shown.

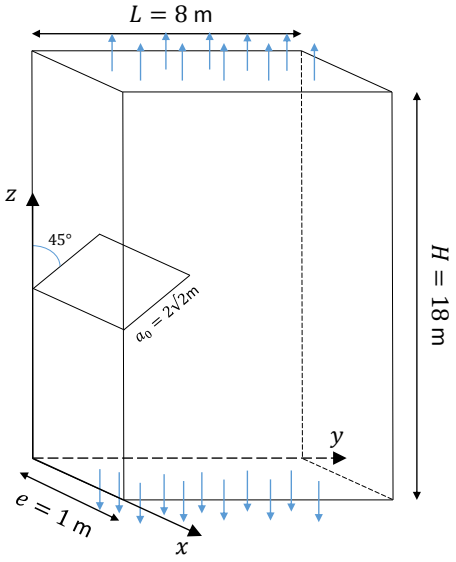


Fig. 2 Geometry and loading for the mode I-II crack growth.

2.3 Numerical tests

2.3.1 A mode I-II crack growth

We propose to compare the results obtained using the Simplex Method, the Upwind Method and the Geometric Method on a simple case. We consider the cracked plate depicted on Fig. 2. The length of the plate is $L = 8$ m, its height is $H = 18$ m and its thickness is $e = 1$ m. The initial crack is inclined with respect to the loading symmetry plane (xOy) by an angle of 45°. The initial crack front is located at a quarter of the

G (J/m^2)	K_I (MPa/$\sqrt{\text{m}}$)	K_{II} (MPa/$\sqrt{\text{m}}$)
45.9905	2.72750	1.38547

Table 2 Tabulated valued of the energy release rate and the stress intensity factors.

length and at half the height. The initial crack length is $a_0 = 2\sqrt{2}$ m. The plate is submitted to a symmetric uniform traction t applied on the top face and the bottom face (see Fig. 2). Since the pre-crack is inclined with respect to the loading symmetry plane, a mode I-II crack growth is expected.

We consider two meshes. The first one is a rectangular grid with twenty eight elements along the length, sixty elements along the height and four elements in the thickness. The second one is obtained by splitting each hexaedron of the first mesh in six tetrahedra. Both meshes share the same nodes, so that the Geometric Method gives the same results with both meshes. The Upwind Method is applied to the first mesh, made of hexaedra. The Simplex Method is applied to the second mesh, made of tetrahedra. The energy release rate and the stress intensity factors are smoothed to make them uniform along the crack front. This way, all points on the crack front travel the same distance. Each simulation consists in one growth increment and the distance travelled by each point of the crack front is $\Delta a^{\max} = 0.4$ m. Paris' law exponent is $m = 1$.

In order to test only the level set update, we use tabulated values of the energy release rate and of the stress intensity factors (*cf.* Table 2). These values were obtained using the mesh made of tetrahedra, a Young's modulus equal to 205000 MPa, a Poisson's ratio equal to 0.3 and a load amplitude $t = 1$ MPa. The energy release rate and the stress intensity factors are computed using the energetic approach in a domain restricted to a torus surrounding the crack front of radius $R_S = 1.16$ m, which corresponds approximately to four times the size of an element.

The domain in which the normal and tangential level set functions are updated is restricted to a torus surrounding the crack front. The radius of this torus is automatically computed by the software from a minimal radius given by the user. We chose a minimal radius $R = R_S + \Delta a^{\max} = 1.56$ m. The radius of the torus computed by the software is 2.31 m.

In this test the problem does not depend on the x coordinate. Consequently, we expect the normal and tangential level set functions to be uniform along axis Ox.

Figure 3 shows the results we obtained using the Simplex Method, the Upwind Method and the Geometric Method to update the level set functions. We computed level sets of the normal and tangential level set functions and we represented the projection of these level sets onto the plane (yOz).

The projection onto the plane (yOz) of the level sets of the level set functions obtained using the Simplex Method (*cf.* Fig. 3(a)) and the Upwind Method (*cf.* Fig. 3(b)) do not result in curved lines but surfaces. The level set functions obtained using the Simplex Method and the Upwind Method are thus not uniform along axis Ox. Conversely, the projection onto the plane (yOz) of the level sets of the level functions obtained using the Geometric Method (*cf.* Fig. 3(c)) result in curved lines so that the level functions obtained using the Geometric Method are uniform along axis Ox.

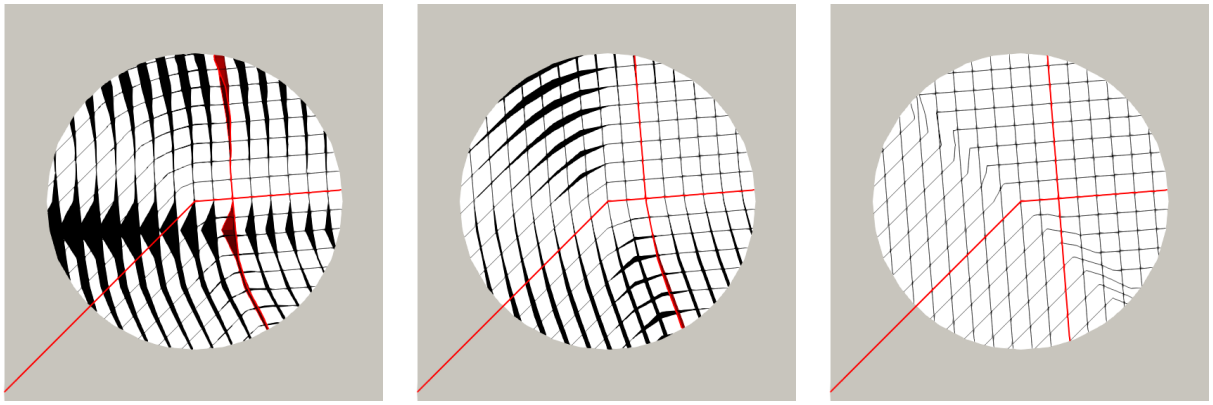
The level sets computed from the normal level set function obtained using the Geometric Method are not parallel to each other. Actually, due to the presence of the bifurcation of the crack, level sets of the normal level set function computed using the Geometric Method are discontinuous, since the normal level set function is not updated where the initial tangential level set function is negative. The post-processing is not able to represent these discontinuities in the general case. However, it is able to represent the kink of the zero level set of the normal level set function. Taking into account this artifact, we conclude the level sets computed from the normal level set function obtained using the Geometric Method are parallel to one another. The level sets computed from the tangential level set function obtained using the Geometric Method are parallel to one another. Furthermore, the level sets of the normal level set function are orthogonal to the level sets of the tangential level set function, where the normal level set function has been updated.

This analysis emphasizes the difficulty to obtain accurate results by integrating the Hamilton-Jacobi equations to perform the reinitialization and orthogonalization steps. The level set functions obtained using the Geometric Method are quite good. Nevertheless, this method faces other issues, as we shall see in the analysis of the Brokenshire test.

2.3.2 Brokenshire test

Brokenshire test is a torsion test first proposed by Barr and Brokenshire (1996). The initial work of Barr and Brokenshire consisted in an experimental analysis. The setup of Brokenshire test is depicted on Fig. 4.

Several authors investigated this test with various approaches. Jefferson et al. (2004) used a plastic-damage-contact model for concrete. Su et al. (2010) used cohesive elements embedded between solid elements. Baydoun and Fries (2012) used X-FEM and studied dif-



(a) Simplex Method (b) Upwind Method (c) Geometric Method

Fig. 3 Representation of the level sets of the normal and tangential level set functions computed at the end of the crack growth using different implementations for the update step: (a) Simplex Method, (b) Upwind Method and (c) Geometric Method. The zero level sets of the two level set functions are shown in red. The update of the level set functions is restricted to a torus surrounding the crack front.

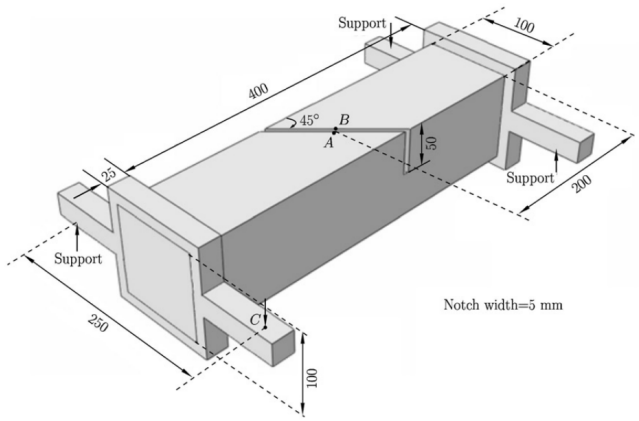


Fig. 4 Geometry and loading of Brokenshire test (Su et al. 2010).

ferent propagation criteria. We expose in this section the behaviour of the Simplex, Upwind and Geometric Methods applied to the Brokenshire test.

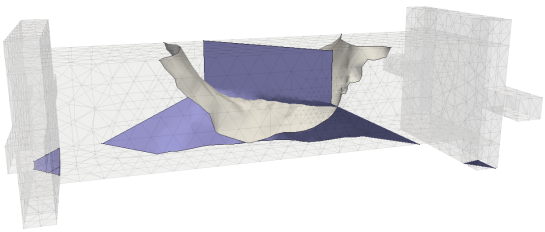
The vertical displacement prescribed at point C, on one arm of the loading collar, is $u = 10$ mm. The vertical displacement of the support which maintains the other arm of the loading collar is prescribed to zero. The two remaining supports, located on the other collar, are fixed. The specimen is made of concrete, with Young's modulus $E = 35000$ MPa and Poisson's ratio $\nu = 0.2$. The collars are made of steel, with Young's modulus $E = 210000$ MPa and Poisson's ratio $\nu = 0.3$.

We consider an unstructured tetrahedral mesh. In order to accurately calculate the energy release rate and the stress intensity factors the mesh is refined at each propagation step in a region surrounding the crack front. To do so we use HOMARD, an automatic mesh adaptation tool distributed along with code_aster (Nicolas et al. 2016). The element size in the initial mesh is $h_0 = 20$ mm. HOMARD splits the edges such that the element size is divided by two in the refined region. We call HOMARD three times in a row, so that the element size is $h = h_0/2^3 = 2.5$ mm in a torus surrounding the crack front of radius $R_S = 5h = 12.5$ mm. The stress intensity factors and the energy release rate are computed using the displacement jump extrapolation technique. We extract the displacement of points lying on the crack faces, within a distance $4h = 10$ mm from the crack front. The maximal crack growth size during a growth increment is $\Delta a^{\max} = 4h = 10$ mm and the Paris' law exponent we arbitrarily chose is $m = 1$. The normal and tangential level set functions are updated in the whole domain.

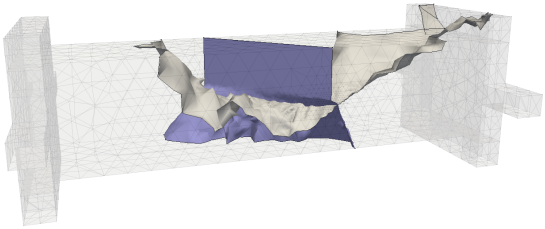
Simplex and Upwind Methods fail to converge during the first increment step (the maximum number of iterations is fixed to five hundred). Geometric Method is able to perform at least the first two growth increments. The zero level sets of the resulting normal and tangential level set functions are shown in Fig. 5.

After the second growth increment, oscillations of the zero level set of the tangential level set function are observed (*cf.* Fig. 5(b)). As a result, the crack front becomes discontiunous and the computation stops.

We conclude none of the three approaches we presented so far is robust enough. We thus propose a new implementation of the Fast Marching Method and we expect it to be robust enough to be able to perform the simulation until the total failure of the specimen is reached.



(a) After one growth increment.



(b) After two growth increments.

Fig. 5 Representation of the zero level set of the normal level set function (in grey) and of the zero level set of the tangential level set function (in purple) during the growth of the crack, obtained using the Geometric Method.

3 Fast Marching Method for crack growth

Sukumar et al. (2003, 2008) proposed to couple the Fast Marching Method and X-FEM to simulate planar and non-planar crack growth. They used the Fast Marching Method to perform the extension and reinitialization steps. In this section, we propose a new approach in which only the reinitialization step is performed by means of the Fast Marching Method.

A level set function Φ satisfies the following eikonal equation:

$$\|\boldsymbol{\nabla}\Phi\| = 1. \tag{19}$$

The Fast Marching Method proposes to solve this eikonal equation, assuming the values of Φ in the neighborhood of the zero level set of Φ are known. Let $d = |\Phi|$ be the distance to the zero level set of Φ. The idea of the method is to propagate the information outward from smaller values of d to larger values. To ensure a monotonic propagation of the information, we split the process in two steps:

1. The function $d = \Phi$ is computed in the domain $\{M : \Phi(M) > 0\}$ from known values of d in the neighborhood of the zero level set of Φ. We then take $\Phi = d$.

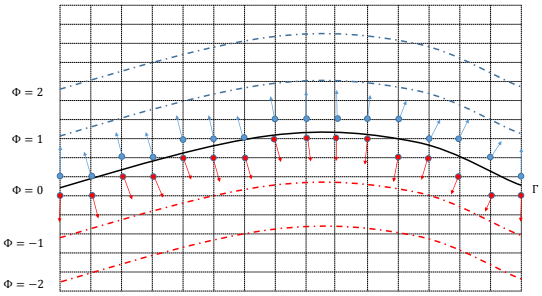


Fig. 6 Propagation of the information during the Fast Marching Method process applied to the computation of the signed distance function Φ to the interface Γ.

2. The function $d = -\Phi$ is computed in the domain $\{M : \Phi(M) < 0\}$ from known values of d in the neighborhood of the zero level set of Φ. We then take $\Phi = -d$.

Each step propagates the (positive) distance function d from smaller to larger values. The propagation of the information during the Fast Marching Method process is illustrated in Fig. 6.

In practice, the neighborhood of the zero level set of Φ refers to a set of nodes $\mathcal{I}$ computed as follows. We loop over the cells and identify the ones which contain the zero level set of Φ. If the sign of the level set function Φ is not the same at all the nodes of a given cell, this cell contains the zero level set of Φ and the nodes belonging to this cell are added to $\mathcal{I}$.

We now expose our new approach:

1. *Extension step.* The extension step discussed in section 2.1.1 is performed.
2. *Update step.* The update step discussed in section 2.1.2 is performed. We recall the updated level set functions $\widetilde{\Phi}_n$ and $\widetilde{\Phi}_t$ obtained at the end of this step are neither expected to be signed distance functions nor expected to be orthogonal.
3. *Reinitialization of the normal level set function.* The normal level set function at the end of growth increment Φ_n^{k+1} is computed as the solution of the eikonal equation:

$$\|\boldsymbol{\nabla}\Phi_n\| = 1, \tag{20}$$

by means of the Fast Marching Method. We recall $\mathcal{I}_n$ is the set of nodes belonging to cells which contain the zero level set of $\widetilde{\Phi}_n$. Initial data consists in the known values of level set function Φ_n^{k+1} at nodes of set $\mathcal{I}_n$. Level set function Φ_n^{k+1} is built so that the zero level set of Φ_n^{k+1} coincides with the zero level set of $\widetilde{\Phi}_n$. Thus, level set function Φ_n^{k+1} at a given node of set $\mathcal{I}_n$ is the distance from this node to the

zero level set of $\widetilde{\Phi}_n$, where $\widetilde{\Phi}_n$ is positive, and the opposite of the distance from this node to the zero level set of $\widetilde{\Phi}_n$, where $\widetilde{\Phi}_n$ is negative. Computation of the distance from each node of $\mathcal{I}_n$ to the zero level set of $\widetilde{\Phi}_n$ is based on the projection algorithm proposed in (Colombo and Massin 2011).

4. *Reinitialization of the tangential level set function.* The tangential level set function at the end of growth increment Φ_t^{k+1} is computed as the solution of the eikonal equation:

$$\|\boldsymbol{\nabla}\Phi_t\| = 1, \tag{21}$$

by means of the Fast Marching Method. We recall $\mathcal{I}_t$ is the set of nodes belonging to cells which contain the zero level set of $\widetilde{\Phi}_t$. Initial data consists in the known values of level set function Φ_t^{k+1} at nodes of set $\mathcal{I}_t$. Since level set function $\widetilde{\Phi}_t$ and level set function Φ_n^{k+1} are not orthogonal, level set function Φ_t^{k+1} at a given node of set $\mathcal{I}_t$ cannot be deduced from the distance from this node to the zero level set of $\widetilde{\Phi}_t$. We thus apply the Geometric Method discussed in section 2.2 to compute level set function Φ_t^{k+1} at nodes of set $\mathcal{I}_t$. This way, the reinitialization of the tangential level set function and the orthogonalization step are condensed in one step.

Table 1 gives a concise comparison of our new approach based on the Fast Marching Method, with respect to the three other methods to update the level set functions (Simplex Method, Upwind Method and Geometric Method).

During each growth increment, the normal level set function and the tangential level set function are reinitialized. Each time a level set function is reinitialized, the zero level set of this level set function is altered (an illustration of this phenomeom is given in (Chopp 2001)). This behavior does not depend on the method used to solve the reinitialization problem and may occur using the Fast Marching Method. The numerical tests reported in section 3.3 show that despite the alteration of the zero level set of the level set functions induced by this phenomenon, our new approach gives acceptable results for our applications.

The Fast Marching Method travels across the nodes of the mesh and updates the level set value of each visited node. The way the Fast Marching Method selects the next node to be updated and the way this node is updated depend on the mesh structure. In the following sections, we expose an approach dedicated to unstructured grids, and we then extend this approach from tetrahedra to all types of elements.

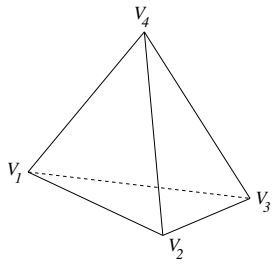


Fig. 7 Representation of the tetrahedron $V_1V_2V_3V_4$.

3.1 Fast Marching Method applied to unstructured grids

We discuss in this section our implementation of the Fast Marching Method dedicated to unstructured grids, which consists in an extension to the tridimensional case of the method proposed by Kimmel and Sethian (1998). The distance function d represents the distance to the zero level set of a given function Φ, which can be either the normal level set Φ_n^{k+1} or the tangential level set Φ_t^{k+1}. The computation is performed for a given set of nodes $\mathcal{N}$, which can correspond to the domain where Φ is positive or the domain where Φ is negative.

3.1.1 Estimation of the distance function at a given node

We consider a non degenerate tetrahedron in an unstructured grid. The vertices of this tetrahedron are denoted V_1, V_2, V_3 and V_4 (see Fig. 7). The value of the distance function d at vertex V_i is denoted d_i. Distances $d_1 = d(V_1)$, $d_2 = d(V_2)$ and $d_3 = d(V_3)$ are known and distance $d_4 = d(V_4)$ is unknown. The first order Taylor series expansion of distance function d around the vertex V_4 evaluated at point P reads:

$$d(P) \approx d(V_4) + \mathbf{V_4P} \cdot \boldsymbol{\nabla} d(V_4). \tag{22}$$

We assume that this approximation is exact for $P = \mathbf{V}_1$, for $P = \mathbf{V}_2$ and $P = \mathbf{V}_3$. We obtain the following system of three equations:

$$\begin{aligned} d_1 &= d_4 + \mathbf{V_4V_1} \cdot \boldsymbol{\nabla} d(V_4), \\ d_2 &= d_4 + \mathbf{V_4V_2} \cdot \boldsymbol{\nabla} d(V_4), \\ d_3 &= d_4 + \mathbf{V_4V_3} \cdot \boldsymbol{\nabla} d(V_4). \end{aligned} \tag{23}$$

Since the tetrahedron is not degenerate, $(\mathbf{V_4V_1}, \mathbf{V_4V_2}, \mathbf{V_4V_3})$ is a basis of $\mathbb{R}^3$. Thus we can write:

$$\mathbf{\nabla d(V_4)} = n^1\mathbf{V_4V_1} + n^2\mathbf{V_4V_2} + n^3\mathbf{V_4V_3}, \tag{24}$$

where n^1, n^2 and n^3 are the coordinates of $\mathbf{\nabla d(V_4)}$ over $(\mathbf{V_4V_1}, \mathbf{V_4V_2}, \mathbf{V_4V_3})$. System (23) can then be

rewritten as follows:

$$\begin{Bmatrix} d_1 \\ d_2 \\ d_3 \end{Bmatrix} = \begin{Bmatrix} d_4 \\ d_4 \\ d_4 \end{Bmatrix} + \mathbf{G}\widehat{\mathbf{n}}, \tag{25}$$

where $\widehat{\mathbf{n}} = (n^1, n^2, n^3)^{\mathrm{T}}$ and matrix $\mathbf{G}$ reads:

$$\mathbf{G} = \begin{bmatrix} \mathbf{V_4V_1} \cdot \mathbf{V_4V_1} & \mathbf{V_4V_1} \cdot \mathbf{V_4V_2} & \mathbf{V_4V_1} \cdot \mathbf{V_4V_3} \\ \mathbf{V_4V_2} \cdot \mathbf{V_4V_1} & \mathbf{V_4V_2} \cdot \mathbf{V_4V_2} & \mathbf{V_4V_2} \cdot \mathbf{V_4V_3} \\ \mathbf{V_4V_3} \cdot \mathbf{V_4V_1} & \mathbf{V_4V_3} \cdot \mathbf{V_4V_2} & \mathbf{V_4V_3} \cdot \mathbf{V_4V_3} \end{bmatrix}.$$

Since d is a distance function, the gradient of the distance at vertex V_4 is a unit vector. We thus have : $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) \cdot \boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) = 1$. Matrix $\mathbf{G}$ represents the metric tensor in the basis $(\mathbf{V_4V_1}, \mathbf{V_4V_2}, \mathbf{V_4V_3})$. Since the metric tensor represents the scalar product, we have:

$$\widehat{\mathbf{n}}^{\mathrm{T}}\mathbf{G}\widehat{\mathbf{n}} = \boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) \cdot \boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) = 1. \tag{26}$$

Equations (25) and (26) represent a system of four equations involving four unknowns: distance d_4 at vertex V_4 and coordinates n^1, n^2 and n^3 of the gradient of distance at vertex V_4. We define two criteria to accept a solution of this system of equations:

1. the consistency condition $d_4 \geq \max(d_1, d_2, d_3)$;
2. the monotonicity condition $n^1 \leq 0$, $n^2 \leq 0$ and $n^3 \leq 0$.

The consistency condition states that, since we want the update to propagate the information outward from smaller values to larger values, we want d_4 to be larger than d_1, d_2 and d_3. The monotonicity condition states that $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ lies in the tetrahedron, *i.e.* $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) \in \mathcal{C}$, where $\mathcal{C}$ is the convex cone defined as follows :

$$\mathcal{C} = \{\alpha_1\mathbf{V_4V_1} + \alpha_2\mathbf{V_4V_2} + \alpha_3\mathbf{V_4V_3}, \quad \alpha_1 \geq 0, \alpha_2 \geq 0, \alpha_3 \geq 0\}.$$

We suppose the solid angle at vertex V_4 is acute, so that we have: $\mathbf{V_4V_1} \cdot \mathbf{V_4V_2} \geq 0$, $\mathbf{V_4V_1} \cdot \mathbf{V_4V_3} \geq 0$ and $\mathbf{V_4V_2} \cdot \mathbf{V_4V_3} \geq 0$. As a consequence, if the monotonicity condition is fulfilled, each component of the vector $\mathbf{G}\widehat{\mathbf{n}}$ is negative so that system (25) gives:

$$\begin{aligned} d_1 - d_4 &\leq 0, \\ d_2 - d_4 &\leq 0, \\ d_3 - d_4 &\leq 0. \end{aligned} \tag{27}$$

The monotocity condition then appears as a sufficient condition to fulfill the consistency condition. Nevertheless, we accept a solution of the system of equations if, and only if, these two conditions are fulfilled simultaneously.

The way to solve the system of equations, depends on a *a priori* hypothesis on $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$:

1. The tridimensional update corresponds to the case where $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie in the interior of convex cone $\mathcal{C}$.
2. The bidimensional update corresponds to the case where $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie in a face of convex cone $\mathcal{C}$.
3. The unidimensional update corresponds to the case where $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie on an edge of convex cone $\mathcal{C}$.

Tridimensional update In this case, $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie in the interior of convex cone $\mathcal{C}$. System (25) is rewritten as follows:

$$\mathbf{d} - d_4\mathbf{1} = \mathbf{G}\widehat{\mathbf{n}}, \tag{28}$$

where $\mathbf{d} = (d_1, d_2, d_3)^{\mathrm{T}}$ and $\mathbf{1} = (1, 1, 1)^{\mathrm{T}}$. Since matrix $\mathbf{G}$ represents a meric tensor, $\mathbf{G}$ is symmetric and positive-definite. We deduce from (28):

$$\mathbf{G}^{-1}(\mathbf{d} - d_4\mathbf{1}) = \widehat{\mathbf{n}}, \tag{29}$$

$$(\mathbf{d} - d_4\mathbf{1})^{\mathrm{T}} = \widehat{\mathbf{n}}^{\mathrm{T}}\mathbf{G}. \tag{30}$$

We thus have:

$$(\mathbf{d} - d_4\mathbf{1})^{\mathrm{T}}\mathbf{G}^{-1}(\mathbf{d} - d_4\mathbf{1}) = \widehat{\mathbf{n}}^{\mathrm{T}}\mathbf{G}\widehat{\mathbf{n}} = 1. \tag{31}$$

Finally, d_4 is solution of the following quadratic equation:

$$ad_4^2 - 2bd_4 + c = 0, \tag{32}$$

where:

$$\begin{aligned} a &= \mathbf{1}^{\mathrm{T}}\mathbf{G}^{-1}\mathbf{1}, \\ b &= \mathbf{1}^{\mathrm{T}}\mathbf{G}^{-1}\mathbf{d}, \\ c &= \mathbf{d}^{\mathrm{T}}\mathbf{G}^{-1}\mathbf{d} - 1. \end{aligned} \tag{33}$$

We assume the discriminant Δ of (32) is non-negative, so that this equation admits two solutions δ_1 and δ_2, such that $\delta_1 \geq \delta_2$. We consider only the solution $d_4 = \delta_1$ to enforce the consistency of the update. Since d_4 is known, we can solve (28) to obtain coordinates n^1, n^2 and n^3. If distance d_4 fulfills the consistency condition and if coordinates n^1, n^2 and n^3 of the gradient of distance $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ fulfill the monotonicity condition, we accept distance d_4. Otherwise, we try the bidimensional update.

Bidimensional update In this case, $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie in a face of convex cone $\mathcal{C}$, *i.e.* the intersection of convex cone $\mathcal{C}$ and one of the faces $V_1V_2V_4$, $V_2V_3V_4$ or $V_3V_1V_4$. One of the coordinates n^1, n^2 and n^3 is set to zero : n^3 is set to zero if we consider face $V_1V_2V_4$, n^1 is set to zero if we consider face $V_2V_3V_4$ and n^2 is set to zero if we consider face $V_3V_1V_4$. As a consequence, one of the equations of system (25) is removed.

Without loss of generality, we consider face $V_1V_2V_4$, so that n^3 is set to zero and the third equation of system (25) has to be removed. System (25) is rewritten as follows:

$$\mathbf{d}' - d_4\mathbf{1}' = \mathbf{G}'\widehat{\mathbf{n}}', \tag{34}$$

where

$$\mathbf{G}' = \begin{bmatrix} \mathbf{V_4V_1}\cdot\mathbf{V_4V_1} & \mathbf{V_4V_1}\cdot\mathbf{V_4V_2} \\ \mathbf{V_4V_2}\cdot\mathbf{V_4V_1} & \mathbf{V_4V_2}\cdot\mathbf{V_4V_2} \end{bmatrix},$$

$\mathbf{d}' = (d_1, d_2)^{\mathrm{T}}$, $\mathbf{1}' = (1,1)^{\mathrm{T}}$ and $\widehat{\mathbf{n}}' = \left(n^1, n^2\right)^{\mathrm{T}}$. We deduce from (34):

$$(\mathbf{d}' - d_4\mathbf{1}')^{\mathrm{T}} \left(\mathbf{G}'\right)^{-1} (\mathbf{d}' - d_4\mathbf{1}') = \left(\widehat{\mathbf{n}}'\right)^{\mathrm{T}} \mathbf{G}'\widehat{\mathbf{n}}'. \tag{35}$$

Since $n^3 = 0$, equation (26) gives:

$$\left(\widehat{\mathbf{n}}'\right)^{\mathrm{T}} \mathbf{G}'\widehat{\mathbf{n}}' = \widehat{\mathbf{n}}^{\mathrm{T}}\mathbf{G}\widehat{\mathbf{n}} = 1. \tag{36}$$

Finally, d_4 is solution of the following quadratic equation:

$$a'd_4^2 - 2b'd_4 + c' = 0, \tag{37}$$

where a', b' and c' are obtained by replacing $\mathbf{1}$, $\mathbf{d}$, $\mathbf{G}$ with $\mathbf{1}'$, $\mathbf{d}'$, $\mathbf{G}'$ in (33).

We proceed similarly to the tridimensionnal update to determine distance d_4 for face $V_1V_2V_4$ but we try also to compute distance d_4 from faces $V_2V_3V_4$ and $V_3V_1V_4$ and we keep the smallest obtained value. If a value of distance d_4 cannot be computed from any face, we try the unidimensionnal update.

Unidimensional update In this case, $-\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is supposed to lie on an edge of convex cone $\mathcal{C}$, *i.e.* the intersection of convex cone $\mathcal{C}$ and one of the edges V_1V_4, V_2V_4 or V_3V_4. Two of the coordinates n^1, n^2 and n^3 are set to zero : n^2 and n^3 are set to zero if we consider edge V_1V_4, n^1 and n^3 are set to zero if we consider edge V_2V_4 and n^1 and n^2 are set to zero if we consider edge V_3V_4. As a consequence, two of the equations of system (25) are removed.

System (25) is rewritten as follows:

$$d_1 - d_4 = \mathbf{V_4V_1}\cdot\mathbf{V_4V_1}n^1. \tag{38}$$

The gradient of distance $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ reads:

$$\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4}) = n^1\mathbf{V_4V_1}. \tag{39}$$

Since $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ is a unit vector, the coordinate n^1 fulfilling the monoticity condition is:

$$n^1 = -\frac{1}{\|\mathbf{V_4V_1}\|}. \tag{40}$$

Distance d_4 fulfilling (38) is then:

$$d_4 = d_1 + \|\mathbf{V_4V_1}\|. \tag{41}$$

A value of d_4 can be computed from edges V_4V_1, V_4V_2 and V_4V_3. We select the smallest distance from the three possible values:

$$d_4 = \min\{d_1 + \|\mathbf{V_4V_1}\|, d_2 + \|\mathbf{V_4V_2}\|, d_3 + \|\mathbf{V_4V_3}\|\}. \tag{42}$$

A pathological case for which the tridimensional update fails Fig. 8 illustrates a case for which the consistency condition is fullfilled but not the monotonicity condition. The coordinates of the vertices over the reference frame $(O, \mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$ are $V_1(0, 1/2, 1)$, $V_2(0, -1/2, 1)$, $V_3(1, 0, 1)$ and $V_4(-1/2, 0, 2)$. The distances associated to vertices V_1, V_2 and V_3 are respectively $d_1 = 1$, $d_2 = 1$ and $d_3 = 1$. The largest root of (32) in this case is $d_4 = 2 > 1 = \max(d_1, d_2, d_3)$, so that the consistency condition is fullfilled. The coordinates of $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ over the basis $(\mathbf{V_4V_1}, \mathbf{V_4V_2}, \mathbf{V_4V_3})$ are $n^1 = -3/4$, $n^2 = -3/4$ and $n^3 = 1/2$. Since $n^3 > 0$, the monotonicity condition is not fullfilled. The tridimensional update thus fails.

We now focus on face $V_1V_2V_4$. Fig. 9 illustrates this problem. The largest root of (37) in this case is $d_4 = 1 + \sqrt{5}/2 > 1$, so that the consistency condition is fullfilled. The coordinates of $\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ over basis $(\mathbf{V_4V_1}, \mathbf{V_4V_2}, \mathbf{V_4V_3})$ are $n^1 = -1/\sqrt{5}$, $n^2 = -1/\sqrt{5}$ and $n^3 = 0$, so that $n^1 \leq 0$, $n^2 \leq 0$, $n^3 \leq 0$ and the monotonicity condition is fullfilled. The bidimensional update is thus successful.

3.1.2 Sweeping process

The initial data consists in the known values of the distance function d at nodes of a given set $\mathcal{I}$. The set of nodes for which a candidate value of distance d is known is denoted $\mathcal{K}$ and the set of nodes with a minimal distance is denoted $\mathcal{P}$. Algorithm 1 outlines the way the nodes are visited by the Fast Marching Method applied to an unstructured grid.

In Algorithm 1 $\mathcal{D}$ refers to the set of candidate values for distance d_j. There are three cases:

Algorithm 1: *Fast Marching Method on an unstructured grid*

INITIALIZATION:
$\mathcal{K} = \mathcal{I}$.
$\mathcal{P} = \emptyset$.

PROCESSING:
WHILE $\mathcal{K} \backslash \mathcal{P} \neq \emptyset$
  Find the node i_0 with the smallest value for the distance in $\mathcal{K} \backslash \mathcal{P}$.
  Add node i_0 to $\mathcal{P}$.
  FOR *each tetrahedron* T *adjacent to the node* i_0
    FOR *each node* j *of the tetrahedron* T
      IF $j \notin \mathcal{I}$ *and* $j \notin \mathcal{P}$ THEN
        IF *all the other nodes adjacent to* T *are in* $\mathcal{K}$ THEN
          The vertices of T are denoted V_1, V_2, V_3, V_4; V_4 refers to node j.
          $\mathcal{D} = \emptyset$
          IF *the discriminant of equation* (32) *is non-negative* THEN
            Compute d_4 as the largest root of (32).
            Compute the coordinates of $\mathbf{\nabla d(V_4)}$ by solving (28).
            IF *the consistency condition and the monotonicity condition are fulfilled* THEN
              Add d_4 to $\mathcal{D}$.
            END IF
          END IF
          IF $\mathcal{D} = \emptyset$ THEN
            FOR *each face* $\in \{V_1V_2V_4, V_2V_3V_4, V_3V_1V_4\}$
              IF *the discriminant of equation* (37) *is non-negative* THEN
                Compute d_4 as the largest root of (37).
                Compute the coordinates of $\mathbf{\nabla d(V_4)}$ by solving (34).
                IF *the consistency condition and the monotonicity condition are fulfilled* THEN
                  Add d_4 to $\mathcal{D}$.
                END IF
              END IF
            END FOR
          END IF
          IF $\mathcal{D} = \emptyset$ THEN
            FOR *each* $k \in \{1, 2, 3\}$
              Compute $d_4 = d_k + \|\mathbf{V}_k\mathbf{V}_4\|$.
              Add d_4 to $\mathcal{D}$.
            END FOR
          END IF
          IF $j \notin \mathcal{K}$ THEN
            Take $d_j = \min(\mathcal{D})$.
            Add the node j to $\mathcal{K}$.
          ELSE
            Take $d_j = \min(\mathcal{D} \cup \{d_j\})$.
          END IF
        END IF
      END IF
    END FOR
  END FOR
END WHILE

1. If distance d_j can be computed from the tetrahedron, $\mathcal{D}$ is a singleton.
2. If distance d_j can be computed from at least one of the faces, $\mathcal{D}$ contains from one to three candidate values.
3. If distance d_j must be computed from the edges, $\mathcal{D}$ contains three candidate values.

Traditional exposition of the Fast Marching Method involves three sets of nodes (Sethian 1996, 1999; Kimmel and Sethian 1998; Sethian and Vladimirsky 2000; Bronstein et al. 2007):

- The distance function at a node in the set of "accepted" nodes is known and frozen.
- The distance function at a node in the set of "tentative" nodes (the so called "Narrow band") is known but can be recomputed.
- The distance function at a node in the set of "distant" nodes is unknown.

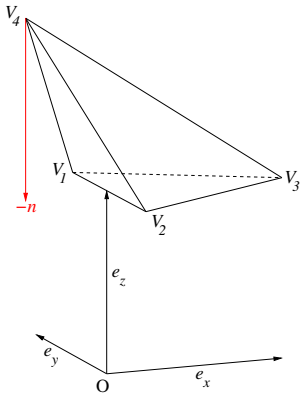


Fig. 8 Illustration of a configuration for which distance d_4 can not be computed using the tridimensional update. The monotonicity condition is not fullfilled, since $-\mathbf{n} = -\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ does not lie in convex cone $\mathcal{C}$.

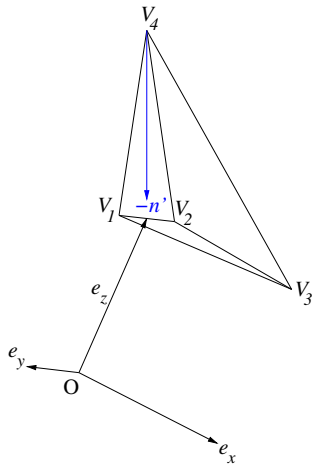


Fig. 9 Illustration of the result of the bidimensional update for face $V_1V_2V_4$ in the configuration for which the tridimensional update fails. Distance d_4 can be computed because the monotonicity condition is fullfilled, since $-\mathbf{n}' = -\boldsymbol{\nabla}\mathbf{d}(\mathbf{V_4})$ lies in convex cone $\mathcal{C}$.

With our notations, the set of "accepted" nodes is $\mathcal{I}\cup\mathcal{P}$, the set of "tentative" nodes is $\mathcal{K}\backslash(\mathcal{I}\cup\mathcal{P})$ and the set of "distant" nodes is $\mathcal{N}\backslash\mathcal{K}$. Our algorithm does not distinguish the update of the distance function at a "tentative" node and the computation of an initial distance function at a node going from the set of "distant" nodes to the set of "tentative" nodes. In the same way, our algorithm does not perform the initialization of the nodes in the set of "tentative" nodes in a distinct block. Actually, while the node with the smallest value for the distance in $\mathcal{K}\backslash\mathcal{P}$ belongs to $\mathcal{I}$, the algorithm populates the initial set of "tentative" nodes. When the initial set of "tentative" nodes is fully populated, the node with the smallest value for the distance in $\mathcal{K}\backslash\mathcal{P}$ belongs to the set of "accepted" nodes.

3.2 Extension of the Fast Marching Method from tetrahedra to all types of elements.

The method we just exposed is designed for a tetrahedral volume mesh. The simplest way to handle any type of element is to build a tetrahedral mesh by splitting the element of the given mesh in a preprocessing step. This solution introduces a new step, a new tool and a new source of problems. We thus propose in this subsection a way to extend the method to arbitrary volume meshes.

In the algorithm, the mesh structure is used to iterate over the tetrahedra adjacent to the node to update. The idea is to build a set of tetrahedra from an arbitrary cell. The algorithm then consists in two nested loops: the outer loop iterates over the cells adjacent to the node to update, while the inner loop iterates over the tetrahedra.

We recall the method computes an updated value of the distance function at the given node, from a given tetrahedron adjacent to the node, by computing an approximation of the gradient of the distance function. We thus propose to iterate over all the tetrahedra we can build from the vertices of the cell, so as many approximated gradients of the distance function as possible are explored. Obviously, some of the tetrahedra built this way can not be used to compute an updated value of the distance function at the given node, since the value of the distance function at some vertices of the cell can be not known yet. Such tetrahedra are simply omitted, as will be explained in the following.

We consider three types of volumic cell, except the tetrahedron: the pyramid, the pentahedron and the hexaedron. We now expose how we compute a set of tetrahedra from each of these cell types.

Pyramid Let $\{1,2,3,4,5\}$ be the connectivity of the reference pyramid (*cf.* Fig. 10). It is possible to build the four following tetrahedra from the pyramid:

$$\begin{array}{l}\{1,3,4,5\},\\ \{1,2,3,5\},\\ \{1,2,4,5\},\\ \{2,3,4,5\}.\end{array} \tag{43}$$

where $\{a,b,c,d\}$ is the connectivity of the tetrahedron based on vertices a, b, c and d.

There are three different cases in which we can use the pyramid to update the value of the distance function at some vertex of the pyramid:

1. If the distance function is known at the four vertices of the base (*i.e.* vertices 1, 2, 3 and 4) and we want to update the value of the distance function at the apex

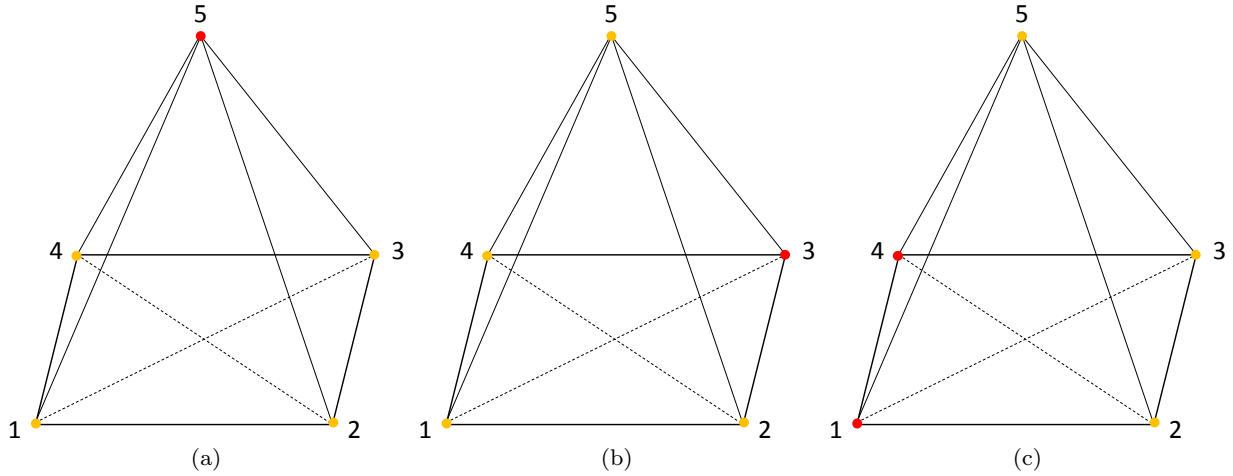


Fig. 10 Spliting a pyramid into tetrahedra. Orange nodes represent known level sets values and red nodes represent level sets values to be determined.

(*i.e.* vertex 5), we iterate over the four tetrahedra of (43). Fig. 10(a) illustrates this case.

2. If the distance function is known at four nodes of the pyramid, including the apex, we iterate over only three tetrahedra. Fig. 10(b) illustrates the case of the update of the distance function at vertex 3 from known values of the distance function at vertices 1, 2, 4 and 5. In this case, we iterate over tetrahedra $\{2,3,4,5\}, \{1,3,4,5\}$ and $\{1,2,3,5\}$.
3. If the distance function is known at three vertices of the pyramid, we consider only one tetrahedron. Fig. 10(c) illustrates the case of the update of the distance function at vertex 1 or 4 from known values of the distance function at vertices 2, 3 and 5. If we want to update the distance function at vertex 1 we consider only tetrahedron $\{1,2,3,5\}$ while if we want to update the distance function at vertex 4 we consider only tetrahedron $\{2,3,4,5\}$.

Pentahedron Let $\{1,2,3,4,5,6\}$ be the connectivity of the reference pentahedron (*cf.* Fig. 11). We first remark we can build the six following pyramids from the pentahedron:

$$\begin{array}{l}\{1,2,3,4,5\},\\ \{1,2,3,4,6\},\\ \{1,4,5,6,2\},\\ \{1,4,5,6,3\},\\ \{2,3,5,6,4\},\\ \{2,3,5,6,1\}.\end{array} \tag{44}$$

From each of these pyramids, we can build four tetrahedra. We then obtain the twenty four tetrahedra listed in Table 3. Each column in Table 3 refers to one of the six pyramids. In each column, the first row gives the connectivity of the pyramid, while the other rows give the connectivities of four tetrahedra corresponding to that pyramid.

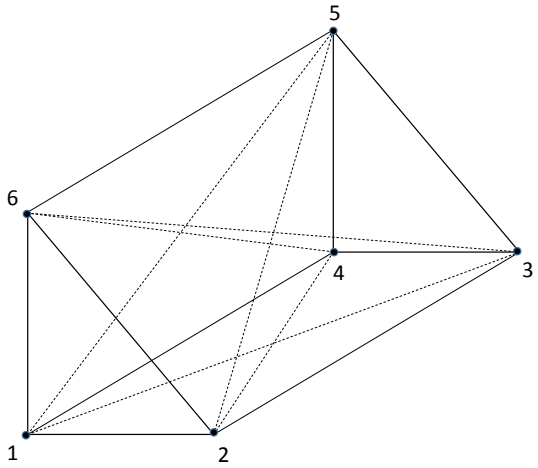


Fig. 11 Spliting of a pentahedron into pyramids.

Some of these twenty four tetrahedra can be built from two pyramids. For example, we can build terahedron $\{1,2,4,5\}$ from both pyramids $\{1,2,3,4,5\}$ and $\{1,4,5,6,2\}$. By removing the duplicates, we end up with the following list of twelve tetrahedra:

$$\begin{array}{l}\{1,2,4,5\},\ \{1,2,4,6\},\ \{1,2,5,6\},\\ \{1,2,3,5\},\ \{1,2,3,6\},\ \{2,4,5,6\},\\ \{2,3,4,5\},\ \{2,3,4,6\},\ \{1,3,5,6\},\\ \{1,3,4,5\},\ \{1,3,4,6\},\ \{3,4,5,6\}.\end{array} \tag{45}$$

Hexahedron Let $\{1,2,3,4,5,6,7,8\}$ be the connectivity of the reference hexahedron. We remark we can build twelve pentahedra from the hexahedron, by splitting the faces two by two. Figure 12(a) illustrates a first way to build four pentahedra from the hexaedron, Fig. 12(b) illustrates a second one and Fig. 12(c) illustrates a third

$\{1,2,3,4,5\}$	$\{1,2,3,4,6\}$	$\{1,4,5,6,2\}$	$\{1,4,5,6,3\}$	$\{2,3,5,6,4\}$	$\{2,3,5,6,1\}$
$\{1,2,4,5\}$	$\{1,2,4,6\}$	$\{1,2,4,5\}$	$\{1,3,4,5\}$	$\{1,2,5,6\}$	$\{2,4,5,6\}$
$\{1,2,3,5\}$	$\{1,2,3,6\}$	$\{1,2,4,6\}$	$\{1,3,4,6\}$	$\{1,2,3,5\}$	$\{2,3,4,5\}$
$\{2,3,4,5\}$	$\{2,3,4,6\}$	$\{1,2,5,6\}$	$\{1,3,5,6\}$	$\{1,3,5,6\}$	$\{3,4,5,6\}$
$\{1,3,4,5\}$	$\{1,3,4,6\}$	$\{2,4,5,6\}$	$\{3,4,5,6\}$	$\{1,2,3,6\}$	$\{2,3,4,6\}$

Table 3 The tetrahedra obtained from the six pyramids one can build from the reference pentahedron. The duplicates are in red.

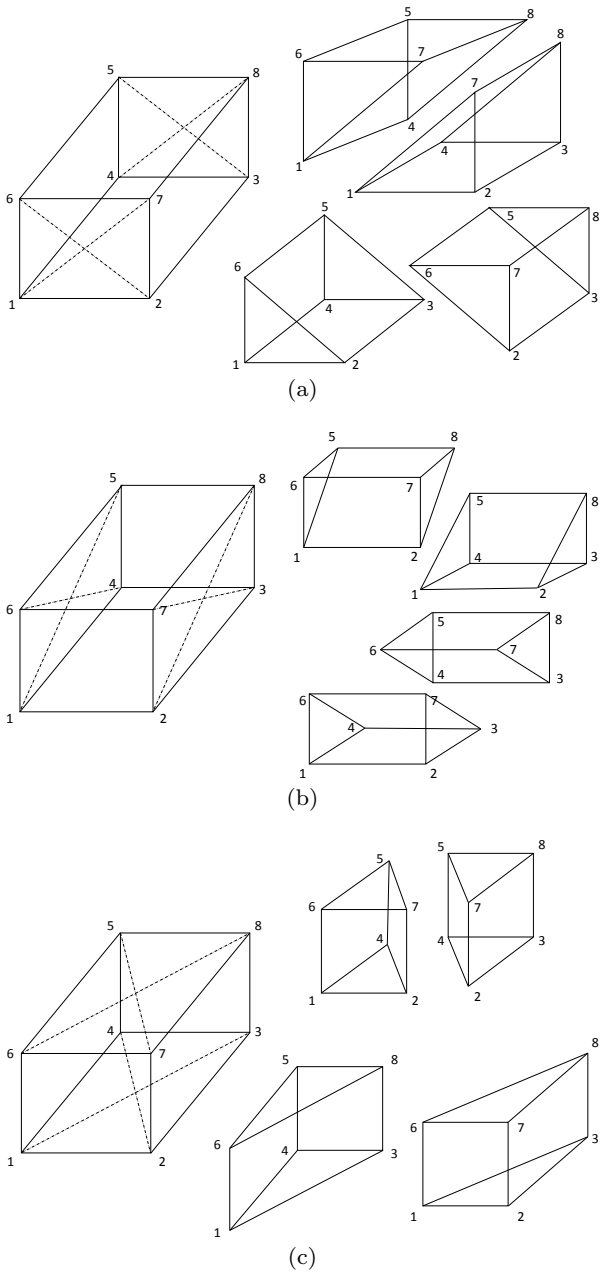


Fig. 12 Spliting of a hexahedron into pentahedra.

one. Since we showed a pentahedron generates twelve tetrahedra, one hundred forty-four (144) tetrahedra (including duplicates) can be built.

In practice, we obviously do not store the connectivities of all these tetrahedra. We implemented a function which returns the connectivities of the twelve tetrahedra one can build from a pentahedron. We call this function for each of the twelve pentahedra one can build from a hexahedron to dynamically build the connectivities of the one hundred forty-four tetrahedra.

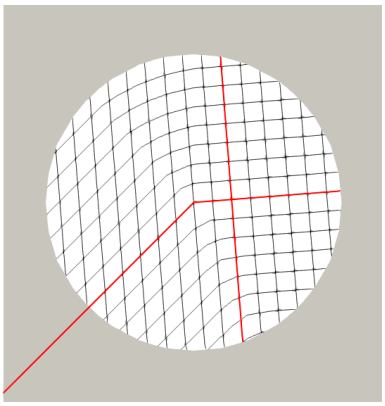
3.3 Numerical tests

3.3.1 A mode I-II crack growth

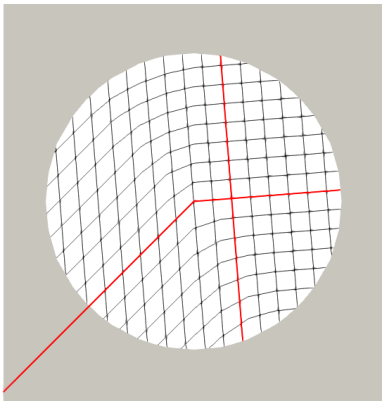
We propose to apply the Fast Marching Method to the mode I-II crack growth problem we already investigated using the Simplex Method, the Upwind Method and the Geometric Method in section 2.3.1. The results are depicted on Fig. 13.

The results obtained using the mesh made of tetrahedra or the mesh made of hexaedra are almost identical, which validates the extension of the method from tetrahedra to other types of elements. The level sets computed from the normal level set function obtained using the Fast Marching Method are parallel to each another. We remark the level sets of the normal level set function computed using the Fast Marching Method are smooth where the level sets of the normal level set function computed using the Geometric Method are discontinuous. The level sets computed from the tangential level set function obtained using the Fast Marching Method are parallel to each another. Furthermore, the level sets of the normal level set function are orthogonal to the level sets of the tangential level set function, where the normal level set function has been updated.

We conclude that the Fast Marching Method is as robust as the Geometric Method, since both methods lead to similar results where the level sets of the normal level set function are continuous. However the Fast Marching Method has the advantage of building smoother level set functions.



(a) Tetrahedra



(b) Hexaedra

Fig. 13 Representation of the level sets of the normal and tangential level set functions computed at the end of the crack growth using the Fast Marching Method for the update step and applied to: (a) the mesh made of tetrahedra and (b) the mesh made of heaxaedra. The zero level sets of the two level set functions are shown in red.

3.3.2 Convergence study

Our new approach based on the Fast Marching Method involves approximations at each step. In order to evaluate *a posteriori* the convergence of our new approach (involving the four steps of the update of the two level set functions), we propose a convergence study. We consider the uniform planar ($\beta = 0$) growth of a circular planar crack. The structure occupies domain $\Omega = [-1,+1]\times[-1,+1]\times[-1/8,+1/8]$. The radius of the initial crack is $R = 1/2$. The initial crack occupies domain $\Gamma^0 = \{M(x,y,z) \in \Omega : x^2+y^2 \leq R^2, z = 0\}$. Each simulation consists in three growth increments and the distance travelled by each point of the crack front during one growth increment is $\Delta a^{\max} = \Delta R = 1/20$. Paris' law exponent is $m = 1$. At the end of growth increment k, normal level set function Φ_n^k and tangential level set function Φ_t^k read:

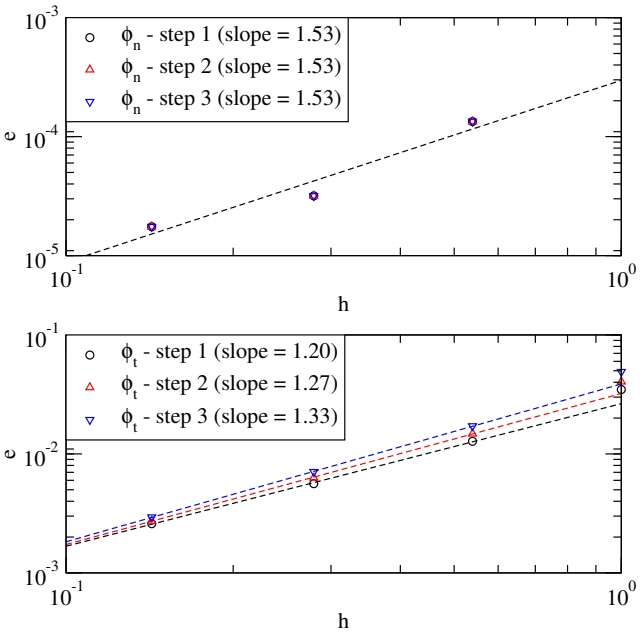


Fig. 14 Error in terms of L^1 norm for the level set functions as a function of the normalized mesh size, obtained with the treatment of obtuse tetrahedra: normal level set function (top) and tangential level set function (bottom).

$$\Phi_n^k(M) = z, \tag{46}$$

$$\Phi_t^k(M) = \sqrt{x^2+y^2} - (R + k\Delta R). \tag{47}$$

A tetrahedron for which the solid angle at each vertex is acute is called an acute tetrahedron. A tetrahedron which is not an acute tetrahedron is called an obtuse tetrahedron. A procedure to compute the solid angle at each vertex of a tetrahedron, based on the formula given by Writh and Dreiding (2014), has been implemented. The caracteristics of the unstuctured meshes used in this convergence study are reported in table 4. Each mesh used in the convergence study includes a small part of obtuse tetrahedra.

The convergence of the Fast Marching Method is ensured only if the triangulation is acute, *i.e.* each tetrahedron of the triangulation is acute (Sethian and Vladimirsky 2003). To handle general triangulations, one can either use a method to split the obtuse tetrahedra (Bronstein et al. 2007; Fu et al. 2013) or the Order Upwind Methods proposed by Sethian and Vladimirsky (2003).

We want to determine if a specific treatment of the obtuse tetrahedra is needed for our applications. We thus perform two convergence studies. In the first one, each time we need to compute the absolute value of a level set function at a vertex, from a tetrahedron such that the solid angle at the vertex is obtuse, we apply the

mesh size	normalized mesh size (h)	# nodes	# tetrahedra	# obtuse tetra.	obtuse tetra. (%)	max. angle (°)
1.47E−1	1.00E+0	892	2893	20	0.69	104.19
7.92E−2	5.40E−1	4771	19520	30	0.15	115.89
4.10E−2	2.79E−1	30564	152574	119	0.08	120.95
2.09E−2	1.43E−1	212917	1183921	622	0.05	128.34

Table 4 Statistics of the unstructured meshes used in the convergence study.

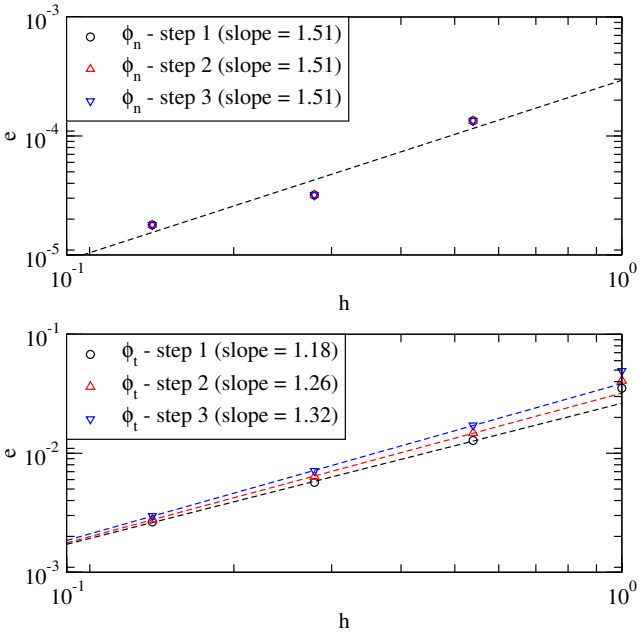


Fig. 15 Error in terms of L^1 norm for the level set functions as a function of the normalized mesh size, obtained without the treatment of obtuse tetrahedra: normal level set function (top) and tangential level set function (bottom).

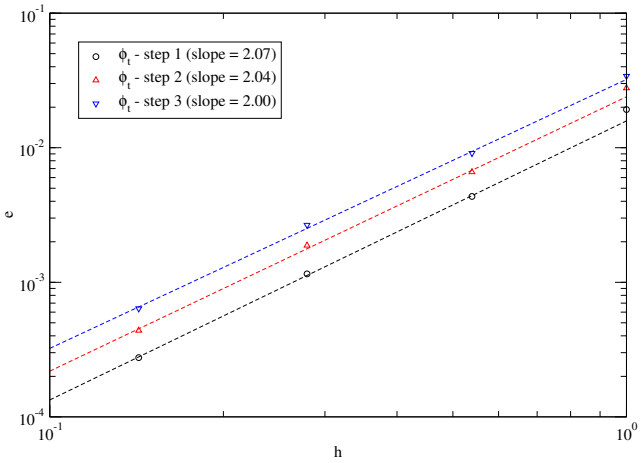


Fig. 16 Error in terms of L^1 norm for the tangential level set function as a function of the normalized mesh size, obtained using the Geometric Method in the whole domain.

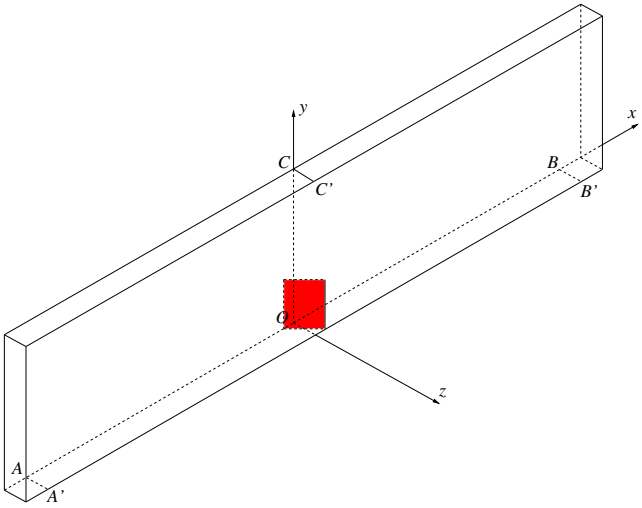


Fig. 17 3D cracked plate subjected to a three-point bending test (the initial crack is in red).

Geometric Method instead of the procedure discussed in 3.1.1 (*cf.* Fig. 14). In the second one, no specific treatment is applied to obtuse tetrahedra (*cf.* Fig. 15).

With or without the treatment of obtuse tetrahedra, we obtain the same order of convergence for both level set functions. We conclude no specific treatment of obtuse tetrahedra is needed for our applications.

In our approach the Geometric Method is used to initialize the Fast Marching Method. As a consequence, we have to consider two convergence mechanisms: 1- the convergence of the distance function at nodes used to intialize the Fast Marching Method and 2- the convergence of the distance function at the other nodes of the domain. The first convergence mechanism caracterizes the convergence of the Geometric Method while the second one caracterizes the convergence of the Fast Marching Method. Since the approximation of the distance at a given node in the Fast Marching Method is based on a first order Taylor series expansion, we expect the error in terms of L^1 norm to converge at order 1, as reported by Fu et al. (2013). Using the Geometric Method in the whole domain, the error in terms of L^1 norm converges at order 2 (*cf.* Fig. 16). For each growth step, the convergence order of the whole process is in between these two values (1 and 2), which seems consistent.

3.3.3 A three-point bending test

A cracked plate subjected to three-point bending is depicted Fig. 17. The length of this beam, associated to direction x, is $L = 260$ mm. Its thickness, associated to direction z, is $t = 10$ mm and its width, associated to

direction y, is $W = 60$ mm. The initial crack length, associated to direction y, is $a = 20$ mm. The initial crack is inclined with respect to plane (yOz) by an angle of 45°. We have $AB = 2L_e = 240$ mm. Line segments AA' and BB' correspond to the supports. The loading consists in a compressive force distribution applied on line segment CC'.

This configuration has been investigated experimentally by Lazarus et al. (2008) and numerically by Citarella and Buchhloz (2008). It has already been used by Colombo and Massin (2011) and Geniaut and Galenne (2012) to challenge crack growth methods available in code_aster. Since the pre-crack is inclined with respect to the loading symmetry plane, the crack is subjected to a mixed mode loading condition and a twisted propagation is observed. We expose in this section the behaviour of the Fast Marching Method applied to this three-point bending test.

The displacement along direction y is prescribed to zero on both line segments AA' and BB'. The displacement along direction z is prescribed to zero at both points A and B. The displacement along direction x is prescribed to zero at point C. The resultant of the load applied on line segment CC' is $F = 2.4$ kN. The material is PMMA with Young's modulus $E = 2800$ MPa and Poisson's ratio $\nu = 0.38$.

We consider an unstructured tetrahedral mesh. In order to accurately calculate the energy release rate and the stress intensity factors we use HOMARD to refine the mesh at each propagation step in a region surrounding the crack front. The element size in the initial mesh is $h_0 = 2.5$ mm. We call HOMARD five times in a row, so that the element size is $h = h_0/2^5 = 0.078125$ mm in a torus surrounding the crack front of radius $R_S = 5h = 0.390625$ mm. The energy release rate and the stress intensity factors are computed using the energetic approach in a domain restricted to this torus.

The maximal crack growth size during a growth increment is $\Delta a^{\max} = 4h = 0.3125$ mm and the Paris' law exponent we arbitrarily chose is $m = 1$. The domain in which the normal and tangential level set functions are updated is restricted to a torus surrounding the crack front. We asked the radius of this torus to be greater than $R = R_S + \Delta a^{\max} = 0.703125$ mm.

At the end of the simulation, the total failure of the specimen has almost been reached (*cf.* Fig. 18). We compare the evolution of the crack front position during the simulation we obtained using the Fast Marching Method with the same evolution of the crack front position reported in Citarella and Buchhloz (2008). Fig. 19 shows the projection of the crack fronts on plane (yOz) while Fig. 20 shows the projection of the crack fronts on plane (xOz). Citarella and Buchholz extracted the crack fronts at the beginning of some growth increments. We extracted from our data the crack fronts corresponding to the ones reported by Citarella and Buchholz. Each curve is indexed by the growth increment to which it corresponds. The growth increments we extracted do not correspond to the ones reported by Citarella and Buchholz because they used a value of the maximal crack growth size during a growth increment $\Delta a^{\max} = 1$ mm, which corresponds to about three times the value we used. One also remarks Citarella and Buchholz used criteria to compute the direction of the crack growth direction and the growth size different from the ones we used. Nevertheless, the solution we obtained with the Fast Marching Method is, qualitatively, in good agreement with the one reported in Citarella and Buchhloz (2008).

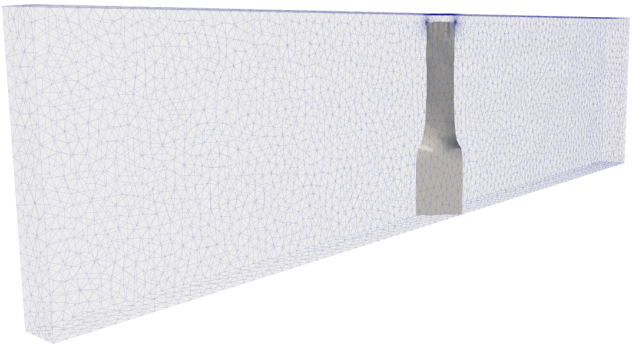


Fig. 18 Crack path obtained at the end of the simulation of the three-point bending test.

3.3.4 Brokenshire test

We expose in this section the behaviour of the Fast Marching Method applied to the Brokenshire test we introduced in 2.3.2. The zero level sets of the normal and tangential level set functions computed after the first two growth increments are depicted Fig. 21.

Unlike the result we obtained using the Geometric Method, no oscillations are observed on the zero level set of the tangential level set function after the first two growth increments. The simulation went on after the second growth increment and the total failure of the specimen was reached. We conclude our implementation of the Fast Marching Method is more robust than the one of the Geometric Method.

The fracture surfaces observed experimentally and obtained by means of the Fast Marching Method are depicted on Fig. 22. They are similar. This qualitative comparaison allows us to conclude our implementation of the Fast Marching Method is able to reproduce a complex crack path.

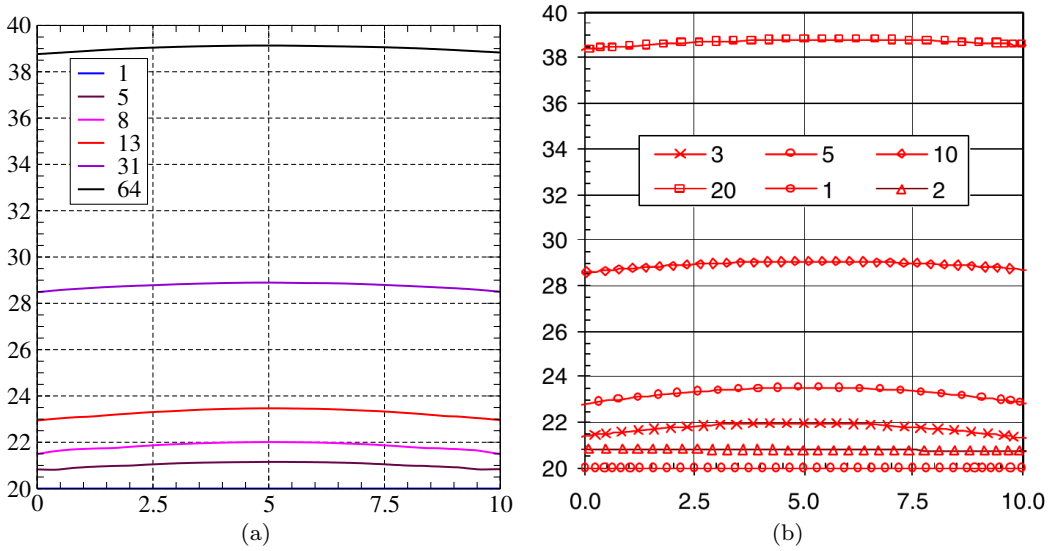


Fig. 19 Crack front plots, coordinate y as a function of the coordinate z: (a) results obtained using the Fast Marching Method, (b) results obtained by Citarella and Buchhloz (2008). The legend gives the growth increment corresponding to each position of the crack front.

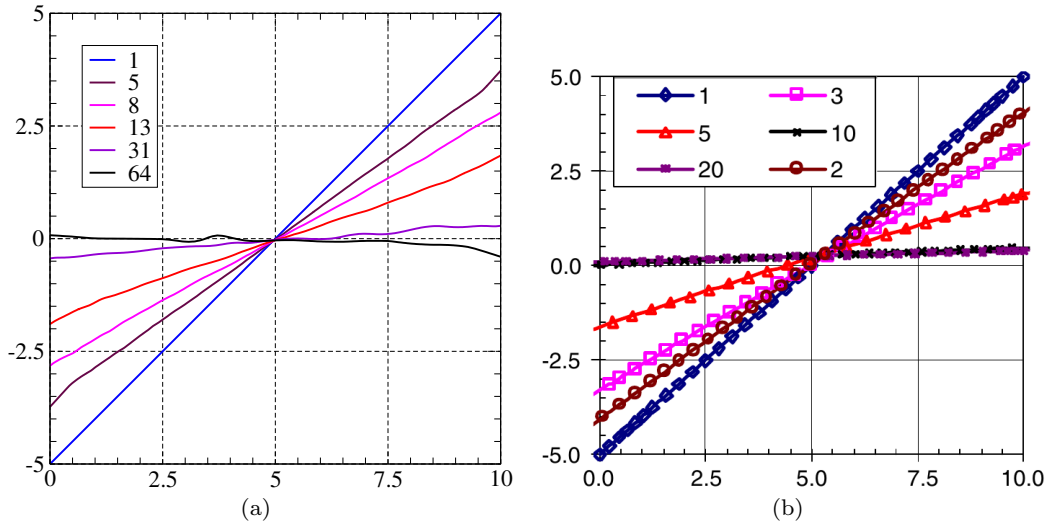


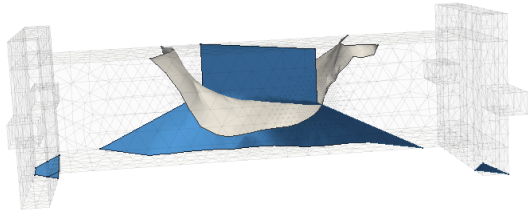
Fig. 20 Crack front plots, coordinate x as a function of the coordinate z: (a) results obtained using the Fast Marching Method, (b) results obtained by Citarella and Buchhloz (2008). The legend gives the growth increment corresponding to each position of the crack front.

4 Conclusion

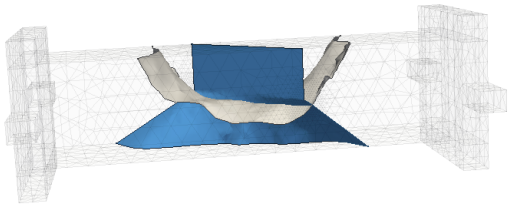
We proposed in this paper a new crack growth method based on X-FEM and Fast Marching Method. We used a Fast Marching Method designed for tetrahedral volume meshes and extended to all types of linear elements (pyramids, pentahedra and hexahedra). Thus, our approach allows us to use the same mesh to solve the mechanical problem and to update the level set functions. The price to pay for this ease of use is an order one discretization scheme of the Eikonal equation used to update the level set functions. For their part, Sukumar et al. (2003, 2008) chose to update the level set functions on an auxiliary rectangular grid to benefit from a second order scheme.

We proposed a new implementation of the Fast Marching Method applied to tetrahedral volume meshes, which consists in an extension of the method proposed by Kimmel and Sethian (1998). Our implementation is suitable for a finite element software. We also proposed a new presentation of the algorithm which is straightforward to implement.

We proposed several numerical examples to show the capabilities of our approach. We used a simple mode



(a) After one growth increment.



(b) After two growth increments.

Fig. 21 Representation of the zero level set of the normal level set function (in grey) and the zero level set of the tangential level set function (in blue) during the growth of the crack, obtained using the Fast Marching Method.

I-II crack growth to show we are able to compute smooth level set functions in the presence of kinks. We performed a convergence study of the proposed approach on a crack growth problem for which an analytical solution, in terms of the level set functions, is known. We investigated the three point bending test introduced by Lazarus et al. (2008) to demonstrate the approcah is able to handle non planar cracks. Finally, this approach was able to simulate the Brokenshire test (Barr and Brokenshire 1996) until the total failure of the specimen while all other methods available in code_aster stopped after two growth increments.

We showed the approach we propose is able to compute smooth level set functions. We now want to get rid of the robustness issues coming from accumulated errors in the position of the crack front during the crack growth. The position of the crack front is computed, at the end of each growth increment, from the energy release rate and the stress intensity factors. Consequently, our next objective is to robustify the computation of the energy release rate and the stress intensity factors.

Acknowledgements We sincerely thank the two anonymous reviewers whose suggestions helped us to improve this manuscript.

References

Barr BIG, Brokenshire DR (1996) Torsion fracture tests. BRE Digest

Barth TJ, Sethian JA (1998) Numerical schemes for the Hamilton-Jacobi and Level Set equations on triangulated domains. Journal of Computational Physics 145(1):1–40

Baydoun M, Fries TP (2012) Crack propagation criteria in three dimensions using the XFEM and an explicit–implicit crack description. International Journal of Fracture 178(1-2):51–70

Belytschko T, Black T (1999) Elastic crack growth in the finite elements with minimal remeshing. International Journal for Numerical Methods in Engineering 45(5):601–620

Bronstein AM, Bronstein MM, Kimmel R (2007) Weighted distance maps computation on parametric three-dimensional manifolds. Journal of Computational Physics 225(1):771–784

Chan SK, Tuba IS, Wilson WK (1970) On the finite element method in linear fracture mechanics. Engineering Fracture Mechanics 2(1):1–17, DOI https://doi.org/10.1016/0013-7944(70)90026-3

Chopp D (2001) Some improvements of the fast marching method. Society for Industrial and Applied Mathematics 23:230–244

Citarella R, Buchhloz FG (2008) Comparison of crack growth simulation by DBEM and FEM for SEN-specimen undergoing torsion or bending loading. Engineering Fracture Mechanics 75(3-4):489–509

Colombo D (2012) An implicit geometrical approach to level sets update for 3D non planar X-FEM crack propagation. Computer Methods in Applied Mechanics and Engineering 237–240:39–50

Colombo D, Massin P (2011) Fast and robust level set update for 3D non-planar X-FEM crack propagation modelling. Computer Methods in Applied Mechanics and Engineering 200(25-28):2160–2180

Destuynder P, Djaoua M (1981) Sur une Interprétation Mathématique de l'Intégrale de Rice en Théorie de la Rupture Fragile. Mathematical Methods in the Applied Sciences 3(1):70–87, DOI 10.1002/mma.1670030106

EDF (1989–2021) Finite element *code_aster*, Analysis of Structures and Thermomechanics for Studies and Research. Open source on www.code-aster.org

Erdogan F, Sih GC (1963) On the Crack Extension in Plates Under Plane Loading and Transverse Shear. Journal of Basic Engineering 85(4):519–525

Fu Z, Kirby RM, Whitaker RT (2013) A fast iterative method for solving the eikonal equation on tetrahedral domains. SIAM Journal on Scientific Computing 35(5):C473–C494

Geniaut S, Galenne E (2012) A simple method for crack growth in mixed mode with X-FEM. International Journal of Solids and Structures 49(15-16):2094–2106, DOI https://doi.org/10.1016/j.ijsolstr.2012.04.015

Gosz M, Moran B (2002) An interaction energy integral method for computation of mixed-mode stress intensity factors along non-planar crack fronts in three dimensions. Engineering Fracture Mechanics 69(3):299–319, DOI https://doi.org/10.1016/S0013-7944(01)00080-7

Gravouil A, Moes N, Belytschko T (2002) Non-planar 3D crack growth by the extended finite element and level sets – Part II: Level set update. International Journal for Numerical Methods in Engineering 53(11):2569–2586

Irwin GR (1957) Analysis of stresses and strains near the end of a crack traversing a plate. Journal of Applied Mechan-

(a) (b)

Fig. 22 Fracture surface observed experimentally (Barr and Brokenshire 1996) (a), and obtained by means of the Fast Marching Method (b).

ics 24:361–364
Jefferson AD, Barr BIG, Bennett T, Hee SC (2004) Three dimensional finite element simulations of fracture tests using the Craft concrete model. Computers and Concrete 1(3):261–284
Kimmel R, Sethian JA (1998) Computing geodesic paths on manifolds. Proceedings of the National Academy of Sciences 95(15):8431–8435
Lazarus V, Buchholz FG, Fulland M, Wiebesiek J (2008) Comparison of predictions by mode II or mode III criteria on crack front twisting in three or four point bending experiments. International Journal of Fracture 153(2):141–151
Li FZ, Shih CF, Needleman A (1985) A comparison of methods for calculating energy release rates. Engineering Fracture Mechanics 21(2):405–421, DOI https://doi.org/10.1016/0013-7944(85)90029-3
Moes N, Dolbow J, Belytschko T (1999) A finite element method for crack growth without remeshing. International Journal for Numerical Methods in Engineering 46(1):131–150
Moes N, Gravouil A, Belytschko T (2002) Non-planar 3D crack growth by the extended finite element and level sets – Part I: Mechanical model. International Journal for Numerical Methods in Engineering 53(11):2549–2568
Nicolas G, Fouquet T, Geniaut S, Cuvilliez S (2016) Improved adaptive mesh refinement for conformal hexahedral meshes. Advances in Engineering Software 102:14–28, DOI https://doi.org/10.1016/j.advengsoft.2016.07.014
Osher S, Sethian JA (1988) Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. Journal of Computational Physics 79(1):12–49
Paris P, Erdogan F (1963) A Critical Analysis of Crack Propagation Laws. Journal of Basic Engineering 85(4):528–533
Prabel B, Combescure A, Gravouil A, Marie S (2007) Level set X-FEM non matching meshes: application to dynamic crack propagation in elastic-plastic media. International Journal for Numerical Methods in Engineering 69(8):1553–1569
Sethian JA (1996) A marching level set method for monotonically advancing fronts. Proceeding of the National Academy of Sciences 93(4):1591–1595
Sethian JA (1999) Fast Marching Methods. SIAM Review 41(2):199–235
Sethian JA, Vladimirsky A (2000) Fast methods for the Eikonal and related Hamilton-Jacobi equations on unstructured meshes. Proceedings of the National Academy of Sciences 97(11):5699–5703
Sethian JA, Vladimirsky A (2003) Ordered upwind methods for static hamilton–jacobi equations: Theory and algorithms. SIAM Journal on Numerical Analysis 41(1):325–363
Shi J, Chopp D, Lua J, Sukumar N, Belytschko T (2010) Abaqus implementation of extended finite element method using a level set representation of three-dimensional fatigue crack growth and life predictions. Engineering Fracture Mechanics 77(14):2840–2863
Stolarska M, Chopp DL, Moes N, Belytschko T (2001) Modelling crack growth by level sets in the extended finite element method. International Journal for Numerical Methods in Engineering 51(8):943–960
Su X, Yang Z, Liu G (2010) Finite Element Modelling of Complex 3D Static and Dynamic Crack Propagation by Embedding Cohesive Elements in Abaqus. Acta Mechanica Solida Sinica 23(3):271–282, DOI https://doi.org/10.1016/S0894-9166(10)60030-4
Sukumar N, Chopp DL, Moran B (2003) Extended finite element method and fast marching method for three-dimensional fatigue crack propagation. Engineering Fracture Mechanics 70(1):29–48
Sukumar N, Chopp DL, Béchet E, Moes N (2008) Three-dimensional non-planar crack growth by a coupled extended finite element and fast marching method. International Journal for Numerical Methods in Engineering 76(5):727–748
Writh K, Dreiding AS (2014) Relations between edge lengths, dihedral and solid angles in tetrahedra. Journal of Mathematical Chemistry 52:1624–1638