Skeleton curve and phantom node method for the Thick Level Set approach to fracture

Mororo, L.A.T.; Poot, A.; van der Meer, F.P.

DOI
10.1016/j.engfracmech.2022.108443

Publication date
2022

Document Version
Final published version

Published in
Engineering Fracture Mechanics

Engineering Fracture Mechanics 268 (2022) 108443

Contents lists available at ScienceDirect

Engineering Fracture Mechanics

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

Skeleton curve and phantom node method for the Thick Level Set approach to fracture

L.A.T. Mororó [a,b,*], A. Poot [b], F.P. van der Meer [b]

[a] Federal Institute of Education, Science and Technology of Ceará, Morada Nova, Brazil
[b] Delft University of Technology, Faculty of Civil Engineering and Geosciences, PO Box 5048, 2600 GA Delft, The Netherlands

ARTICLE INFO

Dataset link: https://doi.org/10.4121/19400849.v1

Keywords:
Thick Level Set
Skeleton curve
Phantom node method
Damage mechanics
Fracture mechanics

ABSTRACT

The Thick Level Set method (TLS) is an approach for non-local damage modeling in which the damage evolution is linked to the movement of a damage front described with the level set method. More recently, a new version of the TLS, designated as the TLSV2, has been proposed as a new concept for coupling continuum damage modeling and discrete cohesive crack modeling for failure analysis in solids. The main objective of this new framework is to profit from both modeling approaches. The continuum part allows for handling crack initiation, branching and merging, whereas the cohesive part brings the capability to handle discrete cracks with large crack opening or sliding without heavily distorted elements, and with the possibility to model stiffness recovery upon contact. In this paper, a generalized framework for the TLSV2 is introduced. Two major issues with the TLSV2 method that have not been dealt with since its inception are addressed in this study, and solutions are proposed. Firstly, the method depends on the location of skeleton curve of the level set field, on which the discontinuity in the displacement field is evaluated. The problem of locating the skeleton curve can be a complicated task, mainly because topological events may emerge as the analysis progresses, such as crack branching. The skeleton curve is determined through a combination of ball-shrinking and graph-based algorithms and then mapped onto the finite element mesh. Secondly, the cohesive forces and displacement discontinuity of the TLSV2 are modeled using the phantom node method. Furthermore, a new approach to compute the averaged values of local quantities is introduced, and model calibration is discussed. The degree of stiffness recovery under compression that is still needed for the continuum part is investigated. Numerical experiments demonstrate the accuracy and ability of the proposed model to handle simulation of failure analysis presenting complex topological crack patterns.

1. Introduction

The Thick Level Set (TLS) method, first introduced by Moës et al. [1], is a non-local damage model that couples damage and fracture mechanics within a single regularized framework. In this method, the damage variable that describes continuum stiffness degradation is made into a function of a level set field. The level set method with a signed distance function is used to construct the level set field whose the zero level set is used to keep track of the damage front. The level set method gives the TLS method the natural ability of representing complex crack events, such as branching and merging [2,3]. In the TLS, the damage variable

* Corresponding author at: Federal Institute of Education, Science and Technology of Ceará, Morada Nova, Brazil.
E-mail addresses: l.a.taumaturgomororo@tudelft.nl, luiz.mororo@ifce.edu.br (L.A.T. Mororó), a.poot-1@tudelft.nl (A. Poot), f.p.vandermeer@tudelft.nl (F.P. van der Meer).

https://doi.org/10.1016/j.engfracmech.2022.108443
Received 16 January 2022; Received in revised form 25 March 2022; Accepted 3 April 2022
Available online 18 April 2022

Nomenclature

c	spread of the damage front movement
c_1, c_2, c_3, c_4	constants for the arc-tangent formula f
d, D	interfacial and continuum damage variables
E	potential energy, Young's modulus, and set of edges of a graph
f, f_t	arc-tangent formula, and strength of the material
G, G_c	graph representation, and fracture energy
h	dimension associated to the smallest element in the whole mesh
I_0	complete set of points that define the front
l, l_c	generic width of a thick damaged band, and characteristic length of the TLS
N_i	shape functions associated with nodes i
k, K	quantity related to the damage front movement, and penalty stiffness
q, Q, $\mathbf{Q}$	interfacial and bulk damage profiles, and transformation matrix
r_{init}	initial radius for ball-shrinking algorithm
t_i, $\bar{t}_i$, T_0	cohesive traction i, cohesive traction i in the local frame, and KD-tree data structure
$\mathbf{u}$	displacement field
v_n, $v_{\max}$	front velocity and maximum front velocity
V	set of vertices of a graph
y, Y	cohesive and bulk driving forces
Y_c, Y_c^0, Y_c^G	material resistance to crack growth, and its strength-based and energy-based limits
$\bar{Y}$, $\bar{Y}_c$	averaged fields
α_i, α_v, $\bar{\alpha}_{ni}$	stiffness recovery parameter for principal strain component i, volumetric strain, and displacement jump i
β	parameter that controls the degree of stiffness recovery under compression
γ	load factor
Γ_0, Γ_s, $\Gamma_\star$	iso-0 curve, skeleton curve, and iso-critical curve for the interfacial part
$\delta_{\min}$, $\delta_{\max}$, δ_{ij}	distance thresholds for the Skeletonizer algorithm, and Kronecker delta
ε_i	principal strain value i of the elastic strain tensor $\boldsymbol{\epsilon}$
η	compression factor
θ_1	denoising angle
κ	stabilization parameter
λ, μ, ν	Lamé's elastic constants, and Poisson's ratio
ξ	user-defined parameter to compute $v_{\max}$
σ_i	stress component i in principal stress space
ϕ, ϕ_s, $\phi_\star$, ϕ_0	level set field, skeleton level set, critical level set, and size of a new damage nucleus
$\bar{\phi}$, $\bar{\phi}_{\text{init}}$, $\bar{\phi}_{\max}$	measure of the size of damaged zones, and its limits to evaluate Y_c
$\bar{\phi}_{\text{init},\beta}$, $\bar{\phi}_{\max,\beta}$	limits of $\bar{\phi}$ to evaluate β
ψ, Ψ	free energy for the interfacial and the bulk parts
Ω, Ω^{d}	entire domain, and damaged subdomain
TLS	Thick Level Set
TLSV1, TLSV2	first and second versions of the TLS method
$\mathcal{N}$, $\mathcal{N}_0$	complete set of nodes, and complete set of nodes that contains the front
$(\cdot)'$, $[\![\cdot]\!]$, $\text{tr}(\cdot)$	derivative of $(\cdot)'$ with respect to its argument, displacement jump of $(\cdot)$ field, and trace of a tensor $(\cdot)$
$\mathbf{K}$, $\mathbf{L}$, $\bar{\mathbf{Y}}$, $\mathbf{l}$, $\mathbf{f}^i$	quantities of the system of equations used for averaging procedure of field i

gradually varies over a thick band of the material located behind the damage front until fully degraded regions arise. The presence of a characteristic length represented by the width of this damaging band gives the TLS a non-local nature that prevents spurious localization in the strain field. The update of damage is related to the averaged configurational force, which is obtained by integrating the local values of energy release rate over this characteristic length.

In the original method (the TLSV1), stress-free macro-cracks are determined by zones where the level set value is greater than the critical length, and damage is equal to one. In a domain discretized with regular finite elements, these zones need to be at least one row of elements wide to represent this stress-free state; as a result, mesh dependency might be present [4]. In order to circumvent this issue, Bernard et al. [5] proposed an enrichment strategy for those elements that are cut by the iso-critical curve of the level set field, which allows for a discontinuity in the strain field across such iso-critical curve.

One application where the robustness of the TLSV1 has been a significant advantage is the simulation of cusp formation in resin-rich regions of composite materials loaded in mode II loading conditions [4,6,7]. For this particular problem, Van der Meer and Sluys [4] showed the necessity of using an asymmetric constitutive law with different behavior under tension and compression in order to avoid unrealistic 'X-shaped' crack configuration in the material loaded in shear. On the other hand, they also found that this constitutive model leads to unrealistic stiffness recovery and stress transfer across the crack along material interfaces. In order to allow for traction-free sliding deformation, Van der Meer and Sluys [4] proposed a special interphase constitutive law that prevents stiffness recovery on the direction of the material interface.

The TLS has been compared with alternative approaches, such as the phase-field method [8]. It is important to emphasize that the same issues found in the shear test case by Van der Meer and Sluys [4] with symmetric and asymmetric constitutive models may be present in simulations with phase-field models equipped with the same constitutive models.

More recently, a new version of the TLS, referred to as TLSV2 hereafter, has been proposed by Lé et al. [9]. The main objective of this method is to couple both continuum damage modeling and cohesive zone modeling within a single framework, and consequently, profit from the advantages of both: the directionality for crack propagation as well as crack branching and merging ability of the former, and the capacity to model discrete cracks of the latter. Building on the basic premise of the TLS, both damage variables are described by the same level set field. An advantage that has been highlighted by Lé et al. [9] with respect to the TLSV1 is the possibility to introduce complex interfacial behavior at the crack faces, such as frictional contact. To this we would like to add the possibility to describe free sliding deformation without need to include information of the orientation of a nearby material interface in the constitutive relation.

The TLSV2 method evaluates the cohesive forces and displacement jump on the so-called skeleton curve, whose representation is dictated by the level set field. It is important to emphasize that Lé et al. [9] only investigated test cases with trivial skeleton curves at *a priori* known locations. A general implementation of the TLSV2 requires the extraction of the skeleton curve from an arbitrary level set field with corresponding damage front.

The objective of this paper is to introduce a more general framework for the TLSV2 method. The main requirements that are addressed are the extraction of free-form skeleton curves, and the construction of a discontinuity in the displacement field at the position of that skeleton as the analysis progresses. The concepts of the original paper on TLSV2 by Lé et al. [9] are taken as starting point. To determine the location of skeleton curve, an algorithm based on a combination of ball-shrinking [10,11] and graph-based algorithms is designed. The resulting skeleton curve is mapped onto the finite element mesh in order to define the cohesive segments, by determining intersection points between the skeleton curve and finite element edges. Subsequently, the cohesive segments are used to define overlapping elements that are used in a phantom node approach [12,13] in order to evaluate the cohesive contribution of the TLSV2. An ad hoc approach to compute the averaged values of local quantities related to crack growth is introduced. In addition, a generalization of two constitutive models for the bulk is introduced to profit from the advantages of both.

The paper is organized as follows. In Section 2, the main concepts of the TLSV2 method as proposed by Lé et al. [9] are outlined, highlighting the differences with the TLSV1. Section 3 is devoted to the algorithm for obtaining the skeleton curve for a given level set field. In Section 4, the main features of the phantom node method are outlined. Several numerical examples are presented in Section 5 to assess the accuracy of the proposed model, and to demonstrate its ability to deal with various crack growth scenarios. Finally, conclusions are presented in Section 6.

2. The Thick Level Set V2 method

This section is dedicated to outlining the main features of the TLSV2 method. As in the TLSV1 method, the location of the damage front Γ_0 is tracked as the zero level set (or the 'iso-0') of an auxiliary field $\phi(\mathbf{x})$, the level set field, which is constructed on the entire domain Ω as the signed distance function to Γ_0, such that the level set field satisfies the eikonal equation [2]:

$$\|\nabla\phi\| = 1 \quad \text{on} \quad \Omega. \tag{1}$$

On a discretized finite element domain, the definition of ϕ at a given point $\mathbf{x}$ is determined by interpolating the values of ϕ from nodes to $\mathbf{x}$ using finite element shape functions.

The TLSV2 implementation in this study inherits the main framework that has been adopted in earlier TLSV1 models [4–7,9]: a staggered solution scheme in which displacements and damage are computed separately, as schematically depicted in Fig. 1. More precisely, it inherits most of the sequential version detailed in [7] where every time step consists of three main modules (or *analysis phases*, as coined in [7]): `LSModule`, `EquilModule`, and `VelocityModule`. In Fig. 1, the highlighted functions indicate the new operations related to the TLSV2 method. For more background on the remaining functions, the reader is referred to [7].

The global solution scheme is described as follows. Firstly, the level set field is updated. As a new task in the `LSModule`, the skeleton curve is determined from the position of the damage front through the `skeletonizer` function. Subsequently, overlapping nodes and elements are introduced in order to accommodate at a later time the phantom node method, for which `updateMesh` is responsible. Then, ϕ is reinitialized, evaluation of damage initiation given the elastic strain field $\boldsymbol{\varepsilon}$ is performed, possibly leading to insertion of a new damage nucleus, and the size of damaged zones, $\bar{\phi}$, is computed, for which the `computePhiBar` function is responsible.

Next, with a given damage distribution associated with ϕ (which could initially consist of negative values throughout Ω to represent an undamaged specimen), the displacements, and consequently, strains and stresses are computed in a standard finite element analysis performed in the `EquilModule`. In this analysis phase, the `IntSchemeUpdate` function (see Fig. 1) is responsible for defining displacement degrees of freedom to the new nodes that has been created in the `updateMesh` function

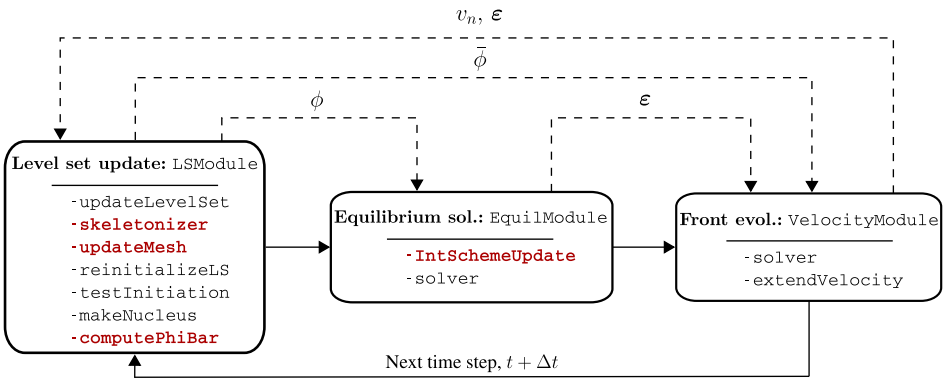


Fig. 1. The global sequential staggered solution scheme for a single time step. Dashed arrows represent the data exchange among the three modules (cf. [7]).

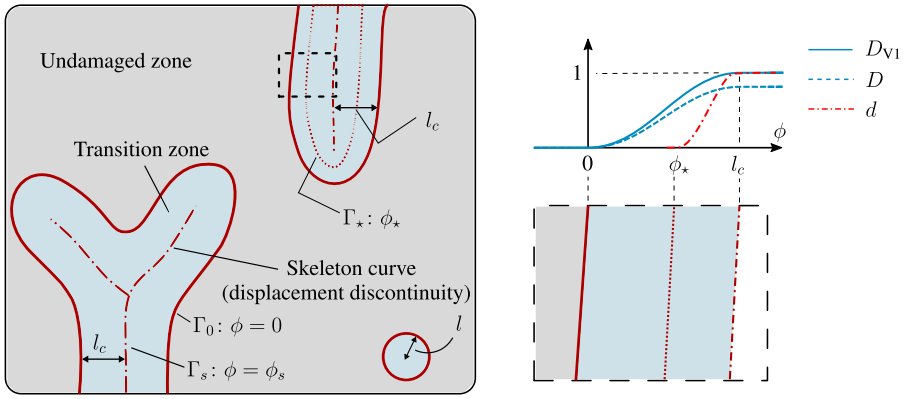


Fig. 2. The TLSV2, as the TLSV1, makes use of a single level set function to describe multiple damaged zones. As illustrated on the right, the damage variables D and d related to the bulk and interfacial models are functions of ϕ and ϕ_s, respectively. For comparison, the damage variable associated to the TLSV1, D_{V1}, is shown.

and for updating the integration scheme by means of a subtriangulation scheme for those elements that contain the iso-0 or skeleton curves.

Finally, the displacement field and $\bar{\phi}$ are used to compute the configurational force. In this part, the actual load for a time step is determined by scaling the unit-load solution with a load factor, γ, such that the maximum scaled value for the non-local averaged configurational force, $\bar{Y}$, equals the averaged material resistance to damage growth, $\bar{Y}_c$, at one point along the front without exceeding it anywhere. With $\bar{Y}$ and $\bar{Y}_c$ along the front, the front velocity, v_n, is computed, which is subsequently extended throughout Ω by means of a fast marching algorithm. Knowing v_n everywhere, the resulting new ϕ can be obtained and used for the next time step.

Apart from the phantom node method in the equilibrium solution phase, the two functions `reinitializeLS` and `extendVelocity` in `LSModule` and `VelocityModule` that are, respectively, responsible for the reinitialization of ϕ and front velocity extension, as well as the two functions `computePhiBar` and `solver` in the same modules that are, respectively, responsible for the computation of $\bar{\phi}$, and the non-local fields ($\bar{Y}$ and $\bar{Y}_c$), are performed on the mesh that has been updated by the `updateMesh` function.

2.1. Damage definition

In the TLSV2, there are two damage variables instead of one, introducing an interfacial damage variable d next to the bulk damage variable D from the TLSV1. In line with the TLS concept, both damage variables are defined as functions of a unique level set field ϕ, as illustrated in Fig. 2. Both damage variables are constrained to follow user-defined profiles within material layers with distinct widths and bounds. As such, the TLSV2 couples both bulk and cohesive damage models within a single framework.

Unlike to the TLSV1 method, where D varies from zero to one as ϕ goes from zero to l_c, D in the TLSV2 no longer reaches one at $\phi = l_c$. Mathematically, the continuum damage variable is expressed by:

$$D(\phi) = \begin{cases} 0, & \phi \le 0 \\ Q(\phi), & 0 < \phi \le l_c \\ Q(l_c), & \phi > l_c \end{cases}, \tag{2}$$

where Q is a function that has the properties of $Q(0) = 0$, $Q(l_c) < 1$ and $Q' \geq 0$. As D is always lower than one, stress-free deformation in the bulk is not possible. In the middle of the damage band, *i.e.*, on the skeleton curve of the level set field, a displacement discontinuity is included, with cohesive tractions acting across it related to a second damage variable, d. The cohesive damage variable depends also on ϕ, more precisely on ϕ evaluated at the skeleton curve, Γ_s, $\phi_s = \phi(\mathbf{x}_s)$, with $\mathbf{x}_s$ being a point on Γ_s. The following set of equations is used for the relation between d and ϕ_s:

$$d(\phi_s) = \begin{cases} 0, & \phi_s \leq \phi_\star \\ q(\phi_s), & \phi_\star < \phi_s \leq l_c \\ 1, & \phi_s > l_c \end{cases}, \tag{3}$$

where q is a function that has the properties of $q(\phi_\star) = 0$, $q(l_c) = 1$ and $q' \geq 0$, and $\phi_\star$ is a user-defined constant. Observe that d does reach one at $\phi_s = l_c$ to allow for a fully degraded material.

Note that the cohesive zone model in the TLSV2 method changes in three stages as ϕ (and consequently ϕ_s) evolves. When $\phi_s < \phi_\star$ and $\phi > 0$, the TLSV2 behaves equivalently to the TLSV1, hence, $D \geq 0$, $d = 0$, and there is no cohesive forces acting in the system. The displacement discontinuity with cohesive tractions arises at $\mathbf{x}_s$, and d kicks in when $\phi_s > \phi_\star$. Finally, the crack is traction-free when $\phi_s = l_c$, *i.e.*, when the damage band has a width of $2l_c$.

In order to fulfill the conditions in Eqs. (2) and (3), the following equations for Q and q are, respectively:

$$Q(\phi) = \eta f(\phi) \tag{4}$$

and

$$q(\phi_s - \phi_\star) = f(\phi_s - \phi_\star), \tag{5}$$

where f is an arc-tangent formula given by [4–6]:

$$f(\phi) = c_2 \arctan\left(c_1\left(\frac{\phi}{l_c} - c_3\right)\right) + c_4, \tag{6}$$

with $c_1 = 10$ and $c_3 = 0.5$ and the other coefficients given by: $c_4 = -c_2 \arctan\left(-c_1 c_3\right)$ and $c_2 = \left(\arctan\left(c_1\left(1 - c_3\right)\right) - \arctan\left(-c_1 c_3\right)\right)^{-1}$ [5]. The user-defined parameter $0 < \eta < 1$ in Eq. (4) defines the value for $Q(l_c)$ in Eq. (2). Observe that q in Eq. (5) is shifted to the right on the horizontal axis by making the argument of f equal to $(\phi_s - \phi_\star)$ (see Fig. 2). Note that Lé et al. [9] used a different strategy for obtaining Q and q based on a 1D model.

2.2. Equilibrium problem

For the present study, the following potential energy definition is adopted:

$$E(\mathbf{u}, \phi) = \int_\Omega \Psi(\boldsymbol{\varepsilon}(\mathbf{u}), D(\phi))\, \mathrm{d}\Omega + \int_{\Gamma_s} \psi(\llbracket \mathbf{u} \rrbracket, d(\phi_s))\, \mathrm{d}\Gamma_s - \int_{\Gamma_N} \mathbf{t}_N \cdot \mathbf{u}\, \mathrm{d}\Gamma_N, \tag{7}$$

where $\boldsymbol{\varepsilon}$ is the elastic tensor defined as the symmetric part of the gradient of the displacement field $\mathbf{u}$, Γ_N is the surface on which Neumann boundary conditions are considered, and $\llbracket \mathbf{u} \rrbracket$ is the displacement jump over the crack surface Γ_s.

Following the previous TLS models [1,4,5], the free energy for the bulk part is adopted:

$$\Psi(\boldsymbol{\varepsilon}, D) = \mu(1 - \alpha_i D)(\varepsilon_i)^2 + \frac{\lambda}{2}(1 - \alpha_v D)\mathrm{tr}(\boldsymbol{\varepsilon})^2, \tag{8}$$

where λ and μ are Lamé's elastic constants, ε_i the principal strain values and $\mathrm{tr}(\boldsymbol{\varepsilon})$ the trace of $\boldsymbol{\varepsilon}$, and

$$\alpha_i = \begin{cases} 1, & \varepsilon_i > 0 \\ \beta, & \varepsilon_i < 0 \end{cases}, \quad \text{and} \quad \alpha_v = \begin{cases} 1, & \mathrm{tr}(\boldsymbol{\varepsilon}) > 0 \\ \beta, & \mathrm{tr}(\boldsymbol{\varepsilon}) < 0 \end{cases}, \tag{9}$$

with $0 \leq \beta \leq 1$ being a user-defined parameter that allows for equal stiffness loss under tension and compression ($\beta = 1$), full stiffness recovery under compression ($\beta = 0$) or anything in between. The influence of β on the global response of a shear test will be assessed in Section 5.3.

With Eq. (8), the stress–strain relation in principal stress space, and the driving force for damage growth can respectively be expressed as:

$$\sigma_i = \frac{\partial \Psi}{\partial \varepsilon_i} = 2\mu(1 - \alpha_i D)\varepsilon_i + \lambda(1 - \alpha_v D)\mathrm{tr}(\boldsymbol{\varepsilon}) \tag{10}$$

and

$$Y = -\frac{\partial \Psi}{\partial D} = \mu\alpha_i(\varepsilon_i)^2 + \frac{\lambda}{2}\alpha_v \mathrm{tr}(\boldsymbol{\varepsilon})^2. \tag{11}$$

Regarding the cohesive part, the free energy based on the displacement jump expressed in the local frame (n, s) defined on the crack face with normal and shear components (see Fig. 12), $\llbracket \bar{\mathbf{u}} \rrbracket = \{\llbracket \bar{u} \rrbracket_n; \llbracket \bar{u} \rrbracket_s\}^{\mathrm{T}}$, is considered [14]:

$$\psi(\llbracket \bar{\mathbf{u}} \rrbracket, d) = (1 - \bar{\alpha}_{ni} d)\frac{1}{2} K \llbracket \bar{u} \rrbracket_i^2, \tag{12}$$

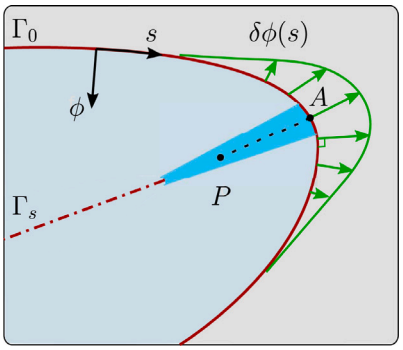


Fig. 3. The curvilinear coordinate system (ϕ, s). Point P at (ϕ_P, s) is affected as point A at $(0, s)$ on the front experiences a front advance.

where the positive constant K is a penalty stiffness, and $\bar{\alpha}_n$ is defined as:

$$\bar{\alpha}_{ni} = \begin{cases} 1, & [\![\bar{u}]\!]_n > 0 \\ (1 - \delta_{ni}), & [\![\bar{u}]\!]_n < 0 \end{cases} \tag{13}$$

so that the possibility of interpenetration between crack faces is prevented by contact along the n-axis, where δ_{ij} is the Kronecker delta. Hence, the state equations associated with the cohesive model become:

$$\bar{t}_i = \frac{\partial \psi}{\partial [\![\bar{u}]\!]_i} = (1 - \bar{\alpha}_{ni} d) K [\![\bar{u}]\!]_i, \tag{14}$$

and

$$y = -\frac{\partial \psi}{\partial d} = \frac{1}{2} \bar{\alpha}_{ni} K [\![\bar{u}]\!]_i^2. \tag{15}$$

2.3. Configurational force

The non-locality of the TLS method appears when a portion δs of the damage front, Γ_0, moves outwards of a distance $\delta\phi$ [5] (see Fig. 3). Because of the signed distance function, all the points having the same curvilinear coordinate s[1] are affected when Γ_0 experiences this front advance. The equation that expresses the amount of dissipated energy as the front moves of $\delta\phi$ on δs can be obtained by differentiating Eq. (7) with respect to $\delta\phi$ (cf. [1,5]):

$$\delta E = -\int_\Omega Y D'(\phi) \delta\phi \, \mathrm{d}\Omega - \int_{\Gamma_s} y d'(\phi_s) \delta\phi_s \, \mathrm{d}\Gamma_s, \tag{16}$$

where D' is the derivative of D with respect to ϕ, and d' is the derivative of d with respect to ϕ_s.

Following Bernard et al. [5], we introduce an averaged quantity $\bar{Y}$ to construct a discretized measure for δE as a function of ϕ. Here, the cohesive part of the TLSV2 is added to the formulation, such that $\bar{Y}$ is the solution of the problem:

$$\int_{\Omega^\mathrm{d}} \bar{Y} D'(\phi) \hat{\bar{Y}} \, \mathrm{d}\Omega = \int_{\Omega^\mathrm{d}} Y D'(\phi) \hat{\bar{Y}} \, \mathrm{d}\Omega + \int_{\Gamma_s} y d'(\phi_s) \hat{\bar{Y}} \, \mathrm{d}\Gamma \quad \forall \hat{\bar{Y}} \in \bar{\mathcal{Y}}, \tag{17}$$

with $\bar{\mathcal{Y}}$ being the space of constant fields along the gradient of ϕ and along the coordinate s in the damaged domain Ω^d, *i.e.* in the region where $\phi > 0$.

This problem is discretized as a field on the nodes of those elements that are at least partially inside Ω^d with $\bar{Y}$ as unknown. The constraint that $\bar{Y}$ is constant along the level set gradient, *i.e.*, $\nabla\bar{Y} \cdot \nabla\phi = 0$, is weakly enforced with Lagrange multipliers. Galerkin's method is employed to find nodal values of $\bar{Y}$ from the discretized version of Eq. (17) giving rise to the following system of equation:

$$\begin{bmatrix} \mathbf{K} & \mathbf{L} \\ \mathbf{L} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \bar{\mathbf{Y}} \\ \mathbf{l} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}^\mathrm{Y} \\ \mathbf{0} \end{Bmatrix}, \tag{18}$$

in which $\bar{\mathbf{Y}}$ and $\mathbf{l}$ are vectors with $\bar{Y}$ and Lagrange multiplier degrees of freedom, respectively. The matrices and the right-hand side vector are defined as (cf. [4]):

$$K_{ij} = \int_{\Omega^\mathrm{d}} D' N_i N_j + \frac{\kappa h^2}{l_c} \frac{\partial N_i}{\partial x_k} \frac{\partial N_j}{\partial x_k} \, \mathrm{d}\Omega, \tag{19}$$

[1] The change in variable is obtained by means of $\mathrm{d}\Omega = \left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi \, \mathrm{d}s$, following Moës et al. [1]. However, in the final equations as implemented, the curvilinear system are not used, or even constructed.

$$L_{ij} = \int_{\Omega^{\mathrm{d}}} l_c \left(\frac{\partial N_i}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \left(\frac{\partial N_j}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \mathrm{d}\Omega, \quad \text{and} \tag{20}$$

$$f_i^{\mathrm{Y}} = \int_{\Omega^{\mathrm{d}}} N_i D' Y \, \mathrm{d}\Omega + \int_{\Gamma_s} N_i d' y \, \mathrm{d}\Gamma, \tag{21}$$

where N_i and N_j are the shape functions associated with nodes i and j, κ is a stabilization parameter, h is the length of the diagonal of the smallest possible rectangle around an individual element in the whole mesh, Y is the local configurational force which depends on the current elastic strain field through Eq. (11), and y is the configurational force related to the interfacial model evaluated through Eq. (15). It is interesting to note that this system of equation keeps the same solution procedure already found in several TLS-based models [4–6,15] simply adding the interfacial contribution in Eq. (21). This does come at the cost of losing the direct relation between $\bar{Y}$ and the energy released per unit crack growth as existed in the TLSV1. Observe that Lé et al. [9] made use of a different strategy to compute non-local fields; instead, they followed the approach based on approximation functions (modes), as explained in [16].

Finally, it is remarked that Eq. (16) as measure for total energy dissipation is only valid when it is assumed that no energy is dissipated where the displacement discontinuity is introduced, *i.e.* that the displacement jump is zero at $\phi_\star$. This is only accurate if K is chosen sufficiently high to mimic initially rigid behavior. However, very high K may affect how y is distributed along the cohesive crack [15,17]. Perhaps, an initially rigid formulation [13,18] can be adopted, but this is considered out of scope for the present investigation.

2.4. Front movement

The front propagation criterion at a given point at the front is obtained by comparing the averaged configurational force $\bar{Y}$ with $\bar{Y}_c$, the averaged value of a resistance parameter against the damage growth, Y_c. For the general case where Y_c can be a function of spatial coordinates, the averaged resistance $\bar{Y}_c$ is computed by solving a system of equations similar to Eq. (18), but replacing Y by Y_c, and $\bar{Y}$ by $\bar{Y}_c$ [4] while the surface contribution (over Γ_s) is left out. If a simulation has a single material and a single value of Y_c, the averaging procedure is not performed and $\bar{Y}_c = Y_c$. The change in the level set field is related to the normal velocity (or level set field increment), v_n, as:

$$\phi_i \leftarrow \phi_i + v_{ni} \tag{22}$$

with $i \in \mathcal{N}$, the complete set of nodes in the mesh.

Before computing the front increment along Γ_0, the configurational force $\bar{Y}$ has been computed with the displacement field from the unit load boundary condition. The actual load level for a time step is then determined by scaling the unit-load solution with a load scale factor γ such that the maximum scaled value for $\bar{Y}$ along the front is equal to $\bar{Y}_c$:

$$\gamma^2 \max_{i \in \mathcal{N}_0} \left\{ \frac{\bar{Y}_i}{\bar{Y}_{ci}} \right\} = 1, \tag{23}$$

where $\mathcal{N}_0$ is the complete set of nodes of those elements that contains the front. Finally, the front velocity v_n for every node in $\mathcal{N}_0$ is obtained through [4]:

$$v_{ni} = k \left\langle \frac{c\gamma^2 \bar{Y}_i}{\bar{Y}_{ci}} - 1 \right\rangle_+ \quad \text{with} \quad k = \frac{v_{\max}}{c-1}, \tag{24}$$

where $v_{\max}$ is the maximum growth the front can experience for a time step. Brackets $\langle \cdot \rangle_+$ are used to denote the positivity condition, which reflects the irreversibility of damage growth. In order to guarantee the numerical stability of the staggered scheme, a value $v_{\max} = \xi h$ is used in this paper, following [4]. The parameter c influences the spread of the front movement to nodes with lower values for the ratio $\bar{Y}/\bar{Y}_c$. For $c \to 1$, only the node with the highest value $\bar{Y}/\bar{Y}_c$ undergoes a front advance. On the other hand, for higher values of c, nonzero front movement is found in more nodes.

As Eq. (24) for v_n is only calculated along the front Γ_0, and due to the fact that the level set update with Eq. (22) requires the velocity to be known throughout the domain Ω, v_n has to be determined on Ω. Therefore, the velocity computed at the nodes of elements that contain the front is propagated through Ω by solving:

$$\nabla \phi \cdot \nabla v_n = 0, \tag{25}$$

which is carried out with a fast marching algorithm. In theory, the updated level set field obtained with Eq. (22) remains a signed distance function [2,3]. However, the discrete nature of the model may cause it to deviate from being an accurate representation of a signed distance function. Thus, a reinitialization procedure is periodically performed with another fast marching algorithm [4,19].

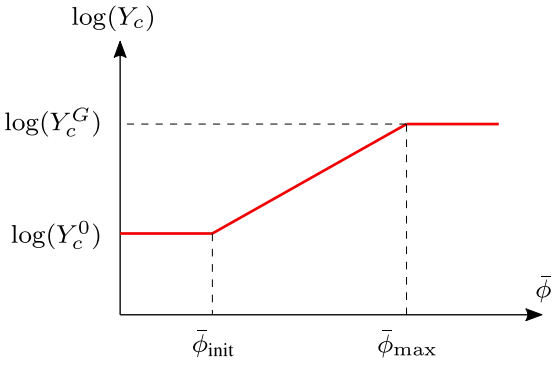


Fig. 4. Interpolation of Y_c between Y_c^0 and Y_c^G.
Source: Adapted from [15].

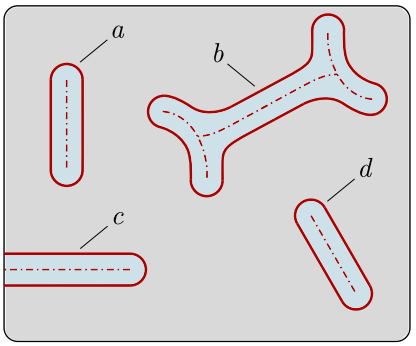


Fig. 5. Computation of $\bar{\phi}$ for four distinct damaged regions. For each of the four damaged regions, $\bar{\phi}$ is computed as the length of associated damage front.

2.5. Initiation

In order to take into account independent input parameters for damage initiation and crack propagation, Y_c is made into a function of $\bar{\phi}$, a quantity that measures the size of a damaged zone. With values for Y_c related to an initial strength-based value, Y_c^0, and to an energy-based value, Y_c^G, the following interpolation is adopted [15] (see Fig. 4):

$$\log(Y_c) = \log(Y_c^0) + \frac{\bar{\phi} - \bar{\phi}_{\text{init}}}{\bar{\phi}_{\text{max}} - \bar{\phi}_{\text{init}}} \left(\log(Y_c^G) - \log(Y_c^0)\right), \tag{26}$$

where $\bar{\phi}_{\text{init}}$ and $\bar{\phi}_{\text{max}}$ are, respectively, the initial size of the damaged zone and the size for which the damaged zone is considered a crack. The quantity Y_c^0 is related to the strength of the material, in which we follow the work in [4]:

$$Y_c^0 = \frac{f_t^2}{2E}, \tag{27}$$

where f_t and E are, respectively, the tensile strength and Young's modulus. Unlike to earlier studies in [4,5] where Y_c^G is a direct function of the fracture energy, G_c, and l_c, Y_c^G in this study is obtained through a fitting procedure based on mode I specimens so that the area under the load displacement curve produces an accurate value of G_c (see Section 5.1).

In contrast to what was proposed in [4], in which $\bar{\phi}$ is computed in a similar way as $\bar{Y}$ in Eq. (18), $\bar{\phi}$ is related to the front length of each closed damaged subdomain making use of additional information on the damage front that is anyway needed for the skeleton curve construction (see the function `computePhiBar` in Fig. 1). The computation of $\bar{\phi}$ encompasses two stages. First, for a given damage front, its length is computed, and its value is stored at the set of nodes of those elements which contains this damage front. Then, this nodal value is propagated in the same way the front velocity is extended in Eq. (25), *i.e.* by a fast marching algorithm (see Fig. 5); however, $\bar{\phi}$ is only propagated through the region where $\phi > 0$. As a result, we have a single value of $\bar{\phi}$ for each damaged subdomain.

In order to deal with damage initiation, the criterion $Y \geq Y_c^0$ is used. If this inequality is met at any undamaged point, a circle with radius $\phi_0 < l_c$ is inserted around the point $\mathbf{x}_{\text{nucl}}$ with the highest ratio Y/Y_c^0. Note that Y used in this criterion is purely local, but still consistent with the non-local growth criterion, since $\bar{Y} \to Y$ and $\bar{Y}_c \to Y_c$ as $l \to 0$.

3. Skeleton curve

In a setting where the damage front can evolve in arbitrary directions, the TLSV2 requires identification of the skeleton curve of the level set field. Several approaches for determining the location of the skeleton curve have been proposed in the literature, based on the direct information on non-uniqueness of the gradient of the level set field $\nabla\phi$ [20], and on Voronoi cell-based algorithms [21,22]. In this work, we make use of the ball-shrinking method proposed by Ma et al. [10] due to its implementation simplicity, and robustness.

This method relies on the concept of maximally inscribed ball (or *interior medial ball*) of a given region delimited by a contour (or surface), such as damaged regions bounded by the damage fronts found in the TLS-based methods. By definition of the signed distance function, the center of such maximal inscribed ball lies at the skeleton curve [10,11,21]. As such, the skeleton curve consists of the set of centers of all interior medial balls of a given region. In a discrete setting, one can obtain a point-based representation of a skeleton curve with a finite set of maximal inscribed balls. This method only requires the set of points that form Γ_0, *i.e.*, the iso-0 curve, and the gradient $\nabla\phi$ at these points as input.

Fig. 6 schematically illustrates the *Skeletonizer* algorithm that has been designed in order to determine the skeleton curve for the TLSV2 method. The Skeletonizer is comprised of three parts. Firstly, for a given accurate damaged subdomain enclosed by its iso-0 curve, a ball-shrinking algorithm is executed in order to obtain discrete points that lie at the 'theoretical' skeleton curve, which are called *atoms* or *atom points*, as depicted in Fig. 6(a). Secondly, these atoms are connected in a manner that corresponds to the connectivity of the skeleton curve, which is a first approximation of the skeleton curve, as illustrated in Fig. 6(b). This is done by means of a graph-based algorithm. Thirdly, once this approximate skeleton curve is fully known, it can be mapped onto the mesh in order to determine the intersection between the skeleton curve and the finite element edges, as shown in Fig. 6(c). These intersection points are used to construct the segments that are later used for the phantom node method.

The remainder of this section outlines these three parts of the Skeletonizer algorithm. For the sake of simplicity, Ω^{d} refers to a single damaged region enclosed by Γ_0, unlike elsewhere in this paper, where it refers to the union of all damaged regions. Moreover, let I_0 be the set of all the points that defines Γ_0 in a finite element domain, *i.e.* the points that are determined by the intersection between the iso-0 curve and finite element edges. Although the Skeletonizer algorithm is outlined through a single generic damaged domain, it can be applied to problems that consist of multiple damage fronts. In this case, one can apply the proposed skeletonizer algorithm for each damaged subdomain (for instance, see Figs. 5 and 13).

3.1. Ball-shrinking algorithm

The atom points are computed with the ball-shrinking algorithm proposed by Ma et al. [10] as improved by Peters [11]. The skeleton curve of Ω^{d} consists of the set of centers of all interior medial balls in Ω^{d}. According to Peters [11], a ball is an interior medial ball of Ω^{d} if it is maximal, namely, if a ball is a subset of Ω^{d} and any ball that contain such ball is not contained in Ω^{d}, as exemplified in Fig. 7. Furthermore, a medial ball does not contain any part of Γ_0; however, a medial ball does touch Γ_0 at least two points.

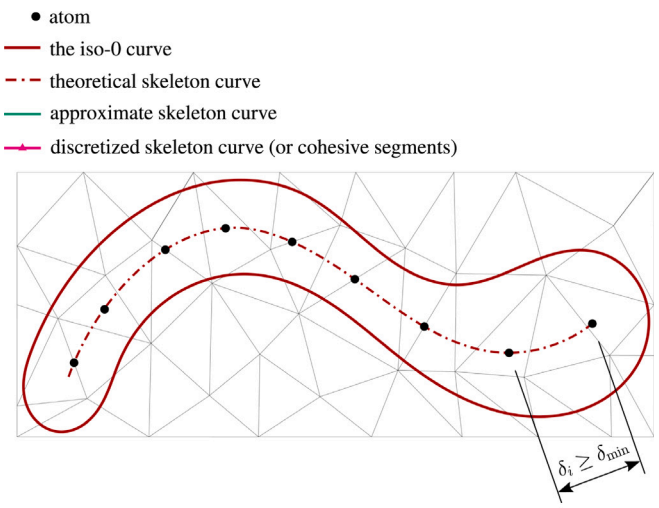
Fig. 8 and Algorithm 1 schematically show the ball-shrinking algorithm for a point $\mathbf{p}$ from I_0 with normal vector $\mathbf{n} = \nabla\phi$. The main premise of the algorithm is that the maximal inscribed ball associated with point $\mathbf{p}$ must have its center point on line L, which is the line through $\mathbf{p}$ parallel to $\mathbf{n}$. To find this center point, an initial huge ball is iteratively shrunk along the line L until an interior medial ball is obtained. The algorithm constructs a new candidate ball that is smaller than the previous one and closer to the final interior medial ball at each iteration [11]. Initially, a huge ball with radius r_{init} is generated touching $\mathbf{p}$. For the center of the current ball, $\mathbf{c}$, its closest point to Γ_0, denoted $\mathbf{q}_{\mathrm{next}}$, is computed. With $\mathbf{p}$, $\mathbf{n}$ and $\mathbf{q}_{\mathrm{next}}$, the ball touching the points $\mathbf{p}$ and $\mathbf{q}_{\mathrm{next}}$ is defined for the next iteration. The algorithm terminates when a maximal inscribed ball is found, *i.e.*, when it is not possible to shrink the ball any further (see line eight in Algorithm 1).

Algorithm 1 makes use of three special functions: `nearestNeighbour`, `computeRadius` and `computeCenter`. The `nearestNeighbour` function simply returns the closest point from the set I_0 to a given query point, which is efficiently implemented using a KD-tree data structure [24,25], represented by T_0 in Algorithm 1. Note that this tree-like data object has to be built prior to the iterative loop of the ball-shrinking algorithm [11]. The remaining two functions are schematically illustrated in Fig. 9. The function `computeRadius` determines the radius of a new ball for a given triplet: $\mathbf{p}$, $\mathbf{n}$, and $\mathbf{q}_{\mathrm{next}}$. The function `computeCenter` gives the center of a new ball for a given triplet: $\mathbf{p}$, $\mathbf{n}$, and r_{next}.

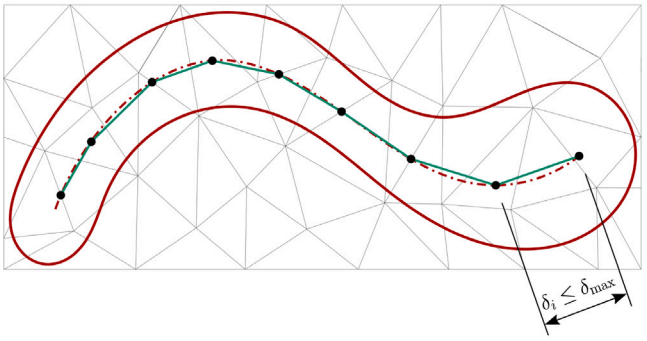
Building on the original algorithm by Ma et al. [10], Peters [11] proposed a denoise approach in order to avoid *spurious* interior medial balls due to the presence of noisy points at Γ_0. The main premise of this procedure is that even in the presence of a noisy point, a proper medial interior ball is usually found before shrinking it further to become a spurious one. In order to determine at which iteration the ball-shrinking algorithm has to be interrupted, Peters [11] suggested to use a separation angle, θ, observing that a point $\mathbf{q}$ that varies at each iteration may be shifted from one side to the other side of Γ_0; as a result, θ suddenly becomes smaller, as illustrated in Fig. 10.

The threshold angle θ_1 is used to detect a spurious ball to obtain a balance between robustness and sensitivity to small noise in Γ_0 [11]. Because exterior medial balls are not considered in this work, unlike in [11], only one such condition is included in the algorithm.

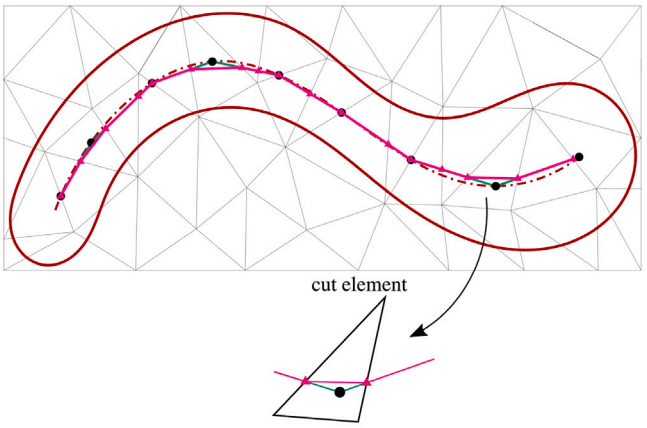
Note that Algorithm 1 computes the interior medial ball for a single point $\mathbf{q}$ in I_0. Hence, to obtain all the atoms associated to the region Ω^{d}, it is necessary to loop over all the points in I_0, as shown in Algorithm 2. In order to avoid overlapping atoms and atoms too close to one another (also avoiding elements with multiple atoms), a distance threshold, δ_{min} (see Fig. 6(a)), is imposed. Therefore, before storing a center of a medial ball this distance restriction is checked via `distNearestAtom` on line four in Algorithm 2, where the distance between a given center $\mathbf{c}$ and its closest existing neighbor atom is computed.



(a) Atom points.



(b) Approximate skeleton curve.



(c) Discretized skeleton curve (cohesive segments).

Fig. 6. The *Skeletonizer* algorithm for a given damaged region.

3.2. Approximate skeleton curve

In order to connect the atoms to form a skeleton curve (see Fig. 6(b)), a spanning-minimum-tree problem is solved. In this problem, a weighted and undirected graph is given, for which the smallest possible tree (an acyclic graph) is found which still connects all vertices of the original graph [26].

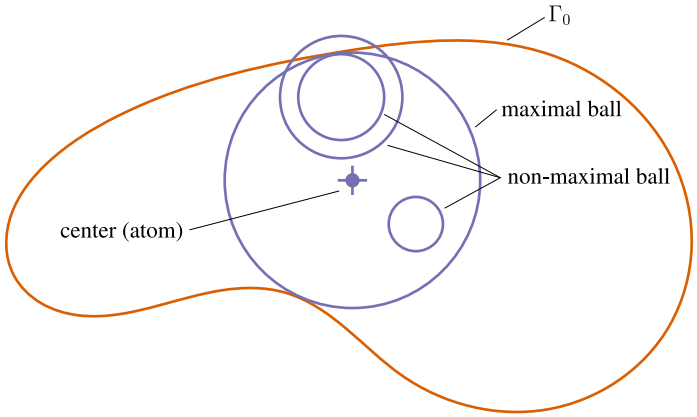


Fig. 7. Examples of maximal and non-maximal inscribed balls (disc in 2D).
Source: Adapted from Blum [23], as cited in Peters [11].

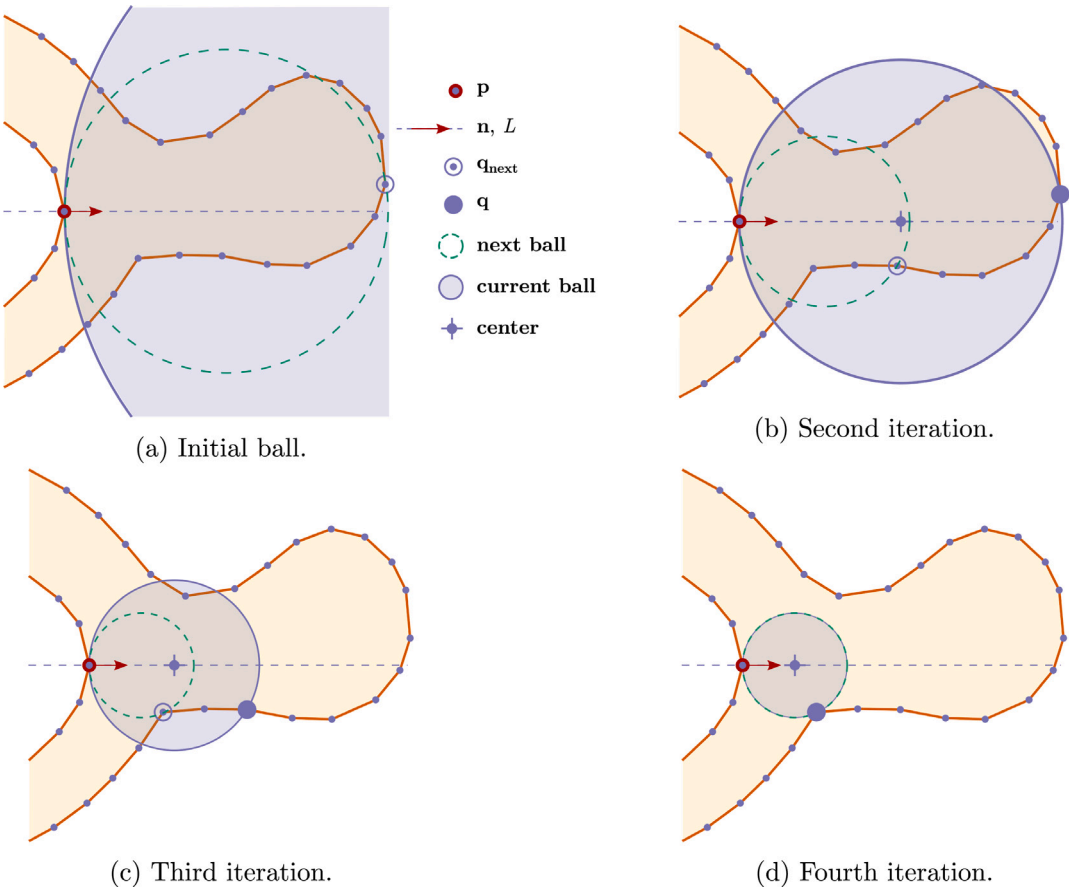


Fig. 8. Ball-shrinking algorithm: an interior medial ball is obtained at the fourth iteration.
Source: Adapted from [11].

Therefore, before solving this problem, a graph data type needs to be constructed, $G = (V, E)$, consisting of:

- a complete set of vertices, V, which corresponds to the atoms;
- a complete set of edges connecting the atoms, E. For each edge, an edge weight, the distance between two linked neighbor atoms, is set.

Algorithm 1: The `shrinkBall` algorithm (adapted from [11]).

Input : a set of points I_0 and its corresponding KD-tree data structure T_0, point $\mathbf{p} \in I_0$ and its normal vector $\mathbf{n}$, and denoising angle θ_1
Output: medial ball center $\mathbf{c}$ and radius r associated to $\mathbf{p}$

$i \leftarrow 0$
$r \leftarrow r_{init}$
$\mathbf{c} \leftarrow$ `computeCenter` $(\mathbf{p}, \mathbf{n}, r)$
while *true* **do**
  $\mathbf{q}_{\text{next}} \leftarrow$`nearestNeighbour` $(T_0, \mathbf{c})$
  $r_{\text{next}} \leftarrow$`computeRadius` $(\mathbf{p}, \mathbf{n}, \mathbf{q}_{\text{next}})$
  $\mathbf{c}_{\text{next}} \leftarrow$`computeCenter` $(\mathbf{p}, \mathbf{n}, r_{\text{next}})$
  if $r_{next} \geq r - \epsilon_{tol}$ **then**
    break
  end
  if $i > 0$ *and* $\theta_1 > \angle \mathbf{p}\mathbf{c}_{next}\mathbf{q}_{next}$ **then**
    break
  end
  $\mathbf{c} \leftarrow \mathbf{c}_{\text{next}}$
  $r \leftarrow r_{\text{next}}$
end

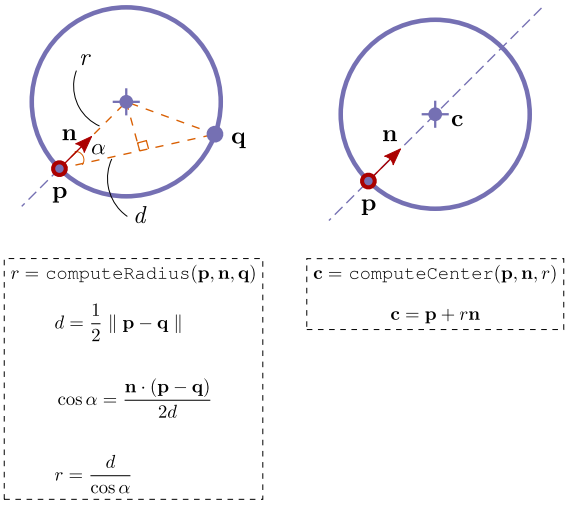


Fig. 9. The auxiliary functions, `computeRadius` and `computeCenter`, used in Algorithm 1.
Source: Adapted from [11].

Algorithm 3 shows how the minimum spanning tree associated with the atom points is built. The function `initAtomsGraph` on line one is responsible for connecting atoms whose outcome is a graph data representation of atoms. Once this graph is constructed, Prim's algorithm is used to obtain its corresponding minimum spanning tree, as represented by the function `primsAlgorithm` on line two. This greedy algorithm finds a minimum tree from a graph representation by constructing this tree one atom at a time, from an arbitrary starting atom, at each step adding the closest possible connected atom from the graph to another atom [24,26]. After determining the minimum spanning tree associated to atom points, a loop over such tree is executed in order to check if the edges of tree are too long through a maximum distance value for threshold edge selection, $\delta_{\max}$ (Fig. 6(b)), or are crossing Γ_0 at any point. The helper function `distEdge` computes the length of a given tree edge. If one of these conditions is met, the corresponding edge is removed from the tree via the function `removeEdge`.

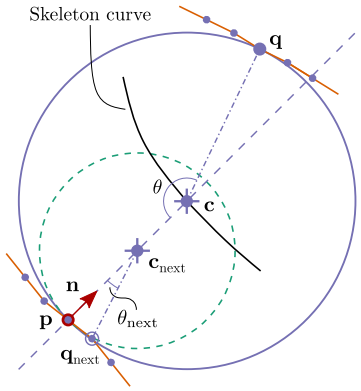


Fig. 10. Example of the concept of separation angle when a disk flips side, from one side of the Γ_0 to the other. The spurious ball can be detected by the small separation angle θ_{next}.
Source: Adapted from [11].

Algorithm 2: The `makeAtoms` algorithm.

Input : a set of points I_0
Output: interior medial balls store in M_{atom} for a region whose boundary are formed from all the points in I_0

$T_0 \leftarrow$ `makeKDTree` (I_0)
foreach $\mathbf{p} \in I_0$ **do**
  $\mathbf{c} \leftarrow$ `shrinkBall` $(\mathbf{p}, \mathbf{n}, T_0)$
  $\delta \leftarrow$ `distNearestAtom` $(M_{atom}, \mathbf{c})$
  if $\delta \geq \delta_{\min}$ **then**
    $M_{atom} \leftarrow$ `push` $(\mathbf{c})$
  end
end

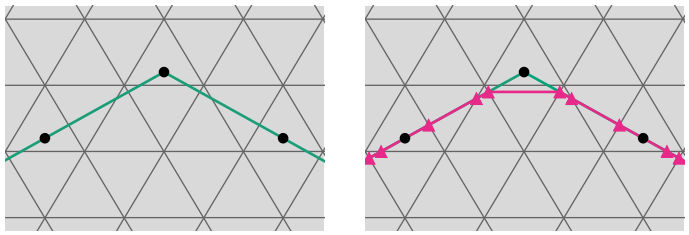
Algorithm 3: The `makeAtomsGraph` algorithm.

Input : a set of medial balls M_{atom}
Output: a graph representation of atoms, G

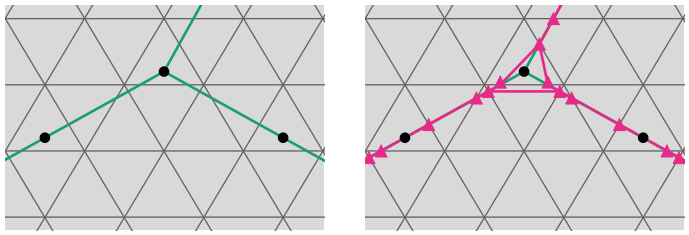
$G \leftarrow$ `initAtomsGraph` (M_{atom})
$G \leftarrow$ `primsAlgorithm` (G)
foreach *edge* $\in G$ **do**
  $\delta \leftarrow$ `distEdge` $(edge)$
  if $\delta \geq \delta_{\max}$ **or** *edge crosses* Γ_0 **then**
    `removeEdge` $(G, edge)$
  end
end

3.3. Discretized skeleton curve

The skeleton curve obtained with the previous two steps connects atoms that are generally not on element edges. However, for the phantom node method, we are looking to construct segments that cross elements in a single straight line. The location of the cohesive segments can be determined by projecting the approximate skeleton curve onto the finite element mesh (see Fig. 6(c)). This operation is performed in two stages. Firstly, crack branches (end-to-end, end-to-junction, or junction-to-junction branches) are generated from the connectivity of the approximate skeleton curve. Next, looping over these branches, the intersection points between the approximate skeleton curve and the elements are determined. As a result, the approximate skeleton curve is mapped on the mesh. This last stage can be efficiently carried out by a R-tree data structure used for spatial searching. An R-tree can store any set of objects (polygons, line segments, points, for instance), and it can give intersection queries with any other object. For an extensive overview of R-trees and their implementation, see [24,27].



(a) Corner element.



(b) Junction element.

Fig. 11. Example of corner and junction elements, which contain local cycles or loops. Triangle markers indicate intersection points between the approximate skeleton curve and finite element edges used to define the discretized skeleton curve.

The resulting intersection points and finite elements crossed by the cohesive segments can again be stored as a graph, in which the elements are treated as vertices and the intersection points are treated as edges. This storage scheme facilitates determining for any given finite element which intersection points it contains, and to which finite element it is connected.

Two points of extra attention exist in the element-intersection graph representation. First, one atom might give rise to a 'corner', as exemplified in Fig. 11. Second, unlike the graph representation of the approximate skeleton curve that has acyclical characteristic (see Fig. 13), the element-intersection graph representation might present local cycles at locations where one atom contains a junction, as exemplified in Fig. 11. Corners are considered to be elements with more than one intersection points lying on the same element edge (see Fig. 11(a)); on the other hand, junctions are considered to be elements containing more than two intersection points (see Fig. 11(b)). Furthermore, the approximate skeleton segment might cross the same element edge twice, which in turn produce two edges that both connect the same of pair of vertices since these intersection are stores as edges in the element-intersection graph. In order to deal with such scenarios, shortcuts are made in the discretized skeleton curve, as illustrated by the magenta curves in Fig. 11.

4. Phantom node method

Once the discretized skeleton curve has been located, *i.e.* when Γ_s has been determined, the phantom node method is used to introduce a discontinuity in the solution basis at that location. Fig. 12 schematically illustrates the phantom node method. When an element is crossed by a crack at Γ_s, such element is divided into two complementary subdomains, A and B, as illustrated in Fig. 12. Phantom nodes are added over the original ones, *i.e.* $\tilde{n}_1$, $\tilde{n}_2$ and $\tilde{n}_3$. Thus, the original element is substituted by two new elements, A and B, which have the exact same geometry but different connectivity. The shaded area indicates which part of a new element is active, Ω_A and Ω_B. The internal force vector and stiffness matrix contribution of each new element to the global system of equations are obtained by integrating only over the active part of the elements.

The discontinuous displacement field in the pair of overlapping elements is defined as:

$$\mathbf{u}(\mathbf{x}) = \begin{cases} \mathbf{N}(\mathbf{x})\mathbf{u}_A, & x \in \Omega_A \\ \mathbf{N}(\mathbf{x})\mathbf{u}_B, & x \in \Omega_B \end{cases}, \tag{28}$$

where $\mathbf{N}$ contains the standard finite element shape functions, and $\mathbf{u}_A$ and $\mathbf{u}_B$ are the nodal displacements from element A and B. The displacement jump over the crack is computed as the difference between the displacement fields of the two overlapping elements:

$$[\![\mathbf{u}]\!](\mathbf{x}) = \mathbf{N}(\mathbf{x})(\mathbf{u}_A - \mathbf{u}_B), \quad \mathbf{x} \in \Gamma_s. \tag{29}$$

Observe that Eq. (29) gives the displacement jump in the global frame, while the constitutive-related expressions in Eqs. (13)–(15) are evaluated in the local frame (n, s). The transformation from global to local coordinate frame is performed as:

$$[\![\bar{\mathbf{u}}]\!] = \mathbf{Q}[\![\mathbf{u}]\!], \tag{30}$$

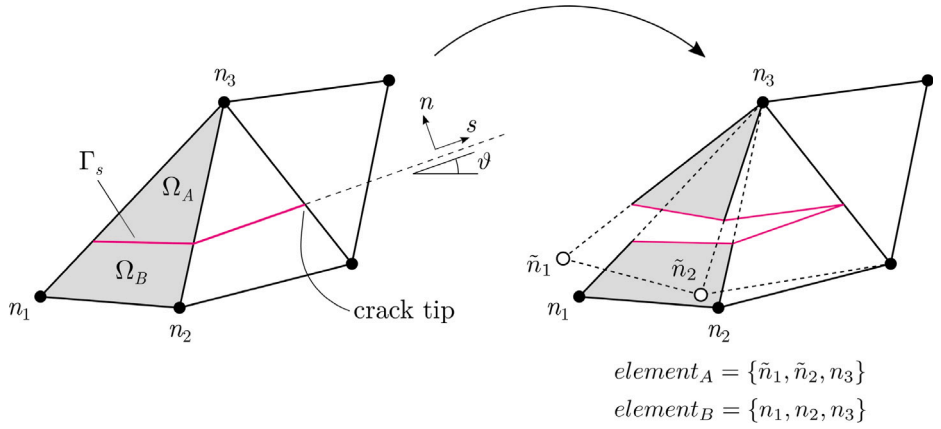


Fig. 12. Connectivity and active parts of two overlapping triangular finite elements in phantom node method. Local frame (n, s) and its orientation angle ϑ.

where the transformation matrix $\mathbf{Q}$ is given as:

$$\mathbf{Q} = \begin{bmatrix} -\sin\vartheta & \cos\vartheta \\ \cos\vartheta & \sin\vartheta \end{bmatrix}. \tag{31}$$

It is also possible to relate:

$$\bar{\mathbf{t}} = \mathbf{Q}\mathbf{t}. \tag{32}$$

More details on how the internal force vector and linearized stiffness matrix are assembled and their contribution to the global system of equations can be found in [12,13,28].

For a known discretized skeleton curve, possibly containing junctions and multiple disconnected parts, the phantom nodes and overlapping elements are generated in the `updateMesh` function (see Fig. 1). This is done by looping over the vertices of the element-intersection graph turning 'right' wherever it meets a junction. Fig. 13 gives a general view of how these loops are constructed. Looping along each of the dashed lines, the part of the elements on the right side of the skeleton curve is taken as the active part, while phantom nodes are introduced for all nodes on the left side. During this procedure, end vertices are regarded as anchor points, at which a crack path begins and ends. This approach facilitates consistency in the connectivity of neighboring pairs of elements. An example of implementation of these two tasks on the element-intersection graph and phantom node constructions, and shortcut operations in the discretized skeleton curve can be found in [29].

It can be observed from Fig. 13 that at junctions, small triangles are created. Elements are defined along the dashed lines, taking the closest magenta skeleton line as the edge of the active domain. As a consequence, the triangles inside the junctions do not belong to the active part of any element. It can occur that junction elements are split in three active parts (*e.g.*, at the junction of paths 1, 2, and 3 in Fig. 13). With the proposed procedure for defining the phantom nodes and overlapping elements, no special treatment is required to allow for presence of three or more overlapping element, except that no cohesive segments are introduced in elements with more than two overlapping elements. Near the junctions it may also happen that there are two overlapping elements that do not touch (*e.g.*, between paths 8 and 10 in Fig. 13). Also in this case, no cohesive element is introduced. In other words, elements that contains junction, their contribution to the system of equation for equilibrium solution phase and to the system of equation in Eq. (18) are not taken into account.

5. Results and discussion

The performance of the proposed TLSV2 method is assessed with numerical examples in this section. The method has been developed with Jem/Jive [30], which is an open-source toolkit for finite element analysis, along with the Boost Graph Library [24], which contains minimum spanning tree algorithms, tools to manage graph data structures, and R-tree functionality.

For all numerical examples, unstructured meshes of linear triangles generated with Gmsh [31] are considered. For nucleation, the size of a new damage nucleus ϕ_0 is about the effective element size. The stabilization parameter from Eq. (18) is set to $\kappa = 1.2$. Furthermore, the constant ξ used to compute $v_{\max}$ in Eq. (24) is set to $\xi = 0.5$, the constants $\bar{\phi}_{\text{init}}$ and $\bar{\phi}_{\max}$ for Y_c-interpolation expression in Eq. (26) are, respectively, set to $\bar{\phi}_{\text{init}} = 0$ and $\bar{\phi}_{\max} = 2\pi l_c + 2l_c$. The compression factor in Eq. (4) is $\eta = 0.92$, and the user-defined level set value $\phi_\star = 0.5l_c$ (see Eqs. (3) and (5) and Fig. 2). The penalty stiffness is $K = 8 \times 10^4\ \text{N/mm}^3$ (see Eqs. (12), (14) and (15)).

Regarding the parameters used for the Skeletonizer algorithm, the denoising angle is $\theta_1 = 125°$, the initial ball has $r_{\text{init}} = 50l_c$ (see Algorithm 1). Besides, the threshold parameter $\delta_{\min}$ is about three times the effective element size, *i.e.*, $\delta_{\min} \approx 3h$, and $\delta_{\max} \approx 4\delta_{\min}$ (see Algorithms 2 and 3 and Fig. 6).

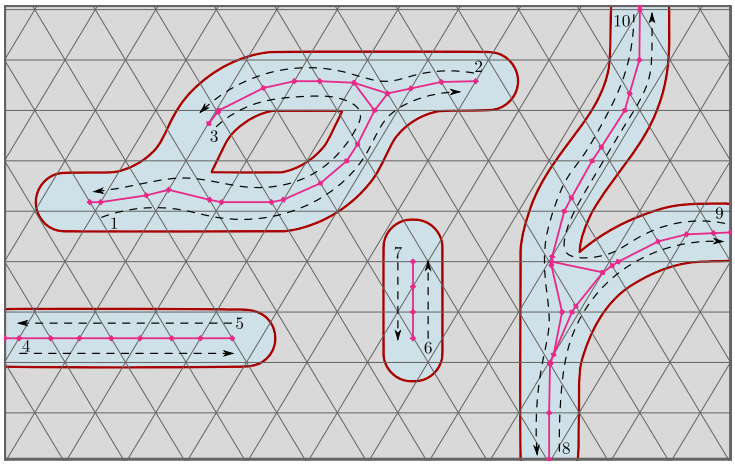


Fig. 13. Example of phantom node construction. The top left damaged subdomain illustrates the 'global' acyclical property of the atom graph after executing Prim's algorithm.

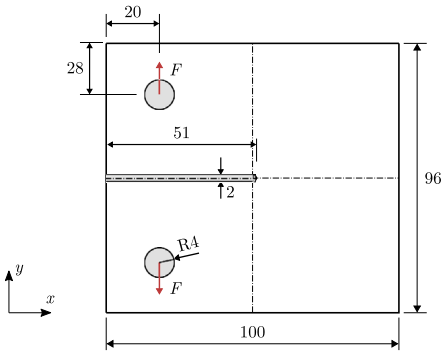


Fig. 14. Compact tension test: boundary condition and geometry (dimensions in mm).

5.1. Compact tension test

As a first example, the response of a compact tension test under plane stress is considered. Boundary conditions and geometry of the specimen are shown in Fig. 14. Young's modulus, Poisson's ratio, and tensile strength are, respectively, $E = 7000\,\text{MPa}$, $\nu = 0.3$, $f_t = 79\,\text{MPa}$. The critical length l_c is equal to 3 mm. This example is performed with $c = 2$ (see Eq. (24)). A refined mesh, with effective element size $h = 0.25\,\text{mm}$, is applied in the region where the crack is expected to develop. The nucleation check is only performed around the notch tip. As this example is a tension dominated problem, the parameter β in Eq. (9) is equal to one, which associated with Eq. (8) lead to a symmetric material law that presents the same behavior in tension and compression.

In Fig. 15, the load–displacement curves from Van der Meer and Sluys [4], obtained with a cohesive zone analysis with $G_c = 40\,\text{N/mm}$, and the TLSV2 are drawn together, and the deformed specimen for the final time step is shown. The result verifies the accuracy of the proposed model. Unlike earlier TLS studies [4,5], in which the parameter related to crack growth Y_c^G (see Eq. (26)) was a direct function of the fracture energy G_c, and l_c, Y_c^G in this work is obtained via a fitting procedure at $Y_c^G = 15.6\,\text{MPa}$. The match with cohesive zone analysis in the descending branch shows that during crack propagation the effective fracture energy obtained with the TLSV2 is constant.

Fig. 16 shows the load–displacement graphs for different values of l_c: 2 mm, 3 mm, and 4 mm. Observe that the curves (the graph on the left-hand side) obtained with l_c equal to 2 mm and 4 mm, and with the fixed value of $Y_c^G = 15.6\,\text{MPa}$ drift away from the corresponding simulation with $l_c = 3\,\text{mm}$, since they are simulated without their corresponding fitted Y_c^G-value.

The graph on the right-hand side shows results from simulations where Y_c is varied along with l_c in a inversely proportional way. The inverse proportionality is based on the fact that Y_c^G is a direct function of G_c and l_c in TLSV1 models (cf. [4,5], where $Y_c^G = \frac{G_c}{l_c}$). It can be observed that also with TLSV2, the same effective fracture energy is obtained for different combinations of Y_c^G, and as long as the product of the two is equal.

Fig. 17 shows the two fields of the averaged quantities $\bar{Y}$ and $\bar{Y}_c$ obtained through Eq. (18). It is important to emphasize the consistency of these results, in which $\bar{Y}$ is zero in the region behind the crack tip, indicating the presence of a traction-free crack and $\bar{Y}_c$ does reach the constant maximum value of 15.6 MPa, the value set in Y_c^G.

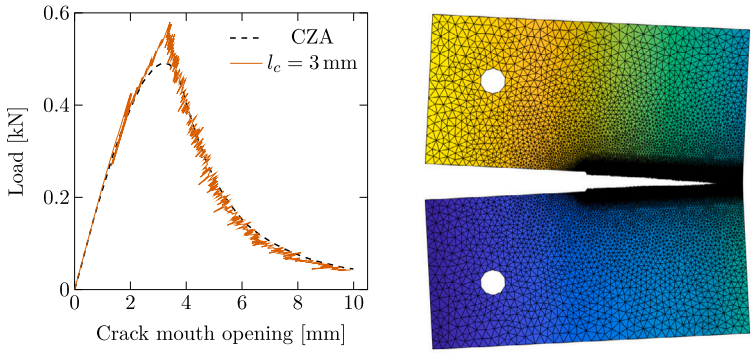


Fig. 15. Compact tension test: load–displacement curve in comparison with the cohesive zone analysis (CZA) as presented in [4] (left), and deformed specimen with plotted vertical displacement field (right) considering $l_c = 3\,\text{mm}$.

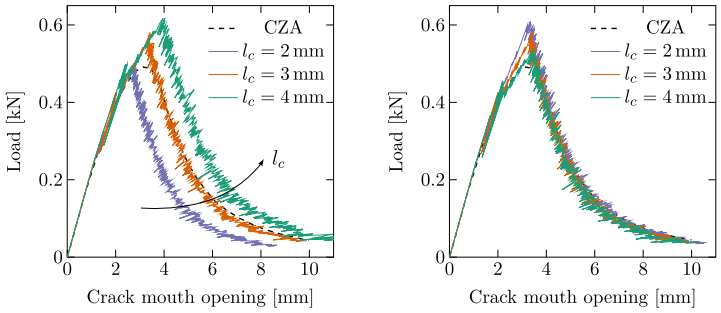


Fig. 16. Compact tension test: load–displacement curves with fixed values of $Y_c^G = 15.6\,\text{MPa}$ (left), and with inversely proportionally adapted values of Y_c^G (right).

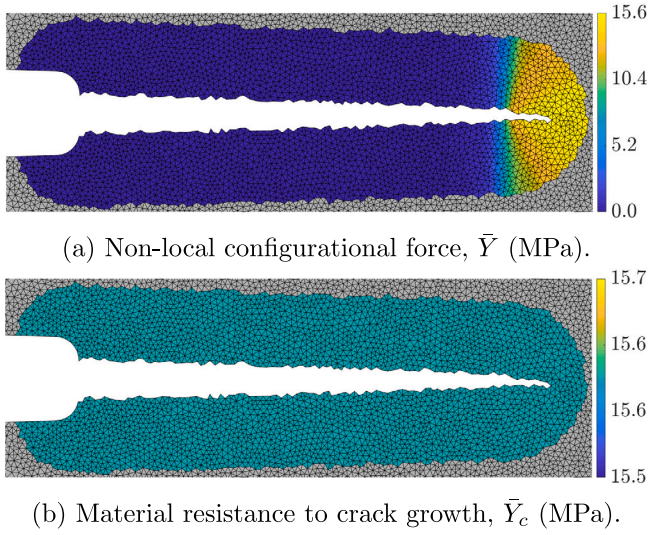
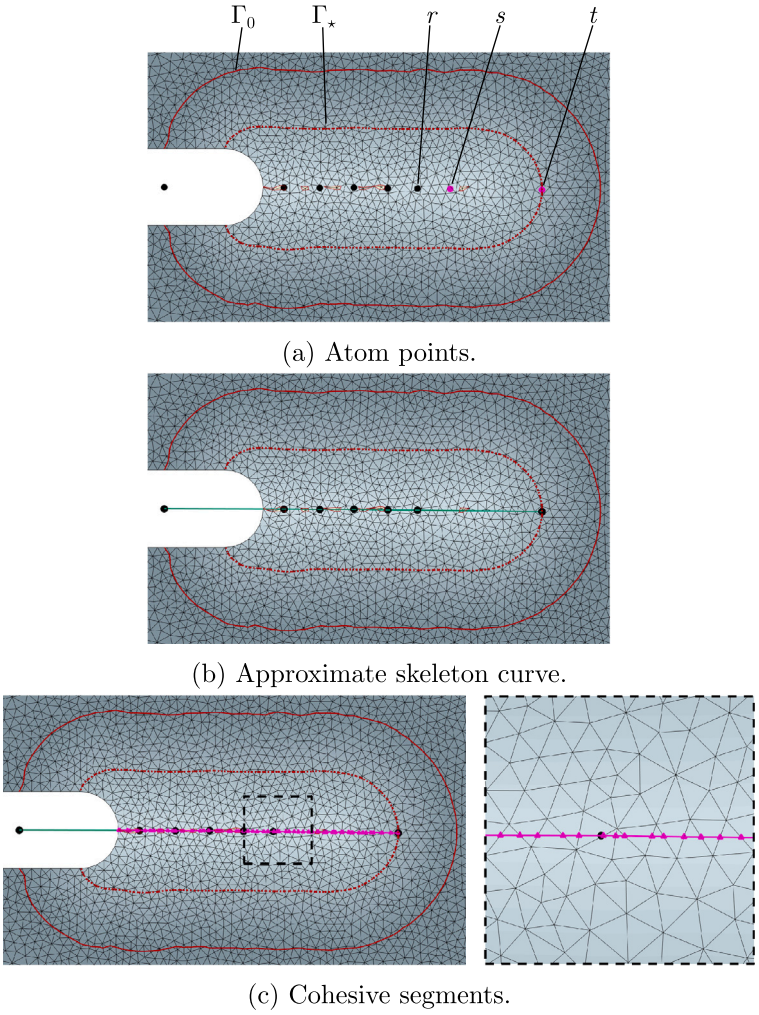


Fig. 17. Compact tension test: averaged quantities at given load step in the post-peak response.

Fig. 18 shows the outcome of main stages of the Skeletonizer algorithm for one time step from the compact tension simulation. Note that the last atom point obtained by Algorithm 2 is the atom s located at $\mathbf{c}_s$; as a result, it would have a lack of cohesive segments between s and the curve $\Gamma_\star$, which in turn gives rise to a non-gradual evolution of traction forces in this region. In order to avoid this absence of cohesive segments in this region, the atom s is moved until it reaches $\Gamma_\star$, whose new location is indicated by the atom t located at $\mathbf{c}_t$. This shifting procedure is accomplished by moving s on the line oriented by the normal vector defined as $\mathbf{n} = \frac{\mathbf{c}_s - \mathbf{c}_r}{\|\mathbf{c}_s - \mathbf{c}_r\|}$; therefore, the intersection point between such line and $\Gamma_\star$ defines the location of t.



(a) Atom points.

(b) Approximate skeleton curve.

(c) Cohesive segments.

Fig. 18. Compact tension test: stages of Skeletonizer algorithm with iso-0, iso-$\phi_\star$, and iso-l_c curves of the level set field.

5.2. Modified compact tension test

In the second example, the compact tension test is modified, as shown in Fig. 19, in order to show the ability of the proposed framework to deal with crack branching and consequently a junction in the skeleton curve. This numerical example is inspired by crack branching case studied by Moës et al. [1] in which the specimen is made of two different Young's moduli. The soft material has the same properties used previously in the first example, and for the stiff material, the properties are $E = 70\,000$ MPa, $\nu = 0.3$, and $Y_c^G = 468$ MPa. The critical length l_c is equal to 1 mm, and the parameter c is increased, $c = 4$, in order to promote more modes along the damage front. The typical element size h is equal to 0.15 mm around the region where the crack is expected to develop. Making use of the previously calibrated value and because Y_c^G is inversely proportional to l_c, Y_c^G is equal to 46.8 MPa in this example.

Fig. 20 shows the load–displacement curve and crack evolution along with its approximate skeleton curve for some time steps. The crack is initiated in the soft material and grows without touching the region where the stiff material is defined.

Fig. 21 shows the deformed specimen and a close-up figure of the junction. Crack branching naturally takes place as the iso-0 curve evolves, and consequently, its corresponding skeleton curve. This case would clearly be more complicated to represent with classical XFEM-based models (see Fig. 22).

5.3. Rail shear test

The final example is a case where traction-free sliding deformation is desired. The case is inspired by the plane strain rail shear test for mode II failure analysis following Van der Meer and Sluys [4]. The case consists of a weak core sandwiched between two stiff arms, as illustrated in Fig. 23. The arms are loaded in opposite direction so that the core is sheared. This setup leads to the

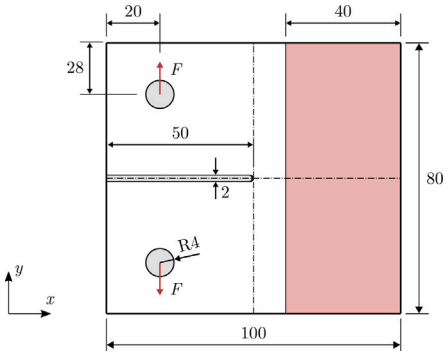


Fig. 19. Modified compact tension test: boundary condition and geometry (dimensions in mm). Colored area indicates where the stiffer material is set.

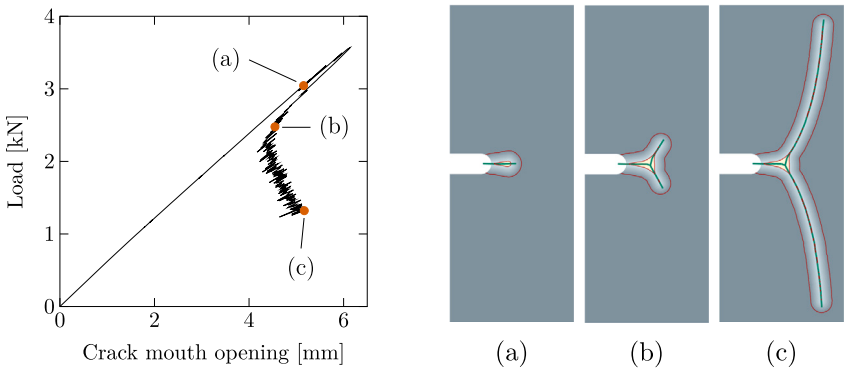


Fig. 20. Modified compact tension test: load–displacement curve, and crack evolution (close-up) and its corresponding approximate skeleton (green curve).

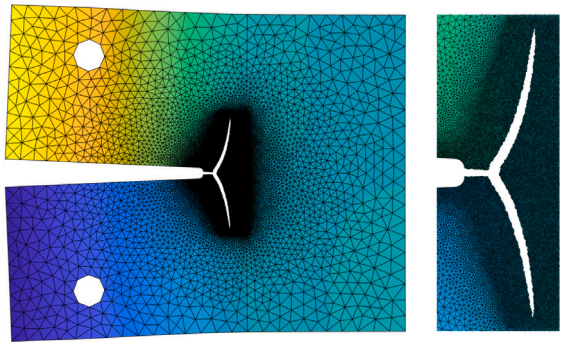


Fig. 21. Modified compact tension test: deformed specimen (left), and a close-up figure of the crack (right).

crack distribution observed in experimental tests of cusps forming in resin-rich regions of composite material in mode II loading conditions.

Young's modulus, Poisson's ratio, and tensile strength of the core material are the same as mentioned in the first example in this paper. For the face material, the properties are $E = 2.1 \times 10^3$ GPa (three hundred times stiffer than the core material), $\nu = 0.3$, and $f_t = 79$ MPa. The typical element size h is 0.15 mm throughout the core region, and l_c and Y_c^G are, respectively, 1 mm and 46.8 MPa. This simulation is performed with $c = 2$, the same value used in the compact tension test.

5.3.1. Single damage nucleus

Following the approach used to investigate a similar setup with TLSV1 in [4], the problem is first considered to have an initial damage nucleus at the mid-height plane of the core, 15 mm from the left side edge, with radius 0.95 mm, which is slightly smaller than l_c, in order to assess the material laws obtained through Eq. (8) by varying the parameter β in Eq. (9).

Three different choices for β are investigated. Firstly, β is set to one, which gives rise to a symmetric constitutive model for the bulk in the sense that stiffness degradation is equal under tension and compression. Fig. 24 shows the load–displacement curve

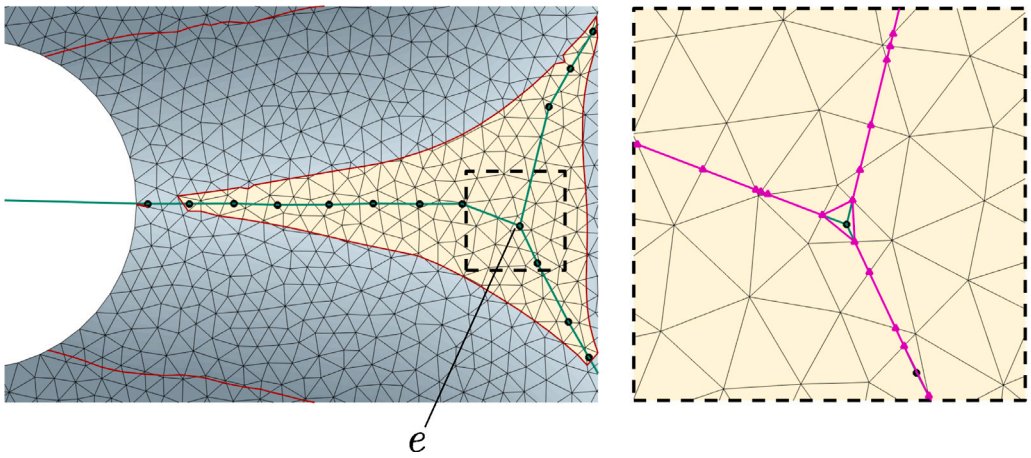


Fig. 22. Modified compact tension test: zoom of the junction located at element e that has been divided into three overlapping elements.

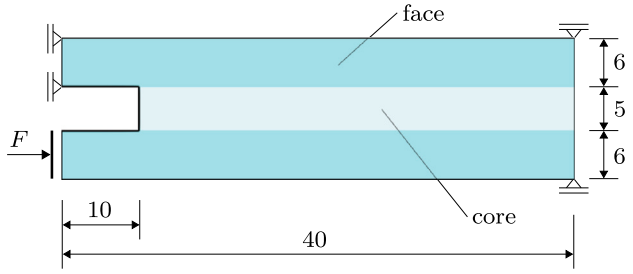


Fig. 23. Rail shear test: boundary condition and geometry (dimensions in mm).

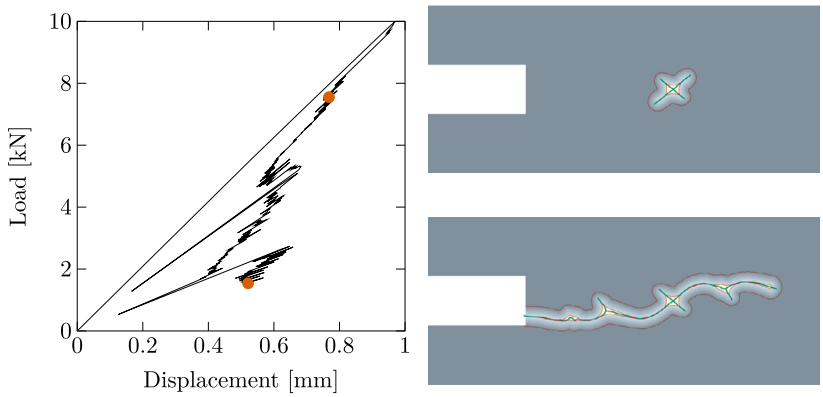


Fig. 24. Rail shear test with an initial damage nucleus and $\beta = 1$: equilibrium curve, and crack distribution for two different time steps are marked on the load–displacement curve.

and crack distributions for this case. It is clear that this constitutive model leads to an unrealistic behavior. Similar to what was observed with the TLSV1, initially an X-shaped crack appears, with unphysical compressive branches. However, with the TLSV2, the tensile branches are eventually favored, because there is still tension/compression asymmetry in the cohesive model. Nevertheless, secondary compressive branches appear, and the overall final crack pattern is not realistic. It is concluded that the asymmetry in the cohesive part of the TLSV2 is not sufficient to prevent compressive damage.

The second simulation is performed with $\beta = 0$ where the degradation of material behaves asymmetrically, with complete stiffness recovery under compression. In Fig. 25, the equilibrium curve obtained with β equal to zero is plotted, and the final crack distribution and its corresponding deformed configuration are shown. Unlike to what is shown with the TLSV1 in [4], where a hardening phenomenon is obtained with the same constitutive model due to the fact that one of the principal strains in the shear band between arm and core become negative leading to stiffness recovery and stress transfer across the crack, the TLSV2 model

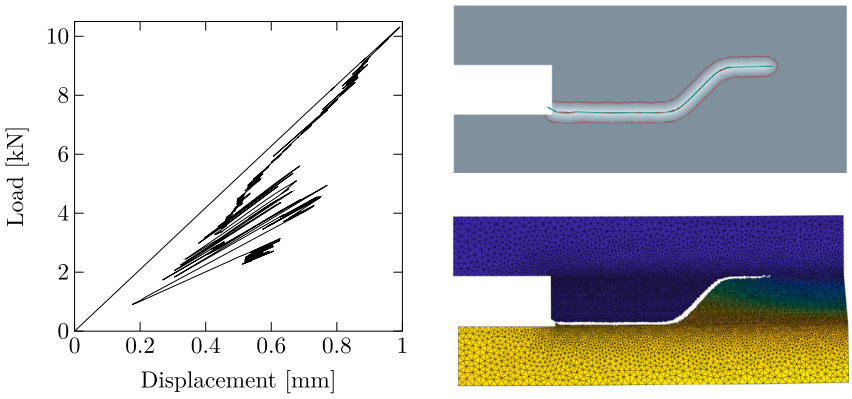


Fig. 25. Rail shear test with an initial damage nucleus and $\beta = 0$: load–displacement curve, and final crack distribution and its corresponding deformed specimen with horizontal displacement field.

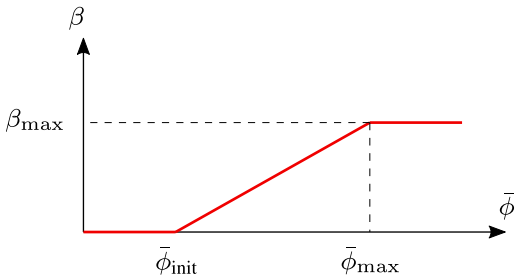


Fig. 26. Variability of parameter β as a function of the size of damaged domains $\bar{\phi}$.

proposed does not show such undesired behavior. The contact condition in the discontinuity does not prevent sliding deformations, and the surrounding bulk material can completely unload. However, the choice for $\beta = 1$ does lead to more oscillatory behavior in the post-peak response.

The last one-nucleus simulation uses a constitutive model that combines the benefits from both models that have been investigated previously. The choice for $\beta = 1$ brings robustness, while $\beta = 0$ gives the correct directionality for crack propagation. For the initial transition from a circular damage nucleus to a clean tensile crack under shear loading, full stiffness recovery under compression ($\beta = 0$) is essential. Later, when the asymmetry in tension/compression behavior is partially represented across the displacement discontinuity, stiffness recovery in the bulk becomes less important, although a value of $\beta = 1$ may still lead to undesirable secondary compressive crack branches. For large cracks, an intermediate value of β is sufficient and still provides an improvement in stability over the stricter $\beta = 0$. Because we have information on the size of the damaged zone through $\bar{\phi}$, we can achieve a transition for β from 0 to a higher value as the damaged zone grows. Therefore, β is made into a function of the size of damaged zone $\bar{\phi}$ as follows:

$$\beta(\bar{\phi}) = \begin{cases} 0, & \bar{\phi} \le \bar{\phi}_{\text{init}} \\ \dfrac{\bar{\phi}-\bar{\phi}_{\text{init},\beta}}{\bar{\phi}_{\max,\beta}-\bar{\phi}_{\text{init},\beta}}\beta_{\max}, & \bar{\phi}_{\text{init},\beta} < \bar{\phi} \le \bar{\phi}_{\max,\beta} \\ \beta_{\max} \le 1, & \bar{\phi} > \bar{\phi}_{\max,\beta} \end{cases}, \tag{33}$$

where $\bar{\phi}_{\text{init},\beta}$ and $\bar{\phi}_{\max,\beta}$ are, respectively, the initial size of the damaged zone and the size for which the material is considered to behave as a fully constitutive law with $\beta = \beta_{\max}$ (see Fig. 26).

Therefore, the problem is simulated again with $\bar{\phi}_{\text{init},\beta} = 2\phi l_c + 3l_c$, $\bar{\phi}_{\max,\beta} = 2\phi l_c + 6l_c$, and $\beta_{\max} = 0.88$ for the equation above. Fig. 27 shows the load–displacement curve with the variability of β, and crack distribution along with its deformed configuration for the last time step. It is clear that the approach proposed for evaluation of β through Eq. (33) influences the overall response on this problem. The initial nucleus does not grow into a X-shaped crack, because the material is dictated by the asymmetric constitutive law. In addition, this simulation does not suffer from substantial oscillations when the crack reaches the interface region between core and arms due to the fact that the constitutive law at this point has much less stiffness recovery under compression with $\beta = \beta_{\max}$. Also, note the absence of secondary branches when compared to the final crack configuration obtained with fixed value of $\beta = 1$.

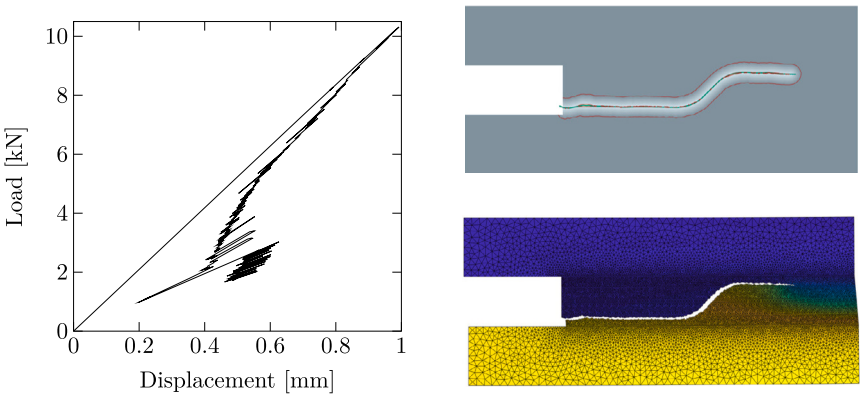


Fig. 27. Rail shear test with an initial damage nucleus and varying $\beta = \beta(\bar{\phi})$: load–displacement curve, and final crack distribution and its corresponding deformed specimen with horizontal displacement field.

5.3.2. Multiple damage nuclei

With this new way to address the value of β, the shear test is revisited. Now, the TLSV2 model is also applied to a case without predefined initial damage nucleus. The nucleation check is only performed around the mid-height of the core. Besides, the distance between a new damage radius and existing damage fronts is set to be at least 7 mm.

The evolution of damage in the shear test with the varying value of β is shown in Fig. 28. As already detailed in a similar test case with the TLSV1 [4], the first nuclei appear before the peak load is reached. As the load increases, the inclined cracks start to form; however, only the left most crack, the main crack, reaches the interface between the two materials, and this crack eventually extends over the whole length of the core. As the analysis progresses, the load eventually drops as the main crack reaches the right end of the specimen. Furthermore, it can be observed that the skeleton curve of the main crack does not join up with others even though its iso-0 curve does merge with the damage fronts of all inclined cracks. The skeleton curves do not merge because of the fact that when the fronts join up and the main crack moves towards to the right edge of the specimen, these regions experience unloading, after which the front stops evolving.

6. Conclusions and discussion

In this paper, the TLSV2 method is extended to provide the ability of dealing with free-form skeleton curves for crack growth problems in solids that bring branching and merging events. The TLSV2 method by Lé et al. [9] has been used as a basis model for the proposed version. A skeletonizer algorithm equipped with the ball-shrinking algorithm in the version by Peters [11] and Prim's algorithm has been designed. The phantom node method has been used to account for the discontinuous part of the TLSV2. Once the skeleton curve has been mapped into the mesh, the construction of the phantom node method including the possibility of crack branching is achieved by traversing along crack branches one side after the other.

After implementing these features, the TLSV2 method proposed was successfully assessed in three numerical examples. The first example, the compact tension test, was used to define some input parameters, and to verify the ability of the TLSV2 to represent crack growth at a constant fracture energy. Even though the fracture energy is not a direct input in the proposed method, this shows that the model can be calibrated to fracture energy measurements. It was demonstrated that it is necessary to extend the last atom point obtained from the ball-shrinking algorithm.

The proposed framework was shown to work for the case of crack branching and merging in the second and third test cases, although merging was only found for the damage front and not for the skeleton. In the shear test, the influence of the stiffness recovery parameter β on the global response was investigated. Variability of β as a function of the size of damaged zone was proposed to benefit from the advantages of the constitutive laws with and without stiffness recovery under compression preventing unrealistic compressive crack branches as obtained with $\beta = 0$, as well as undesirable oscillations as encountered with $\beta = 1$. Furthermore, the special interphase material used in the earlier TLS versions [4,6,7] was not necessary, since the crack could grow in a mode II manner without artificial hardening.

The work in this paper is restricted to unstructured meshes of linear triangles and to 2D simulations. However, the proposed framework, especially the algorithm to locate the skeleton curve in conjunction with the phantom node approach, is expected to work as well for meshes of quadrilaterals without substantial changes.

On the other hand, a 3D version of the proposed framework would require extra attention, mainly for construction of crack surface. Although the ball-shrinking algorithm behaves the same in 3D as it does in 2D [11], it still provides a point-based representation of the medial surface of the iso-0 surface in 3D simulations, *i.e.* an unstructured set of medial atoms, which would need to be transformed into surfaces with boundaries in order to represent cracks in 3D. An extensive overview of methods for constructing this surface can be found in [32]. Once the medial surface of the iso-0 surface has been determined, the corresponding discretized skeleton surface can be determined, which can subsequently be used in conjunction with the phantom node method for constructing a discontinuity in the displacement field.

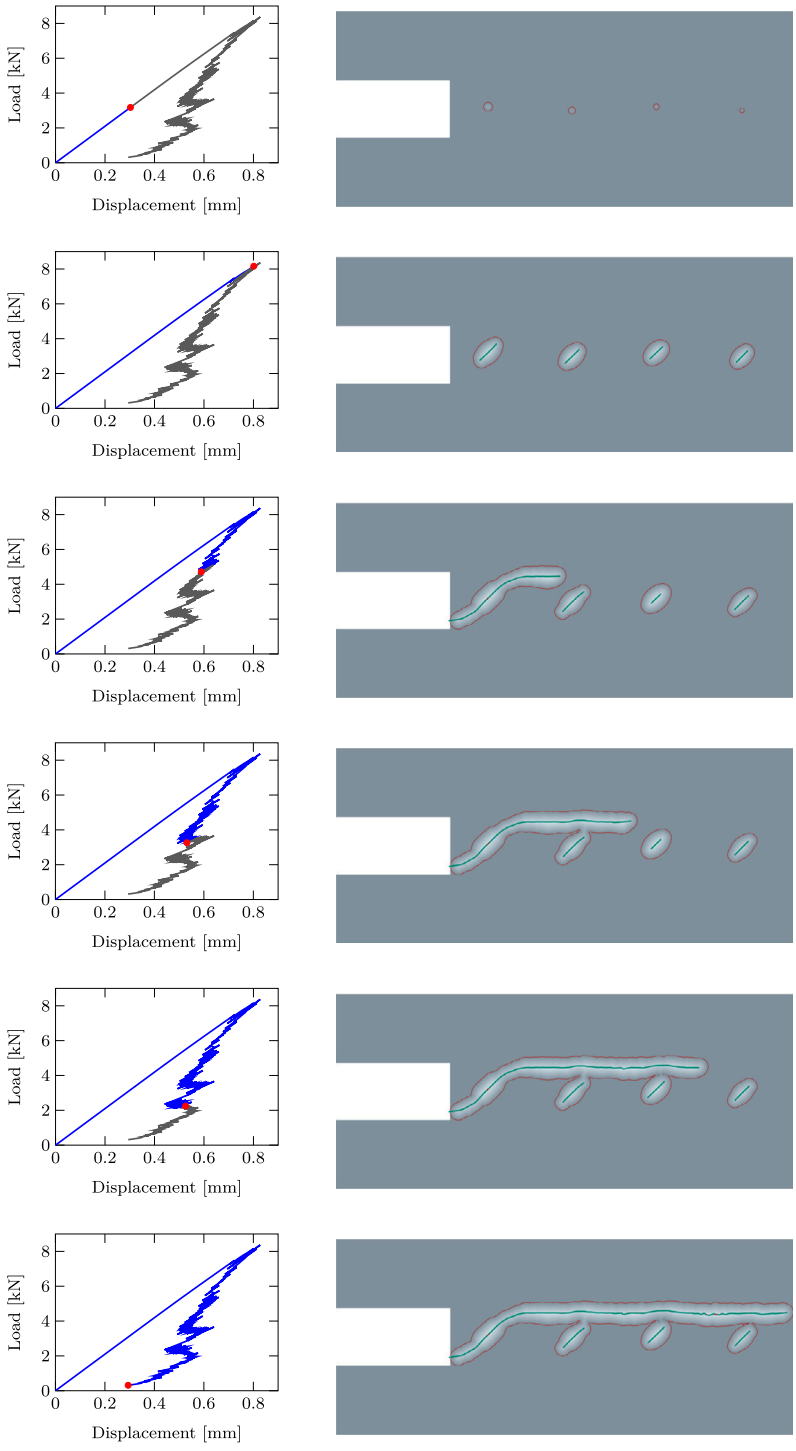


Fig. 28. Rail shear test: damage front and approximate skeleton curve evolution.

CRediT authorship contribution statement

L.A.T. Mororó: Conceptualization, Formal analysis, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **A. Poot:** Formal analysis, Investigation, Software, Validation, Conceptualization, Writing – review & editing. **F.P. van der Meer:** Data curation, Funding acquisition, Methodology, Project administration, Resources, Supervision, Writing – review & editing.

Data availability

The data used to generate the graphs and some figures of each example in this article is available at the 4TU.Research Data repository through https://doi.org/10.4121/19400849.v1.

Acknowledgments

FM acknowledges financial support from the Netherlands Organization for Scientific Research (NWO) under Vidi grant nr. 16464. LATM acknowledges the financial support by the Federal Institute of Education, Science and Technology of Ceará (IFCE).

References

[1] Moës N, Stolz C, Bernard P-E, Chevaugeon N. A level set based model for damage growth: The thick level set approach. Internat J Numer Methods Engrg 2011;86(3):358–80.
[2] Sethian J. Level set methods and fast marching methods: envolving interfaces in computational geometry, fluid mechanics, computer vision, and material science. 2nd ed.. Cambridge, United Kingdom: Cambridge University Press; 1999.
[3] Osher S, Fedkiw R. Level set methods and dynamic implicit surfaces. 1st ed.. New York, United States of America: Springer-Verlag; 2003.
[4] van der Meer FP, Sluys LJ. The Thick Level Set method: Sliding defomations and damage initiation. Comput Methods Appl Mech Engrg 2015;285:64–82.
[5] Bernard PE, Moës N, Chevaugeon N. Damage growth modeling using the Thick Level Set (TLS) approach: Efficient discretization for quasi-static loadings. Comput Methods Appl Mech Engrg 2012;233–236:11–27.
[6] Mororó LAT, van der Meer FP. Combining the thick level set method with plasticity. Eur J Mech A Solids 2020;79:pp. 14.
[7] Mororó LAT, van der Meer FP. Parallel computing with the Thick Level Set (TLS) method. SIAM J Sci Comput 2021;43(6):C386–C410.
[8] Cazes F, Moës N. Comparison of a phase-field model and of a thick level set model for brittle and quasi-brittle fracture. Internat J Numer Methods Engrg 2015;103:114–43.
[9] Lé B, Moës N, Legrain G. Coupling damage and cohesive zone models with the Thick Level Set approach to fracture. Eng Fract Mech 2018;193:214–47.
[10] Ma J, Bae SW, Choi S. 3D Medial axis point approximation using nearest neighbors and the normal field. Vis Comput 2012;28:7–19.
[11] Peters R. Geographical point cloud modelling with the 3D medial axis transform (Ph.D. thesis), Delft University of Technology; 2018.
[12] Hansbo A, Hansbo P. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. Comput Methods Appl Mech Engrg 2004;193(33):3523–40.
[13] van der Meer FP, Sluys LJ. A phantom node formulation with mixed mode cohesive law for splitting in laminates. Int J Fract 2009;158(2):107–24.
[14] Turon A, Camanho P, Costa J, Dávila C. A damage model for the simulation of delamination in advanced composites under variable-mode loading. Mech Mater 2006;38(11):1072–89.
[15] Latifi M, van der Meer FP, Sluys LJ. An interface thick level set model for simulating delamination in composites. Internat J Numer Methods Engrg 2017;111:303–24.
[16] Moreau K, Moës N, Chevaugeon N, Salzman A. Concurrent development of local and non-local damage with the Thick Level Set approach: Implementation aspects and application to quasi-brittle failure. Comput Methods Appl Mech Engrg 2017;327:306–26.
[17] Voormeeren L, van der Meer F, Maljaars J, Sluys L. A new method for fatigue life prediction based on the Thick Level Set approach. Eng Fract Mech 2017;182:449–66.
[18] van der Meer FP, Sluys LJ, Hallett SR, Wisnom MR. Computational modeling of complex failure mechanisms in laminates. J Compos Mater 2012;46(5):603–23.
[19] van der Meer FP, Moës N, Sluys LJ. A level set model for delamination – modeling crack growth without cohesive zone or stress singularity. Eng Fract Mech 2012;79:191–212.
[20] Levet F, Granier X. Improved skeleton extraction and surface generation for sketch-based modeling. In: Proceedings of graphics interface 2007. GI '07, 2007, p. 27–33.
[21] Tagliasacchi A, Delame T, Spagnuolo M, Amenta N, Telea A. 3D skeletons: A state-of-the-art report. Comput Graph Forum 2016;35(2):573–97.
[22] Siddiqi K, Pizer SM. Medial representation: mathematics, algorithms and applications. Springer; 2008.
[23] Blum H. Discussion paper: A geometry for biology. Ann New York Acad Sci 1974;231(1):19–30.
[24] Boost C++ libraries. 2021, http://www.boost.org/. [Accessed 20 July 2021].
[25] de Berg M, Cheong O, van Kreveld M, Overmars M. Computational geometry: algorithm and applications. 3rd ed.. Springer; 2008.
[26] Kepner J, Gilbert JR. Graph algorithms in the language of linear algebra. Vol. 22. Philadelphia, PA: Society for Industrial and Applied Mathematics; 2011.
[27] Samet H. Foundations of multidimensional and metric data structures. San Francisco, CA: Morgan Kaufmann; 2006.
[28] Song J-H, Areias PMA, Belytschko T. A method for dynamic crack and shear band propagation with phantom nodes. Internat J Numer Methods Engrg 2006;67(6):868–93.
[29] Poot A. Laying the Groundwork for a New Thick Level Set Method. the Netherlands: Delft University of Technology; 2020.
[30] Jem-jive - software development kit for advanced numerical simulations. 2019, http://jive.dynaflow.com. [Accessed 10 December 2019].
[31] Geuzaine C, Remacle J-F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. Internat J Numer Methods Engrg 2009;79(11):1309–31.
[32] Delame T, Kustra J, Telea A. Structuring 3D medial skeletons: A comparative study. In: Proceedings of the conference on vision, modeling and visualization. VMV '16, Eurographics Association; 2016, p. 1–8.