Available online at www.sciencedirect.com
SCIENCE DIRECT®
Engineering Fracture Mechanics 73 (2006) 1176–1201
Engineering Fracture Mechanics
www.elsevier.com/locate/engfracmech

Enriched finite elements and level sets for damage tolerance assessment of complex structures

Stéphane Bordas [a,*], Brian Moran [b]

[a] Ecole Polytechnique Fédérale de Lausanne (EPFL) Institut des Structures, Laboratoire de Mécanique des Structures et des Milieux Continus, Station 18, CH-1015 Lausanne, Switzerland
[b] Northwestern University, Mechanical Engineering Department, 2145 Sheridan Road, Evanston, IL 60208, United States

Received 10 August 2005; received in revised form 5 January 2006; accepted 9 January 2006
Available online 28 February 2006

Abstract

The extended finite element method (X-FEM) has recently emerged as an alternative to meshing/remeshing crack surfaces in computational fracture mechanics thanks to the concept of discontinuous and asymptotic partition of unity enrichment (PUM) of the standard finite element approximation spaces. Level set methods have been recently coupled with X-FEM to help track the crack geometry as it grows. However, little attention has been devoted to employing the X-FEM in real-world cases. This paper describes how X-FEM coupled with level set methods can be used to solve complex three-dimensional industrial fracture mechanics problems through combination of an object-oriented (C++) research code and a commercial solid modeling/finite element package (EDS-PLM/I-DEAS®). The paper briefly describes how object-oriented programming shows its advantages to efficiently implement the proposed methodology. Due to enrichment, the latter method allows for multiple crack growth scenarios to be analyzed with a minimal amount of remeshing. Additionally, the whole component contributes to the stiffness during the whole crack growth simulation. The use of level set methods permits the seamless merging of cracks with boundaries. To show the flexibility of the method, the latter is applied to damage tolerance analysis of a complex aircraft component.

Keywords: Enriched/extended finite elements; Object-oriented programming (C++); Damage tolerance assessment; Industrial problems

1. Introduction

Many industries have the need for simple and accurate tools to study crack growth scenarios and assess the damage tolerance of various structures. Examples range from the aeronautical industry to the automotive industry. Standard finite element methods require remeshing of the cracks as they grow, which costs additional computer time. This can be very burdensome in 3D and for cracks with very complex geometries. In this

* Corresponding author. Tel.: +41 21 693 2403; fax: +41 21 693 6340.
E-mail address: stephane.bordas@alumni.northwestern.edu (S. Bordas).
URL: http://people.epfl.ch/stephane.bordas (S. Bordas).

0013-7944/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engfracmech.2006.01.006

paper, a methodology based on the extended finite element method [1] is presented that enables the analyst to assess various damage tolerance scenarios easily with a minimal amount of remeshing. The implementation of the methodology is performed in the context of an object-oriented C++ code [2] that is interfaced with a commercial code, here EDS-PLM/I-DEAS. The only requirement on the commercial code is that it be able to output all the algebraic information of a problem (stiffness matrix, load vector, and the associated equation numbers). A super-element technique is used where the stiffness matrix of the uncracked structure[1] is assembled to the stiffness matrix of a substructure.[2] Therefore, as opposed to widely used global–local methods, the *whole component* participates to the stiffness of the component during the *whole crack growth process*.

The most salient features of the method is that the whole component participates to the stiffness computation, at all times during the crack growth simulation. Partition of unity enrichment of the standard finite element spaces allows for cracks to be grown without meshing or remeshing the crack faces. The coupling of enriched finite elements with level set methods makes explicit geometrical tracking of the discontinuities not necessary. Level set methods also allow cracks to naturally intersect boundaries as they grow, without any additional modification to the code.

In the next section of this paper, the linear elastic fracture mechanics (LEFM) problem is stated, for which a Galerkin weak form is given. The Paris law is also briefly recalled. The basic concepts of X-FEM and of the level set method for crack growth are then touched upon. Section 3 presents the proposed crack growth methodology for complex components. In Section 4 the key notions pertaining to object-oriented programming are presented and advantages of the approach are described. The most important classes of objects created in the implementation of the method are also quickly reviewed. Finally, in Section 5 numerical examples are given in 2D and 3D fracture mechanics. In particular, it is shown on an industrial example (a Boeing 757 Electrical Equipment (EE) Access door) how the proposed methodology may be exercised on complex industrial 3D components.

2. Problem statement

2.1. Numerical simulation of crack growth in a linear elastic body

2.1.1. Governing equations

Consider a domain Ω, bounded by Γ. The boundary is partitioned into three sets: Γ_u, Γ_t and Γ_c as shown in Fig. 1. Displacements are prescribed on Γ_u, tractions are prescribed on Γ_t and all the n^3 Γ_c^i are assumed to be a traction free surfaces.

The equilibrium conditions and boundary conditions for this problem are

$$\nabla \cdot \sigma + \mathbf{b} = \mathbf{0} \quad \text{in } \Omega \tag{1}$$

$$\sigma \cdot \mathbf{n} = \bar{\mathbf{t}} \quad \text{on } \Gamma_t \tag{2}$$

$$\sigma \cdot \mathbf{n} = \mathbf{0} \quad \text{on } \Gamma_c \tag{3}$$

$$u = \bar{u} \quad \text{on } \Gamma_u \tag{4}$$

where σ is the Cauchy stress tensor, $\mathbf{u}$ is the displacement field, $\mathbf{b}$ is the body force per unit volume and $\mathbf{n}$ is the unit outward normal. It is assumed that displacements remain small and the kinematics equations consist of the strain–displacement relation:

$$\varepsilon = \varepsilon(\mathbf{u}) = \nabla_s \mathbf{u} \tag{5}$$

where $\nabla_s(\cdot)$ is the symmetric part of the gradient operator. The constitutive relation for the elastic material under consideration is given by Hooke's law

$$\sigma = \mathbf{C} : \varepsilon \tag{6}$$

[1] Also noted "outer" structure, and possibly containing several element types such as shells, beams, continuum, multiple-point constraints, etc.

[2] Also noted "inner" structure in the following, this structure contains the cracks and is treated using the extended finite element method to allow their growth without remeshing.

[3] Number of cracks in the domain.

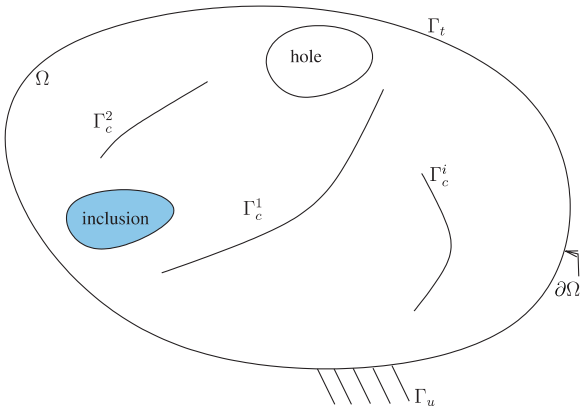


Fig. 1. Body Ω with external boundary $\partial\Omega = \Gamma$, with cracks, holes, and inclusions. Γ_u is the Dirichlet boundary, where displacements are prescribed. Γ_t is the Neumann boundary, where tractions are prescribed, and such that $\Gamma = \Gamma_u \oplus \Gamma_t$, i.e. $\Gamma = \Gamma_u \cup \Gamma_t$ and $\Gamma_u \cap \Gamma_t = \emptyset$. The n cracks in the body are such that $\forall i \in \{1, \ldots, n\}, \Gamma_c^i$ is a traction free crack. Holes can also be present as well as inclusions. Since the X-FEM is used here, neither holes, nor inclusions, nor cracks need to be meshed.

2.1.2. Variational formulation or weak form

Let the space of admissible displacement fields (trial function space) be defined by

$$\mathscr{U} = \{\mathbf{u} \in \mathscr{S} | \mathbf{u} = \bar{\mathbf{u}} \text{ on } \Gamma_u \text{ and } \mathbf{u} \text{ discontinuous through } \Gamma_c\} \tag{7}$$

In [3,4] the choice of the space of admissible displacements $\mathscr{S}$ when the body contains internal boundaries or re-entrant corners is discussed. Similarly, the test function space may be defined as

$$\mathscr{U}_0 = \{\mathbf{v} \in \mathscr{S} | \mathbf{v} = \mathbf{0} \text{ on } \Gamma_u \text{ and } \mathbf{v} \text{ discontinuous through } \Gamma_c\} \tag{8}$$

A weak formulation of the equilibrium equations is given by

$$\text{Find } \mathbf{u} \in \mathscr{U} | \forall \mathbf{v} \in \mathscr{U}_0, \quad \int_\Omega \sigma(\mathbf{u}) : \varepsilon(\mathbf{v}) \, \mathrm{d}\Omega = \int_\Gamma \mathbf{b} \cdot \mathbf{v} \, \mathrm{d}\Gamma + \int_{\Gamma_t} \bar{\mathbf{t}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \tag{9}$$

or, using the constitutive relation,

$$\text{Find } \mathbf{u} \in \mathscr{U} | \forall \mathbf{v} \in \mathscr{U}_0, \quad \int_\Omega \varepsilon(\mathbf{u}) : \mathbf{C} : \varepsilon(\mathbf{v}) \, \mathrm{d}\Omega = \int_\Gamma \mathbf{b} \cdot \mathbf{v} \, \mathrm{d}\Gamma + \int_{\Gamma_t} \bar{\mathbf{t}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \tag{10}$$

Define the bilinear form $\mathscr{B}$

$$\forall \mathbf{u} \in \mathscr{U} \ \ \forall \mathbf{v} \in \mathscr{U}_0, \quad \mathscr{B}(\mathbf{u}, \mathbf{v}) = \int_\Omega \epsilon(\mathbf{u}) : \mathbf{C} : \varepsilon(\mathbf{v}) \, \mathrm{d}\Omega \tag{11}$$

and the linear form $\mathscr{L}$

$$\forall \mathbf{v} \in \mathscr{U}_0, \quad \mathscr{L}(\mathbf{v}) = \int_\Gamma \mathbf{b} \cdot \mathbf{v} \, \mathrm{d}\Gamma + \int_{\Gamma_t} \bar{\mathbf{t}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \tag{12}$$

With these notations, the above can be rewritten

$$\text{Find } \mathbf{u} \in \mathscr{U} | \forall \mathbf{v} \in \mathscr{U}_0, \quad \mathscr{B}(\mathbf{u}, \mathbf{v}) = \mathscr{L}(\mathbf{v}) \tag{13}$$

2.1.3. Numerical computation of fracture parameters

Among the numerical methods for calculating fracture parameters, boundary integral methods [5,6] and the domain integral method [7–9] have proved adequate tools. In this work, the domain integral method, in conjunction with interaction energy integrals, is used to determine mixed-mode stress intensity factors. In the interaction energy integral method, auxiliary fields are introduced and superimposed onto the actual fields satisfying the boundary value problem. By suitably selecting these auxiliary fields, a relationship can be found

between the mixed-mode stress intensity factors and the interaction energy integrals. These integrals can be represented in so-called domain forms and evaluated in a post-processing step, once the solution to the boundary value problem is known. In the general, non-planar [10], three-dimensional case, the influence of the curvature of the crack front on the evaluation of the domain integrals should be noted [11]. Refer to [12–15] for more details.

2.2. Modelling cracks with the extended finite element method and the level set method

The level set method and the extended finite element method were first used to model crack growth in two dimensions [16], and in three dimensions [17–19,12,13]. The extended finite element method (X-FEM) is a partition of unity method (PUM) [20,21].

When cracks are present in a linear elastic body the strain field has a $1/\sqrt{r}$ singularity when r approaches zero, while the displacement field behaves like $\sqrt{r}$ as r approaches zero. *Enriching* the displacement approximation space with the function $\sqrt{r}$ increases the capability of the finite element method in approximating the exact solution to the crack problem. The notion of local approximability can be also exploited by adding a function discontinuous across the crack faces to the displacement approximation space. The finite element method thereby becomes able to model the discontinuity without needing to conform the discretization to the discontinuity. This incorporation of the asymptotic fields of linear elastic fracture mechanics as well as a discontinuous function in a standard finite element approximation in this fashion was first used in [2] and the associated finite element method was coined the extended finite element method (X-FEM). In [22] the application of X-FEM to introduce *arbitrary discontinuities in finite elements* is presented. The enrichment scheme in this paper is identical to that of [12,13].

2.2.1. Extended finite element approximation

Consider a point $\mathbf{x}$ that lies inside a finite element e. Denote the element's nodal set as $\mathcal{N}_e = \{n_1, n_2, \ldots, n_{m_e}\}$, where m_e is the number of nodes of element e. The enriched displacement approximation for a vector-valued function $\mathbf{u}^h : \mathbb{R}^d \to \mathbb{R}^d$ (d is the number of space dimensions) assumes the form:

$$\mathbf{u}^h(\mathbf{x}) = \sum_{I, n_I \in \mathcal{N}_e} N_I(\mathbf{x})\mathbf{u}_I + \sum_{J, n_J \in \mathcal{N}^g} N_J(\mathbf{x})\mathscr{E}(\mathbf{x})\mathbf{a}_J \tag{14}$$

where the nodal set $\mathcal{N}^g$ is the set of nodes whose support is intersected by the domain Ω_g associated with a geometric entity such as a hole, crack surface, or crack front and $\mathcal{N}$ is the set of nodes that are not enriched. Mathematically,

$$\mathcal{N}^g = \{n_J : n_J \in \mathcal{N}_e | \omega_J \cap \Omega_g \neq \mathrm{O}\} \tag{15}$$

In the above equation, $\omega_J = \mathrm{supp}(n_J)$ is the support of the nodal shape function $N_J(\mathbf{x})$, which consists of the union of all elements with n_J as one of its vertices; and Ω_g is the domain associated with a geometric entity such as a hole, crack surface, or crack front. The choice of the function $\mathscr{E}$: $\mathbf{x} \mapsto \mathscr{E}(\mathbf{x})$ depends on the geometric entity under consideration (crack, hole, material interface, etc.).

For crack problems, two sets of enrichment functions are used:

- The function $H(\mathbf{x})$, i.e. the modified Heaviside function which takes on the value $+1$ above the crack and -1 below the crack.
- The functions $B_\alpha(\mathbf{x})$, which is a basis that spans the near-tip asymptotic fields:[4]

$$\mathbf{B} \equiv [B_1, B_2, B_3, B_4] = \left[\sqrt{r}\sin\frac{\theta}{2}, \sqrt{r}\cos\frac{\theta}{2}, \sqrt{r}\sin\frac{\theta}{2}\cos\theta, \sqrt{r}\cos\frac{\theta}{2}\cos\theta\right] \tag{16}$$

[4] $r \in \mathbb{R}_+$ and $\theta \in [-\pi, +\pi]$ are the usual polar coordinates defined at the crack tip.

2.2.2. Discretized equilibrium equations

In the extended finite element method, the additional unknowns $\mathbf{a}_J$ associated with the enrichment functions simply augment the conventional unknown displacement vector $\mathbf{u}$ and are solved for in the same manner:

$$\mathbf{K}\cdot\mathbf{u}=\mathbf{f}^{\text{ext}} \Longleftrightarrow \begin{bmatrix}\mathbf{K}_{\text{uu}} & \mathbf{K}_{\text{ua}}\\ \mathbf{K}_{\text{au}} & \mathbf{K}_{\text{aa}}\end{bmatrix}\begin{bmatrix}\mathbf{u}\\ \mathbf{a}\end{bmatrix}=\begin{bmatrix}\mathbf{f}_{\text{u}}^{\text{ext}}\\ \mathbf{f}_{\text{a}}^{\text{ext}}\end{bmatrix} \tag{17}$$

The size of the stiffness matrix is $N_{\text{total}}=N_e+N_{\text{enr}}$, it is symmetric, positive definite, sparse and banded – since enrichment is local. The implementation of the extended finite element method is thus very closely related to that of the finite element method [23].

2.2.3. The extended finite element method for cracks in 3D

2.2.3.1. Crack and displacement field description. The presentation in this section follows closely [12,13] to which the interested reader is referred for more details. In three dimensions, as in two dimensions, the crack geometry is defined by two signed distance functions named here ϕ and ψ. The intersection of the two zero-level set surfaces represents the crack front, as shown in Fig. 3. A schematic representation of the two level set functions for the case of an elliptical crack is given in Fig. 2 where the actual crack is shaded. The iso-ϕ surfaces are planes parallel to the plane $(OxOy)$ and the iso-ψ surfaces are coaxial cylinders whose axes go through O and are supported by the line (Oz).

The signed distance functions are updated and reorthogonalized by level set methods (see [24–30] for details on finite difference based level set methods).

The signed distance function obeys a hyperbolic conservation law written by setting the total time derivative of ϕ to zero, in order to enforce the condition that the level set function ϕ remain constant on the interface (here the crack surface). The interface motion is then governed by the conservation equation

$$\frac{\mathrm{D}\phi}{\mathrm{D}t}=\frac{\partial\phi}{\partial t}+\boldsymbol{U}\cdot\nabla\phi=0 \tag{18}$$

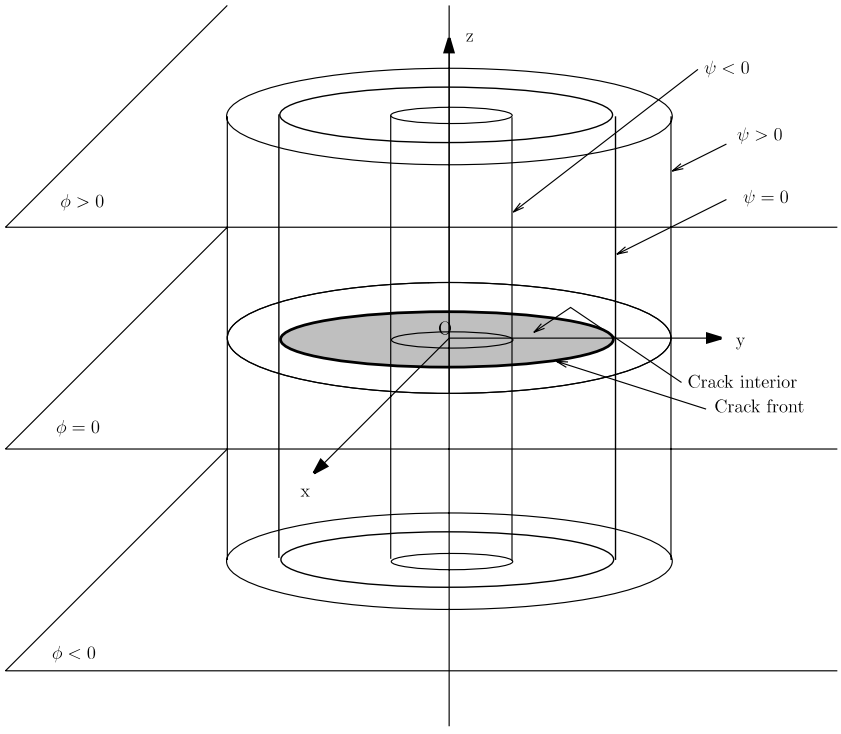


Fig. 2. Level set representation of an elliptical crack in 3D.

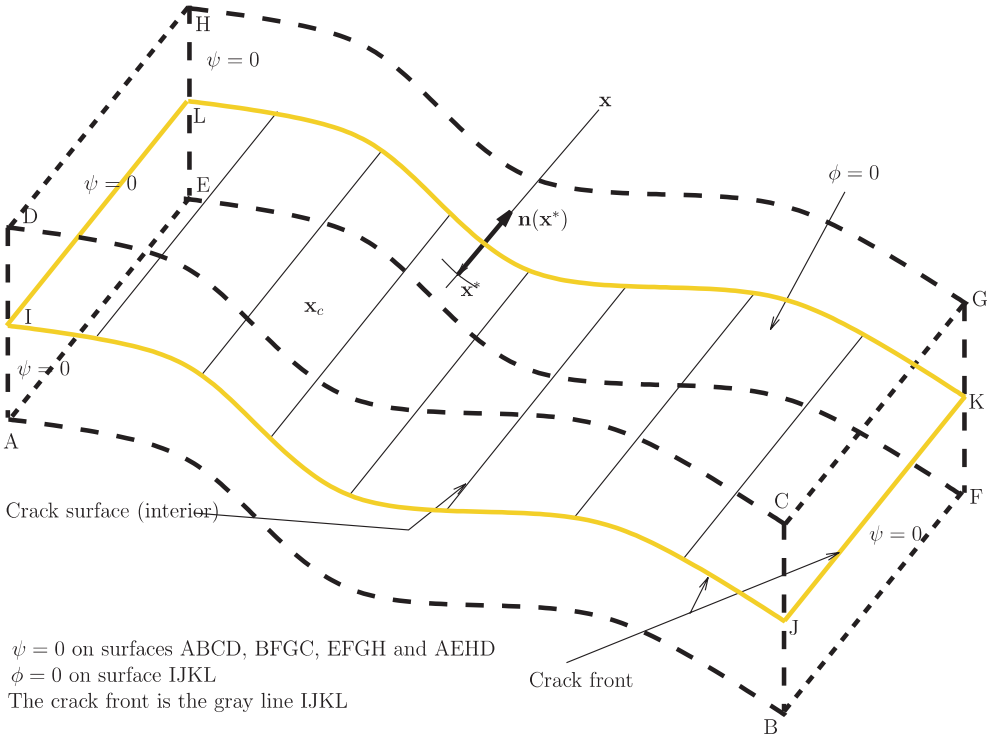


Fig. 3. Level set definition of a crack in 3D. $\mathbf{x}$ is a point in space where the level set functions are computed. $\mathbf{x}^*$ its orthogonal projection on the crack surface, and $\mathbf{x}_c$ a generic point on the crack surface.

where D/Dt is the total (material) time derivative moving with the interface, $\boldsymbol{U}$ is the vector velocity field that advects the level set field, computed based on the crack growth law (including length of increment, and direction).

In this study, the standard displacement approximations are piecewise linear and C^0. Linear tetrahedral elements are used to represent both the mechanical fields and the level set functions and therefore, the geometrical representation of the crack also is C^0 and piecewise linear. The method naturally extends to higher order shape functions [31,32].

2.2.3.2. Summary of the algorithm. A detailed explanation of the procedure is given in [13] and only a brief summary of the algorithm is given here. The level set update equations are identical to their two-dimensional counterpart.

The complete level set update procedure may be summarized as follows:

(1) Extend the crack level set ϕ to the level set sub-domain where $\psi \geqslant 0$.
(2) Extend the ϕ and ψ components of the velocity field to the sub-domain.
(3) Modify the ϕ component of the velocity field to preserve the crack surface's history (prevent its modification by the crack extension). The crack history must be preserved through the update so that the new ϕ level set continues to have a zero velocity field on the previous crack surface ($\psi < 0$).
(4) Update the ϕ level set and reinitialize it.
(5) Update the ψ level set.
(6) Orthogonalize the ψ level set to the ϕ level set and reinitialize it.

3. A methodology for the analysis of crack growth in complex 3D components

In industrial settings, meshing takes up a substantial part of an analyst's time. The technique presented in this paper in the context of damage tolerance analysis allows meshing and remeshing to be kept to a minimum.

Usually, a mesh is generated to perform a stress analysis, which identifies the regions in the structure which are most susceptible to crack initiation and growth. The present approach uses the stress analysis mesh as the starting point for damage tolerance analysis (DTA).

3.1. An approach to damage tolerance assessment of complex structures

The idea of the proposed damage tolerance assessment technique is to split the problem into two parts. The first is the inner, or local domain and consists of the subset of the part in which cracks are assumed to have initiated. The second is the outer, or global, which consists of the remainder of the structure. The former contains the cracks, that are treated with the X-FEM. The latter may contain several element types (here shell, multiple point constraint, and continuum elements) and is treated using standard finite element methods – here the commercial software EDS-PLM–I-DEAS®. The approach is depicted in Fig. 4. The complex geometrical features of the component are handled in the commercial code (I-DEAS® by EDS-PLM) while the fracture-related issues are handled by the extended finite element code.

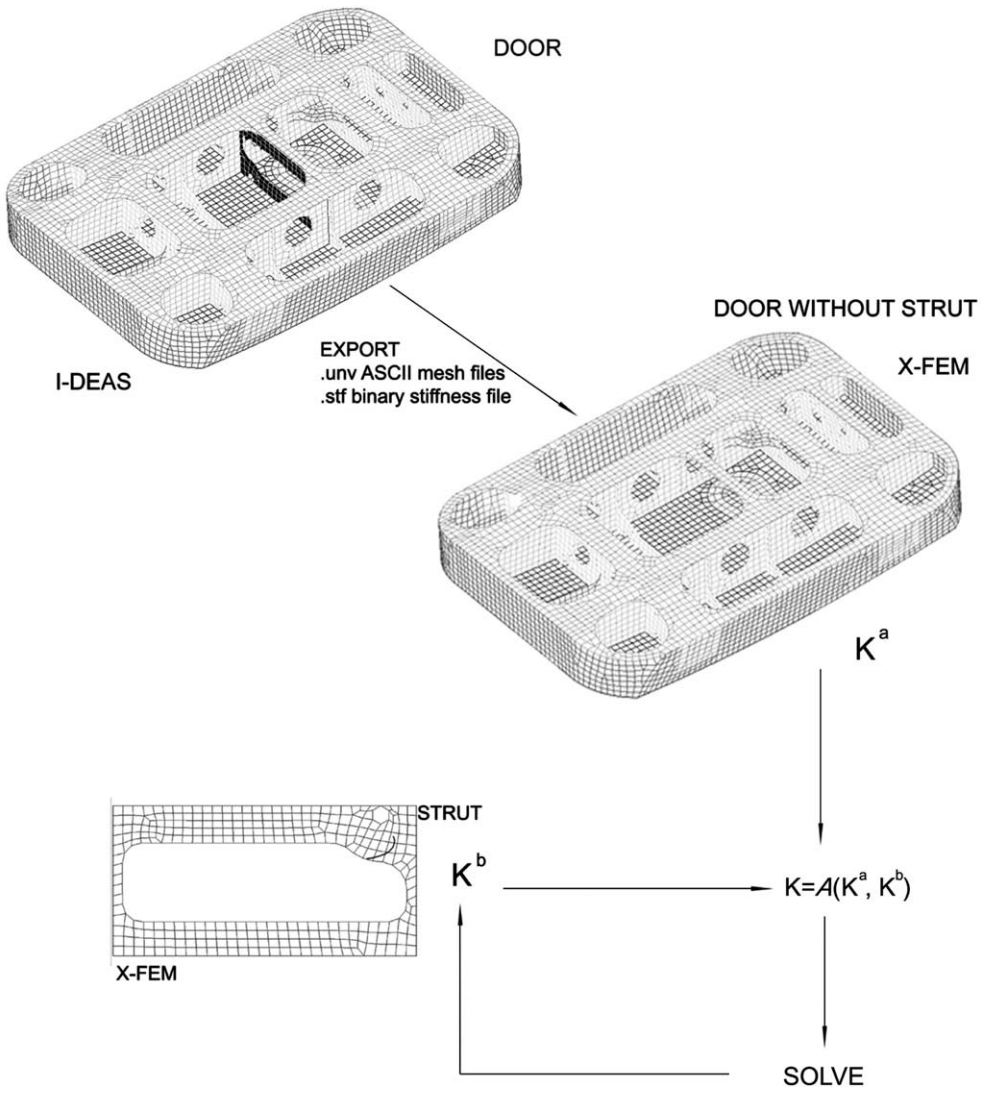


Fig. 4. Superelement/X-FEM methodology, conceptual representation.

In the following, the X-FEM mesh/domain in which the crack is introduced is referred to as the *inner mesh* or *inner domain* and the superelement mesh/domain, the remainder of the structure when the inner mesh is removed, is referred to as the *outer mesh* or *domain* or the *superelement*.

3.2. Operations in the commercial software

The procedure may be split into two sets of actions. The first pertains to the commercial code, the second to the X-FEM code.

In the commercial code, the following operations are performed:

(1) Create the geometry.
(2) Based on design considerations – modeling of manufacturing process, non-destructive evaluation considerations, stress analysis results – identify regions where cracks are likely to initiate and propagate and regions where the probability of detection is low [33].
(3) Mesh the geometry with the exception of the regions of interest isolated in (2).
(4) Create the Dirichlet and Neumann boundary conditions on the outer mesh.
(5) Export the resulting load vector (already modified to account for essential boundary conditions).
(6) Export the stiffness matrix associated with the outer mesh and the vector containing the degree of freedom information. This map shall be called the *EQNNIN map*. Both the EQNNIN map and the stiffness matrix have been modified to account for essential boundary conditions. The EQNNIN map provides a relationship between the degree of freedom number (row and column in the matrix, row in the load and displacement vectors) and the corresponding node number and physical information. This information is required in order to create the degrees of freedom associated with the superelement, which will be used at the assembly stage.
(7) For each region identified in (2), create an initial coarse mesh. This initial mesh should be sufficiently refined in areas where the cracks will be added. As a rule of thumb the mesh size can be taken as one-twentieth of the characteristic size of the crack. Export the resulting mesh – inner mesh.

3.3. Operations in the in-house X-FEM code

Then, in the X-FEM code, the following operations are carried out:

(1) Input the stiffness matrix associated with the outer mesh or superelement.
(2) Input the EQNNIN map for the outer mesh.
(3) Input the load vector for the outer mesh.
(4) Create the superelement data structure. This data structure includes the stiffness matrix and load vector for the outer mesh and the degree of freedom information necessary for their assembly, which is built using the EQNNIN map from step (2).
(5) Create the free degrees of freedom associated with the outer mesh based on the EQNNIN map. Note that the fixed degrees of freedom, those associated with the nodes on the essential boundary need not be created since such boundary conditions are already accounted for in the EQNNIN vector, the stiffness matrix and the load vector. The equations associated with those degrees of freedom are simply suppressed from the system of equations.
(6) Create the degrees of freedom information associated with the inner mesh. Note that the *interface degrees of freedom* (degrees of freedom which belong to both the inner and the outer meshes) will only be created at step (5) and will not be modified during step (6).
(7) Assemble the superelement (superelement arrays may also be condensed onto the inner–outer interface degrees of freedom). This assembles the load vector input in (3) and the stiffness matrix input in (1) based on the degree of freedom information input in (5) into the global stiffness matrix.
(8) Through mesh-geometry interaction techniques, find the enriched nodes and create the associated function spaces. For those elements that are enriched, those function spaces are added to the conventional displacement function spaces. This involves finding the enriched nodes and creating the extra degrees of freedom (enrichment) associated with the enrichment functions.

(9) Assemble the inner stiffness matrix. It accounts for the presence of cracks through the enrichment information set in (8).
(10) Solve the linear system of equations after renumbering to optimize band width, yielding the displacement field in the inner region. Compute the stresses from those displacement fields using the X-FEM approximation.
(11) Compute the stress intensity factors on the front; deduce the crack velocities using the fatigue crack propagation law. Given this velocity field, find the critical time step for the mesh and the current position of the crack front. This time step will be used to update the level sets and find the new location of the crack front.
(12) Extend the velocity field to the narrow band around the crack and update the two level sets defining the crack. Re-orthogonalize the two level set functions and preserve the history of the crack.
(13) If the simulation is not complete, i.e., if the current time is less than the final simulation time, repeat (8)–(13), the newly obtained crack geometry.

3.4. Crack growth methodology

The general methodology used in this paper to advance the cracks at each time step is described next:

(1) Compute the displacement and strain/stress field in the structure using the X-FEM approximation.
(2) Compute the stress intensity factors using domain integral techniques at each point on the crack front. In this case, points on the front are taken at the intersection of the zero level set function describing the crack and the elements of the mesh used for the description of the mechanical and level set fields.
(3) Compute the crack advance "velocity" at each of the points on the front. Here, the "velocity" at point i on the front is given by the Paris law as $C(\Delta K_i)^m$. In case of mixed mode loading, ΔK_i is a mixed-mode equivalent stress intensity factor.
(4) Advance the crack by an amount $\Delta a_i = C(\Delta K_i)^m \Delta t$, where Δt is chosen smaller than the critical time step associated with the level set update, i.e., the smallest time that it takes to the crack to cross the elements cut by the front.
(5) Go to (1).

For the above to be feasible, the stress analysis software must have the ability to export the stiffness matrix and load vector for the superelement as well as the "degree of freedom (dof) information" (given a dof number, recover the associated node number and nature of the dof). This information is read into an inhouse code, which handles the X-FEM calculations. Since the superelement information remains unchanged throughout the crack growth simulation, it may be assembled once and for all or even condensed onto the interface degrees of freedom.

The present work was performed using the commercial code EDS-PLM/I-DEAS® and a C++ implementation of 3D crack growth based on the X-FEM [2].

4. C++ implementation and data structures

4.1. Introduction

As opposed to procedural programming, the object-oriented (OO) programming philosophy segments the problem at hand into objects, thereby increasing code cleanliness and orthogonality. This produces a set of application-specific data types used in writing the code. A large part of the programming process resides in deciding which objects are useful, what data they should hold, which actions they should perform, and how they should interact with each other. Then, the logic of the program in terms of those objects and the kinds of operations they allow should be set up. These concepts should make a well-written object-oriented program more modular, easier to understand, maintain and evolve, and therefore more reusable. C++ is an example of an object-oriented programming language and is the language in which the present X-FEM

code is written. An excellent reference on C++ is the book by its creator [34]. The reader interested in more details on object-oriented designs of extended finite element code are encouraged to read Ref. [23].

In the extended finite element framework, there are objects such as a *finite element object*, which has to "know" its type (triangular, quadrilateral), its dimension, the coordinates of its nodes, the type of integration used and the coordinates of the integration points, the type of interpolation functions defined on the element, which of its nodes are enriched, etc. A finite element object should also be able to perform some operations; it would therefore contain functions to operate on it such as: compute its area or volume, its Jacobian matrix, get its stiffness matrix or mass matrix, etc. This packaging of data values and functions within an object is referred to as *encapsulation* and is one of the corner stones of object-oriented programming.

The first successful attempts to organizing a finite element program using the object-oriented paradigm dates back to [35–37] with the SMALLTALK programming language. Theresince, the ease of maintenance, reliability, and reusability of the OO generated software have given birth to multiple object-oriented finite element packages. C++, and, more recently JAVA, have emerged as the two languages of choice for OO numerical analysis.

In the following, objects and classes will be denoted in **bold font**, while methods will be denoted in *italics*.

4.2. The superelement class

In the following, widely accepted OO vocabulary is used. "Classes" are the building blocks of the OO program. Instances of a class are known as "Objects". Classes have "data members", which represent the data they hold, and "methods", which implement the actions they are able to perform.

The **SuperElement** class *derives* from (i.e., inherits properties from) a standard **Element** class and stores the following private members:

(1) nodalLoadVector (its load vector, read from input file),
(2) stiffnessMatrix (its stiffness matrix, read from input file),
(3) dofKeys (physical significance of its dofs, created using EQNNIN map),
(4) data (inherited from the Element class).

Now, a **SuperElement** should be able to perform a few tasks (coded in member functions):

(1) Create itself and allocate memory dynamically based on input data.
(2) Destroy itself and free the memory that was allocated for it.
(3) Find the meaning of its dofs (dofKeys), using the EQNNIN data.
(4) Set and get its load or EQNNIN vector.
(5) Set and get its stiffness matrix.
(6) Find the size of its arrays.

The main action in the above is the creation of its member dofKeys based on the EQNNIN map, as described below. The **SuperElement** has access to the EQNNIN map (through the data object above), relating a node number to its associated equation numbers. This map is a container from the C++ Standard Template Library (STL):

```
map⟨int,vector⟨int⟩⟩,
```

where the first integer (called first value of the map) is the node number and the vector of integers (called second value of the map) contains all dof types associated with this node number. For instance, a node belonging to a tetrahedral element has three dofs of types $DISP_X$, $DISP_Y$ and $DISP_Z$. A shell element node has three additional rotational dofs R_X, R_Y and R_Z. The pseudo-code for the creation of the degrees of freedom associated with the **SuperElement** follows:

```
Locate iterator at beginning of EQNNIN map
Traverse the EQNNIN map (increment iterator) {
```

```
    get the first value of the map (node number N)
    get the second value of the map (list of dof type)
      Loop on the values in the vector of dof types {
        Switch on the dof type {
          DISP_X set the physical nature of dof to DISP_X
          DISP_Y set the physical nature of dof to DISP_Y
          ...
          R_Z set the physical nature of dof to R_Z
        }
        Create and store a dofKey with the physics
        determined above
      }
    return a vector⟨DofKey⟩
  }
```

The actual creation of the dofs associated with the **SuperElement** is performed at the **Formulation** level, as well as its assembly. The **Formulation** class is described next.

4.3. The Formulation class

The **Formulation** class is an abstract class for all formulations. It serves as an interface to ensure that all classes derived from it implement the methods required for any **Formulation**. For instance, a **Formulation** object must implement the method *TreatmentOfFormulation*, which actually sets and solves the numerical problem.

The key data member of a **Formulation** is a **DofData** object, which allows it to store the information about all dofs in the problem and an **Assembler** object, which deals with the assembly of the problem's elements based on the **DofData**.

The tasks that a **Formulation** object must be able to perform are:

(1) Create the **Dofs** associated with the **Elements** in the domain, including the **SuperElement** (derived from the **Element** class). It also handles information pertaining to enriched **Dofs**.
(2) Account for essential boundary conditions. In particular, modify the **SuperElement** arrays accordingly.
(3) Assemble all element arrays, including the **SuperElement** arrays in the left hand side (LHS) and right hand side (RHS) – using its data member of class **Assembler**.

It was seen how a **SuperElement** object can construct a **std::vector** of **DofKeys**, which stores the physical meaning of its **Dofs**. This **std::vector** is, in turn, used by the **Formulation** class to create the associated **Dofs**:

```
Loop on the std::list of SuperElement in the Data object {
  Get the std::vector of DofKeys for the current SuperElement
  Loop on DofKeys {
    Create a symmetrical Dof (DofSym) based on the DofKey
  }
Update its DofData member as made necessary by the above.
}
```

Dirichlet boundary conditions are read from the input file and stored in the form of two Standard Template Library STL maps [34] (**std::map**). Accounting for Dirichlet boundary conditions:

```
map⟨int, vector⟨double⟩⟩* dirichletContainer;
map⟨int, vector⟨int⟩⟩* dirichletDirections;
```

The first maps a node number value (int) to a **std::vector** of prescribed values (**std::vector** ⟨**double**⟩). The second specifies which directions are subjected to the prescribed values. The account for boundary conditions in the **DofData** object of the **Formulation** class is now straightforward

```
Get the EQNNIN map
Get the dirichletContainer
Get the dirichletDirections
Traverse the dirichletContainer {
Get first value of dirichletContainer (node number I)
Get second value of dirichletContainer (values of prescribed dofs)
Find node I in the EQNNIN map
  deduce (EQNNIN map) dof types associated with node I
Find node I in the dirichletDirections map
  deduce (dirichletDirections) which dofs are prescribed
  this is called directionVector
Loop on entries in directionVector {
  if entry == 1
    create DISP_X FIXED dof for node I with value given by
    corresponding entry in dirichletContainer
    Update DofData with this dof
  else if entry == 2
    create DISP_Y FIXED dof for node I with value given by
    corresponding entry in dirichletContainer
    Update DofData with this dof
  ...
  else if entry == 6
    create R_Z FIXED dof for node I with value given by
    corresponding entry in dirichletContainer
    Update DofData with this dof
  }
}
```

Once the **Formulation** object knows the updated **DofData** object, accounting for all boundary conditions, it can perform the assembly of the **SuperElement** object using an **Assembler** object, as shown in the following section.

Note that since the abstract (base) class **Formulation** implements the above methods, they are also available to all classes derived from it, namely, the **Elasticity**, **Mechanics** or **Thermics** formulations.

4.4. The Assembler class

In I-DEAS, all information pertaining to a model is exported in the so-called "universal" (UNV) format. The role of the **Assembler** class is to read the algebraic information from the UNV file obtained from I-DEAS and assemble them in the global arrays.

An **Assembler** object has the main tasks of assembling local element arrays into the global arrays for the problem. It can assemble a term in a vector, a vector in another vector, or a matrix in another matrix.

For instance, the assembly of the **SuperElement** arrays requires for the **Assembler** to know the locations where the arrays are to be assembled. This knowledge is transmitted through a vector of **DofKeys** (built by the **SuperElement** object of interest). The stiffness matrix and the load vector are also given by the **SuperElement** object.

4.5. The Mechanics class

The **Mechanics** class is derived from the **Formulation** class. Consequently, it inherits the public methods and protected data members from the base class **Formulation**. A **Mechanics Formulation** is a specialized version of a **Formulation**. It specializes in the treatment of crack growth problems in linear elasticity.

The main method of class **Formulation** sets up and solves the numerical problem:

```
void Mechanics :: TreatmentOfSuperFormulation (Data *data)
```

A pseudo-code for this fundamental method is as follows:

```
SET UP PROBLEM
Find enriched nodes
Export geometry of cracks and other interfaces
Create function spaces for the standard FE interpolation on all
elements of the inner mesh region (displacement in x-, y- and z-directions,
Lagrange interpolation of degree one, say)
Defines function spaces for superelement on the superelement region
Find enriched degrees of freedom
```

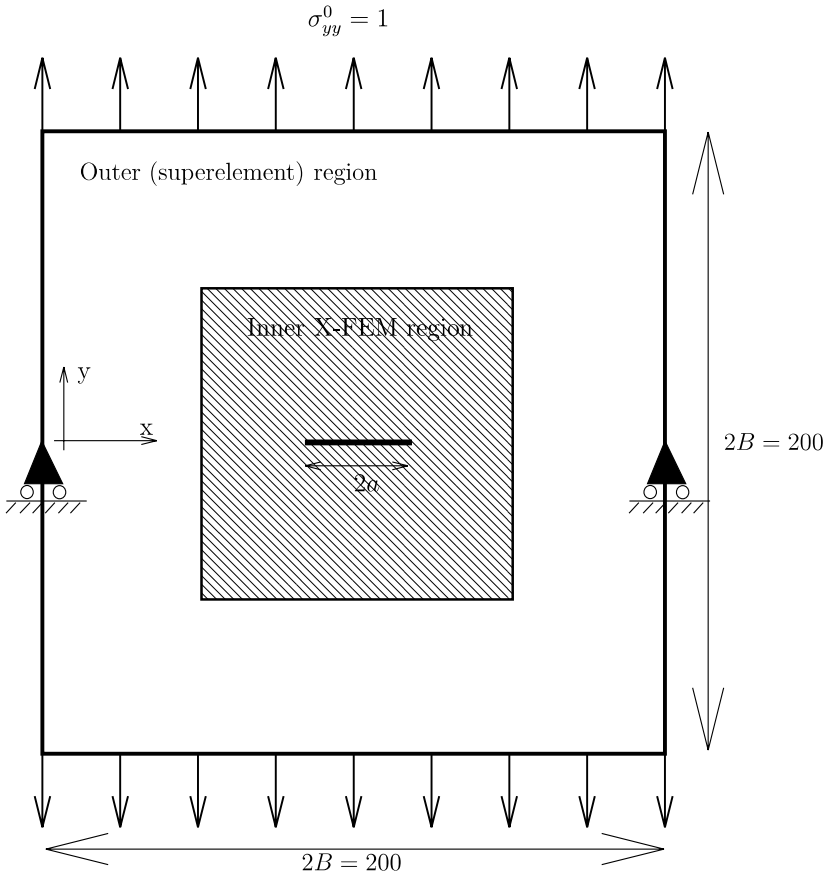


Fig. 5. Griffith crack in a plate: schematic representation of the inner and outer domains. The figure is not to scale and $B \gg a$.

```
Create degrees of freedom for super-element
Treat the presence of holes
Create degrees of freedom for the inner region
SOLVE PROBLEM
Define unknown vectors and stiffness matrix
Create Linear System object
Create assembler object
Enforce Neumann boundary conditions
If (timeStep == 1) Assemble the super element (outer stiffness matrix)
Assemble the inner stiffness matrix
Solve System
Store solution
POSTPROCESS
Compute stress intensity factors (domain integrals)
Advance the cracks (update level sets)
```

It can be seen that once the problem has been decomposed into suitable objects, the actual solution process is merely a succession of calls to methods, whose implementation depend on the problem at hand. This modularity makes seamless the process of adding new formulations. To do so, it is only necessary to locate the methods implemented by the base **Formulation** class, which need to be modified, and to reimplement them accordingly. The rest of the code being left unchanged.

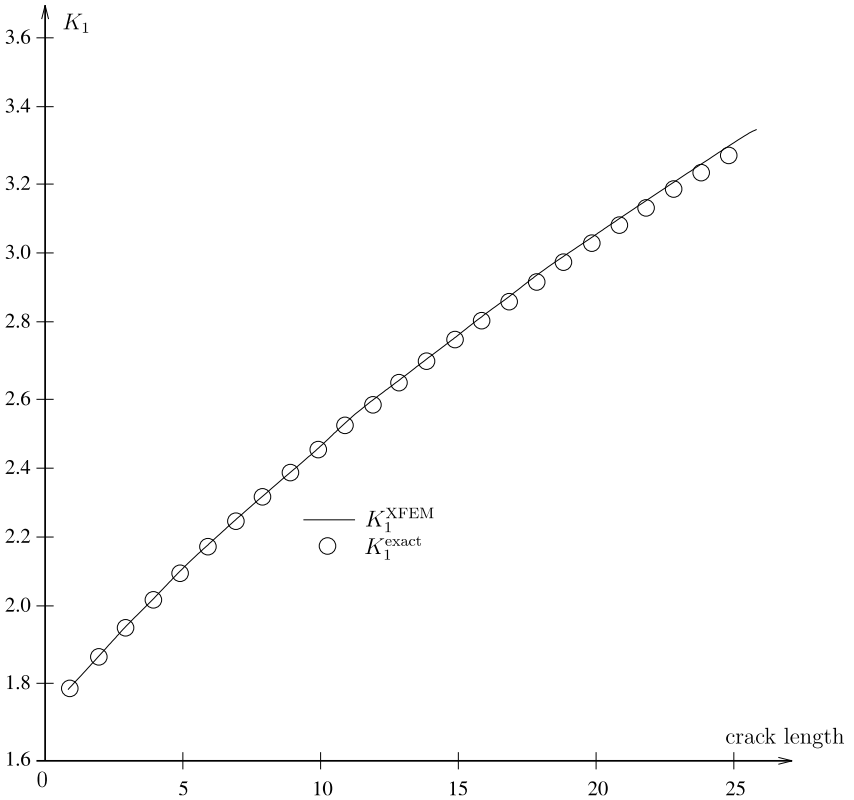


Fig. 6. Evolution of the mode I stress intensity factor versus crack length for a Griffith crack in a plate.

5. Examples

5.1. Griffith crack in an infinite plate

In this section, a Griffith crack problem in a two-dimensional body assuming plane strain conditions is considered. Fig. 5 shows the problem definition. The exact stress intensity factors for a Griffith crack of length $2a$ in an infinite medium subjected to uniaxial tension σ^0_{yy} are:

$$K_{\mathrm{I}} = \sigma^0_{yy}\sqrt{\pi a} \quad K_{\mathrm{II}} = 0 \tag{19}$$

The effects of finite size of the specimen on the stress intensity factors are very small for $B \gg a$ and may be neglected here since $a/B = 10^{-2}$. Thus, (19) will be used as a reference solution for this problem and, with the above assumptions, the exact mode I stress intensity factor is $K_{\mathrm{I}} = \sqrt{\pi}$.

The meshes used for the stress intensity calculation as well as the outer mesh are comprised of a mix of linear quadrilateral and triangular elements. In all the calculations shown in this section, the outer mesh holds 17,514 degrees of freedom.

Results show that for a mesh as coarse as 1022 dofs, error on the computation of K_{I} is 0.48%.

The original Griffith crack is grown through the plate. Fig. 6 shows the evolution of the *mode I* stress intensity factor with crack length. In this example, the crack is assumed to advance an amount fixed to $a/10$ at each

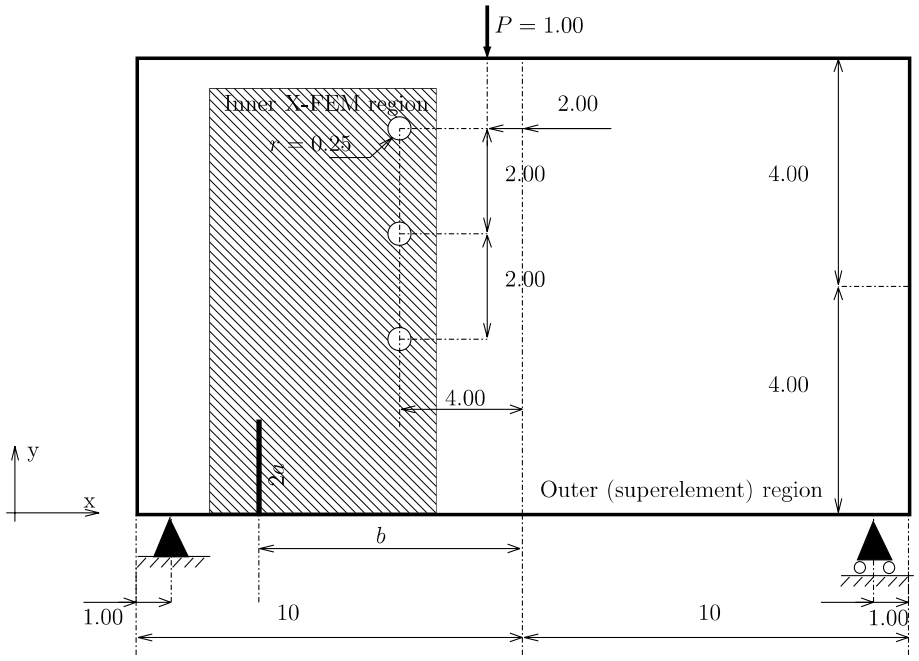


Fig. 7. Schematic of a beam with three holes subjected to three-point bending showing the X-FEM and the superelement parts of the domain.

Table 1
Two different setups for the cracked beam in three-point bending conditions

Configuration	Crack length a (in.)	Crack offset b (in.)
Setup A	1.5	5.0
Setup B	1.0	6.0

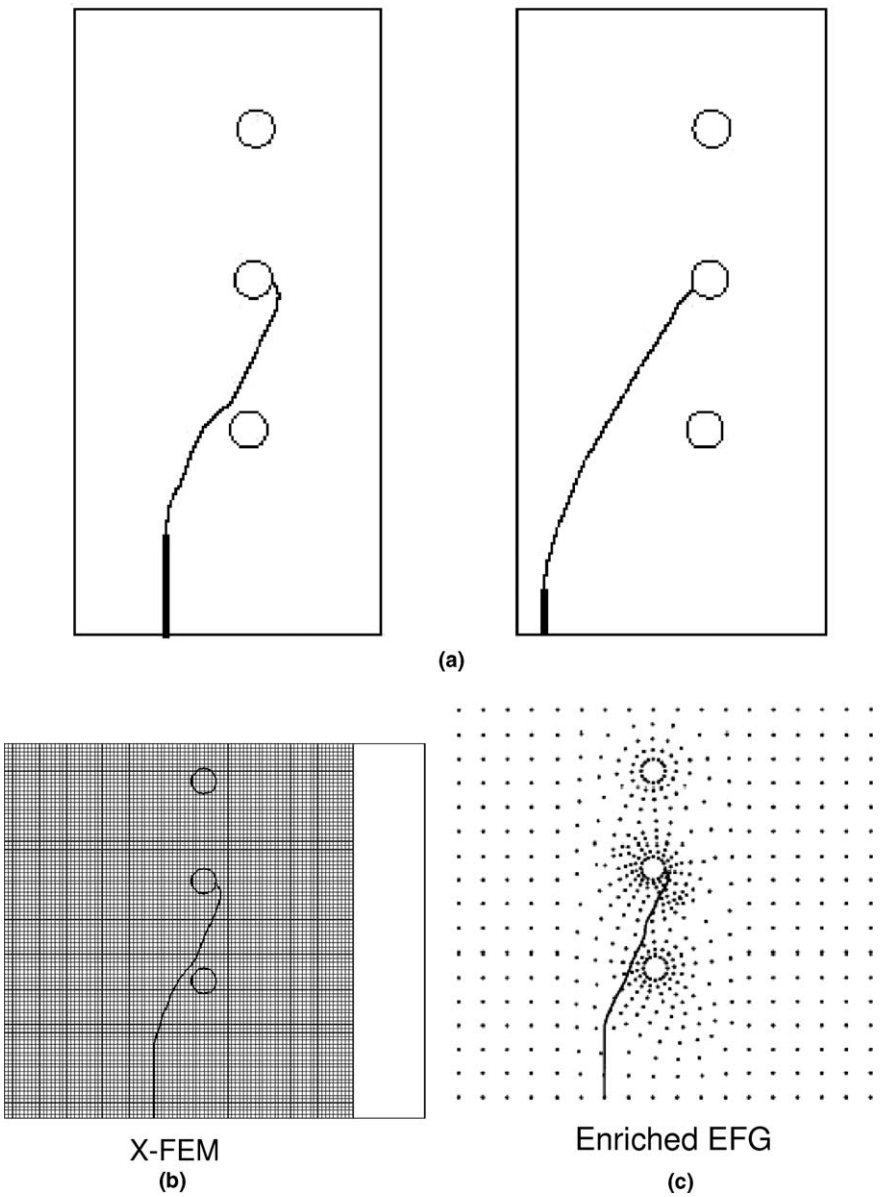


Fig. 8. Comparison of enriched EFG and Superelement/X-FEM and experimental results [38] for 2D crack growth in a perforated plate. (a) Digitized photographs of crack trajectories for setup A: left and setup B: right. (b) Superelement/X-FEM for setup A. (c) Enriched EFG for setup A.

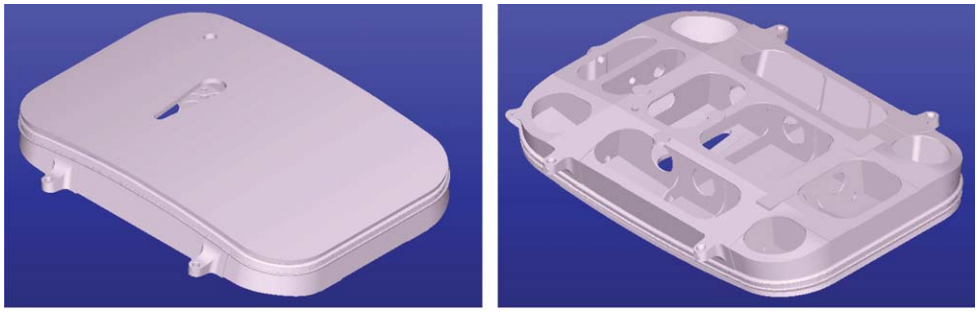
step, where a is the original crack length. As should be the case in this loading mode, the crack is observed to propagate parallel to its initial direction, i.e., perpendicular to the major principal stress direction.

5.2. Crack propagation in a panel with rivet holes

In this benchmark problem, the crack growth in a beam under three-point bending is examined. Three holes are located in a vertical line having an offset from the centerline, and a crack is seeded at the bottom of the beam and allowed to grow (Fig. 7). In [38], an experimental and numerical treatments of this problem are proposed and it is shown that, depending on the location and length of the original crack, the path of the crack

would either intersect one of the holes or pass between them. Table 1 gives the parameters for two different setups that are examined.

The X-FEM mesh and crack growth path are shown in Fig. 8. Different crack growth increments are used and results are compared qualitatively with the experimental and numerical results of [38] and those of [39]. It is emphasized that the same inner mesh may be used for the two cracked configurations. The location and size of the holes (modeled via enrichment) can easily be modified without modifying the inner mesh, which permits examining different scenarios easily.



(a) Top view (b) Bottom view

Fig. 9. Geometry of the Boeing 757 EE Access door.

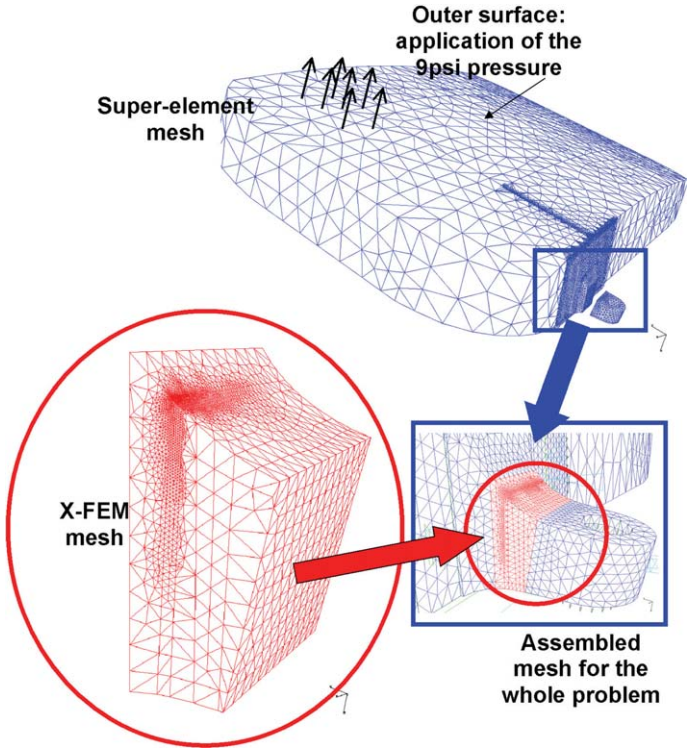


Fig. 10. Superelement model and X-FEM model.

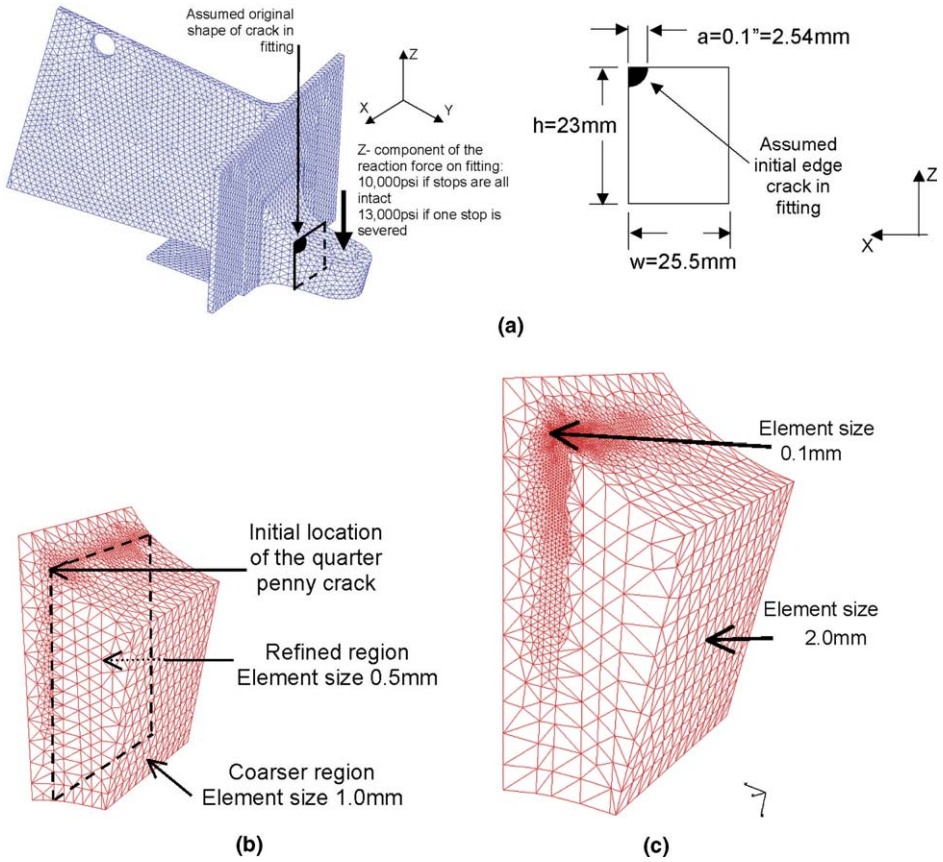


Fig. 11. X-FEM models of the Boeing 757 EE Access Door stop and location of the initial crack. (a) Continuum model and initial location of the crack. (b) X-FEM model: refinement in the crack region, and (c) X-FEM model with 0.1 mm elements in the crack region.

It is interesting to note that the superelement/X-FEM simulations yield crack trajectories that are qualitatively very close to those observed in Fig. 8. It is also interesting to notice that while a relatively coarse mesh suffices to capture the crack behavior, sufficiently small increments in crack length are required for an accurate prediction of the crack path. The same numerical behavior is noted in [39], where the element free Galerkin method [40] is used along with discontinuous enrichment as shown in Fig. 8.

5.3. Industrial example, the Boeing 757 EE Access Door

To further illustrate the application of the superelement/XFEM/level set method as the case of the *Boeing 757 EE Access Door* is examined. The interested reader is referred to [15,41,42] for more details on any question touched upon in this section. The *Boeing 757 EE Access door* is a monolithic casting with an integral skin, many thin ribs and complex cross-sections as shown in Fig. 9.

The model of the door mixes shell elements and solid elements, linked using multiple point constraint (MPC) elements [41].

The door is subjected to a cyclic loading of amplitude 9 psi (normal operating pressure), which is the differential pressure load applied on the outer surface directed from the inside to the outside of the aircraft and due to the pressurization of its fuselage when it is airborne. The pressure load is applied on the curved surface.

A substructure of the door is chosen as the inner structure, based on other design considerations [33,14,42]. Here, this substructure is chosen to be part of one of the stops, where the stresses are highest [41]. Fig. 10 shows the superelement and X-FEM meshes as well as the full, combined mesh.

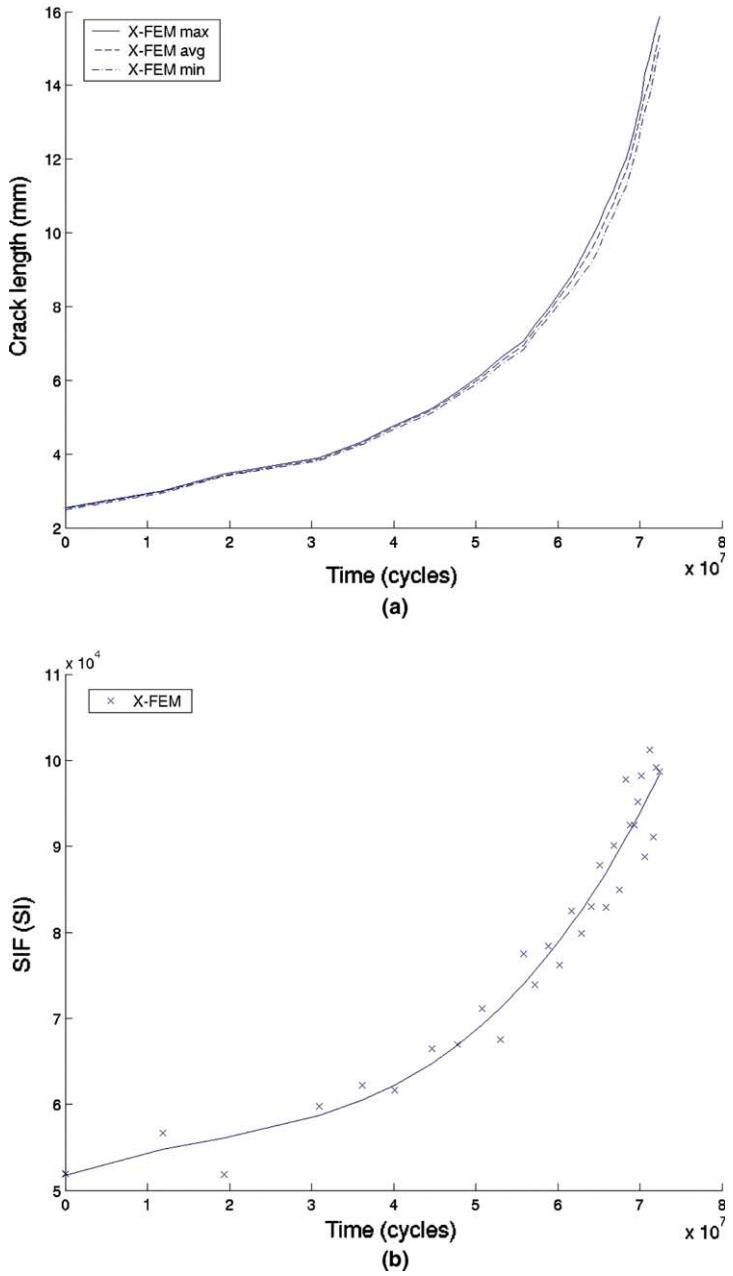


Fig. 12. Crack length and SIF versus number of cycles for a corner edge crack in a stop with a constant SIF assumed on the front for growth. (a) Crack length (mm) versus time (cycles). (b) SIF ($\mathrm{mN/mm^2}\ \sqrt{\mathrm{mm}}$) versus time (cycles).

The material the door is made of is *A356-T60*. The material properties for this aluminum grade are taken as $E = 72.4$ GPa and Poisson's ratio $v = 0.33$. The shear modulus μ is $\mu = E/2(1+v) = 27.2$ GPa. Fatigue properties,[5] for the aluminum were given by the NASA program NASGRO [43] and are available in the code AFGROW [44]:

[5] Paris law is assumed, in which $\mathrm{d}a/\mathrm{d}N = C(\Delta K)^m$: $\mathrm{d}a$ is the increment in crack length, $\mathrm{d}N$ is the increment in the number of loading cycles, ΔK is the stress intensity factor range, C and m are experimentally determined material parameters.

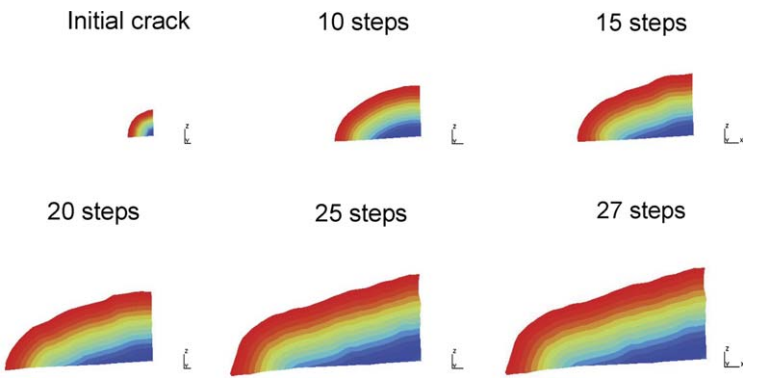


Fig. 13. Crack growth of an edge crack in a stop. In this figure, the horizontal direction corresponds to the $-z$ direction in Fig. 11.

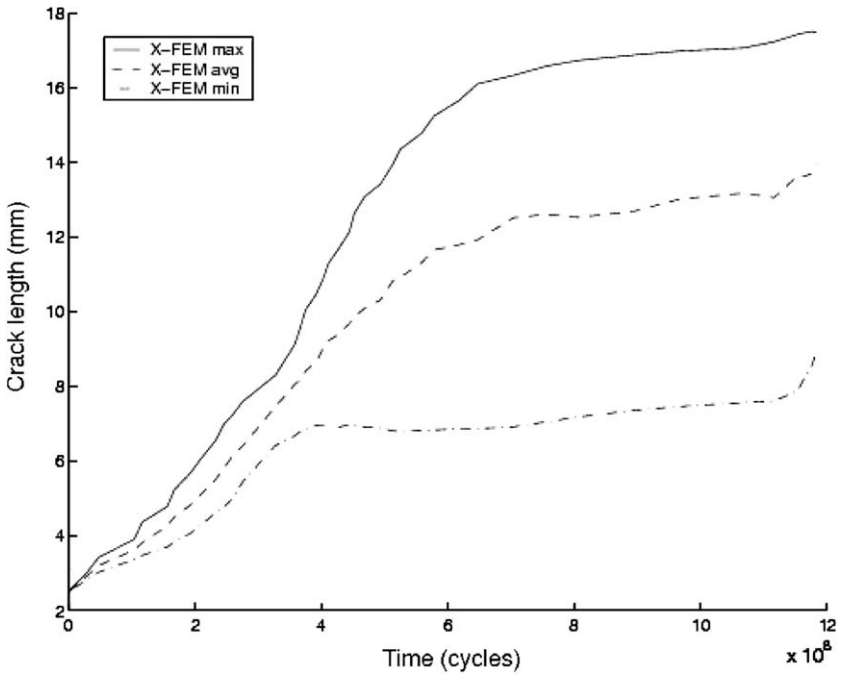


Fig. 14. Crack length versus cycles for the free SIF case.

- $m = 4.98$.
- $C = 2.25 \times 10^{-12}\left(\frac{\text{m}}{\text{cycles}}\right)\left(\frac{1}{\text{MPa}\ \sqrt{\text{m}}}\right)^{4.98}$.

In the analyses presented in this work constant amplitude load cycles (no load spectrum effect) are assumed, and fatigue crack propagation is governed by the Paris law [45]. Threshold effects are neglected.

5.3.1. Damage tolerance evaluation

The method proposed here permits multiple analyses to be carried out with minimum user intervention. Two such analyses are presented here:

(1) An edge crack is present at the maximum stress location (previously determined by a stress analysis) on one of the four fittings.
(2) A large pore (3–4 mm) is present in the bulk of one of the fittings.

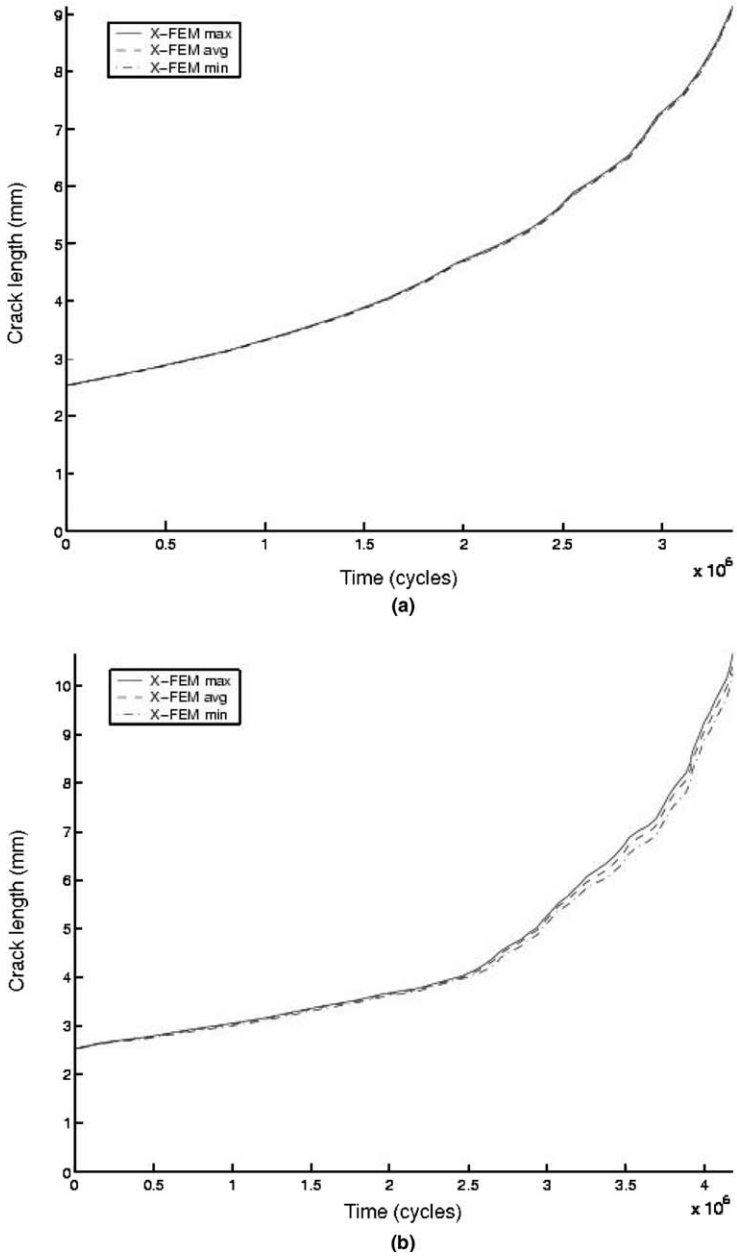


Fig. 15. Severed stop, quarter corner edge crack length versus cycles for two time steps. (a) Crack length (mm) versus time (cycles) with crack advances on the order of 2.5 elements per step. (b) Crack length (mm) versus time (cycles) with crack advances on the order of 1 element per step.

The case of the corner edge crack is chosen to assess the time between two inspections. The initial inspection interval is determined using an initial flaw of 0.1 in. = 2.54 mm. To determine the recurring inspection intervals, an initial flaw of size 0.1 in. = 2.54 mm and a severed stop are assumed.[6]

[6] In the crack growth plots, the minimum, average, and maximum distance from the crack front to the center of the initial flaw are given. Initially, for a circular flaw, all points on the front are equidistant from the center of the flaw. If the SIF is not constant on the front, some points on this front may grow faster than others thereby creating a divergence in those three values.

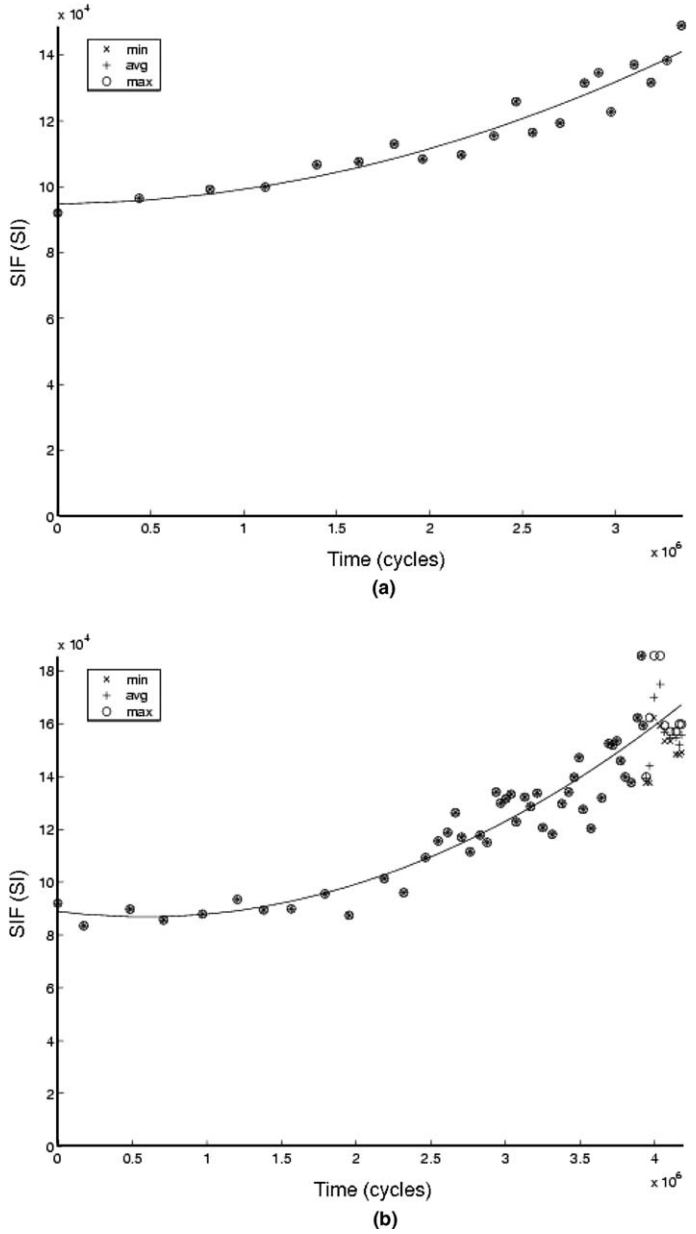


Fig. 16. Severed stop, equivalent SIF on the front of the quarter corner edge crack for two time steps. (a) SIF ($mN/mm^2\sqrt{mm}$) with crack advances on the order of 2.5 elements per step. (b) SIF ($mN/mm^2\sqrt{mm}$) with crack advances on the order of 1 element per step.

5.3.2. Fatigue growth of a quarter corner penny-shape edge crack in a stop

The superelement mesh used for the following analyses is shown in Fig. 10. Fig. 11 shows the X-FEM mesh and the initial location of the quarter corner penny-shape edge crack in the stop.

In a first approach, the stress intensity factor on the crack is forced to be constant and equal to the maximum SIF on the front along the whole crack front, yielding a crack length and SIF versus time curves shown in Fig. 12.

In a second approach, the actual value of the equivalent stress intensity factor at each point on the front is used to advance the crack yielding a crack length versus cycle curve shown in Fig. 14. The crack configurations at different stages of the growth process are presented in Fig. 13. Note that when the crack is left free to evolve,

Table 2
Effect of a severed stop on crack growth rates with a Paris exponent equal to 4.98; SIFs in $mN/mm^2\ \sqrt{mm}$, crack growth rate in mm/cycle, number of cycles in millions

	SIF	Crack growth rate	Fatigue life	SIF (3 million cycles)
Intact	5.25×10^4	2.8×10^{-8}	70	6×10^4
Severed	9.5×10^4	5.4×10^{-7}	3.5	13×10^4
$\frac{\text{Severed}}{\text{Intact}}$	1.8	19.2	20	2.1

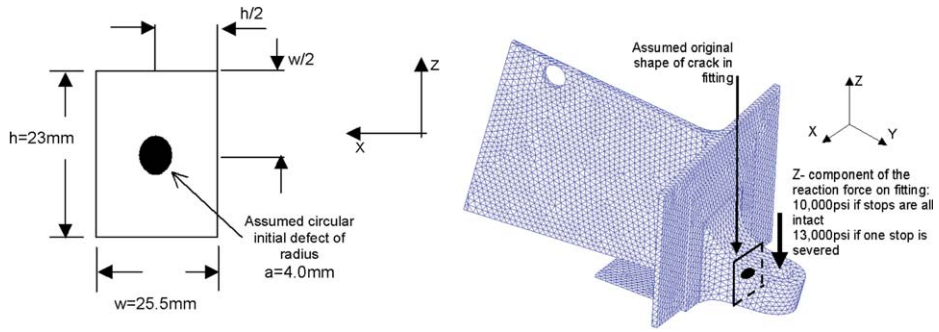


Fig. 17. Large pore assumed present in the stop for damage tolerance analysis of the Boeing 757 EE Access door.

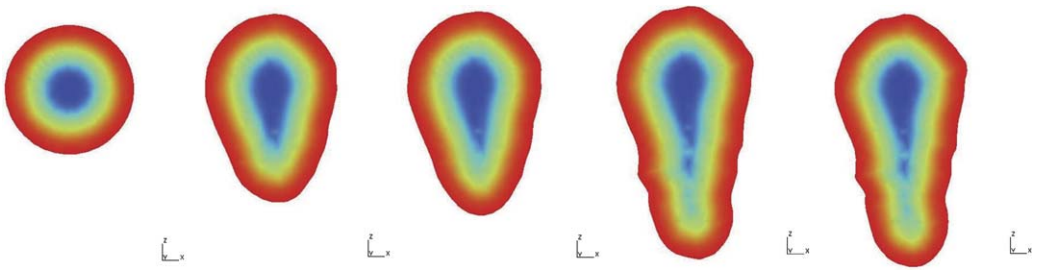


Fig. 18. Crack evolution from an initial large pore in a stop.

it tends to grow faster on the vertical side of the stop than on the horizontal side because of the compressive state of stress that reigns in the bottom part of the stop.

In the second scenario, detectable damage is considered in the form of a missing stop. Results for two time steps, are shown in Figs. 15 and 16. Fig. 16 shows that the stress intensity factors on the front do not reach the fracture toughness of the material. Even after three million cycles and a length of 8 mm, the maximum value of the equivalent SIF on the front is approximately $12.5 \times 10^4\ mN/mm^2\ \sqrt{mm} = 3.95\ MPa\ \sqrt{m} < 17.581\ MPa\ \sqrt{m} = K_c$. The fatigue life given by this method is approximately 70 million cycles. Applying a factor of safety of 4.0, the predicted inspection interval for the component is about 750,000 cycles and is well beyond the expected economic life of a commercial aircraft. For the case of the severed stop, the predicted fatigue life is around 4 million cycles. This severed/intact ratio of about 1–20 is due to the large Paris exponent (4.98) used for the analysis. Table 2 compares the severed to the intact case.

5.3.3. Fatigue growth of a crack originating from a large pore in the bulk of a fitting

The meshes used in this analysis are identical to those presented in Section 5.3.2. The initial defect is assumed to be a relatively large (3–4 mm) circular pore located in the bulk of the stop (Fig. 17).

Fig. 18 shows the evolution in time of the crack. Note that the bottom half part of the crack is under tension (opening mode) and therefore tends to open and propagate, while the top half part of the crack is located

in a compressive region and propagates a much smaller amount. The method resolves this phenomenon naturally, without any additional modification. Contact between the crack faces is not accounted for here.

6. Conclusions

In the technique proposed in this paper, the linear elastic fracture problem in a complex structure is split into two domains: the outer and the inner domain – several inner domains may coexist. In the former, structural complexities are encompassed and entirely treated within the commercial code. In the latter, cracks are introduced and treated by a hybrid extended finite element level set method:

- Complex meshing features are handled in the commercial software and enter the computation through the outer stiffness.
- The entire structure participates to the crack growth process: no additional assumption is needed to grow the crack in any complex structure.
- The same outer mesh can be used while only varying the inner mesh in the vicinity of the cracks, for analysis of various scenarios.

Within the inner domain, the presence of the crack is *only* accounted for by discontinuous and asymptotic enrichment functions and two level set functions, which allows for:

- No (re)meshing of the crack faces.
- Coarser meshes for the same accuracy as with finite elements (asymptotic enrichment).
- No tedious geometrical tracking of the crack, even for non-planar crack surfaces.
- Alleviates the labor associated with mesh data processing.

The methodology presented was successfully applied to a real-world aerospace component, which proved its simplicity and accuracy, and the authors believe the proposed paradigm to be a promising tool for damage tolerance assessment of complex three-dimensional components.

Acknowledgments

The authors gratefully acknowledge the support of the FAA through FAA Contract DTFA03-98-F-IA025 *Design and Quality Assurance of Premium Quality Aerospace Castings*. Special thanks is extended to Xiaogong Lee, John Backuckas and Terry Khaled of the FAA for their support and availability throughout the project. The availability of Ed. Nichols for many helpful discussions on industry practice pertaining to DTA is underlined. The kind help of Jay Terry and Professor Jack Chessa is very much appreciated.

References

[1] Belytschko T, Black T. Elastic crack growth in finite elements with minimal remeshing. Int J Numer Meth Engng 1999;45:601–20.
[2] Moës N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. Int J Numer Meth Engng 1999;46:131–50.
[3] Babuška I, Rosenzweig M. A finite element scheme for domains with corners. Numer Math 1972;20:1–21.
[4] Grisvard P. Elliptic problems in nonsmooth domains. Boston: Pitman Publishing, Inc.; 1985.
[5] Forth S, Keat W. Three-dimensional nonplanar fracture model using the surface integral method. Int J Fract 1996;77:243–62.
[6] Sladek J, Sladek V, Atluri SN. Local boundary integral equation (LBIE) method for solving problems of elasticity with nonhomogeneous material properties. Comput Mech 2000;24(6):456–62.
[7] Shih CF, Moran B, Nakamura T. Energy release rate along a three-dimensional crack front in a thermally stressed body. Int J Fract 1986;30:79–102.
[8] Nikishkov GP, Atluri SN. Calculation of fracture mechanics parameters for an arbitrary three-dimensional crack by the 'equivalent domain integral' method. Int J Numer Meth Engng 1987;24:851–67.
[9] Moran B, Shih CF. Crack tip and associated domain integrals from momentum and energy balance. Engng Fract Mech 1987;127:615–42.

[10] Gosz M, Moran B. An interaction energy integral method for the computation of mixed-mode stress intensity factors along non-planar crack fronts in three dimensions. Engng Fract Mech 2002;29:299–319.
[11] Gosz M, Dolbow J, Moran B. Domain integral formulation for stress intensity factor computation along curved three-dimensional interface cracks. Int J Solids Struct 1998;35:1763–83.
[12] Moës N, Gravouil A, Belytschko T. Non-planar 3D crack growth by the extended finite element and level sets. Part I. Mechanical model. Int J Numer Meth Engng 2002;53:2549–68.
[13] Gravouil A, Moës N, Belytschko T. Non-planar 3D crack growth by the extended finite element and level sets. Part II. Level set update. Int J Numer Meth Engng 2002;53(11):2569–86.
[14] Bordas SPA. Extended finite element method and level set methods with applications to the growth of cracks and biofilms, Ph.D. thesis, Northwestern University, December 2003.
[15] Moran B, Bordas S, Conley JG. Damage tolerance assessment of complex aerospace structures, design and quality assurance of premium quality aerospace castings. FAA contract DTFA03-98-F-IA025 Northwestern University, Center for Quality Engineering and Failure Prevention, McCormick School of Engineering and Applied Science.
[16] Stolarska M, Chopp DL, Moës N, Belytschko T. Modelling crack growth by level sets and the extended finite element method. Int J Numer Meth Engng 2001;51(8):943–60.
[17] Sukumar N, Moës N, Belytschko T, Moran B. Extended finite element method for three-dimensional crack modelling. Int J Numer Meth Engng 2000;48(11):1549–70.
[18] Sukumar N, Chopp DL, Moran B. Extended finite element method and fast marching method for three-dimensional fatigue crack propagation. Engng Fract Mech 2003;70(1):29–48.
[19] Chopp DL, Sukumar N. Fatigue crack propagation of multiple coplanar cracks with the coupled extended finite element/fast marching method. Int J Engng Sci 2003;41(8):865–9.
[20] Melenk JM, Babuška I. The partition of unity finite element method: Basic theory and applications. Comput Meth Appl Mech Engng 1996;139:289–314.
[21] Babuška I, Melenk I. Partition of unity method. Int J Numer Meth Engng 1997;40(4):727–58.
[22] Belytschko T, Moës N, Usui S, Parimi C. Arbitrary discontinuities in finite elements. Int J Numer Meth Engng 2001;50(4):993–1013.
[23] Bordas S, Nguyen P, Dunant C, Nguyen DH, Guidoum A. An extended finite element library. Int J Numer Meth Engng, submitted for publication.
[24] Sethian JA. Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge, UK: Cambridge University Press; 1999.
[25] Osher S, Sethian J. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulations. J Comput Phys 1988;79(1):12–49.
[26] Barth TJ, Sethian JA. Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains. J Comput Phys 1998;145(1):1–40.
[27] Sussman M, Almgren A, Bell JB, Colella P, Howell LH, Welcome ML. An adaptive level set approach for incompressible fluid flow. Comput Phys 1999;148:81–124.
[28] Sussman M, Fatemi E. An efficient interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. J Sci Comput 1999;20(4):1165–91.
[29] Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flows. J Comput Phys 1994;114:146–59.
[30] Sussman M, Fatemi E, Smereka P, Osher S. An improved level set method for incompressible two-phase flows. Comput Fluids 1997;27(5):663–80.
[31] Stazi F, Budyn E, Chessa J, Belytschko T. An extended finite element method with higher-order elements for crack problems with curvature. Comput Mech 2003;31:38–48.
[32] Legay A, Belytschko T. Strong and weak arbitrary discontinuities in spectral finite elements. Int J Numer Meth Engng 2005;94(8):991–1008.
[33] Bordas S, Conley JG, Moran B, Gray J, Nichols E. A design paradigm for integrating casting modeling, damage tolerance and non-destructive evaluation. Engng Comput., submitted for publication.
[34] Stroustrup B. C++ programming language. 3rd ed. Addison Wesley; 2002.
[35] Rehak DR, Baugh JJW. Alternative programming techniques for finite element program development, In: IABSE (Ed.), IABSE colloquium on expert systems in civil engineering, Bergamo, Italy, 1989.
[36] Forde BWR, Foschi RB, Stiemer SF. Object-oriented finite element analysis. Comput Struct 1990;34:355–74.
[37] Zimmermann T, Dubois-Pèlerin Y, Bomme P. Object-oriented finite element programming. I. Governing principles. Comput Meth Appl Mech Engng 1992;92:291–303.
[38] Bittencourt TN, Wawrzynek PA, Ingraffea AR, Sousa JL. Quasi-automatic simulation of crack propagation for 2D LEFM problems. Engng Fract Mech 1996;55:321–34.
[39] Ventura G, Xu JX, Belytschko T. Level set crack propagation modelling in the element-free galerkin method. In: European conference on computational mechanics, June 2001.
[40] Belytschko T, Lu YY, Gu L. Element-free Galerkin methods. Int J Numer Meth Engng 1994;37:229–56.
[41] Moran B, Bordas S, Conley JG. Static strength analysis of aerospace castings, design and quality assurance of premium quality aerospace castings. FAA contract DTFA03-98-F-IA025 Northwestern University, Center for Quality Engineering and Failure Prevention, McCormick School of Engineering and Applied Science.

[42] Conley JG, Moran B, Bordas S. Integrated design approach of aerospace castings, design and quality assurance of premium quality aerospace castings. FAA contract DTFA03-98-F-IA025 Northwestern University, Center for Quality Engineering and Failure Prevention, McCormick School of Engineering and Applied Science.
[43] NASA, Nasgro, NASA Johnson Space Center and Southwest Research Institute (SwRI). Available from: http://www.nasgro.swri.org/, 2003.
[44] Harter J. AFGROW. Available from: http://fibec.flight.wpafb.af.mil/fibec/afgrow.html, 1996.
[45] Paris P, Erdogan F. A critical analysis of crack propagation laws. J Basic Engng, Trans Am Soc Mech Engrs 1963(December):528–34.