INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING

Int. J. Numer. Meth. Engng 2009; **1**:1 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

An XFEM method for modelling geometrically elaborate crack propagation in brittle materials

Casey L. Richardson[1,*], Jan Hegemann[2], Eftychios Sifakis[1], Jeffrey Hellrung[1], Joseph M. Teran[1]

[1] *Department of Mathematics, University of California, Los Angeles, Box 951555, Los Angeles, CA 90095-1555*

[2] *Institute for Computational and Applied Mathematics, University of Münster, Einsteinstrasse 62, D-48149 Münster*

SUMMARY

We present a method for simulating quasistatic crack propagation in 2-D which combines the extended finite element method (XFEM) with a general algorithm for cutting triangulated domains, and introduce a simple yet general and flexible quadrature rule based on the same geometric algorithm. The combination of these methods gives several advantages. First, the cutting algorithm provides a flexible and systematic way of determining material connectivity, which is required by the XFEM enrichment functions. Also, our integration scheme is straightfoward to implement and accurate, without requiring a triangulation that incorporates the new crack edges or the addition of new degrees of freedom to the system. The use of this cutting algorithm and integration rule allows for geometrically complicated domains and complex crack patterns. Copyright © 2009 John Wiley & Sons, Ltd.

*Correspondence to: Casey Richardson, Department of Mathematics, University of California, Los Angeles, Box 951555, Los Angeles, CA 90095-1555, e-mail: clr@math.ucla.edu

1. INTRODUCTION

Simulating the propagation of cracks using traditional finite element methods is challenging because the topology of the domain changes continuously. The Extended Finite Element Method (XFEM) has been used very successfully to model cracks because the finite element mesh can be created independent from the crack geometry, and in particular the domain does not have to be remeshed as the crack propagates.

We now summarize the main idea and historical background of XFEM (see [1], [2], and [3] for more complete surveys). The idea is to enrich the usual finite element spaces with additional degrees of freedom, which incorporate the near tip asymptotic solutions and allow the displacements to be discontinuous across the crack face. The application of XFEM to cracks began with the Belytschko and Black [4], where they applied the partition of unity methods (see for instance [5]) to the problem of using finite elements with discontinuous basis functions. In [6] Moes, et. al., used XFEM to create a technique for simulating crack propagation in two dimensions without remeshing the domain. The extension to three dimensions was begun by Sukumar et al. [7], where they used the two dimensional enrichment functions for planar cracks, and then extended in [8].

Since its introduction, XFEM enrichment has been employed in a variety of settings to model fracture. Using special enrichments cohesive fracture can be modelled, this began with the work of Moes and Belytschko [9], extended in [10], and continues to be developed (see for instance [11], [12], [13], [14]). There has been much work in other settings: [15] for fracture with elastodynamics, and [16] to model crack propagation in composite materials. XFEM has been combined naturally with the level set method of Osher and Sethian ([17], [18]) to track the moving discontinuity sets (for cracks see for instance [19], [20], [21], [22], [23] and

for holes and inclusions see [24]) and has also been coupled with fast marching methods by Sukumar, et. al. [25]. There has been some work to study the error estimates [26] and XFEM has also been combined with other techniques to increase the rate of convergence, such as cut off functions and geometric enrichment in [27], [28] and [29]. However, the XFEM approach carries technical challenges: assembling the stiffness matrix requires integration of singular/discontinuous functions and implementing enrichment requires resolving material connectivity (often using a level set representation).

Integrating the gradients of the XFEM basis functions is difficult because of the singularities and discontinuities. As noted in [30], the use of Gauss quadrature or Monte Carlo integration is unstable: since the crack path through a given triangle is unknown a priori, the singularities can move very close to quadrature points. One approach to the problem (see, for instance, [31]) is to perform a Delaunay triangulation on the cut triangle that incorporates the crack edges, and then to use Gauss quadrature on each of the resulting triangles. This triangulation does not provide additional degrees of freedom; it is only used for integration of the basis functions. However, this approach can be difficult to implement when the geometry of the crack is complicated by branching or multiple cracks, and is generally impossible in three dimensions without introducing new vertices. Two other methods, introduced in [32] and [33], use mappings of the crack enrichment functions to domains where Gauss quadrature can be used, but also require meshing of the tip triangle. Another approach is to use higher order Gauss quadrature (see [34]). In [35], Ventura, et al. transformed the area integral required for assembly of the stiffness matrix to a more stable line integral. In [36], Park, et al. also used a mapping technique to remove the singularity for tetrahedral elements (in three dimensions). Also for integrating singularities in tetrahedral elements, Areias and Belytschko used a smoothing

technique in [8]. For integrating the Heaviside functions, Ventura [37] used a map to equivalent polynomials which were integrated using standard quadrature techniques; Holdych, et al. [38] used a similar technique where they introduced a dependence of the Gauss weights on the position of the node within the triangle. In [39], Benvenuti, et al. regularize the Heaviside function for integration with Gauss quadrature, and then prove that the solutions coverge as the regularization parameter goes to zero. We introduce a simple method of integration (see Section 4) that combines naturally with our geometric cutting algorithm (see Section 3).

In order to allow cracks to open, XFEM needs to generate additional degrees of freedom, a process referred to as *enrichment*. In a region that has been unambiguously separated into two pieces (i.e., away from the crack tip), the enrichment is provided by a Heaviside function, defined to be 1 on one side of the crack and -1 on the other side. This is easy in the case of a single straight crack, but more challenging as the geometry of the crack becomes complicated. In [30], Daux et. al., handle the case of branched cracks by using separate enrichments for each crack, and then use another enrichment function to represent the junction itself. They then generalize this technique to cracks that have multiple branches, however their method requires that the cracks have been hierarchically decomposed into a main crack and its branched components, and still involves solving the problem of material connectivity. In [40] and [41] the above work was extended to incorporate multiple cracks and to address the issue of cracks intersecting each other. In [42], Song and Belytschko introduced the cracking node method, which is based on XFEM and is designed to more easily handle complicated crack geometries. The use of virtual or ghost nodes to incorporate discontinuities has become increasingly popular, see for instance [43]. The methods of Hansbo and Hansbo [44] and Song et al. (see [45], [46]), which are equivalent (see [47]), use a notion of ghost or phantom degrees

of freedom to handle displacement discontinuities. Also, Dolbow and Harari [48] use phantom nodes in the context of embedded interface problems. We use a similar notion of virtual nodes, which are created by leveraging a recent computational geometric algorithm to cut domains with crack curves.

In this paper, we present a method for simulating quasistatic crack propagation in 2-D, which combines XFEM with a simple integration technique and a very general algorithm for cutting triangulated domains. Our approach:

- is based on virtual nodes created by a cutting algorithm that incorporates material connectivity,
- can handle complicated crack patterns (including multiple tips in the same element, crack branches, and crack tips in fully cut elements),
- can handle geometrically complex domains,
- does not require remeshing of the domain (which is in the spirit of XFEM),
- employs a quadrature rule that is built on the cutting algorithm, and whose degree of complexity is independent of the crack geometry.

2. GOVERNING EQUATIONS

Under the assumption of a quasistatic evolution, it can be assumed that at each time the material is in elastic equilibrium. Denoting the rest configuration of the material by $\Omega \subset \mathbb{R}^2$ open and bounded (with boundary denoted $\partial\Omega$), the equations of material equilibrium are

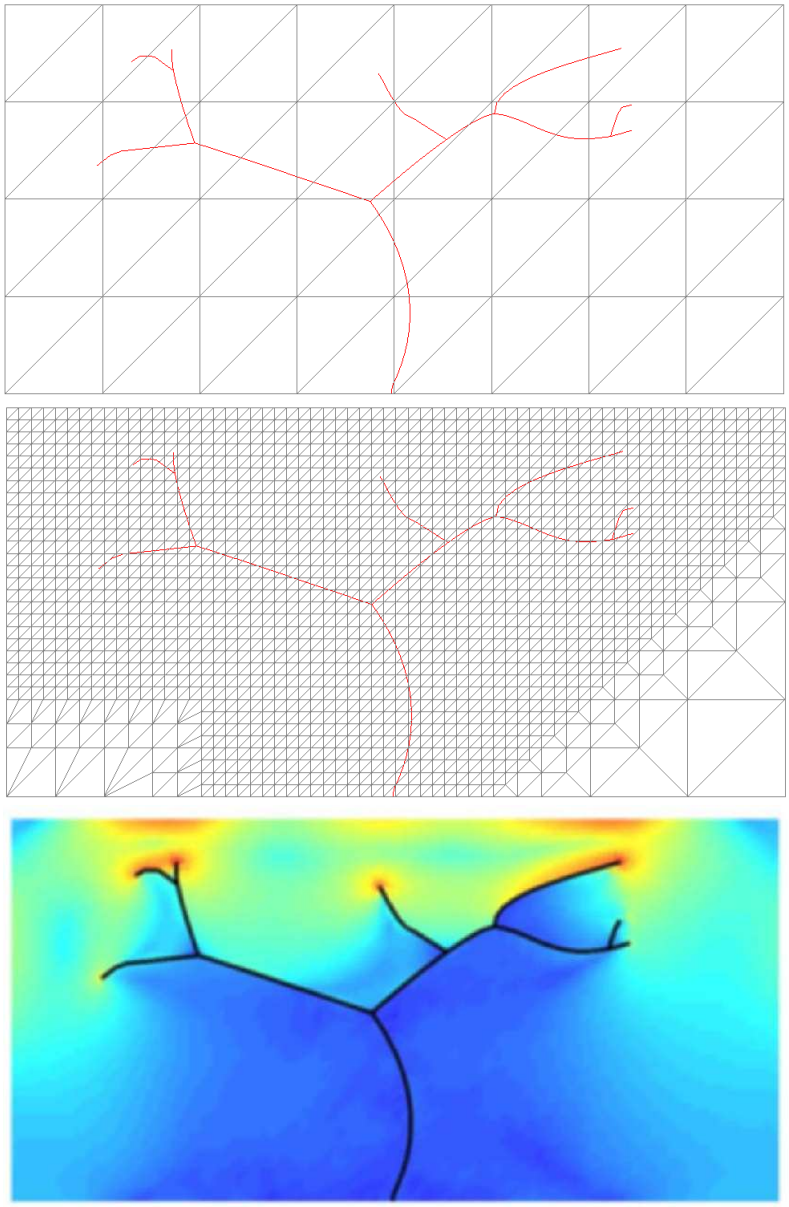

Figure 1. An example of our technique: *top*, a crack cutting the simulation mesh; *center*, cutting of the embedded quadrature mesh; *bottom*, computed stress field, with uniform traction applied to left and right edges.

given by:

$$\begin{aligned} \operatorname{div}(\boldsymbol{\sigma}) + \boldsymbol{b} = 0 \qquad & \text{in } \Omega \setminus \Gamma, \\ \boldsymbol{u} = \boldsymbol{u_0} \qquad & \text{on } \partial\Omega_D, \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = \bar{\boldsymbol{t}} \qquad & \text{on } \partial\Omega_t, \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad & \text{on } \Gamma^+, \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad & \text{on } \Gamma^-. \end{aligned}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{b}$ is the body force per unit volume, $\boldsymbol{u}$ is the displacement, $\boldsymbol{u_0}$ are the Dirichlet values (applied to a subset of the boundary that we write as $\partial\Omega_D$), $\bar{\boldsymbol{t}}$ is the traction (applied to $\partial\Omega_t \subset \partial\Omega$), $\boldsymbol{n}$ denotes the unit outer normal, Γ is the crack surface, and Γ^+, Γ^- represent the two different orientations of the crack surface. In this paper, we will consider the case of small strains and displacements and linear elasticity, which further gives us the relations:

$$\boldsymbol{\varepsilon}(\boldsymbol{u}) = \boldsymbol{\nabla_s} \boldsymbol{u},$$

where $\boldsymbol{\nabla_s} \boldsymbol{u}$ is the symmetric part of the displacement gradient, and

$$\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\varepsilon},$$

where $\boldsymbol{C}$ is the Hooke tensor. Equivalently, equilibrium can be described as minimizing the potential energy, i.e., one would find the displacement u that minimizes

$$\Psi[\boldsymbol{u}] := \frac{1}{2} \int_\Omega \boldsymbol{\varepsilon}(\boldsymbol{u}) : \boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{u}) dx - \int_\Omega \boldsymbol{b} \cdot \boldsymbol{u} dx - \int_{\partial\Omega_t} \bar{\boldsymbol{t}} \cdot \boldsymbol{u} ds \tag{1}$$

subject to $\boldsymbol{u} = \boldsymbol{u_0}$ on $\partial\Omega_D$.

2.1. Extended Finite Elements

Conceptually, the starting point of our work is the extended finite element method (XFEM), which was proposed and studied in the context of fracture by Belytschko, et. al., see for instance [6]. In this method, at each step in the evolution one solves an approximation to the equations of material equilibrium in a finite dimensional subspace. This subspace is formed by taking the usual C_0 conforming finite elements (in our case, on triangles), and *enriching* this space with additional degrees of freedom that allow cracks to open and increase the accuracy of the approximation near the crack tip. Thus, the functions in the XFEM space $\mathcal{U}_h$ have the form:

$$\boldsymbol{u}_h(\boldsymbol{x}) = \sum_i \boldsymbol{u}_i \phi_i(\boldsymbol{x}) + \sum_j \boldsymbol{b}_j \phi_j(\boldsymbol{x}) H(\boldsymbol{x}) + \sum_k \phi_k(\boldsymbol{x}) \left(\sum_{\ell=1}^{4} \boldsymbol{c}_k^\ell F_\ell(r(\boldsymbol{x}), \theta(\boldsymbol{x})) \right), \qquad (2)$$

where $\{\phi_i\}$ are the usual nodal basis functions, $H(\boldsymbol{x})$ is the Heaviside function associated to the current crack geometry, $\{\boldsymbol{b}_j\}$ are enrichment degrees of freedom associated with crack separation away from the tip, $\{\boldsymbol{c}_k^\ell\}$ are enrichment degrees of freedom associated with near-tip displacement, and

$$\{F_\ell(r,\theta)\} := \left\{ \sqrt{r}\sin(\theta/2), \sqrt{r}\cos(\theta/2), \sqrt{r}\sin(\theta/2)\sin(\theta), \sqrt{r}\cos(\theta/2)\sin(\theta) \right\}$$

are the asymptotic crack tip functions (r and θ are the polar coordinates from the crack tip). Notice in (2) that $\boldsymbol{u}_h$ is written as a linear combination of three types of basis functions: the nodal basis functions (which have support local to mesh nodes), Heaviside enrichment functions, and the near-tip enrichment functions (which have support local to the crack tip). The sum over i in (2) is taken over all the mesh nodes, while the second two sums are taken over those nodes whose conforming basis functions have support that overlaps the crack (see [6]). We will henceforth refer to the set of elements in the support of the conforming basis function for a node as the one-ring of that node. The representation in (2) is for a crack with

only one crack tip, but can be generalized to accomodate crack geometries that have multiple tips.

3. CUTTING OF CRACKED DOMAINS

For simplicity of exposition we first focus on the discretization away from any crack tips (thus, ignoring tip enrichment) where (2) takes the simpler form:

$$\boldsymbol{u}_h(\boldsymbol{x}) = \sum_i \boldsymbol{u}_i \phi_i(\boldsymbol{x}) + \sum_{n_j \in J} \boldsymbol{b}_j \phi_j(\boldsymbol{x}) H(\boldsymbol{x}), \tag{3}$$

where J is the set of nodes whose one-ring is cut by the crack. A part of our approach is to replace the Heaviside degrees of freedom with virtual nodes (see [43]), also known as ghost or phantom nodes (see for instance [45], [44], [48]). However, we create our virtual nodes in a different way: we use a cutting algorithm designed to cut triangulated domains with arbitrary curves (see [49]).

3.1. Description of the Cutting Algorithm

We now briefly summarize the cutting algorithm (see [49] for more detail). In two dimensions, the algorithm operates on a triangulated domain and a segmented cutting curve, and produces another mesh whose triangles incident to the crack have been duplicated into materially disconnected counterparts (see Figure 5).

In the first stage, the cutting algorithm processes each individual triangle. We identify the distinct material components that the triangle is split into by the cutting curve and describe each of them as a closed polygonal region (depicted in blue in Figure 2). Then, for each of these material regions, a duplicate copy of the triangle is made, creating new vertices (called virtual nodes) in the non-material regions of these duplicate triangles. In Figure 2, the cutting surface divides the triangle into two distinct material regions, so two copies of the original triangle are made, with each of the copies associated to one of the material regions. In the duplicated triangles, the nodes in the material regions (drawn in the figure with solid blue dots) can be

identified with the three original degrees of freedom. The two triangles are also furnished with virtual nodes (in the right of the figure, the nodes labeled n_4, n_5, n_6 and depicted as unfilled blue circles). A triangle that has been cut by a more geometrically complex crack surface is depicted in Figure 3. In this scenario, the triangle at left is duplicated three times, each copy associated to its own material region.

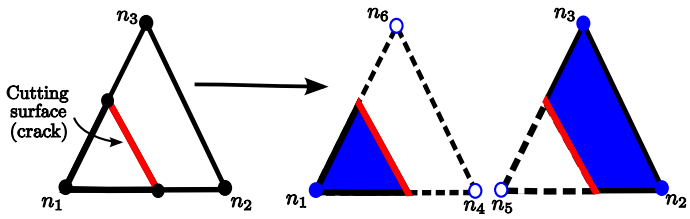

Figure 2. Cutting example: on *left*, original mesh triangle; on *right*, duplicates with material regions (*solid blue*) and virtual nodes (*hollow blue circles*)

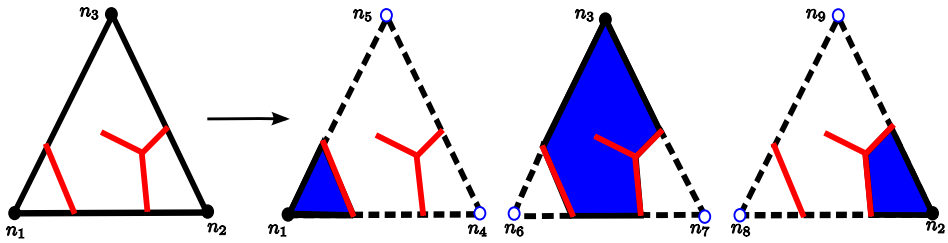

Figure 3. Cutting example for a complex crack: on *left*, original mesh triangle; on *right*, three duplicates with material regions (*solid blue*) and six virtual nodes (*hollow blue circles*)

After processing all individual triangles affected by the crack, the algorithm determines the global material connectivity (see Figure 4). For a given triangle T' of the aforementioned duplication process, each of its vertices is either a material node, meaning it is contained in the material region assigned to that triangle copy, or a virtual node. For each triangle T of

the uncut mesh, let $C(T)$ denote the set of triangles in the duplicated mesh that were created by copying T. Also, let P be the map that takes a triangle T' of the duplicated mesh to its parent triangle in the simulation mesh, i.e., the triangle that was copied in order to create T'. The cutting algorithm then proceeds as follows. For every triangle T' in the duplicated mesh, find $T = P(T')$. For every triangle T_2 that neighbors T, inspect each triangle in $C(T_2)$, and determine if it shares a material connection with the original triangle T'. If so, the relevant nodes are identified as equivalent, and the corresponding degrees of freedom are collapsed.

The entire cutting process is illustrated in Figure 4. The mesh at left, composed of three triangles, is cut by the two red cracks (here the geometry is quite complicated, since the center triangle contains a branch, a tip, and is cut into multiple pieces). First, the cutting algorithm treats each triangle in isolation, and creates a duplicate version for each material region created by the crack, shown at the center of Figure 4. Then, in the second phase, the copies are joined together so that they are hinged on the same degrees of freedom where they share material connectivity along an edge (on the right of the figure).

Figure 5 illustrates the result of the cutting algorithm where the cutting surface completely cuts one triangle and only partially cuts another. Figure 6 illustrates the results of taking the mesh on the left of Figure 5, refining it near the crack, cutting this refined mesh, and then resolving the global connectivity (the circular inset shows how the resulting virtual nodes allow the crack to separate). In our context, the unrefined mesh corresponds to our simulation mesh, while the refined mesh – after being cut as in the diagram – is used for quadrature purposes (see Section 4).

Now, consider a mesh that has been cut as described above. Since some of the nodes are virtual, meaning they do not correspond to material nodes, as in [44] we create nodal basis

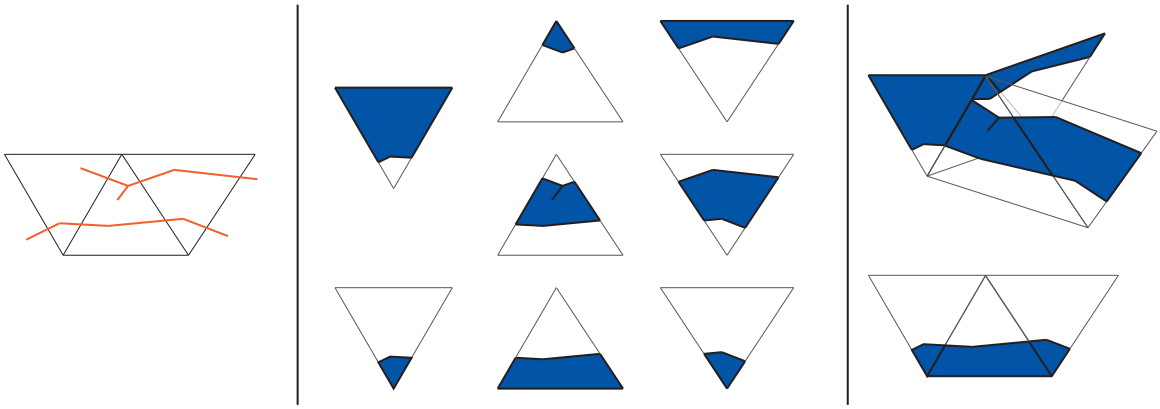

Figure 4. The mesh at left is cut by two cracks, one of which contains a branch. The cutting algorithm first treats each triangle separately, creating duplicates for each material region (*at center*), and then uses the global mesh topology to hinge these duplicates on the proper degrees of freedom (*at right*).

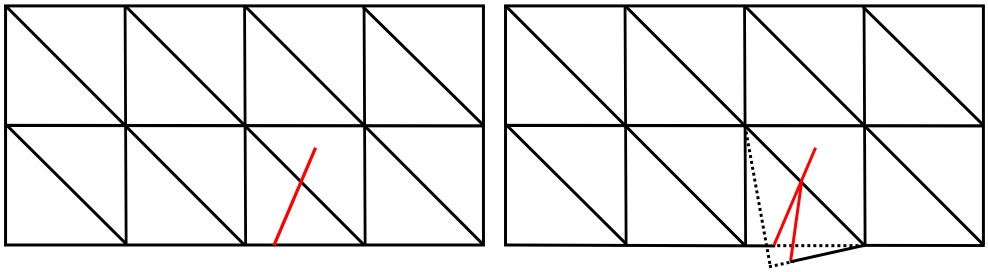

Figure 5. Global cut topology, unrefined mesh. The mesh at *left* is cut by the crack, resulting in the mesh at *right*, with duplicated triangles and virtual nodes.

functions that respect the crack geometry, i.e., we take into account that nodes and triangles may have been duplicated (see Figure 7 for a one-dimensional illustration). Take $\{\tilde{\phi}_i\}$ to be the usual piecewise affine "hat" functions for the simulation mesh, and $\boldsymbol{x_i}$ to be the node of the simulation mesh corresponding to $\tilde{\phi}_i$ (again, this could be a virtual node). Using ω_i to denote the collection of triangles in the one-ring of the mesh node $\boldsymbol{x_i}$, we define a new set of

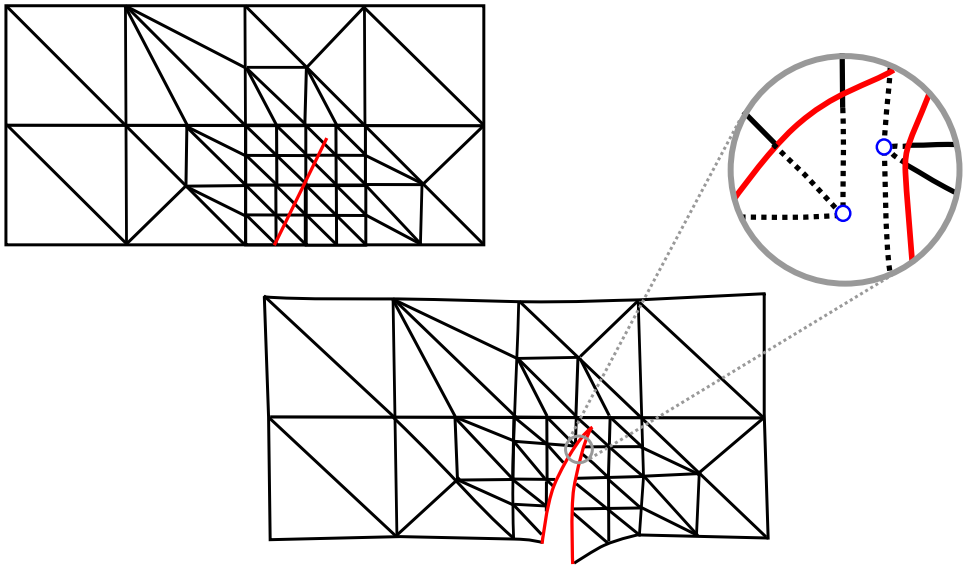

Figure 6. Global cut topology, refined mesh; the mesh at *upper left* is cut, resulting in the geometry at *bottom*; *upper right* shows virtual nodes with a blow up of a region near the crack.

truncated hat functions $\{\phi_i\}$ by setting:

$$\phi_i(\boldsymbol{x}) = \tilde{\phi}_i(\boldsymbol{x}) \sum_{\mathcal{T} \in \omega_i} I_{\mathcal{T}_M}(\boldsymbol{x}), \tag{4}$$

where $I_{\mathcal{T}_M}$ is the characteristic function of the material region for triangle $\mathcal{T}$.

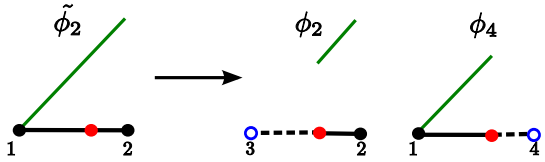

Figure 7. A one-dimensional illustration of our truncated basis functions. At *left*, a mesh element formed by nodes 1 and 2 is depicted with a graph (*in green*) of one of the usual hat functions $\tilde{\phi}_2$; the element is cut (symbolized by the red dot), resulting in two duplicate elements at *right*. The usual hat function $\tilde{\phi}_2$ is then truncated according to the material regions of each copy.

In many cases, this virtual node approach is equivalent to the use of Heaviside functions, i.e., they produce equivalent finite element spaces (see [45]). However, certain crack geometries can produce different finite element spaces. As an example, the configuration in the center of Figure 8 is cut by a crack represented by the red line. The resulting finite element spaces differ, depending on the use of Heaviside enrichment (whose results are pictured in Figure 8, left) or virtual nodes (Figure 8, right). In general, the finite element spaces resulting from the virtual node technique are at least as rich as the spaces resulting from Heaviside enrichment (in terms of the number of degrees of freedom).

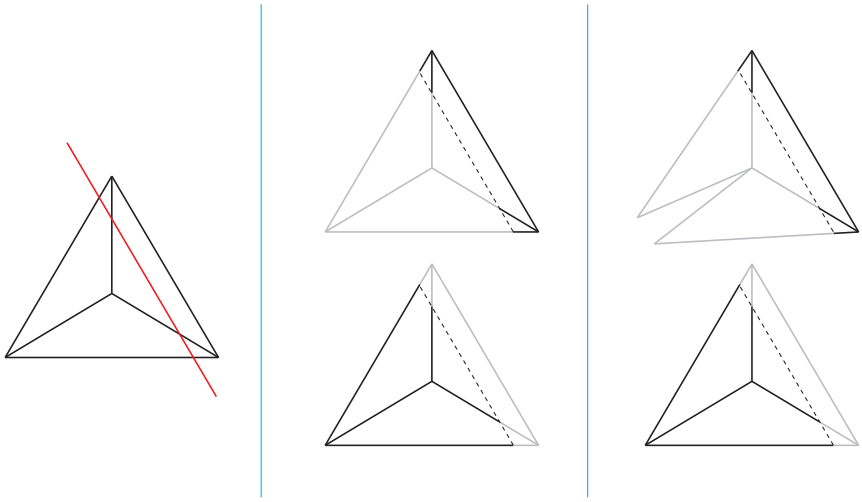

Figure 8. Cutting the mesh at *left* produces different degrees of freedom for Heaviside enrichment (*center*) from virtual nodes (*right*)

3.2. Using the Cutting Algorithm to Mesh Domains

The cutting algorithm described above can also be used to create meshes for domains with geometrically complicated boundaries (see Figure 9). The process is as follows: first, we create

a triangular mesh on a square domain that contains our desired domain. Then, we describe the boundary of our new domain, which can be geometrically quite elaborate, using a closed segmented curve. This curve is provided as input to the cutting algorithm. The result is a mesh that has been cut into disconnected pieces, one piece corresponding to the interior of the domain and the other piece corresponding to the exterior. We then simulate propagation on the mesh that represents the interior of the new domain, by providing the crack at time zero as described by another segmented curve, and then using the crack to cut the domain enterior. We illustrate the use of this technique in the simulation examples of Section 6.4.

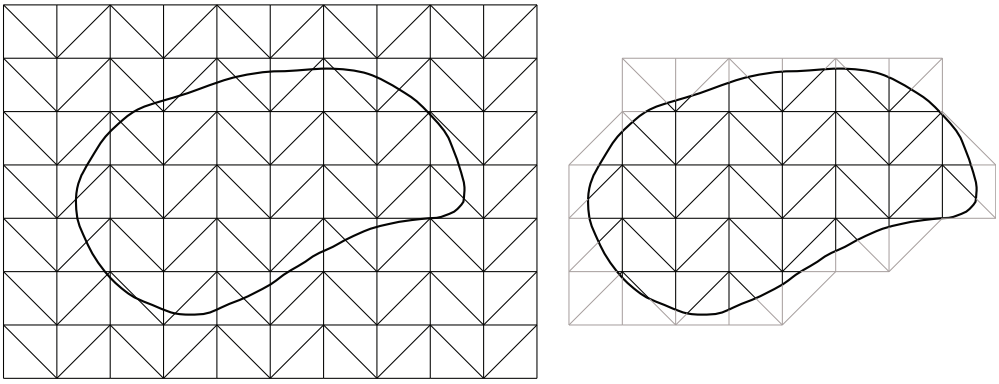

Figure 9. An illustration of using the cutting algorithm to mesh a domain with complex boundary. A square containing the domain is triangulated (shown *at left*), and the cutting algorithm is run with the boundary of the domain as input. This results in two disconnected meshes, one corresponds to the interior of the domain (shown *at right*) and the other is the exterior.

4. INTEGRATION

Our integration scheme uses what we refer to as a hybrid mesh approach, which we describe in this section. In the case of a conforming triangulation, which we assume here for simplicity, we triangulate the domain to create a mesh (see Figure 5, left), and then our process involves further creating the following two triangle meshes:

- The *simulation* mesh, which we create by cutting the original mesh with the crack surface as described in Section 3 (see Figure 5, right). This simulation mesh is used to define the actual degrees of freedom for the system, which include both nodal and crack tip enrichment degrees of freedom.
- The *quadrature* mesh, which we create by first refining the original mesh near the crack and crack tips and then cutting this refined mesh (see Figure 6, bottom). Interpolation for this mesh only uses nodal basis functions, and these nodal positions are fixed by the values of the degrees of freedom of the simulation mesh. We use this mesh only to perform the required integrations, and it does not add additional degrees of freedom to the system.

We solve the equations of equilibrium using the relatively coarse simulation mesh (with fewer degrees of freedom), but perform the required integrations on the finer quadrature mesh, by approximating the (generally nonlinear) basis functions using functions that are piecewise affine on the finer quadrature mesh.

Our quadrature rule is similar to another approach from the XFEM literature (see for instance [31]). This involves performing a Delaunay triangulation of triangles containing the crack, and then assembling the stiffness matrix by using Gauss quadrature on these new

triangles. This method, like our integration rule, uses a finer triangulation of the original mesh to perform the integrations needed for assembly of the stiffness matrix. Also, like our method, this finer triangulation does not add degrees of freedom, the new triangles are only used for integration. However, in our integration technique the cutting algorithm is used to resolve the crack geometry, and so, in contrast to the Delaunay approach, the quadrature mesh does not have to conform to the exact crack geometry. This decouples the resolution of the quadrature mesh from the resolution of the segmented curve used to model the crack. Also, the cutting algorithm can handle complicated crack geometries, which may not be straightforward to re-tessellate with a Delaunay approach.

4.1. Construction of the Simulation Mesh and Quadrature Mesh

We now describe the creation of our hybrid mesh in more detail, and we will use notation consistent with [51]. We first construct the simulation mesh by cutting the original mesh using the techniques of Section 3. Then, using modified hat functions (as in (4)), we define our simulation finite element space $\mathcal{V}_h$ (where h signifies the dependency on the discretization) as those functions $\boldsymbol{u}_h$ with the form:

$$\boldsymbol{u}_h(\boldsymbol{x}) = \sum_i \boldsymbol{u}_i \phi_i(\boldsymbol{x}) + \sum_k \phi_k(\boldsymbol{x}) \left(\sum_{\ell=1}^{4} \boldsymbol{c}_k^\ell F_\ell(r(\boldsymbol{x}), \theta(\boldsymbol{x})) \right), \tag{5}$$

where $\{F_\ell\}$ are the functions given in Section 2.1. This expression is similar to (2), only that the Heaviside enrichment has been replaced with the virtual nodes of the cutting algorithm and the corresponding truncated basis. Let $\mathcal{F}_h$ denote the degrees of freedom of the space $\mathcal{V}_h$, and note that elements of $\mathcal{F}_h$ can be identified with column vectors $\boldsymbol{u} = \left(\boldsymbol{u}_1 \ ... \ \boldsymbol{u}_j \ ... \ \boldsymbol{c}_k^\ell \ ... \ \right)^T \in \mathbb{R}^N$, where N is the total number of degrees of freedom.

Now, we construct the quadrature mesh. We take the original mesh and regularly refine

triangles that are cut by the crack. Away from the crack, this refinement is graded down to the original mesh resolution using a red-green approach (see [52]). We run the cutting algorithm on this refined mesh so that the integration mesh will also account for the crack topology. The result of this procedure is the quadrature mesh (see Figure 6 in Section 3). We then define the quadrature finite element space $\mathcal{V}_h^q$ to be the piecewise affine finite element space that is associated to this quadrature mesh, i.e. we write for $\boldsymbol{u}_h^q \in \mathcal{V}_h^q$

$$\boldsymbol{u}_h^q(\boldsymbol{x}) = \sum_i \boldsymbol{u}_i^q \phi_i^q(\boldsymbol{x}) \tag{6}$$

where $\{\phi_i^q\}$ are the associated nodal basis functions (multiplied by suitable characteristic functions as in (4)). We denote the degrees of freedom of $\mathcal{V}_h^q$ by $\mathcal{F}_h^q$, which is identified with $\mathbb{R}^M$, where M is twice the number of nodes on the quadrature mesh. Note that we do not employ crack tip enrichment in this space, so all the degrees of freedom in $\mathcal{F}_h^q$ are identified with nodes of the quadrature mesh.

We use the quadrature finite element space to integrate the gradients of the basis functions of $\mathcal{V}_h$. This is done by embedding the quadrature mesh onto the simulation mesh, through the establishment of a fixed linear relationship between the degrees of freedom $\mathcal{F}_h^q$ and $\mathcal{F}_h$ (see [50]). Consider $\boldsymbol{u}_i^q \in \mathcal{F}_h^q$ and the position of its corresponding mesh node $\boldsymbol{x}_i^q$. We define the value of $\boldsymbol{u}_i^q$ from the degrees of freedom in $\mathcal{F}_h$ using (5) by:

$$\boldsymbol{u}_i^q = \sum_j \boldsymbol{u}_j \phi_j(\boldsymbol{x}_i^q) + \sum_k \phi_k(\boldsymbol{x}_i^q) \sum_{\ell=1}^{4} \boldsymbol{c}_k^\ell F_\ell(r(\boldsymbol{x}_i^q), \theta(\boldsymbol{x}_i^q)). \tag{7}$$

Note that the sum over j in (7) will involve at most three nonzero terms since $\boldsymbol{x}_i^q$ will be in the support of at most three of the ϕ_j. As seen in (7), the variables $\boldsymbol{u}_i^q$ are functionally constrained to the degrees of freedom in $\mathcal{F}_h$, and do not represent any new degrees of freedom. Also, this defines a linear relationship between the quadrature and simulation degrees of freedom, which

can be represented by a matrix W (see (8)). Binding the quadrature mesh to the simulation mesh results in the following integration scheme: during assembly, we first project the basis functions of $\mathcal{V}_h$ onto the space $\mathcal{V}_h^q$ and then compute the associated matrix using those projected functions. This integration is explained further in the next section.

The quadrature mesh node $\boldsymbol{x}_i^q$ does not necessarily correspond to a material node. It can also be a virtual node created by the cutting algorithm. Special care needs to be taken in computing the polar coordinates $(r(\boldsymbol{x}_i^q), \theta(\boldsymbol{x}_i^q))$ of such virtual nodes. As illustrated in Figure 10, for virtual nodes we reverse the orientation of the angle θ with respect to the crack, essentially associating the virtual nodes with the other side of the crack, where the material for that triangle lies. That way, $\theta(x)$ becomes a continuous map across the triangle, and can take values outside the typical bounds $[-\pi, \pi]$.

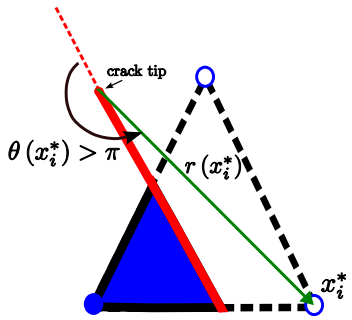

Figure 10. The polar angle of a virtual node in a quadrature triangle is measured accross the crack to ensure continuity of θ inside of the quadrature triangles.

4.2. Integration Scheme

The relation (7) defines the quadrature degrees of freedom in terms of the simulation degrees of freedom. Letting $\mathbf{u} \in \mathcal{F}_h$, then (7) can be written using a matrix W that maps $\mathbf{u}$ to a vector

$\mathbf{u}^q \in \mathcal{F}_h^q$:

$$\mathbf{u}^q = \boldsymbol{W}\mathbf{u}. \tag{8}$$

Since $\mathcal{V}_h^q$ is a piecewise affine finite element space, there is a natural process for assembly of the stiffness matrix in that space, K_h^q. Let a denote the bilinear form associated with the energy (1). Then, our quadrature method approximates the true stiffness matrix of the enriched space used for simulation, K_h, by

$$\begin{aligned} \boldsymbol{u}^T \boldsymbol{K}_h \boldsymbol{u} &= a\left(\boldsymbol{u}_h(\boldsymbol{x}), \boldsymbol{u}_h(\boldsymbol{x})\right) \\ &\approx a\left(\boldsymbol{u}_h^q(\boldsymbol{x}), \boldsymbol{u}_h^q(\boldsymbol{x})\right) \\ &= \left(\boldsymbol{u}_h^q\right)^T \boldsymbol{K}_h^q \boldsymbol{u}_h^q, \\ &= \boldsymbol{u}^T \boldsymbol{W}^T \boldsymbol{K}_h^q \boldsymbol{W} \boldsymbol{u}, \end{aligned} \tag{9}$$

where $\boldsymbol{u}_h$ is the function in $\mathcal{V}_h$ corresponding to the vector $\boldsymbol{u} \in \mathcal{F}_h$ and $\boldsymbol{u}_h^q$ is the function in $\mathcal{V}_h^q$ corresponding to the vector $\boldsymbol{u}^q \in \mathcal{F}_h^q$. This means that

$$\boldsymbol{K}_h \approx \boldsymbol{W}^T \boldsymbol{K}_h^q \boldsymbol{W}.$$

Thus our integration scheme will be integrating only an approximation of the non-smooth basis functions (see Figure 11), but this approximation improves as we refine the embedded quadrature mesh. Note that our sampling of the singular basis functions could also occur at points near the singularity. However, unlike integration schemes based on Gauss quadrature or Monte Carlo methods, those evaluations will get additionally weighted by the area of the smaller quadrature triangle. Thus, no single value, possibly located near the singularity at the tip, contributes disproportionally. Finally, we note that for simplicity and improved stability we further approximate the integrations in (9) by treating cut quadrature triangles as if they were full of material, i.e., we remove the multiplication by the characteristic functions introduced in

(4) and assemble K_h over the usual nodal "hat" functions. Note that this optional modification naturally vanishes under refinement of the quadrature mesh, as the error in the support of the integrated shape functions goes to zero.

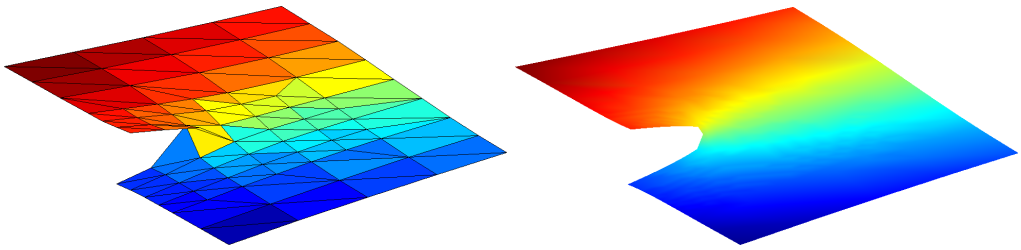

Figure 11. Tow samplings of the enrichment function $F_1 = \sqrt{r}\sin(\theta/2)$: on the *left*, a sampling on a low resolution quadrature mesh, on the *right*, sampling on a high resolution mesh.

5. CRACK PROPAGATION

For a fixed state of the system, i.e., a given crack and an equilibrium displacement for that crack and boundary conditions, there are several different criteria used by engineers to determine the angle at which the crack will propagate. We follow [6] in using the maximum circumferential stress criterion to compute the propagation direction and then move the crack by a small fixed increment. We chose this approach so that the results of using our integration technique can be compared against the tests cases in [6]. This method is fairly standard and the details are found in the references, so we only sketch it here.

The criterion involves computing the stress intensity factors at the crack tip, and then calculating the angle of maximal stress by the relation:

$$\theta_c = 2 \arctan \left(\frac{1}{4} \left[\frac{K_{\mathrm{I}}}{K_{\mathrm{II}}} \pm \sqrt{\left(\frac{K_{\mathrm{I}}}{K_{\mathrm{II}}} \right)^2 + 8} \right] \right). \tag{10}$$

To compute the stress intensity factors, we use the so called interaction J-integral, which is defined for two possible states of the system, whose variables we denote using superscript 1 and 2, by:

$$I^{(1,2)} := \int_\Gamma \left(W^{(1,2)} \delta_{1j} - \left[\sigma_{ij}^{(1)} \frac{\partial u_i^{(2)}}{\partial x_1} + \sigma_{ij}^{(2)} \frac{\partial u_i^{(1)}}{\partial x_1} \right] \right) n_j ds \tag{11}$$

where

$$W^{(1,2)} := \sigma_{ij}^{(1)} \epsilon_{ij}^{(2)}. \tag{12}$$

Choosing the two states to be the current state and a pure Mode I state in the above gives K_{I}:

$$K_{\mathrm{I}} = \frac{E^*}{2} I^{(\mathrm{Curr,Mode\ I})}, \tag{13}$$

where

$$E^* = \begin{cases} \dfrac{E}{1-\nu^2} & \text{plane strain} \\ E & \text{plane stress,} \end{cases} \tag{14}$$

and E is Young's modulus and ν is Poisson's ratio. K_{II} is found similarly. As in [6], to actually compute these interaction integrals requires converting them into area intregrals via multiplication by a suitably smooth test function and applying integration by parts.

The resulting area integral is then computed on the embedded quadrature mesh described in Section 4. Having used finite elements to compute the displacement, the stresses and strains for the current state of the system are piecewise affine on the embedded mesh. We then interpolate the displacements, strains, and stresses for the pure Mode I and pure Mode II solutions used to compute (11) using functions in the space $\mathcal{M}_h$. The required integrations are then simple to compute, since all the quantities in (11) are piecewise affine. The only challenge is to choose an integration area that encompasses enough of the triangles of the simulation mesh to achieve accuracy, but not so large as to introduce more error brought on by the finite domain size. Our experiments show that integrating over the one-ring or two-ring of the simulation triangle that contains the crack tip provides good results (the results of Section 6 use a two-ring). Note that by one-ring of a triangle T we mean the set of all triangles that share a node with T, and by two-ring of a triangle T we mean the set of all triangles that are either in the one-ring of T or share a node with a triangle in the one-ring of T.

6. NUMERICAL EXPERIMENTS

We tested our approach with some examples from the literature (see [6], [4]). First, we chose these examples because the exact stress intensity factors (or good approximations) can be calculated analytically for comparision, and also we can compare our results to the literature.

6.1. Example 1: Straight Crack with Pure Mode I Displacement

The first example involves a straight center crack in a rectangular body with a constant traction applied to part of the boundary of the body, as diagrammed in Figure 12 (as in [6] we use $L = 16$, $W = 7$, $a = 3.5$, $\epsilon = 100$[kpsi] and $\nu = .3$).

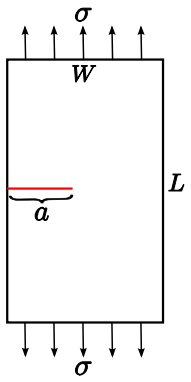

Figure 12. The setup for the first numerical experiment.

In this case, the exact Mode I stress intensity factor is given by

$$K_{\mathrm{I}} = C\sigma\sqrt{a\pi},$$

where C is the finite geometry correction factor:

$$C = 1.12 - 0.231\left(\frac{a}{W}\right) + 10.55\left(\frac{a}{W}\right)^2 - 21.72\left(\frac{a}{W}\right)^3 + 30.39\left(\frac{a}{W}\right)^4.$$

We normalize K_{I} through an appropriate choice of σ, and compare our results over various combinations of granularity of the simulation mesh and refinement levels of the quadrature mesh. The results of this study are found in the table of Figure 13: each column represents a different resolution for the simulation mesh, and data is presented for different levels of regular refinement near the crack ($0 - 5$ levels). Note that the computed intensity factors improve as the simulation mesh is refined, as expected, but also we are able to get good results for coarser simulation meshes by refining the quadrature mesh (of course up to a limit that is determined by the simulation resolution). Also, increasing the refinement levels of the quadrature mesh roughly matches the results for increases in resolution of the simulation mesh, and that accuracy is acheived at less cost since refining the quadrature mesh does not add new degrees of freedom to the system.

K_{I}

Levels	8x4	16x8	32x16	64x32	128x64
0	1.54778000	1.11393000	1.00303000	1.01901000	0.98573100
1	0.89678100	0.91319400	0.97447200	1.01661000	0.98443400
2	0.82495400	0.91514000	0.95464700	1.00335000	0.97799900
3	0.84035000	0.90665100	0.96431200	0.98717700	0.96803500
4	0.83361600	0.91260700	0.95926000	0.98667700	0.96945600
5	0.84292500	0.90972400	0.96276500	0.98118700	0.96666500

K_{II}

Levels	8x4	16x8	32x16	64x32	128x64
0	0.18590300	0.09165160	0.10725100	0.05913580	0.01374570
1	0.04005210	0.03758170	0.03205980	0.00391652	0.00846525
2	0.02733560	0.01045710	0.00299528	0.00849926	0.00699155
3	0.01483750	-0.00186129	0.00743422	-0.00470591	0.00537171
4	0.00600684	-0.00094331	0.00796979	-0.00481388	0.00312401
5	0.01097960	-0.00105607	0.00809645	-0.00851569	-0.00154344

Figure 13. Results for first numerical example; analytical result is $K_{\mathrm{I}} = 1$, $K_{\mathrm{II}} = 0$. Note that increasing refinements in the quadrature mesh roughly match results for increasing resolution of the base mesh.

6.2. Example 2: Straight Crack with Constant Shear Displacement

The second example involved the same geometry as the first example, namely a straight edge crack, but we applied a zero displacement condition to one end and a constant shear (with

respect to the crack frame) to the other end (see the diagram in Figure 14). In this case, the stress intensity factors are known (see [6]): $K_{\mathrm{I}} = 34.0[\mathrm{psi}\sqrt{\mathrm{in}}]$ and $K_{\mathrm{II}} = 4.55[\mathrm{psi}\sqrt{\mathrm{in}}]$. Again, we compared our results when varying the refinement of both the simulation and quadrature meshes, with results summarized in the table of Figure 15.

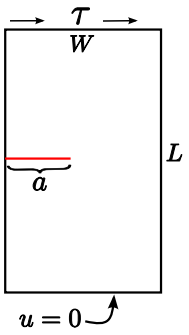

Figure 14. The setup for the second numerical experiment.

K_{I}

Levels	16x8	32x16	64x32	128x64
0	65.41790000	53.60800000	45.35520000	40.20960000
1	34.11440000	35.16530000	35.01640000	34.56440000
2	28.34310000	31.69620000	33.10160000	33.86180000
3	27.20410000	31.08820000	32.96800000	33.75550000
4	27.18120000	31.17290000	32.94070000	33.86550000
5	27.32760000	31.15840000	33.01980000	33.87260000

K_{II}

Levels	16x8	32x16	64x32	128x64
0	35.32480000	24.00590000	16.58030000	11.85800000
1	13.27730000	10.71430000	8.77887000	7.33766000
2	7.56987000	6.70199000	6.00861000	5.46512000
3	5.88499000	5.54668000	5.27518000	5.02933000
4	5.42942000	5.24631000	5.10806000	5.02421000
5	5.29152000	5.14862000	5.12151000	5.02322000

Figure 15. Results for second numerical example; analytical result is $K_{\mathrm{I}} = 34.0$, $K_{\mathrm{II}} = 4.55$. Note that most of the benefits of refinement of the quadrature mesh are realized after only two to three levels of refinement (infinite refinement would correspond to exact quadrature during assembly).

6.3. Angled Center Crack with Mixed Mode Displacement

Following the example in Section 4.3 of [6], we compute the stress intensity factors for a plate with an angled center crack and subjected to a far field constant traction, as pictured in Figure 16. The dimensions of the square plate are taken to be $W = 10$[in] and the crack length is set by $a = .5$[in]. Since the crack size is small compared to the dimensions of the plate, the stress intensity factors can be approximated by the intensity factors corresponding to the solution in the entire plane, which are given by

$$K_{\mathrm{I}} = \sigma\sqrt{\pi a}\cos^2(\beta),$$

$$K_{\mathrm{II}} = \sigma\sqrt{\pi a}\sin(\beta)\cos(\beta).$$

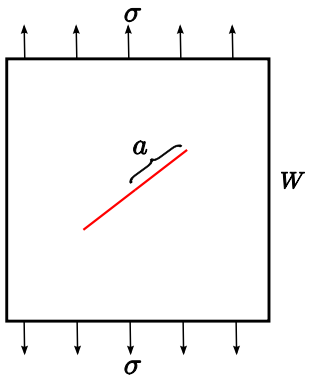

Figure 16. The setup for the angled center crack numerical experiment.

We computed the stress intensity factors as β ranges from 0 to $\pi/2$ in increments of $\pi/20$. We used a simulation mesh with resolution of 64x64 elements, and varied the levels of refinement for the quadrature mesh. We illustrate in Figure 17 the results of a comparison between the "exact" K_{I} and K_{II} for the various values of β (plotted using blue squares and orange diamonds, respectively) and the computed values (yellow triangles for computed K_{I} and green triangles for

K_{II}). On the left of the figure, we plot the results for one level of refinement in the quadrature mesh; on the right of the figure we plot the results for five levels of refinement. While one level of refinement gives decent agreement with the exact values, as we refine the quadrature mesh the agreement with the exact values becomes much stronger (and is comparable to the results of [6]).

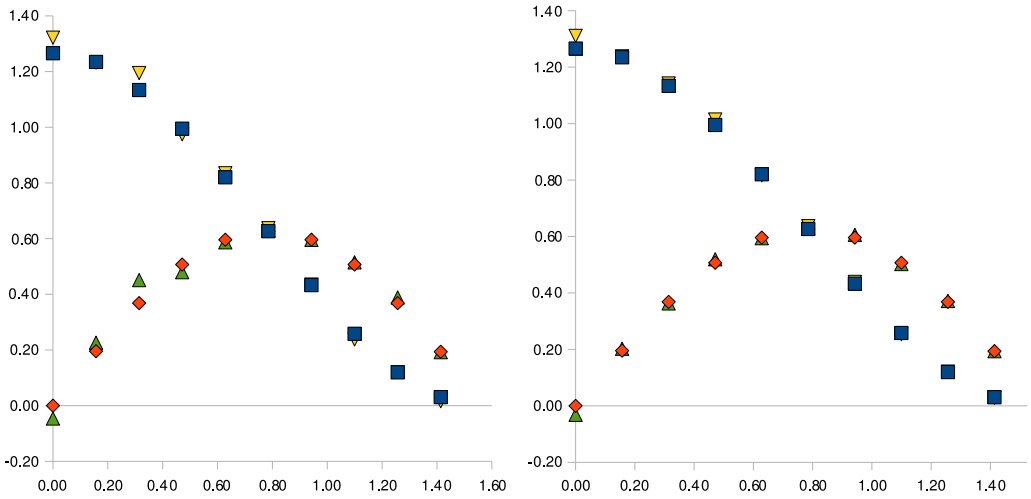

Figure 17. Results for angled crack example: Exact K_{I} (blue square) and K_{II} (orange diamond) as compared to computed values (yellow and green triangles, respectively). The graph at *left* shows results for 1 level of refinement on quadrature mesh; the graph at *right* shows results for 5 levels.

6.4. Propagation Examples

In Figures 18, 20, 21, and 22 we show results of using our method for simulating the propagation of cracks. In the first example, shown in Figure 18, we simulate a rectangular domain that has been initialized with a straight crack. We apply symmetric displacement boundary conditions

to the right and left sides of the domain, and the result is a crack that moves straight, and in the end cuts the domain into two disconnected regions. The colors of the diagrams in Figure 18 represent the Frobenius norm of the stress, red denotes relatively large values and blue denotes small values (we also use this color convention for the remaining figures of this section).

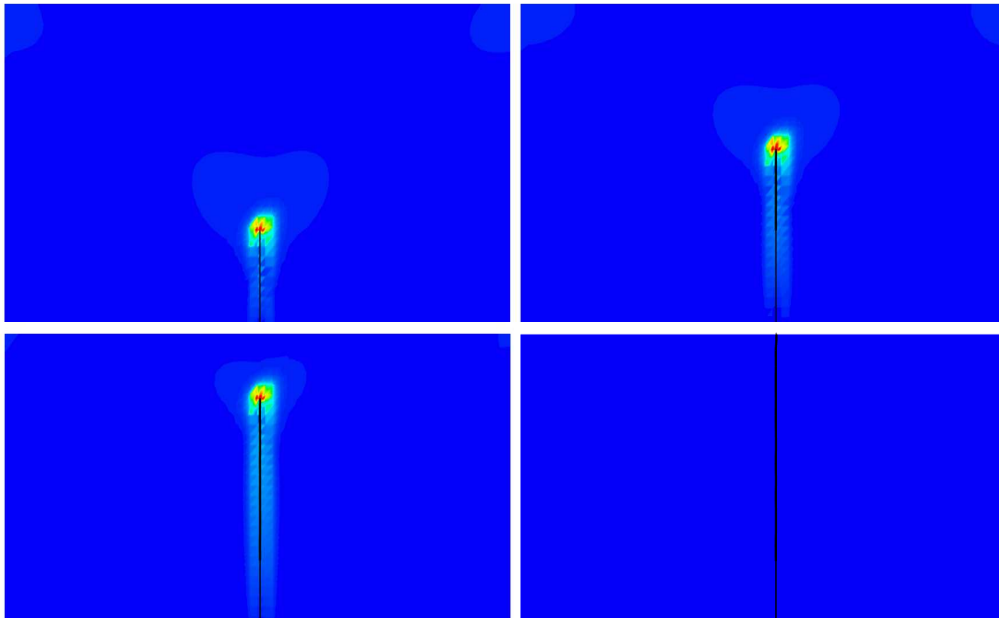

Figure 18. Simulation of a rectangular domain with symmetric boundary displacements; from *upper left*: initial configuration, 5 timesteps, 10 timesteps, 15 timesteps.

Our next propagation example involves the quasistatic propagation of a crack in a beam, as pictured in the diagram of Figure 19. As in [4], we varied the inital perturbation angle θ, using values 1.43, 2.86, and 5.71 (all degrees). The results of simulating the propagation for these three angles is presented in Figure 20, where we have plotted the position of the crack tip at each timestep using blue triangles for 1.43 degrees, orange squares for 2.86, and yellow

triangles for 5.71. Our results are in good agreement with the results in [4].

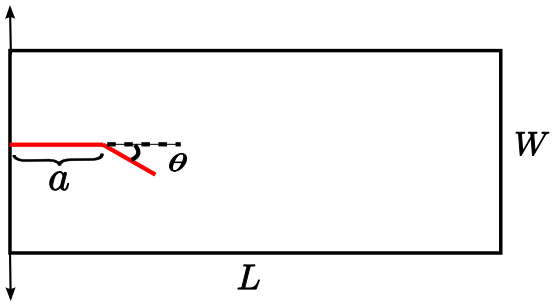

Figure 19. The setup used to show propagation beam for various values of the initial perturbation angle θ.

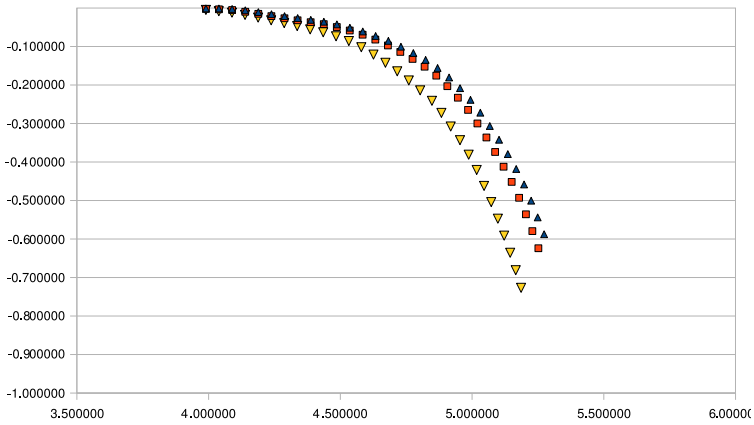

Figure 20. The results of cantilever beam propagation; the position of the crack tip at each timestep is plotted using blue triangles for $\theta = 1.43$ degrees, orange squares for $\theta = 2.86$, and yellow triangles for $\theta = 5.71$.

In Figure 21, we illustrate a more complicated scenario. Our initial setup, pictured in the upper left corner of the figure, is a square that has two holes of the same (small) radius. The holes of the domain are created by employing the cutting algorithm described in Section 3.2,

and we remove the duplicated triangles that are associated with the holes. The simulation is initialized with two cracks, one emerging from each hole, at the same angle with respect to the horizontal, so that the resulting geometry is symmetric (see the zoom-in on the holes showing the cracks, at *upper right*). We subject this domain to constant traction at the right and left side of the domain. The bottom row of the figure shows the results of the crack simultation after twenty timesteps (note that for clarity we removed the disconnected material region in between the two cracks at the final timestep).

In Figure 22 we further complicate the geometry of the domain and cracks. As in the previous example, we constructed a domain with more complicated geometry by cutting a rectangular domain with the cutting algorithm, as described in Section 3.2 (this new domain is the reference configuration of our material body). Then, we create initial cracks in the domain that have junctions, in order to illustrate the geometric flexibility of our algorithm. Note that, in this case, some of the elements will be duplicated into more than two copies, since the cracks will cut triangles near the junction into three materially distinct elements. We subject this new domain to a dirichlet condition at the left and right ends, and also we apply traction conditions to five other parts of the boundary (in the diagram they can be seen by the higher stresses that they produce). We then simulated the progagation of the cracks over twenty timesteps. Note that as the cracks evolve they can join with other cracks, which we accomplish by procedurally merging cracks whose paths intersect.

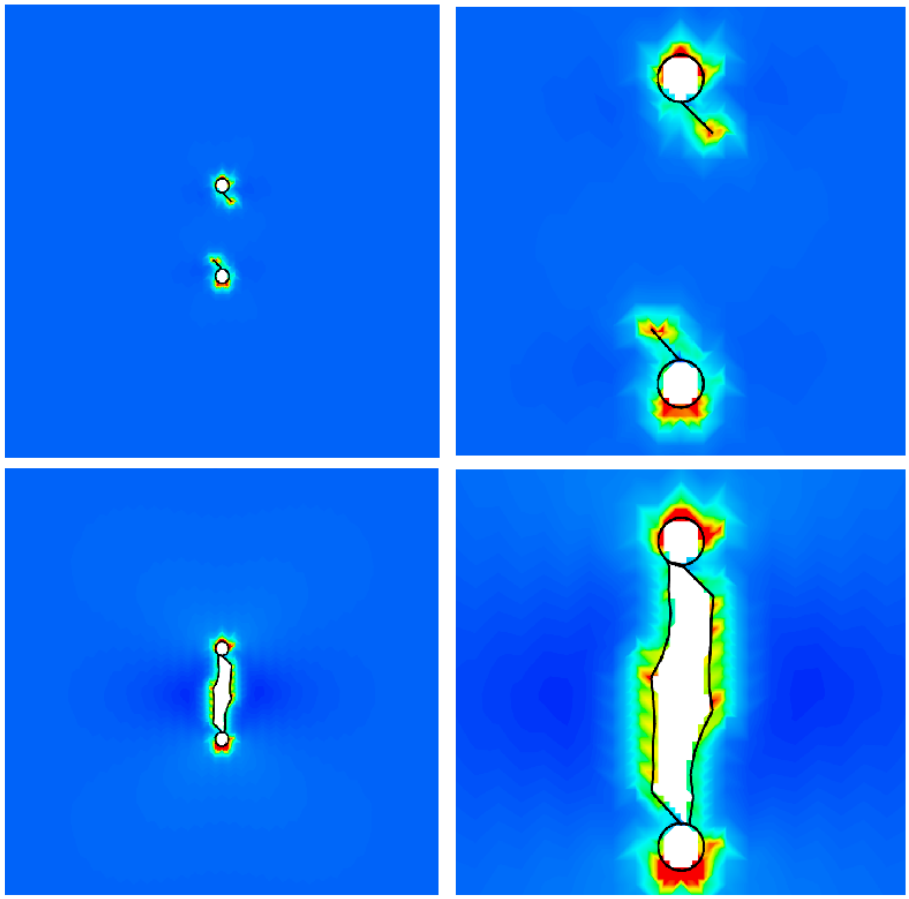

Figure 21. Cracks propagating in a domain with holes; *top row*: initial configuration on *left* and a zoom into the holes on *right*; *bottom row*: analagous diagrams for result after 20 timesteps.

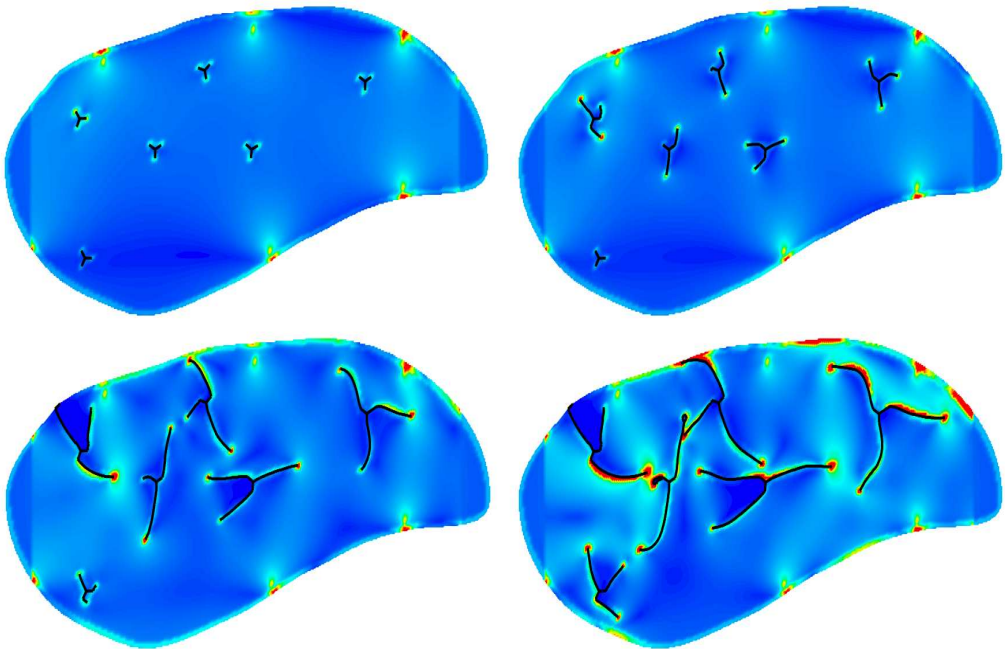

Figure 22. Simulation with complex geometry; from *upper left*: initial configuration, 5 timesteps, 15 timesteps, 25 timesteps.

7. CONCLUDING REMARKS

We presented an XFEM-based method for the simulation of crack propagation. This method uses virtual nodes generated by the cutting algorithm of Sifakis, et.al., [49] to create the extra degrees of freedom that allow the crack to open, in a way that is general and flexible. Our technique gives accurate stress intensity factors, which we use to propagate the crack. Our our discretization and simulation approach can accomodate complex crack and domain geometry. We illustrated the accuracy of our method by comparison with results from the literature, and showed the geometric flexibity with propagation examples in complicated domains.

ACKNOWLEDGEMENTS

This work was partially supported by UC Lab Fees Research Grant 09-LR-04-116741-BERA and Office of Naval Research grant N000140310071. CR was partially supported by the National Science Foundation under grant No. DMS-0714945.

REFERENCES

1. Belytschko T, Gracie R, Ventura G. A review of extended/generalized finite element methods for material modeling. *Modelling and Simulation in Materials Science and Engineering* 2009; **17**(4):043 001.
2. Karihaloo BL, Xiao QZ. Modelling of stationary and growing cracks in fe framework without remeshing: a state-of-the-art review. *Computers and Structures* 2003; **81**(3):119 – 129.
3. Abdelaziz Y, Hamouine A. A survey of the extended finite element. *Computers and Structures* 2008; **86**(11-12):1141 – 1151.
4. Belytschko T, Black T. Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering* 1999; **45**:601–620.
5. Melenk JM, Babuška I. The partition of unity finite element method: Basic theory and applications. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**(1-4):289 – 314.
6. Moes N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering* 1999; **46(1)**:131–150.
7. Sukumar N, Mos N, Moran B, Belytschko T. Extended finite element method for three-dimensional crack modelling. *International Journal for Numerical Methods in Engineering* 2000; **48**:1549–1570.
8. Areias P, Belytschko T. Analysis of three-dimensional crack initiation and propagation using the extended finite element method. *International Journal for Numerical Methods in Engineering* 2005; **63**:760–788.
9. Moes N, Belytschko T. Extended finite element method for cohesive crack growth. *Engineering Fracture Mechanics* 2002; **69(7)**:813–833.
10. Zi G, Belytschko T. New crack-tip elements for xfem and applications to cohesive cracks. *International Journal for Numerical Methods in Engineering* 2003; **57(15)**:2221–2240.
11. de Borst R, Gutirrez M, Wells G, Remmers J, Askes H. Cohesive-zone models, higher-order continuum theories and reliability methods for computational failure analysis. *International Journal for Numerical Methods in Engineering* 2004; **60**:289–315.
12. de Borst R, Remmers JJ, Needleman A. Mesh-independent discrete numerical representations of cohesive-zone models. *Engineering Fracture Mechanics* 2006; **73**(2):160 – 177.
13. Mariani S, Perego U. Extended finite element method for quasi-brittle fracture. *International Journal for Numerical Methods in Engineering* 2003; **58**:103–126.
14. Asferg J, Poulsen P, Nielsen L. A consistent partly cracked xfem element for cohesive crack growth. *International Journal for Numerical Methods in Engineering* 2007; **72**:464–485.

15. Belytschko T, Chen H. Singular enrichment finite element method for elastodynamic crack propagation. *International Journal of Computational Methods, 1(1): 1-15* 2004; **1(1)**:1–15.

16. Huynh D, Belytschko T. The extended finite element method for fracture in composite materials. *International Journal for Numerical Methods in Engineering* 2009; **77(2)**:214–239.

17. Osher S, Sethian JA. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics* 1988; **79**(1):12 – 49.

18. Osher S, Fedkiw R. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.

19. Belytschko T, Moes N, Usui S, Parimi C. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering* 2001; **50(4)**:993–1013.

20. Moes N, Gravouil A, Belytschko T. Non-planar 3d crack growth by the extended finite element and level sets part i: Mechanical model. *International Journal for Numerical Methods in Engineering* 2002; **53(11)**:2549–2568.

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

22. Duflot M. A study of the representation of cracks with level sets. *International Journal for Numerical Methods in Engineering* 2007; **70(11)**:1261–1302.

23. Prabel B, Combescure A, Gravouil A, Marie S. Level set x-fem non-matching meshes: application to dynamic crack propagation in elastic-plastic media. *International Journal for Numerical Methods in Engineering* 2007; **69(8)**:1553–1569.

24. Sukumar N, Chopp DL, Mos N, Belytschko T. Modeling holes and inclusions by level sets in the extended finite-element method. *Computer Methods in Applied Mechanics and Engineering* 2001; **190**(46-47):6183 – 6200.

25. Sukumar N, Chopp D, Bechet E, Moes N. Three-dimensional non-planar crack growth by a coupled extended finite element and fast marching method. *International Journal for Numerical Methods in Engineering* 2008; **76(5)**:727–748.

26. Bordas S, Duflot M, Le P. A simple error estimator for extended finite elements. *Communications in Numerical Methods in Engineering* 2008; **24(11)**:961–971.

27. Chahine E, Laborde P, Renard Y. Crack tip enrichment in the xfem using a cutoff function. *International Journal for Numerical Methods in Engineering* 2008; **75(6)**:626–646.

28. Chahine E, Laborde P, Renard Y. A quasi-optimal convergence result for fracture mechanics with xfem.

Comptes Rendus Mathematique 2006; **342**(7):527 – 532.

29. Lew A, Shen Y. Crack tip enrichment in the xfem using a cutoff function. *International Journal for Numerical Methods in Engineering* 2009; .
30. Daux C, Moes N, Dolbow J, Sukumar N, Belytschko T. Arbitrary branched and intersecting cracks with the extended finite element method. *International Journal for Numerical Methods in Engineering* 2000; **48**:1741–1760.
31. Stazi F, Budyn E, Chessa J, Belytschko T. An extended finite element method with higher-order elements for curved cracks. *Computational Mechanics* 2003; **31**:38–48.
32. Bchet E, Minnebo H, Mos N, Burgardt B. Improved implementation and robustness study of the x-fem for stress analysis around cracks. *International Journal for Numerical Methods in Engineering* 2005; **64(8)**:1033–1056.
33. Laborde P, Pommier J, Renard Y, Salaun M. High order extended finite element method for cracked domains. *International Journal for Numerical Methods in Engineering* 2005; **64**:354–381.
34. Strouboulis T, Babuška I, Copps K. The design and analysis of the generalized finite element method. *Computer Methods in Applied Mechanics and Engineering* 2000; **181**:43–69.
35. Ventura G, Gracie R, Belytschko T. Fast integration and weight function blending in the extended finite element method. *International Journal for Numerical Methods in Engineering* 2009; **77(1)**:1–29.
36. Park K, Pereira J, Duarte C, Paulino G. Integration of singular enrichment functions in the generalized/extended finite element method for three-dimensional problems. *International Journal for Numerical Methods in Engineering* 2009; **78(10)**:1220–1257.
37. Ventura G. On the elimination of quadrature subcells for discontinuous functions in the extended finite-element method. *International Journal for Numerical Methods in Engineering* 2006; **66**:761–795.
38. Holdych D, Noble D, Secor R. Quadrature rules for triangular and tetrahedral elements with generalized functions. *International Journal for Numerical Methods in Engineering* 2008; **73(9)**:1310–1327.
39. Benvenuti E, Tralli A, Ventura G. A regularized xfem model for the transition from continuous to discontinuous displacements. *International Journal for Numerical Methods in Engineering* 2008; **74(6)**:911–944.
40. Budyn E, Zi G, Moes N, Belytschko T. A method for multiple crack growth in brittle materials without remeshing. *International Journal for Numerical Methods in Engineering* 2004; **61**:1741–1770.
41. Zi G, Song J, Budyn E, Lee S, Belytschko T. A method for growing multiple cracks without remeshing and its application to fatigue crack growth. *Modelling Simul. Mater. Sci. Eng.* 2004; **12**:901–915.

42. Song J, Belytschko T. Cracking node method for dynamic fracture with finite elements. *International Journal for Numerical Methods in Engineering* 2009; **77(3)**:360–385.

43. Molino N, Bao Z, Fedkiw R. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. Graph. (SIGGRAPH Proc.)* 2004; **23**:285–392.

44. Hansbo A, Hansbo P. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**(33-35):3523 – 3540.

45. Song J, Areias P, Belytschko T. A method for dynamic crack and shear band propagation with phantom nodes. *International Journal for Numerical Methods in Engineering* 2006; **67(6)**:868–893.

46. Duan Q, Song J, Menouillard T, Belytschko T. Element-local level set method for three-dimensional dynamic crack growth. *International Journal for Numerical Methods in Engineering* 2009; **80(12)**:1520–1543.

47. Areias PM, Belytschko T. A comment on the article 'a finite element method for simulation of strong and weak discontinuities in solid mechanics' by a. hansbo and p. hansbo [comput. methods appl. mech. engrg. 193 (2004) 3523-3540]. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(9-12):1275 – 1276.

48. Dolbow J, Harari I. An efficient finite element method for embedded interface problems. *International Journal for Numerical Methods in Engineering* 2009; **78(2)**:229–252.

49. Sifakis E, Der K, Fedkiw R. Arbitrary cutting of deformable tetrahedralized objects. *Proc. of ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.*, 2007; 73–80.

50. Sifakis E, Shinar T, Irving G, Fedkiw R. Hybrid simulation of deformable solids. *Proc. of ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.*, 2007; 81–90.

51. Braess D. *Finite elements*. Third edn., Cambridge University Press: Cambridge, 2007.

52. Molino N, Bridson R, Teran J, Fedkiw R. A crystalline, red green strategy for meshing highly deformable objects with tetrahedra. *12th Int. Meshing Roundtable*, 2003; 103–114.