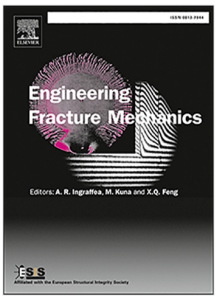
Journal Pre-proof



An X-FEM based framework for 3D fatigue crack growth using a B-spline crack geometry description

Kris Hectors, Wim De Waele

PII: S0013-7944(22)00005-4
DOI: https://doi.org/10.1016/j.engfracmech.2022.108238
Reference: EFM 108238

To appear in: *Engineering Fracture Mechanics*

Received date : 29 June 2021
Revised date : 19 October 2021
Accepted date : 3 January 2022

Please cite this article as: K. Hectors and W.D. Waele, An X-FEM based framework for 3D fatigue crack growth using a B-spline crack geometry description. *Engineering Fracture Mechanics* (2022), doi: https://doi.org/10.1016/j.engfracmech.2022.108238.

An X-FEM based framework for 3D fatigue crack growth using a B-spline crack geometry description

Kris Hectors[a,b], Wim De Waele[b]

[a] *SIM vzw, Tech Lane Ghent Science, Park / Campus A 48, Zwijnaarde, 9052, Belgium*
[b] *OWI-Lab, Ghent University, Faculty of Engineering and Architecture, Department of Electromechanical, Systems and Metal Engineering, Laboratory Soete, Technologiepark 46, Zwijnaarde, 9052, Belgium*

Abstract

This paper presents an X-FEM based iterative framework for numerical simulation of three-dimensional fatigue crack propagation. To accurately describe the crack geometry, B-spline curves and surfaces are used for the crack front and faces respectively. A new crack front extension method based on the Möller-Trumbore algorithm is introduced to achieve fully automated crack growth for complex geometries. To validate the presented work, three different cases are discussed: (1) compact tension specimens with an asymmetrically located hole, (2) a beam with a slanted crack subjected to a three-point bending load and (3) a beam with a slanted crack subjected to a torsion load. Good agreement between the numerical and experimental results is observed.

Keywords: Fatigue, Fracture simulation, Fatigue crack growth, X-FEM, B-spline

1. Introduction

The majority of failures in engineering structures and mechanical components can be attributed to fatigue. Fatigue cracks mostly initiate at local stress raising features, e.g. cope holes or welds. In large structural components, the initiation of a crack does not necessarily imply that the structural integrity is compromised. Significant fatigue crack propagation life may remain [1]. The fatigue life is typically said to be exhausted when the crack reaches a critical size [2]. To estimate the remaining fatigue life after fatigue crack initiation, there is a need for reliable fatigue crack growth (FCG) simulation tools. Some commercial tools such as AFGROW [3] and NASGRO [4] can be used to simulate the FCG of a predefined planar crack based on analytical solutions for the stress intensity factors (SIF) under uni-axial loading [5]. However, industrial structures are mostly subjected to multi-axial loading conditions that result in mixed mode (i.e. non-planar) FCG.

The most widely used computational method for calculation of stress intensity factors in cracked geometries is the finite element method (e.g [6, 7]). In the conventional finite element method (FEM), cracks are modeled as geometrical discontinuities with the crack faces explicitly defined as contact surfaces [8]. A number of commercial software packages for 3D FCG based on conventional FEM are Zencrack [9], FRANC3D [10] and ADAPCRACK3D [11]. Although they have been successfully applied for simulation of mixed mode FCG in complex geometries (e.g. [12, 13, 14]) there are drawbacks to the use of conventional FEM for simulation of FCG. A major drawback is that the FEM mesh needs to conform to the geometry of the crack. Therefore

it is required to update the mesh for each crack increment each time the crack propagates, which is challenging [15, 16]. For an extensive overview of the state-of-the-art on simulation of FCG using conventional FEM the reader is referred to [17].

The extended finite element method (X-FEM) by Belytschko and co-workers was introduced to overcome the need for remeshing when modeling cracks and crack propagation [18, 19]. X-FEM allows modeling of an arbitrary crack path in the mesh through the enrichment of the displacement field based on the partition of unity. The nodes of elements that are completely cut through by a crack have shape functions that are multiplied with the Heaviside function $H(\mathbf{x})$ that represents the gap between the crack surfaces. Elements that are only partially cut, i.e. the crack tip elements, are enriched with the asymptotic crack tip functions that reproduce the asymptotic LEFM fields [18]

$$F_\alpha(r,\theta) = \left[\sqrt{r}\sin\frac{\theta}{2},\ \sqrt{r}\cos\frac{\theta}{2},\ \sqrt{r}\sin\frac{\theta}{2}\sin\theta,\ \sqrt{r}\cos\frac{\theta}{2}\sin\theta \right] \tag{1}$$

where r, θ are local polar coordinates defined at the crack tip. The displacement approximation in X-FEM takes the form [19]

$$\mathbf{u}(\mathbf{x}) = \sum_{i\in S} N_i(\mathbf{x})\mathbf{u}_i + \sum_{i\in S_H} N_i(\mathbf{x})H(\mathbf{x})\mathbf{a}_i + \sum_{i\in S_T}\left[N_i(\mathbf{x}) \sum_{\alpha=1}^{4} F_\alpha(\mathbf{x})\mathbf{b}_{i\alpha} \right] \tag{2}$$

where $\mathbf{u}$ is the nodal displacement vector, S is the set of all nodes in the finite element space, S_H is the set of nodes enriched with the Heaviside function $H(\mathbf{x})$, S_T is the set of nodes enriched with the crack tip functions $F_\alpha(\mathbf{x})$, $\mathbf{a}_i$ and $\mathbf{b}_{i\alpha}$ are the additional nodal DOF's corresponding to the respective enrichment functions. The DOF's $\mathbf{a}_i$ and $\mathbf{b}_{i\alpha}$ are mutually exclusive, the discontinuity behind the crack tip is accounted for by $F_1(r,\theta) = \sqrt{r}\sin\frac{\theta}{2}$ which is discontinuous at $\theta = \pm\pi$.

Combination of X-FEM with the level set method removes the need for remeshing because the crack is implicitly represented. Two level sets, for which signed distance functions are used, define the crack. One level set describes the crack surface and the intersection between the first and second level set defines the crack front. Figure 1 illustrates the representation of an arbitrary nonplanar crack in three dimensions by two signed distance functions ϕ and ψ. With the level set method it is possible to describe the crack entirely with nodal data and as such it is used to determine which nodes have to be enriched and in what way [20, 21].

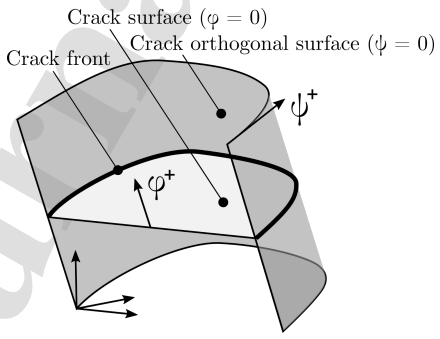


Figure 1: Representation of an arbitrary nonplanar crack in three dimensions by two signed distance functions ϕ and ψ.

The X-FEM method has been implemented in several commercial finite element codes but most do not directly support FCG analysis based on the theory of linear elastic fracture mechanics (LEFM). Independent researchers and institutions have developed plug-ins for commercial finite element codes. Examples are Morfeo/crack and XFA3D [22] which implemented 3D non-planar fatigue crack growth based on the X-FEM method. These tools are based on an implicit description of the crack using the level set method. Fries and Baydoun [23] noted that for a methodology using purely implicit crack description, updating the level set functions is not simple and may introduce inaccuracies. An example is the occurrence of erroneously created crack surfaces because the level set functions are ill-controlled far away from the true crack surface. Although this can be improved with the use of narrow-band techniques, as described by Shi *et al.* [22], it cannot be eliminated. An example of this can be seen in the work of Sadeghirad *et al.* [24]. A hybrid approach that combines implicit and explicit approaches is an effective way to address the shortcomings of the purely implicit approach. In the hybrid implicit/explicit approach the level set method (implicit) that is used in the X-FEM step is combined with an explicit description of the crack to determine the crack increment and to update the crack representation. The hybrid implicit/explicit approach was first introduced by Fries and Baydoun [25]. For the explicit representation of the crack in 3D they used flat triangles. After each X-FEM calculation step, the crack surface is extended at each crack front node based on the corresponding stress state and as such the explicit crack representation is updated. The level sets are then updated based on the new explicit description. A similar crack extension technique was implemented in the simulation software ProCrack of Rabold and Kuna [26]. Sadeghirad *et al.* [24] presented a different hybrid explicit/implicit approach that was implemented in XFA3D. This approach updates the implicit crack description and uses the explicit description of the crack to ensure that the generated level sets are based on a consistent crack description. Although a large number of papers have been published on the description and tracking of cracks, only a small number deals with the three-dimensional implementation thereof. The most important challenge is correctly tracking the crack surface, if continuity is required. Rabczuk *et al.* [27] noted that most methods require C^0 continuity, which means that the curves are continuous. Only a few papers were concerned with higher order continuous crack surfaces such as C^1 continuity. A C^1 continuity means that the first derivatives of the curves are also continuous. Exhaustive reviews on the advancements and applications of the X-FEM method have been published for which the reader is referred to [25, 28, 29, 30].

In this work, an X-FEM based framework for mixed-mode non-planar fatigue crack growth is presented. It makes use of the X-FEM solver of Abaqus. A hybrid implicit/explicit crack description approach is adopted for tracking and extending the crack. Similar to the work of Fries and Baydoun [23] the explicit description of a crack is used to determine the crack increment and update the level sets. The method employed here is different in that the crack face is described using a B-spline surface. This increases the accuracy of the crack front description compared to methods that describe the crack front by a set of flat triangles. This is especially true in simulations where the crack front length increases significantly, because then the fixed discretization of the crack front leads to a strongly discontinuous crack front geometry. In this work the locations of the crack front evaluation points are re-positioned equidistantly along the continuous crack front after each iteration to ensure optimal description of the crack front. Furthermore, with the use of a B-spline surface for the description of the crack faces it is possible to impose a C^1 continuity of the crack faces. To validate the presented work, three different cases are discussed: (1) compact tension specimens with an asymmetrically located hole, (2) a beam with a slanted crack subjected to a three-point bending load and (3) a beam with a slanted crack subjected to a torsion

load.

2. Framework structure

The framework has been developed in Python using an object-oriented programming (OOP) paradigm. As OOP takes advantage of similarities between objects, it allows for easy implementation of different crack growth laws, initial crack types, etc. Python was chosen because of its open-source nature with a large contributing community. The large number of libraries that are available, make implementation of new algorithms generally simple and efficient. In this work the Abaqus (v2021) X-FEM solver and the associated Abaqus scripting interface (ASI), which is an extension of Python, are used. The ASI is based on Python 2.7 which, since January 1, 2020 is no longer supported. To ensure that the developed framework can take full advantage of the latest Python packages and of future developments of Python 3, the framework is split up in two parts such that only the functions that directly interact with the ASI were developed in Python 2.7. A communication protocol has to be used to establish a connection between the two parts, this was realized with a TCP/IP protocol. Here the client (i.e. the main program) runs in a Python 3 environment and sends serialized commands, messages, data, ... to a dedicated network port. The server (the ASI) which is constantly listening on this port will receive the serialized data, act accordingly and if necessary send data back. Additional advantages of using a network protocol are that the Abaqus solver can run on a dedicated machine with more computational resources and that it enhances the compatibility of the framework with other solvers. If the chosen solver or software can communicate over a network port it should be compatible with the developed framework.

The framework structure is presented in Figure 2. It comprises several modules with dedicated tasks which are identified in the flowchart. It also shows where data is being transferred from the client to the server and vice-versa by means of striped arrows. In the following sections, each of the modules and their functions will be discussed.

3. Initialization

The framework requires three input files. The first is a generic ASCII configuration file in which all the simulation parameters have to be defined. This includes the initial crack type and dimensions, crack growth law, size of the crack growth increment, material type, solver settings, etc. The second file is used to define the load spectrum, which means the number of cycles, maximum and minimum load and the corresponding load case. In the finite element model (FEM), different steps are defined that correspond to different loading conditions. Each of these loading conditions is solved for a unit load. In the FCG calculation, the output results (i.e. the stress intensity factors) of the X-FEM simulation can then be scaled with the loads defined in the load spectrum file, which is valid in the scope of linear elastic fracture mechanics. The last file is an Abaqus/CAE file that contains the meshed base model in which the initial cracks have to be inserted. All loading scenarios that should be considered in the FCG analysis also have to be defined in that file.

3.1. TCP server and communication protocol

When a FCG analysis is launched, the program goes through a number of initialization steps. The first step is setting up the TCP connection, this is realised with the standard Python packages `socket` and `SocketServer`. The server is created first, this requires instantiating a request

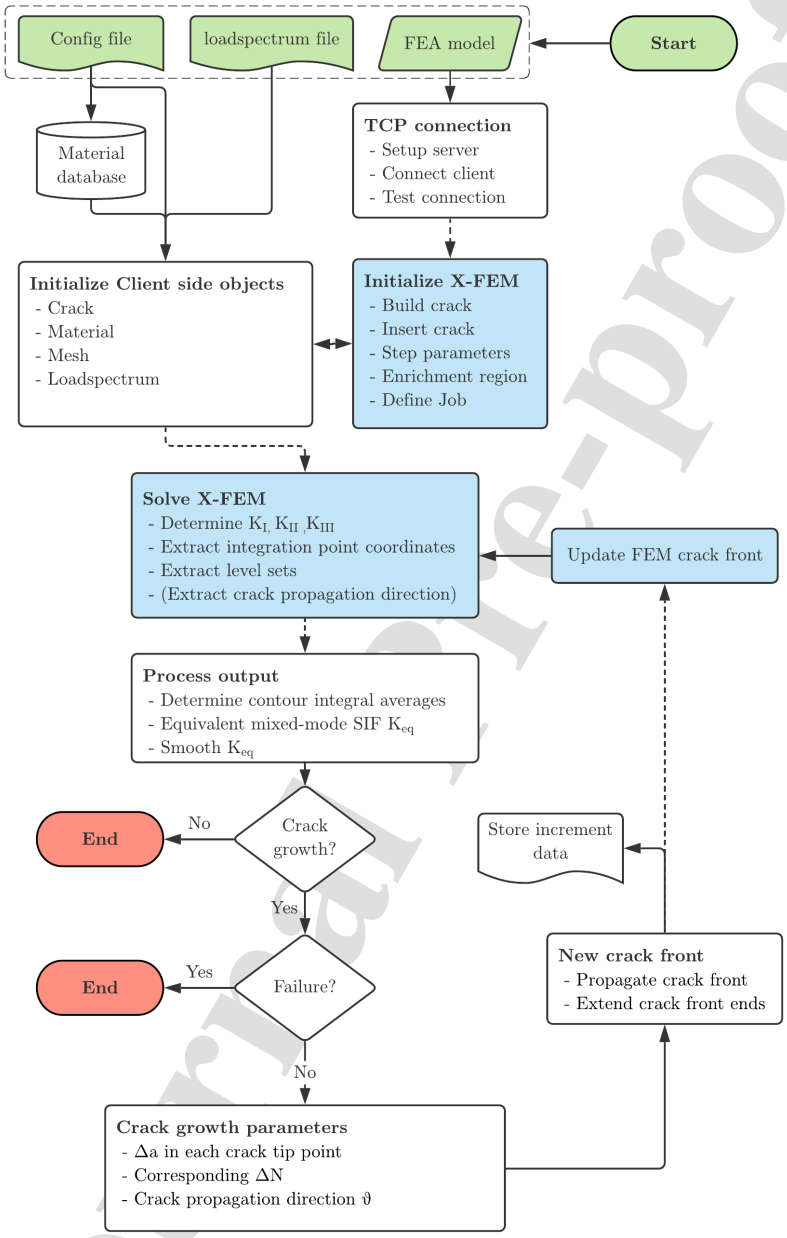


Figure 2: Flowchart of the framework. Green boxes represent the inputs, white boxes represent client-side processes, blue boxes represent server-side processes, red boxes indicate termination criteria. The striped arrows indicate an exchange of data between client and server.

handler class that inherits functionality of an abstract class called `BaseRequestHandler`. The inheriting class is the `AbaqusRequestHandler` which defines how incoming requests are handled. Next the `TCPServer` class is instantiated after which the server starts listening for requests. When the server has been started successfully, a test connection is made from the client side to confirm this. Once the client-server connection is confirmed, the relevant inputs of the configuration file are sent to the server and the FEM initialization is started.

Since TCP/IP is a stream based protocol, it is required to define message-based protocol on top of TCP to differentiate message boundaries. This is achieved by preceding the payload with a message that contains the length of the actual message. It is then possible to determine at the receiving side if the complete package was received. If a file has to be exchanged between the client and server, the length of the file is sent first. At the receiving side, the payload byte-stream is then continuously decoded and written to a temporary file until the length of the temporary file corresponds to the file length specified in the header.

3.2. X-FEM initialization

The first step is exporting the mesh to a predefined file type (e.g. .OBJ), this file is subsequently sent to the client. The mesh topology will be used in the crack front end extension and crack tip branch algorithms that are performed at the client side. These are discussed in later sections. Next, the crack geometry is build in the Abaqus part module and subsequently inserted in the base model in the assembly module. The coordinate system in which the crack is built is independent of the coordinate system used in the assembly. The positioning of the crack in the assembly module is achieved using a translation and a rotation. The rotation is based on Euler's rotation theorem that states that any arbitrary rotation can be composed of three basic rotations. Each rotation, i.e. around the x, y and z axes respectively, is specified by an angle of rotation which is positive for a counterclockwise rotation. The rotation sequence with the accompanying Euler angles and the translation vector are defined in the configuration file. For example, an arbitrary rotation is given in Equation 3, where $\mathbf{P}_{assembly}$ is a $3 \times n$ matrix of the coordinates of points in the assembly, $\mathbf{P}_{crack}$ is a $3 \times n$ matrix containing the coordinates of the points in the coordinate system of the crack geometry. Finally, $\mathbf{R}_x(\alpha)$, $\mathbf{R}_y(\beta)$ and $\mathbf{R}_z(\gamma)$ are 3×3 rotation matrices for a rotation around each axis, $\mathbf{T}$ is a 3×1 translation matrix.

$$\mathbf{P}_{assembly} = \mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma)\mathbf{P}_{crack} + \mathbf{T} \tag{3}$$

Similar to the FRANC3D crack library, a list of predefined crack shapes is available, but custom crack shapes based on non-destructive testing (NDT) could also be implemented. After the initial crack has been inserted in the base model, the enrichment domain has to be defined. The enrichment domain should consist of those elements that are intersected by the crack and of those elements that are likely to be intersected by the crack as it propagates. Only the elements that are part of the enrichment domain are potentially enriched. The enrichment domain has to be specified in Abaqus/CAE. In Abaqus, each crack that is inserted in the FEM has to be associated with an enrichment domain and there should not be any overlap between the enrichment domains associated with different cracks. If only a single crack is inserted in the base model, the whole enrichment domain will be associated with that crack. If multiple cracks are inserted or if crack front branching occurs during the FCG, the enrichment domain of each crack (front) has to be determined accordingly. This will be elaborated further in Section 4.5.

3.3. Client side initialization

Once the mesh topology file is received by the client, the client side objects can be instantiated. This takes place concurrently to the initialization at the server side. The most import client side classes are `Crack`, `Material`, `Mesh` and `LoadSpectrum`. The `Crack` class is an abstract class that contains all generic functions which are related to FCG. The `Crack` class is inherited by the classes `SurfaceCrack` and `EmbeddedCrack` which implement crack type specific functions for surface breaking cracks and embedded cracks respectively. At the initialization, the initial crack front coordinates are stored in the `Crack` object.

The `Material` class stores all material parameters relevant to the fatigue crack growth analysis. Using the material identification tag defined in the configuration file, the material parameters are pulled from a database that is included in the framework. Of course adding custom materials is straightforward. The material parameters are used to instantiate an object of the `Material` class.

An object of the `Mesh` class is constructed using the exported mesh topology file. The mesh is exported as a linear tetrahedral mesh regardless of the element type used in the actual FEM. Quadratic elements are split at their mid-side nodes to create linear elements and quadrilateral elements are split into triangular elements. The mesh connectivity stored at the client side is thus different from the FEM mesh, but the nodal data is not altered. In the framework, the mesh topology will be used to determine where the crack front intersects the mesh boundaries. Therefore the volumetric data of the mesh is non-essential, i.e. only the element faces on the external surfaces of the base model should be stored. An element face lies on an exterior face of the model if that element face is part of only one tetrahedral element. Their identification is implemented as follows. The number of tetrahedrons that share a particular element face can be found by counting the connected nodes for each node of the face. Then the number of connected nodes shared by all three nodes of a face is equal to the number of tetrahedrons containing that face. Only the element faces from which the nodes share only one common neighbour are kept.

Finally the `LoadSpectrum` class is constructed based on the load spectrum file. The load spectrum class is constructed as a list of `LoadBlock` objects. Each `LoadBlock` instance stores the number of cycles, minimum and maximum load and the load case of the FEM it corresponds to. During the FCG analysis, the `LoadSpectrum` class mainly tracks which load blocks have already been applied and how many cycles of the current load block have been applied.

After the initialization, the iterative part of the FCG simulation follows. This will be discussed in the next section.

4. Iterative crack growth implementation

4.1. Determination of the stress intensity factors

The first step after the initialization stage is solving the finite element model with the initial crack. The stress intensity factors corresponding to different modes will be evaluated using contour integrals around the crack tip [19]. Multiple contour integrals around the crack tip are evaluated using the implementation of Abaqus/Standard [31].

As discussed in the introduction, the crack tip elements are enriched with asymptotic crack tip functions. An enrichment scheme where only the crack tip elements are enriched is referred to as topological enrichment. It has been shown that to improve the asymptotic near-tip displacement solutions, the enrichment should be applied to a zone around the crack tip independent of the mesh size [32, 33], which is referred to as geometrical enrichment. In Abaqus the size of this zone

is defined using the enrichment radius R_{enr} as illustrated in Figure 3. Kim *et al.* [34] recommend a normalized enrichment radius $R_{enr}/(\sqrt{3}L_e) \geq 2$ which corresponds to an enrichment of at least two elements near the crack tip.

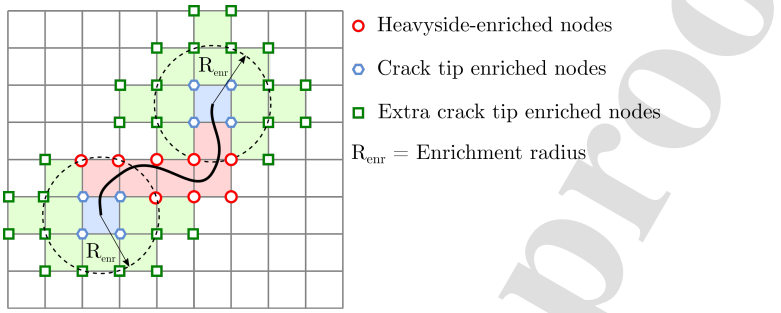


Figure 3: Illustration of the enrichment scheme based on the enrichment radius as defined in Abaqus

When the X-FEM model has been solved, the level set values in the nodes surrounding the cracks and crack tips, the SIF solutions (Mode I, II and III) for all contour integrals along the crack front and the coordinates of the X-FEM integration points are extracted. The extracted output data is stored in a nested dictionary data structure, serialized and sent to the client side.

4.2. X-FEM data extraction and processing

First the contour integral data is processed. The SIF values used for the FCG simulation are obtained by averaging the contour integral values corresponding to the same X-FEM integration point. The SIF's obtained from the first two to four contours often exhibit large deviations when compared to analytical solutions, whilst the other contours converge to the analytical solutions [35]. Therefore the first four contours are omitted when determining the average SIF values using the default framework settings.

In a three-dimensional analysis, the multi-axial stress state influences the crack growth rate. Tanaka [36] proposed a modified Paris law for fatigue crack growth under mixed mode I/II loading

$$\frac{da}{dN} = C\left(\Delta K_{eq}\right)^m \tag{4}$$

$$\Delta K_{eq} = \left(\Delta K_I^2 + 8\Delta K_{II}\right)^{1/4} \tag{5}$$

A number of other equations for an equivalent mixed mode stress intensity factor found in literature have also been implemented. An overview and comparison of a number of these models can be found in [37]; a thorough review on mixed-mode fracture of metals can be found in [38]. The oscillating behaviour of the extracted SIF along the crack front can introduce difficulties when performing a FCG analysis; especially for a simulation that requires a large number of iterations. Therefore the K_{eq} distribution is smoothed. Good results can be obtained by fitting a 5^{th} degree polynomial to the K_{eq} distribution, but in many cases a 3^{rd} degree polynomial is sufficient [34]. The smoothed K_{eq} distribution will be referred to as K_{sm}.

Next, an evaluation is done whether the crack will remain unaffected, propagate in a stable manner, or lead to unstable failure. The former is determined by evaluating the inequality $\Delta K_{sm} \leq$

K_{th}, the latter by evaluating $K_{sm} > K_C$. K_{th} is the threshold SIF and K_{IC} the plane strain fracture toughness. If both inequalities are false, the next stable crack growth increment is determined. If $\Delta K_{sm} \leq K_{th}$ is true, the framework iterates through subsequent load blocks in the load spectrum until this condition is false. If no load block in the load spectrum invalidates the condition, the FCG simulation is terminated. Similarly, if a load block for which $K_{sm} > K_C$ is true is activated, unstable crack growth occurs and thus the FCG simulation is terminated.

If $\Delta K_{sm} \leq K_{th}$ along a part of the crack front, then the locations where the threshold is not exceeded will not advance whilst the ones where the threshold is exceeded will. Thus the crack is allowed to grow irregularly along the crack front. In some cases this can lead to unrealistic crack growth patterns. Therefore the user has the option to set $K_{th} = 0$ or to define a custom behavior for $\Delta K_{sm} \leq K_{th}$ such that a limited amount of crack growth is still allowed.

4.3. Crack growth

4.3.1. Crack front local coordinate system

As mentioned in the introduction, the crack face is defined as a B-spline surface to ensure at least C^1 continuity. The crack surface is spatially three-dimensional (x, y, z) and is described in a two-dimensional parametric space (ξ, η). Thus the B-spline crack surface will be defined as a vector-valued function that maps a two-dimensional space into the three-dimensional space. The function is a tensor product of basis functions and a set of three-dimensional control points [39]. To construct the B-spline based crack surface, the open-source modelling framework NURBS-Python [40] was used.

Let $\Xi = \{\xi_1, ..., \xi_{n+p+1}\}$ be a non-decreasing sequence of real numbers, i.e. $\xi_i < \xi_{i+1}$ with $i = 1, ..., \xi_{n+p+1}$. Ξ is called the knot vector and ξ_i the knots. n is the number of basis functions which comprises the B-spline shape and p is its polynomial order. The basis functions are described with the Cox-deBoor recursion formula starting with piece-wise constants ($p = 0$)

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi \leq \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The i^{th} order basis function $N_{i,p}(\xi)$ with degree p of the B-spline shape for the parametric dimension ξ is described by [41]:

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \tag{7}$$

Through a linear combination of basis functions, a B-spline shape in $\mathbb{R}$ can be constructed. Consider n basis functions $N_{i,p}(\xi)$ and the corresponding control points $\mathbf{P}_i$ then a piece-wise polynomial B-spline curve is given by:

$$C(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\mathbf{P}_i \quad \text{with } i = 1, 2, ..., n \tag{8}$$

Analogously, considering the two-dimensional parametric space (ξ, η) and given a bidirectional net of control points $\mathbf{P}_{i,j}$, a B-spline surface with polynomial order p in ξ and polynomial order q in η can be described as the tensor product [41]:

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \mathbf{P}_{i,j} \tag{9}$$

where $N_{i,p}(\xi)$ and $M_{j,q}(\eta)$ are the basis functions of the parametric dimensions defined on the knot vectors $\Xi = \{\xi_1, ..., \xi_{i+p+1}\}$ and $\mathcal{H} = \{\eta_1, ..., \eta_{i+p+1}\}$ respectively. For the construction of the crack surface, quadratic basis functions are used for both parametric dimensions (ξ, η).

In the input configuration file of the framework, the user defines the number of points k at which the crack front has to be evaluated. This is determinative for the modelling accuracy of the crack front. When the B-spline crack surface is constructed, $(l + 1)k$ control points are used, where l is equal to the number of times the crack front has been extended including the initial crack front, i.e. for the first framework iteration $l = 1$. In the following iterations, the control net for the B-spline crack surface comprises the points that describe the initial crack geometry and all crack front points of previous iterations. This is illustrated in Figure 4. The knot vectors Ξ and $\mathcal{H}$ of the crack surface are generated using $k + p + 1$ and $(l + 1) + q + 1$ knots respectively. The crack front always corresponds to the (normalized) $\xi = 1.0$ knot line. Both knot vectors are clamped and uniform, meaning that the first and last p and q knots for Ξ and $\mathcal{H}$, respectively, are 0 and 1, have a multiplicity of $p + 1$ and that the intermediate knots have a multiplicity of one.

Once the B-spline crack surface has been constructed, k local coordinate systems located on the crack front are determined. Their origin points are equidistantly spaced along the crack front. The local coordinate systems are defined by $(\mathbf{t}_{cf}, \mathbf{n}_{cf}, \mathbf{n}_{cs})$. $\mathbf{t}_{cf}$ are the vectors tangent to the crack front, $\mathbf{n}_{cs}$ the vectors normal to the crack surface and $\mathbf{n}_{cf}$ the vectors that are normal to the crack front and tangent to the crack face. The vectors that are tangent to the B-spline crack surface can be determined by evaluating the first partial derivatives $S_{,\xi}(\xi, \eta)$ and $S_{,\eta}(\xi, \eta)$ of the surface $S(\xi, \eta)$. They are defined in equation 10. Since the crack front corresponds to the $\xi = 1.0$ knot line, the partial derivatives have to be evaluated at $(1.0, \eta_i)$ in the parameter space. $S_{,\xi}(\xi, \eta)$ and $S_{,\eta}(\xi, \eta)$ evaluated at $(1.0, \eta_i)$ give the tangent vectors $\mathbf{n}_{cf,i}$ and $\mathbf{t}_{cf,i}$ at crack front point i, respectively.

$$S_{,\xi}(\xi, \eta) = \frac{\partial}{\partial \xi} S(\xi, \eta)$$
$$S_{,\eta}(\xi, \eta) = \frac{\partial}{\partial \eta} S(\xi, \eta) \tag{10}$$

The normalized surface normal vector $\mathbf{n}_{cs,i}$ at an arbitrary crack front point $(1.0, \eta_i)$ is then given by

$$\mathbf{n}_{cs,i} = \frac{S_{,\xi}(1.0, \eta_i) \times S_{,\eta}(1.0, \eta_i)}{|S_{,\xi}(1.0, \eta_i) \times S_{,\eta}(1.0, \eta_i)|} \tag{11}$$

Figure 4 shows an example of a crack surface in the physical space. The control points used to construct the surface are visualized with white cubes. The black line on the front surface indicates an isoline for $\eta = 0.35$. The partial derivatives and their bi-normal evaluated at $(1.0, 0.35)$ are also shown. The figure illustrates how the local coordinate systems $(\mathbf{t}_{cf}, \mathbf{n}_{cf}, \mathbf{n}_{cs})$ along the crack front are defined. The colormap shows the values of the knot vector Ξ across the B-spline surface.

4.3.2. Crack growth increment

Every iteration, the crack is advanced with a pre-defined fixed crack length Δa. The Δa increment is determined for all points along the crack front in such a way that the crack growth increment in the point with the largest K_{sm} is equal to the fixed increment. It would also be possible to use a fixed number of load cycles ΔN but then the crack growth increment would be small at the start of the simulation and increase rapidly as the crack growth rate exponentially increases.

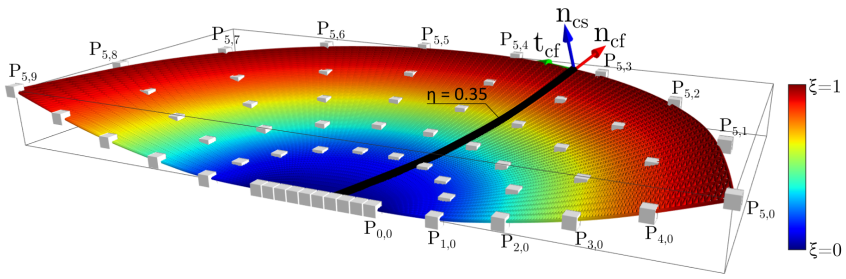


Figure 4: A bi-cubic B-spline crack surface. The black line corresponds to $\eta = 0.35$. The crack face tangent vectors t_{cf} and n_{cf} obtained by evaluation of the first partial derivatives at $\xi = 1.0$, $\eta = 0.35$, are shown. The binormal vector n_{cs} is also illustrated.

For variable amplitude block loading the crack growth is determined as follows. First the K_{min} and K_{max} are determined by scaling the unit SIF's extracted from the model's corresponding load case with the active load block of the load spectrum. Next, the crack growth rate da/dN in each crack front point is determined using the growth law defined in the input file. Different growth laws have been implemented, the most basic one being the Paris law. But if the mean stress has to be accounted for, Walker's equation could for example be used. Crack closure is accounted for by assuming that no crack growth occurs where $\Delta K < K_{th}$. Using the crack growth rate, a value N_{stop} for the crack front point with the largest ΔK is determined which corresponds to the number of cycles until Δa is reached at the load level of the active load block. The length of the active load block is denoted as N_{block}. If $N_{block} < N_{stop}$, the crack growth increment in each crack front point for the active load is determined as $da/dN \times N_{block}$ and stored. N_{stop} is then updated by subtracting the number of applied cycles and the next load block is applied. If then $N_{block} > N_{stop}$, the growth increment corresponds to $da/dN \times N_{stop}$. The total crack growth in each node is then the sum of crack growth increments for each applied load block.

4.3.3. Crack propagation direction

Concurrently with the crack growth increment, the framework also determines the crack propagation direction θ in each crack front point. The vector $\mathbf{n}_{cf}$ corresponds to a crack propagation angle $\theta = 0°$. The crack propagation angle θ can be determined using the maximum tangential stress criterion [42] given in equation 12.

$$\theta = sign(-K_{II}) \cos^{-1}\left(\frac{3K_{II} + \sqrt{K_I^4 + 8K_I^2 K_{II}^2}}{K_I^2 + 9K_{II}^2}\right) \tag{12}$$

Again, other criteria can easily be implemented if needed. The Abaqus solver builds the level set values for the X-FEM analysis based on the crack geometry that is inserted in the mesh. Since the determination of the crack propagation direction in the framework is handled independent of the X-FEM software, it has to be checked if the direction of $\mathbf{n}_{cf,j}$ corresponds to that of the level set ϕ. If they are oriented in the same direction (i.e. dot product > 0) $\mathbf{n}_{cf,j}$ is rotated with a counterclockwise angle θ_j with $\mathbf{t}_{cf,j}$ as the rotation axis. If they are oriented in the opposite

direction, $\mathbf{n}_{cf,j}$ has to rotated with a clockwise angle $-\theta_j$. The counterclockwise rotation of $\mathbf{n}_{cf,j}$ around $\mathbf{t}_{cf,j}$ with angle θ_j has been implemented using Rodrigues' rotation formula. The rotation matrix $\mathbf{R}$ is determined as

$$\mathbf{R} = \mathbf{I} + \sin\theta_j \mathbf{W} + (1 - \cos\theta)\,\mathbf{W}^2 \tag{13}$$

with $\mathbf{I}$ the identity matrix and $\mathbf{W}$ the skew-symmetric matrix for the unit vector $\mathbf{t}_{cf,j}$ expressed as

$$\mathbf{W} = \begin{bmatrix} 0 & -\mathbf{t}_{cf,zj} & -\mathbf{t}_{cf,yj} \\ -\mathbf{t}_{cf,zj} & 0 & -\mathbf{t}_{cf,xj} \\ -\mathbf{t}_{cf,yj} & -\mathbf{t}_{cf,xj} & 0 \end{bmatrix} \tag{14}$$

Then the crack front propagation direction in point j, denoted as $\mathbf{q}_j$, is determined as

$$\mathbf{q}_j = \mathbf{n}_{cf,j} \cdot \mathbf{R} \tag{15}$$

The use of Rodrigues' rotation formula makes vectorization of the code simple, which improves computational efficiency.

4.4. Crack front update

Once the crack growth increment and the crack propagation direction have been determined along the crack front, the crack is updated. The new crack front coordinates in point j can simply be obtained as

$$P_{i+1,j} = \Delta a_j \cdot \mathbf{q}_j + P_{i,j} \tag{16}$$

where Δa_j and B_j are the crack growth increment and crack propagation direction vector in crack front point j, respectively. Once the new crack front has been determined, the new crack geometry is constructed in Abaqus, inserted in the finite element mesh and the next iteration is started by launching the X-FEM solver.

4.5. Definition of multiple cracks and crack front splitting

Each X-FEM crack feature defined in Abaqus has to be associated with the definition of an enrichment region and there should be no overlap between enrichment domains of different crack features. This also implies that a single mesh element cannot contain multiple cracks. The definition of the enrichment regions during the analysis is handled in an automated manner. For the initialization of the analysis the user can manually indicate what regions should be considered when defining the enrichment regions, but this is not required. When a single crack is inserted in the model, the entire region is automatically associated with that crack. However, when multiple independent cracks are inserted or the crack front splits into multiple independent crack fronts, the enrichment region has to be defined accordingly.

When a crack propagates and one of the intermediate crack front points falls outside of the mesh, the crack front has to be split and both parts have to be handled independently from each other. This is illustrated in Figure 5. To determine if a crack front point falls outside of the mesh after propagation, the Möller-Trumbore ray-triangle intersection algorithm is used. It is a fast method for calculating the intersection of a ray and a triangle [43]. A ray with an arbitrary direction is cast from each crack front end point. If a ray intersects with an even number of element faces, then the associated point lies inside the surface mesh (see Figure 6a). Conversely, if the ray intersects with an uneven number of element faces, the point lies outside the mesh (see

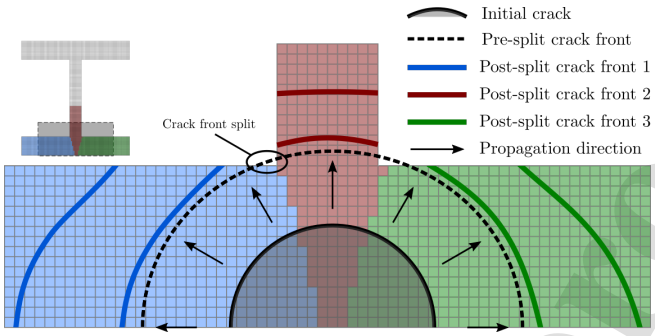


Figure 5: Illustration of the enrichment regions defined for a crack front after splitting. All elements with the same color are associated with the crack front intersecting that region.

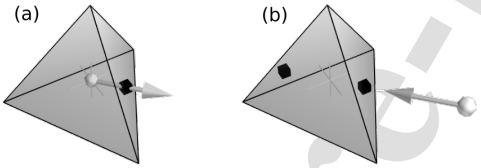


Figure 6: Illustration showing how an arbitrary ray cast from a point can be used to determine if the point lies inside or outside the mesh depending on the number of times the ray intersects with the surface mesh boundary.

Figure 6a). The Möller-Trumbore algorithm was implemented in a vectorized manner (using Numpy).

When a crack front is split or multiple cracks are present in the model, the finite element model has to be sectioned into different enrichment sets. The enrichment sets are automatically determined as follows. First the middle point on each B-spline crack front is determined, this corresponds to the point at $\xi = 1.0$ and $\eta = 0.5$ in the parameter space. Then, for each mesh element, the center point is determined. The enrichment set of a crack front is then defined by all elements from which the center points lie closest to that crack front's center point. Figure 5 shows an example of an elliptical crack that initiated in the bottom flange of an I-profile and has propagated into the web resulting in three independent crack fronts. The colored regions correspond to the three enrichment regions that were determined to be used in the definition of the X-FEM crack feature.

4.6. Robustness measures

4.6.1. Automated crack front widening

The level set values are constructed based on the crack geometry inserted in the finite element model. The crack propagation direction vectors are always normal to the crack front. The crack front end points of the propagated front may fall inside the mesh. This can happen due to the direction of the crack front normal vectors at the end points or because the geometry of the specimen widens. This is illustrated in Figure 7. In that case, the spatial curve which corresponds to the crack front does not completely intersect the mesh, and some elements will not be

completely cut through. This challenge is encountered not only in X-FEM, but in all crack propagation software packages that are based on an explicit representation of the crack. To resolve this, the crack front is typically extended using some type of extrapolation (mostly linear) of the crack front end points. However this can lead to unrealistic scenarios especially when the crack grows around sharp corners or small radii (compared to the crack growth increment). This can be solved through the use of different extrapolation functions for the determination of the crack front ends, however these require operator intervention [10]. Here a new crack front widening algorithm is introduced that is capable of dealing with these types of features automatically.

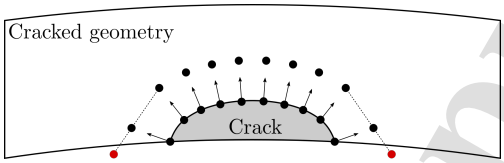


Figure 7: Linear extrapolation of the crack front ends

First it is determined if the crack front end points lie inside or outside the mesh with the Möller-Trumbore algorithm as described in Section 4.5. If a crack front end is not inside the mesh, the corresponding side of the crack front does not have to be extended. If it is inside the mesh, then the corresponding crack front end has to be extended. A circle centered at the crack front end point with a radius equal to the Δa increment and defined in the plane tangent to the crack surface is determined. Let A be an end point of crack front i inside the mesh. $\mathbf{t}_{cf,A}$ and $\mathbf{n}_{cf,A}$ are the crack front tangent and normal in A. Then the circle is defined by the parametric equation 17.

$$\begin{cases} x = A_x + \Delta a \cos(\theta)\mathbf{t}_{cf,A_x} + \Delta a \sin(\theta)\mathbf{n}_{cf,A_x} \\ y = A_y + \Delta a \cos(\theta)\mathbf{t}_{cf,A_y} + \Delta a \sin(\theta)\mathbf{n}_{cf,A_y} \\ z = A_z + \Delta a \cos(\theta)\mathbf{t}_{cf,A_z} + \Delta a \sin(\theta)\mathbf{n}_{cf,A_z} \end{cases} \tag{17}$$

Next, n points B_j (where $j = 0, ..., n-1$) are determined on the circle and projected onto the mesh. Figure 8a illustrates the circle with radius Δa, the crack front end point A and the projected points B_j. With each projected point B_j, an associated vector $\mathbf{w}_j$ is defined according to equation 18. These vectors are also illustrated in Figure 8a.

$$\mathbf{w}_j = \left[B_{xj}, B_{yj}, B_{zj}\right]^T - \left[A_x, A_y, A_z\right]^T \tag{18}$$

Any projected point B_j that falls outside the green region shown in Figure 8b will be discarded. This region is demarcated by the boundary of a sphere with the same radius and center point as the circle and two vectors. Using a widening point that lies outside this region generally leads to an unrealistic crack front propagation. The vectors that demarcate the boundary are determined as follows. The vector $\mathbf{q}$ is defined as:

$$\mathbf{q} = [A_x, Ay, A_z]^T - \left[A'_x, A'_y, A'_z\right]^T \tag{19}$$

where A is the end point of the current crack front and A' the corresponding end point of the previous crack front. Then, for each of the projected points the following conditions are required:

$$\begin{cases} \mathbf{q} \cdot \mathbf{w}_j > 0 \\ \mathbf{t}_{cf,A} \cdot \mathbf{w}_j > 0 \\ distance(A, B_j) < \Delta a \end{cases} \tag{20}$$

If one of these conditions is not full-filled, the projected point corresponding to $\mathbf{w}_j$ is discarded. This is illustrated in Figure 8b. The green region is defined through the three imposed conditions for the projected points. For each of the remaining projected points, the angle between the corresponding $\mathbf{w}_j$ and $\mathbf{t}_{cf,A}$ is calculated. The vector $\mathbf{w}_j$ for which the angle is smallest, denoted $\mathbf{w}_{min}$, is then used to determine the extended crack front end point:

$$W = \left[A_x, A_y, A_l\right]^T + \left(1 + \left[\mathbf{w}_{y,min}, \mathbf{w}_{z,min}, \mathbf{w}_{x,min}\right]^T\right) \tag{21}$$

where W is the extended crack front end point or crack front widening point and $\|\mathbf{w}_{i,min}\| = \text{distance}(A, B_j)$. Equation 18 results in a point that lies 1 mm outside of the mesh such that it ensures that the crack front completely cuts the mesh. In most situations, the algorithm result is equivalent to a simple linear extrapolation of the crack front end points. However, for crack growth around sharp corners and small radii it enables automated determination of the new crack front.

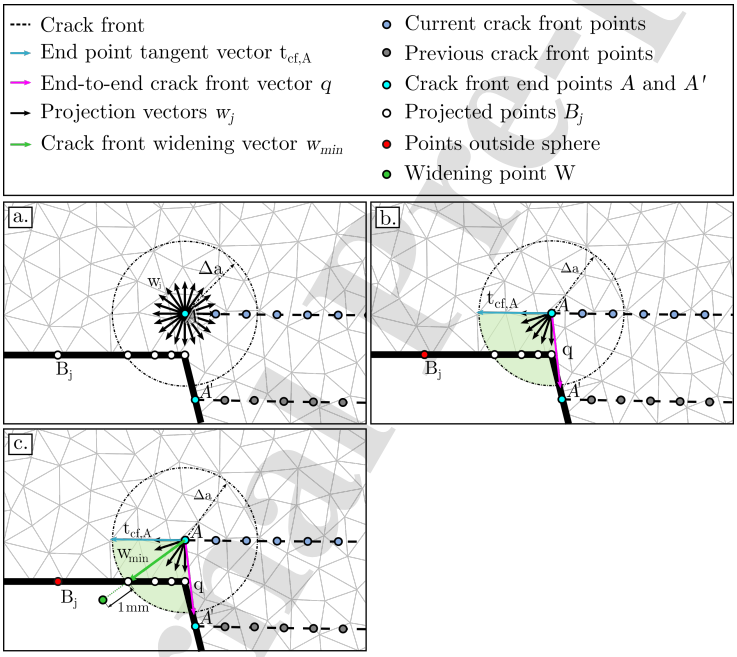


Figure 8: Crack front widening algorithm steps

4.6.2. Handling level set errors

In some cases, Abaqus fails in constructing the level sets. Generally this is due to a numerical incompatibility between the mesh and the inserted crack geometry. One way to solve this is by re-meshing the model, however this defeats the purpose of using X-FEM in the first place. In this framework, when such incompatibility occurs, the crack extension is simply increased with 10%

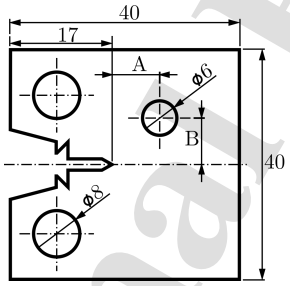
of the defined Δa increment. This has proven to be effective for a large number of simulations with different geometries. Of course the lifetime corresponding to this increase is also determined in the same way as explained in Section 4.3.2. Although it is also possible to develop Abaqus user subroutines (which are mostly written in Fortran) to handle the construction of the level sets, one of the goals of this work was the development of a purely Python based framework.

5. Validation

To validate the 3D crack growth propagation framework, results of the proposed framework are compared with both in-house fatigue crack growth experiments and experimental data available in literature. Three different cases are discussed: (1) compact tension specimens with an asymmetrically located hole, (2) a beam with a slanted crack subjected to a three-point bending load and (3) a beam with a slanted crack subjected to a torsion load. Cases (2) and (3) have been selected to demonstrate the framework's performance for non-planar 3D crack growth.

5.1. Compact tension specimen with an asymmetrically located hole

Based on literature [44], validation specimens were designed in the form of compact tension specimens with an additional, asymmetrically placed hole [45]. The geometry of the specimens is shown in Figure 9. The asymmetrically placed hole introduces mode II, out of plane crack growth. Four different specimens, each with a thickness of 16*mm*, were produced with different *A* and *B* combinations. The different combinations are given on the right in Figure 9. The specimens were made from a high strength low alloy (HSLA) steel, NV F460, developed for use in offshore structures. The chemical composition and mechanical properties can be found in tables 1 and 2, respectively [46].



1. A = 6.77 B = 8.0
2. A = 6.77 B = 6.5
3. A = 8.77 B = 8.0
4. A = 8.77 B = 6.5

Figure 9: Geometry of the tested CT specimens, values of *A* and *B* are given on the right for the four different specimens. All dimensions in mm.

Table 1: Chemical properties of NV F460 in steel wt. %

C	Mn	Si	P	S	Cu	Ni	Cr	Mo
0.08	1.24	0.24	0.01	0.001	0.05	0.21	0.05	0.005

Before actual testing started, the specimen had been fatigue pre-cracked to make sure that a sharp crack tip is created according to ASTM E647. Following pre-cracking, a servo-hydraulic ESH 150 *kN* universal testing machine was used to perform cyclic sinusoidal loading with a

Table 2: Mechanical properties of NV F460 steel

Property	Symbol	Value	Unit
Yield strength	σ_y	560	MPa
Ultimate tensile strength	σ_{UTS}	630	MPa
Paris' law exponent	m	3.064	-
Paris' law coefficient	C	$3.6 \cdot 10^{-12}$	$m/cycle$
Threshold SIF	ΔK_{th}	5.0	MPa $\sqrt{m}$
Mode I critical SIF	K_{Ic}	84.0	MPa $\sqrt{m}$
Young's Modulus	E	209	GPa

maximum value of 13.8 kN and minimum value of 5.2 kN at a frequency of 5Hz. A digital image correlation (DIC) system was used to track the crack path at the specimen surface.

Figure 10 shows the post-processed DIC results of the fatigue crack growth experiments and the associated X-FEM crack path predictions compared to the experimental results. The colormaps on the DIC images correspond to the von Mises strain fields. Overall the X-FEM crack path predictions show good agreement with the experimental results taking inherent spread of experimentally determined material data into account. The largest deviation between the numerical and experimental crack paths can be seen for specimen a. Here the experimental crack path was seemingly unaffected by the influence of the asymmetrically located hole. A slight deviation away from the hole was even observed. On the other hand the X-FEM crack grows slightly towards the hole. The numerically determined final crack length was slightly larger than the experimental one for the second and the third specimens, but good agreement was found for the first and last specimens.

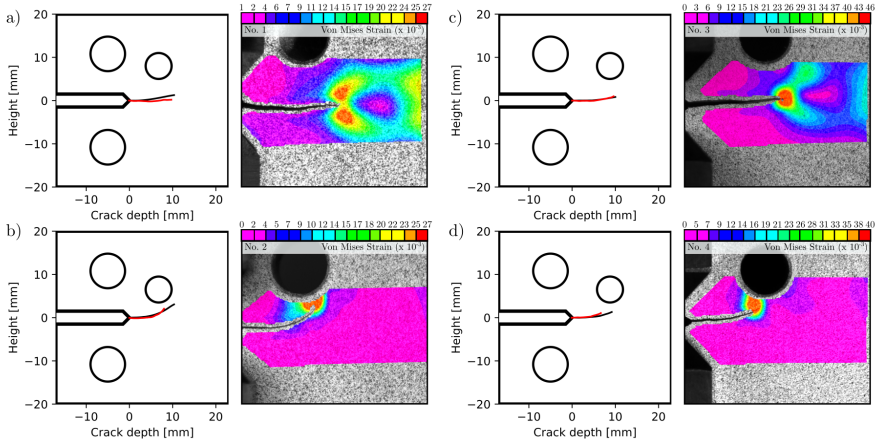


Figure 10: Comparison between experimental (red) and numerical trajectories (black) shown in the left column. Post-processed DIC images at the end of the fatigue crack growth experiments shown in the right column.

5.2. Single edge slanted notch three-point bend test

Based on the numerical/experimental study of Buchholz *et al.* [47], a single edge notch (SEN) specimen with a slanted crack in a three-point bending setup is considered. A drawing of the specimen with dimensions, load and boundary conditions is shown in Figure 11. Identical to the original work, the material parameters used in the simulation are $E = 210\ GPa$ and $\nu = 0.3$. The finite element model is meshed with linear, reduced integration, hexahedral elements. A load ratio $R = 0.1$ is assumed and a $\Delta a_{max} = 1$ mm is used.

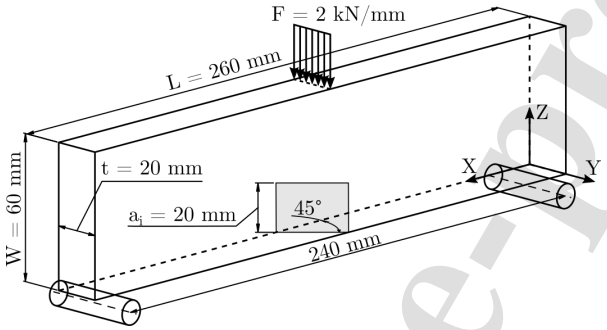


Figure 11: SEN specimen with a slanted crack subjected to three point bending

The results are shown in Figure 12. Figure 12a shows the final crack obtained with the presented framework in the finite element mesh compared to the final experimental crack in the PMMA specimen of Buchholz *et al.* Figure 12b shows the different crack front iterations from the initial to the final fatigue crack growth stage as a top-down view. Quantitative comparison with the experiments of Buchholz *et al.* is not possible as this type of data was not published in [47]. However, a qualitatively good agreement with the experimental results is undeniable as seen in Figure 12a.

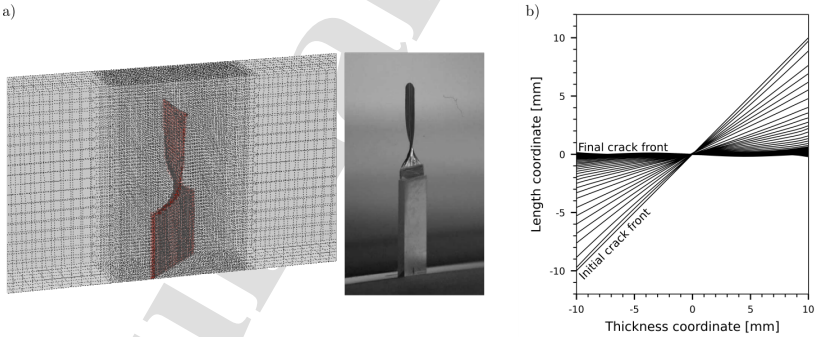


Figure 12: a) Final crack shape in a SEN specimen with a slanted crack subjected to three point bending obtained using the presented framework compared to an experimentally obtained crack in a PMMA beam [47]. b) Fatigue crack front iterations in a top-down view

5.3. Torsion loaded single edge slanted notch specimen

A single edge notch (SEN) specimen with a slanted crack subjected to a torsion load is considered, as reported in [48, 49]. A drawing of the specimen with dimensions, load and boundary conditions is shown in Figure 13. Identical to the original work, the material parameters used in the simulation are $E = 210\ GPa$ and $\nu = 0.3$. The finite element model is meshed with linear, reduced integration, hexahedral elements. A load ratio $R = 0.1$ is assumed and a $\Delta a_{max} = 1\ mm$ is used.

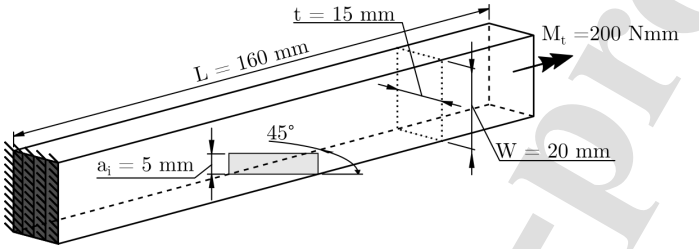


Figure 13: Clamped SEN specimen with a slanted crack subjected a torsion load.

Buchholz *et al.* [48] performed an experiment on a PMMA specimen. These experiments only serve a qualitative purpose as the experiment was a displacement controlled quasi-static test, i.e. not a fatigue crack growth test. Nonetheless, Buchholz *et al.* [47] note that previous comparisons between displacement controlled quasi-static tests and fatigue crack growth tests showed nearly identical crack paths for different materials among which PMMA. Figure 14 compares numerically predicted crack fronts obtained using the presented framework and the experimental results presented by Buchholz *et al.* [48]. Here again a qualitative agreement is found, validating the capability of the presented framework to deal with complex 3D fatigue crack growth.

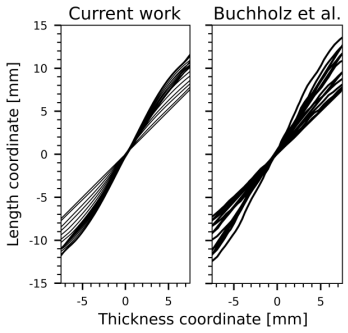


Figure 14: Comparison between the experimental results of the current work compared to the numerical results from the work of Buchholz *et al.* [48] for crack growth in a SEN torsion loaded beam with a slanted crack. [48]

6. Conclusions

This paper introduced a Python based numerical framework for arbitrary three-dimensional fatigue crack propagation based on the X-FEM method. All code related to the description and propagation of the fatigue crack is developed as a stand-alone code base. For this paper, the framework takes advantage of the Abaqus X-FEM solver to determine the stress intensity factor solutions. Combination with the Abaqus scripting interface is achieved through the implementation of a TCP-protocol. To accurately describe the crack front and crack plane in three dimensions, a B-spline formulation with a C^1 continuity for the crack faces is used. The crack advancement is simulated in an iterative manner based on a predefined Δa_{max}. To improve automated crack advancement along sharp corners, a new crack front extension algorithm was introduced. The introduced algorithm combines the use of common linear or polynomial extrapolation extension methods with a mesh boundary detection method based on the Möller-Trumbore algorithm. To further improve the robustness, a pragmatic solution to failure of level set construction from the explicit crack face geometry was implemented.

The proposed framework is validated through comparison of simulation results with three cases: (1) compact tension specimens with an asymmetrically located hole, (2) a beam with a slanted crack subjected to a three-point bending load and (3) a beam subjected to a torsion load with a slanted crack. For all three specimens a good qualitative agreement was found with respect to the final 3D crack front shape showing the feasibility of the framework.

7. Acknowledgments

The authors acknowledge the support of SIM (Strategic Initiative Materials in Flanders) and IBN Offshore Energy. This research was funded by VLAIO, project number 179P04718W.

References

[1] D. B. Brickman, ASM Handbook Volume 11: Failure analysis and prevention, 1997.
[2] J. Schijve, Fatigue of Structures and Materials, 2nd Edition, Springer, Delft, The Netherlands, 2008.
[3] J. Harter, AFGROW User guide and technical manual (Februari 1999).
[4] NASGRO, Fracture mechanics and fatigue crack growth analysis software, NASA Johnson Space Center and Southwest Research Institute, 6th Edition (2010).
[5] H. Dirik, T. Yalçinkaya, Crack path and life prediction under mixed mode cyclic variable amplitude loading through XFEM, International Journal of Fatigue 114 (April) (2018) 34–50. doi:10.1016/j.ijfatigue.2018.04.026.
URL https://doi.org/10.1016/j.ijfatigue.2018.04.026
[6] D. G. Pavlou, G. N. Labeas, N. V. Vlachakis, F. G. Pavlou, Fatigue crack propagation trajectories under mixed-mode cyclic loading, Engineering Structures 25 (7) (2003) 869–875. doi:10.1016/S0141-0296(03)00018-X.
[7] D. Colombo, M. Giglio, A methodology for automatic crack propagation modelling in planar and shell FE models, Engineering Fracture Mechanics 73 (4) (2006) 490–504. doi:10.1016/j.engfracmech.2005.08.007.
[8] M. R. Nikfam, M. Zeinoddini, F. Aghebati, A. A. Arghaei, Experimental and XFEM modelling of high cycle fatigue crack growth in steel welded T-joints, International Journal of Mechanical Sciences 153-154 (January) (2019) 178–193. doi:10.1016/j.ijmecsci.2019.01.040.
[9] ZENCRACK, User manual, issue 6, Zentech inc. (1999).
[10] Franc3D, User manual, version 7.4, Fracture Analysis Consultants Inc. (2019).
[11] M. Schöllmann, M. Fulland, H. A. Richard, Development of a new software for adaptive crack growth simulations in 3D structures, Engineering Fracture Mechanics 70 (2) (2003) 249–268. doi:10.1016/S0013-7944(02)00028-0.
[12] E. Poursaeidi, H. Bakhtiari, Fatigue crack growth simulation in a first stage of compressor blade, Engineering Failure Analysis 45 (2014) 314–325. doi:10.1016/j.engfailanal.2014.06.018.
[13] Q. Wu, X. Chen, Z. Fan, D. Nie, J. Pan, Engineering fracture assessment of FV520B steel impeller subjected to dynamic loading, Engineering Fracture Mechanics 146 (2015) 210–223. doi:10.1016/j.engfracmech.2015.07.045.

[14] T. D. Joy, J. P. Brüggemann, G. Kullmer, Crack growth simulation with Adapcrack3D in 3D structures under the influence of temperature, Procedia Structural Integrity 13 (2018) 328–333. doi:10.1016/j.prostr.2018.12.055.
[15] D. V. Swenson, A. R. Ingraffea, Modeling mixed-mode dynamic crack propagation using finite elements: Theory and applications, Computational Mechanics 3 (6) (1988) 381–397. doi:10.1007/BF00301139.
[16] T. L. Anderson, Fracture Mechanics: Fundamentals and Applications, Fourth Edition, 2017.
[17] R. Branco, F. V. Antunes, J. D. Costa, A review on 3D-FE adaptive remeshing techniques for crack growth modelling, Engineering Fracture Mechanics 141 (2015) 170–195. doi:10.1016/j.engfracmech.2015.05.023.
[18] T. Belytschko, T. Black, Elastic crack growth in finite elements with minimal remeshing, International Journal for Numerical Methods in Engineering 45 (1999) 601–620. doi:10.3760/cma.j.issn.0366-6999.2011.18.023.
[19] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, International Journal for Numerical Methods in Engineering 46 (1) (1999) 131–150. doi:10.1002/(SICI)1097-0207(19990910)46:1¡131::AID-NME726¿3.0.CO;2-J.
[20] N. Sukumar, N. Moës, B. Moran, T. Belytschko, Extended finite element method for three-dimensional crack modelling, International Journal for Numerical Methods in Engineering 48 (11) (2000) 1549–1570. doi:10.1002/1097-0207(20000820)48:11¡1549::AID-NME955¿3.0.CO;2-A.
[21] M. Stolarska, D. L. Chopp, N. Mos, T. Belytschko, Modelling crack growth by level sets in the extended finite element method, International Journal for Numerical Methods in Engineering 51 (8) (2001) 943–960. doi:10.1002/nme.201.
[22] J. Shi, D. Chopp, J. Lua, N. Sukumar, T. Belytschko, Abaqus implementation of extended finite element method using a level set representation for three-dimensional fatigue crack growth and life predictions, Engineering Fracture Mechanics 77 (14) (2010) 2840–2863. doi:10.1016/j.engfracmech.2010.06.009.
[23] T.-P. Fries, M. Baydoun, Crack propagation with the extended finite element method and a hybrid explicit-implicit crack description, International Journal for Numerical Methods in Engineering 89 (February) (2012) 1527–1558. doi:10.1002/nme.
[24] A. Sadeghirad, D. L. Chopp, X. Ren, E. Fang, J. Lua, A novel hybrid approach for level set characterization and tracking of non-planar 3D cracks in the extended finite element method, Engineering Fracture Mechanics 160 (2016) 1–14. doi:10.1016/j.engfracmech.2016.03.027.
[25] T.-P. Fries, T. Belytschko, The extended/generalized finite element method: An overview of the method and its applications, International Journal for Numerical Methods in Engineering 84 (April) (2010) 253–304. doi:10.1002/nme.2914.
[26] F. Rabold, M. Kuna, Automated Finite Element Simulation of Fatigue Crack Growth in Three-dimensional Structures with the Software System ProCrack, Procedia Materials Science 3 (2014) 1099–1104. doi:10.1016/j.mspro.2014.06.179.
[27] T. Rabczuk, X. Zhuang, J.-H. Song, C. Anitescu, Extended Finite Element and Meshfree Methods, 2019. doi:https://doi.org/10.1016/C2017-0-00659-6.
[28] T. Belytschko, R. Gracie, G. Ventura, A review of extended/generalized finite element methods for material modeling, Modelling and Simulation in Materials Science and Engineering 17 (4) (2009). doi:10.1088/0965-0393/17/4/043001.
[29] N. Sukumar, J. E. Dolbow, N. Moës, Extended finite element method in computational fracture mechanics: a retrospective examination, International Journal of Fracture 196 (1-2) (2015) 189–206. doi:10.1007/s10704-015-0064-8.
[30] K. Rege, H. G. Lemu, A review of fatigue crack propagation modelling techniques using FEM and XFEM, IOP Conference Series: Materials Science and Engineering 276 (1) (2017). doi:10.1088/1757-899X/276/1/012027.
[31] M. Smith, ABAQUS/Standard Users Manual, Version 2020, Dassault Systemes Simulia Corp, United States, 2020.
[32] E. Béchet, H. Minnebo, N. Moës, B. Burgardt, Improved implementation and robustness study of the X-FEM for stress analysis around cracks, International Journal for Numerical Methods in Engineering 64 (8) (2005) 1033–1056. doi:10.1002/nme.1386.
[33] P. Laborde, J. Pommier, Y. Renard, M. Salaün, High-order extended finite element method for cracked domains, International Journal for Numerical Methods in Engineering 64 (3) (2005) 354–381. doi:10.1002/nme.1370.
[34] J. S. Kim, H. J. Lee, Y. J. Kim, Y. B. Kim, The mesh density effect on stress intensity factor calculation using ABAQUS XFEM, Journal of Mechanical Science and Technology 33 (10) (2019) 4909–4916. doi:10.1007/s12206-019-0931-8.
[35] J. H. Pang, K. S. Tsang, H. J. Hoh, 3D stress intensity factors for weld toe semi-elliptical surface cracks using XFEM, Marine Structures 48 (2016) 1–14. doi:10.1016/j.marstruc.2016.04.001.
[36] K. Tanaka, Fatigue crack propagation from a crack inclined to the cyclic tensile axis, Engineering Fracture Mechanics 6 (3) (1974). doi:10.1016/0013-7944(74)90007-1.
[37] S. Sajith, K. S. Murthy, P. S. Robi, Fatigue life prediction under mixed-mode loading using equivalent stress intensity factor models, in: MATEC Web of Conferences, Vol. 172, 2018, pp. 2–6. doi:10.1051/matecconf/201817203005.

[38] Y. Wang, W. Wang, B. Zhang, C. Q. Li, A review on mixed mode fracture of metals, Engineering Fracture Mechanics 235 (June) (2020) 107126. doi:10.1016/j.engfracmech.2020.107126.
[39] J. A. Cottrel, T. J. Hughes, Y. Bazilevs, Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons, Ltd, 2009.
[40] O. R. Bingol, A. Krishnamurthy, NURBS-Python: An open-source object-oriented NURBS modeling framework in Python, SoftwareX 9 (2019) 85–94. doi:10.1016/j.softx.2018.12.005.
[41] L. Piegl, W. Tiller, The NURBS book, Springer Science & Business Media, 1996.
[42] F. Erdogan, G. Sih, On the crack extension in plates under plane loading and transverse shear, Journal of Basic Engineering 86 (4) (1963) 519–525. doi:10.1115/1.3656897.
[43] T. Möller, B. Trumbore, Fast, minimum storage ray-triangle intersection, Journal of Graphics Tools 2 (1) (1997) 21–28. doi:10.1080/10867651.1997.10487468.
[44] A. C. Miranda, M. A. Meggiolaro, J. T. Castro, L. F. Martha, T. N. Bittencourt, Fatigue life and crack path predictions in generic 2D structural components, Engineering Fracture Mechanics 70 (10) (2003) 1259–1279. doi:10.1016/S0013-7944(02)00099-1.
[45] Zhang, Jie and Kiekens, Cedric and Hertelé, Stijn and De Waele, Wim, Identification and prediction of mixed-mode fatigue crack path in high strength low Alloy steel, in: Proceedings of The 18th International Conference on Experimental Mechanics, Vol. 2, mdpi, 2018, p. 6.
URL http://dx.doi.org/10.3390/ICEM18-05420
[46] Micone, Nahuel and De Waele, Wim, Experimental evaluation of block loading effects on fatigue crack growth in offshore structural steels, Marine Structures 64 (2019) 463–480.
URL http://dx.doi.org/10.1016/j.marstruc.2018.10.005
[47] F. G. Buchholz, A. Chergui, H. A. Richard, Fracture analyses and experimental results of crack growth under general mixed mode loading conditions, Engineering Fracture Mechanics 71 (4-6) (2004) 455–468. doi:10.1016/S0013-7944(03)00015-8.
[48] F. G. Buchholz, V. Just, H. A. Richard, Computational simulation and experimental results on 3D crack growth in a 3PB-specimen with an inclined crack plane, Key Engineering Materials 251-252 (June) (2003) 85–90. doi:10.4028/www.scientific.net/kem.251-252.85.
[49] F. G. Buchholz, V. Just, H. A. Richard, Computational simulation and experimental findings of three-dimensional fatigue crack growth in a single-edge notched specimen under torsion loading, Fatigue and Fracture of Engineering Materials and Structures 28 (1-2) (2005) 127–134. doi:10.1111/j.1460-2695.2005.00864.x.

Highlights

An X-FEM based framework for 3D fatigue crack growth using a B-spline crack geometry description

Kris Hectors, Wim De Waele

- A Python based iterative framework for three-dimensional fatigue crack propagation.
- B-spline formulation based crack surface and crack face description.
- A new, robust, crack front extension method for simulation of 3D crack growth in complex geometries.
- Good agreement between numerical and experimental results.

Declaration of interests

☐ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Kris Hectors reports financial support was provided by Agency of Innovation and Entrepreneurship.