Advances in Engineering Software 41 (2010) 111–119

Contents lists available at ScienceDirect

Advances in Engineering Software

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

Automatic LEFM crack propagation method based on local Lepp–Delaunay mesh refinement

David Azócar [b], Marcelo Elgueta [b], María Cecilia Rivara [a,*]

[a] Universidad de Chile, Departamento de Ciencias de la Computación, Av. Blanco Encalada 2120, Santiago, Chile
[b] Universidad de Chile, Departamento de Ingeniería Mecánica, Av. Beauchef 850 Torre Central, Santiago, Chile

ARTICLE INFO

Article history:
Received 6 October 2008
Accepted 2 October 2009
Available online 31 October 2009

Keywords:
Crack propagation
Finite element method
Delaunay
Triangulation
Lepp–Delaunay method
Displacement extrapolation
Numerical methods

ABSTRACT

A numerical method for 2D LEFM crack propagation simulation is presented. This uses a Lepp–Delaunay based mesh refinement algorithm for triangular meshes which allows both the generation of the initial mesh and the local modification of the current mesh as the crack propagates. For any triangle t, Lepp(t) (Longest Edge Propagation Path of t) is a finite, ordered list of increasing longest edge neighbor triangles, that allows to find a pair of triangles over which mesh refinement operations are easily and locally performed. This is particularly useful for fracture mechanics analysis, where high gradients of element size are needed. The crack propagation is simulated by using a finite element model for each crack propagation step, then the mesh near the crack tip is modified to take into account the crack advance. Stress intensify factors are calculated using the displacement extrapolation technique while the crack propagation angle is calculated using the maximum circumferential stress method. Empirical testing shows that the behavior of the method is in complete agreement with experimental results reported in the literature. Good results are obtained in terms of accuracy and mesh element size across the geometry during the process.

1. Introduction

The analysis of crack propagation as well as the failure prediction of structural components in engineering applications are important research subjects. In the last decade numerical analysis of fracture problems have become an effective way of approaching this problem due to the development of the computing capacity. Several methods for the numerical analysis of fracture problems have been developed. The finite element based methods are the more recurrent in the literature. Another approaches used are boundary element methods [1], element-free Galerkin methods [2] or extended finite element methods (X-FEM) [3]. Numerical methods have been widely used to calculate fracture parameters, including linear elastic and elastic–plastic fracture mechanics [4], dynamic fracture mechanics [5], fatigue [6] and quasi-static crack growth [7].

In general terms, a finite element based numerical method applied to fracture mechanics proceeds iteratively as follows: an approximate displacement field solution is numerically obtained; then a numerical approximation of the fracture parameters is computed by using appropriate data post processing.

In any case, mesh generation is a critical aspect of an efficient crack propagation method. This should consider at least the following issues: firstly, generation of a good quality initial mesh of the complex geometry such as a cracked one. A crack in a 2D geometry is represented as a 1D entity, where two free surfaces coincides geometrically but are topologically different, thus the mesh generator must be able to take into account these free surfaces, both in the initial mesh generation and in the crack growth steps. Secondly, in a cracked geometry mesh, elements near crack tip are much smaller than elements far from crack tip, so the algorithm must generate a good size transition between these zones, optimizing element size and keeping element quality in all the mesh. Third, crack propagation simulation implies modification of both the object geometry and its associated mesh for every time step. A first crack growth method due to Bittencourt [8] uses a local method to remesh just the zone close to the crack tip avoiding the remeshing of all geometry, in order to minimize time consumption in the meshing step. The same method [6] was presented for triangular meshes using a mesh generator that combines quadtree and advancing front technique, and a back tracking procedure to eliminate bad shape triangles; this method improves mesh quality at each propagation step but mesh modification is very complex. A similar method generating meshes for cracked geometries (but not crack growth) was implemented for 3D [9] using tetrahedral meshes. In general terms, these methods have the disadvan-

* Corresponding author.
E-mail address: mcrivara@dcc.uhile.cl (M.C. Rivara).

0965-9978/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2009.10.004

tage of using several intermediate steps in order to obtain a good quality mesh. Bouchard [10] developed a crack growth method that includes a remeshing technique that optimizes elements size and quantity but the geometry is completely remeshed in each step. The same method has been proved for several linear elastic and elastic–plastic examples in [4]. Phongthanapanich [11] developed a method that completely reconstructs the mesh between refinement steps by using adaptive meshing and Delaunay triangulations. Khoei [7] has developed an adaptive method for crack propagation that includes analytical crack tip expression in the error estimation, optimizing the size of elements and remeshing the whole geometry. Meyer [12] has proposed a crack propagation method that combines an adaptive iterative solver, mesh refinement and mesh coarsening techniques, and optimization of the number of nodes. In exchange, Askes et al. [13] have discussed remeshing strategies, based on relocation of the nodes, for r-adaptive and h/r-adaptive analysis of crack propagation.

In this paper a new method for crack growth simulation in 2D LEFM solids is presented. This method uses a displacement extrapolation technique for calculating K_I and K_{II} and the maximum circumferential stress criteria to calculate crack propagation angle. A Lepp–Delaunay based mesh generation and refinement algorithms are used. This allows the local refinement of the mesh, preserves mesh quality, and generates a smooth transition between elements close to the crack tip and larger elements far from this zone. The whole process is performed by using a Lepp based integrated algorithm, thus there is neither need of a posteriori mesh improvement routines, nor pre processing routines such as quad-tree techniques. The method proposed has the advantage of using local remeshing techniques but in a simpler and more efficient way than previous methods, so a better performance is expected. A quantitative analysis and comparison with other methods reported in the literature is an extensive and complex work and it is out of the scope of this paper. The Lepp based method is a robust and automatic tool that can be generalized both for adaptive finite element methods including mesh coarsening, and more general crack propagation studies. This can be also combined with several more general fracture mechanics formulations (e.g. [7,11,12]).

2. Numerical method for SIF calculation

2.1. Stress intensity factors

The relevant parameters in Linear Elastic Fracture Mechanics are the Stress Intensify Factors (SIFs) which can be calculated by using either a J integral based equivalent domain integral (EDI) [14], a virtual crack closure method (VCCM) [15], a virtual crack extension method [16] or a displacement extrapolation method [17]. For a comparison of these methods see [8]. In this paper a displacement extrapolation method which obtains the SIF's values directly from finite element nodal displacements is used. In polar coordinates (Fig. 1), the following analytical expression for displacements from theoretical fracture mechanics [18] is used:

$$\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \frac{K_I}{2\mu}\sqrt{\frac{r}{2\pi}}\begin{Bmatrix} \cos(\theta/2)[\kappa - 1 + 2\sin^2(\theta/2)] \\ \sin(\theta/2)[\kappa + 1 - 2\cos^2(\theta/2)] \end{Bmatrix} \tag{1}$$

where $\mu = \frac{E}{2(1+\nu)}$ is the shear modulus, $\kappa = \frac{3-\nu}{4-\nu}$ for plane stress and $\kappa = 3 - 4\nu$ for plane strain. This expression can be simplified, for $\theta = \pi$ as follows:

$$K_I = \lim_{r\to 0} \frac{2\mu}{\kappa + 1}\sqrt{\frac{2\pi}{r}}u_2 \tag{2}$$

In order to obtain a better approximation of the field near crack tip, special quarter point finite elements are used [19] where the mid-side node of the element in the crack tip is moved to 1/4 of the length of the element, as shown in Fig. 1.

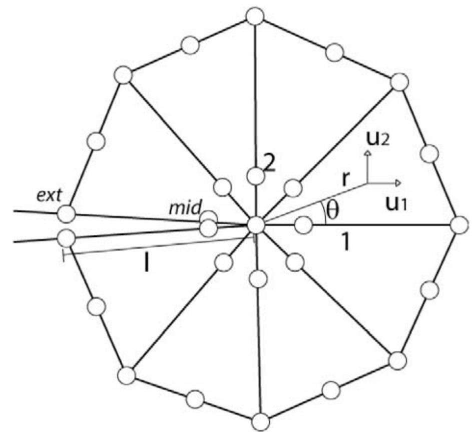


Fig. 1. Special elements used for displacement extrapolation method. The mid-side node (mid) is moved one quarter from its original position in order to modify shape functions.

According to Eq. (2), when this method is used with finite elements, K_I is related with the node displacement as follows:

$$K_I = \frac{2\mu}{\kappa + 1}\sqrt{\frac{2\pi}{l}}(4v_{mid} - v_{ext}) \tag{3}$$

where v_{mid} is the displacement of the mid-side node in the y-direction, and v_{ext} is the displacement of the node opposed to the crack tip in the y-direction, being the crack aligned with the x-direction, as shown in Fig. 1. Analogously, for K_{II}

$$K_{II} = \frac{2\mu}{\kappa - 1}\sqrt{\frac{2\pi}{l}}(4u_{mid} - u_{ext}) \tag{4}$$

where u corresponds to the displacement in the x-direction.

2.2. Angle of propagation

Under complex load conditions, crack growth does not occur in a straight direction, but follows a complex curved crack path, where the change of direction is determined by the relation between K_I and K_{II}. To calculate the instantaneous change of direction of the crack kink, the most common methods used are the maximum circumferential stress [20], the maximum potential energy release rate [21] and the minimum strain energy density [22]. In this paper the maximum circumferential stress method is used.

In polar coordinates, the stress field (2D) near crack tip is written as follows [18]:

$$\sigma_{\theta\theta} = \frac{1}{\sqrt{2\pi r}}\cos\left(\frac{\theta}{2}\right)\left(K_I\cos^2\left(\frac{\theta}{2}\right) - \frac{3}{2}K_{II}\sin(\theta)\right) \tag{5}$$

$$\tau_{r\theta} = \frac{2}{\sqrt{2\pi r}}\cos\left(\frac{\theta}{2}\right)[K_I\sin(\theta) + K_{II}(3\cos(\theta) - 1)] \tag{6}$$

The crack propagation direction is obtained by imposing the condition [20] $\partial\sigma_{\theta\theta}/\partial\theta = 0$ or $\tau_{r\theta} = 0$ which produces the following equation:

$$K_I\sin\theta + K_{II}(3\cos\theta - 1) = 0 \tag{7}$$

where θ is measured with respect to the crack face. This equation can be solved numerically by using the Newthon–Raphson method.

3. Lepp–Delaunay methods for quality triangulation

3.1. Lepp (Delaunay terminal edge) midpoint method

The algorithm was designed to improve the smallest angles in a Delaunay triangulation but can be used in general, for refining (and improving) any target triangle in the mesh. For each target triangle t_0, the algorithm selects a point M which is midpoint of a Delaunay terminal edge (a longest edge for both triangles that share this edge) which is then Delaunay inserted in the mesh. This method uses the longest edge propagating path Lepp(t_0) associated to the target triangle t_0 to determine a terminal edge in the current mesh. The longest edge propagating path associated to t_0, Lepp$(t_0) = \{t_0, t_1, \ldots, t_{n-1}, t_n\}$, is a sequence of neighbor increasing triangles in the mesh (t_i is neighbor of t_{i-1} by the longest edge of t_{i-1}, and longest edge of t_i is greater than the longest edge of t_{i-1}) and where either t_{n-1}, t_n share a terminal edge, or t_n has a boundary/constrained terminal edge. For an illustration of the Lepp(t_0) see Fig. 2a. For an illustration of the ideas of the algorithm, see Fig. 2b, where the processing of t_0 produces the Delaunay insertion of midpoint M of terminal edge AB. In this particular case, the insertion of M destroys t_0 and the process stops. In the general case, the processing of t_0 is repeated until t_0 is destroyed. The algorithm was introduced in a rather intuitive basis as a generalization of previous longest edge algorithms in [23,24] and studied in [25,26]. Given an angle tolerance θ_{tol}, the algorithm can be simply described as follows: iteratively, each bad triangle t_{bad} with smallest angle less than θ_{tol} in the current triangulation is eliminated by finding Lepp(t_{bad}), a pair of terminal triangles t_1, t_2, and associated terminal edge l. If non-constrained edges are involved, then the midpoint M of l is Delaunay inserted in the mesh. Otherwise a constrained point insertion criterion is used. The process is repeated until t_{bad} is destroyed in the mesh, and the algorithm finishes when the minimum angle in the mesh is greater than or equal to an angle tolerance θ_{tol}.

Fig. 3 shows the use of Lepp–Delaunay terminal edge algorithm to perform a localized refinement near a corner in a concave geometry.

The algorithm is given below:

Lepp midpoint algorithm
Input = a CDT, τ, and angle tolerance θ_{tol}
Find S_{bad} = the set of bad triangles with respect to θ_{tol}
for each t in S_{bad} **do**
  while t remains in τ **do**
    Find Lepp (t_{bad}), terminal triangles t_1, t_2 and terminal edge l. Triangle t_2 can be null for boundary l.
    Select Point (P, t_1, t_2, l)
    Perform constrained Delaunay insertion of P into τ
    Update S_{bad}
  end while
end for
Select Point (P, t_{term1}, t_{term2}, l_{term})
if (second longest edge of t_{term1} is not constrained and second longest edge of t_{term2} is not constrained) or l_{term} is constrained **then**
  Select P = midpoint of l_{term} and return
else
  for j = 1, 2 **do**
    if t_{termj} is not null and has constrained second longest edge l^* **then**
      Select P = midpoint of l^* and return
    end if
  end for
end if

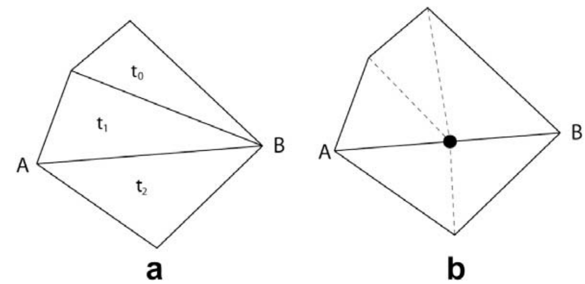


Fig. 2. (a) For target triangle t_0, Lepp$(t_0) = \{t_0, t_1, t_2\}$. (b) The processing of t_0 implies the selection and insertion of M, midpoint of terminal edge AB.

3.2. Lepp-centroid algorithm

In order to improve the performance of the previous Lepp midpoint algorithm, in [27,28] a new Lepp-centroid algorithm for quality triangulation is introduced. For any general (planar straight line graph) input data, and a quality threshold angle θ, the algorithm constructs constrained Delaunay triangulations that have all angles at least θ as follows: for every bad triangle t with smallest angle less than θ, a Lepp-search is used to find an associated convex terminal quadrilateral formed by the union of two terminal triangles which share a local longest edge (terminal edge) in the mesh. The centroid of this terminal quad is computed and Delaunay inserted in the mesh. The process is repeated until the triangle t is destroyed in the mesh. In [27] the new Lepp centroid algorithm and geometrical results which explain the better performance of the Lepp centroid method, are discussed. Also an empirical study that compares the behavior of Lepp-centroid and Lepp-midpoint methods is presented. The centroid method computes significantly smaller triangulation than the terminal edge midpoint variant, produces globally better triangulations, and terminates for higher threshold angle θ (up to 36°). It is also shown that the Lepp centroid method behaves better than the off-center algorithm for $\theta > 25°$.

4. Mesh generation for cracked geometries

Initial mesh generation is a key step in fracture mechanics problems. 2D cracked geometries are complex to mesh because a very fine mesh is needed near the crack tip while a coarse mesh up to two orders of magnitude larger suffices far from crack tip. Also the mesh methodology must be able to correctly generate elements in crack faces, because a crack is a 1D entity where the two faces are geometrically coincident but topologically different. To deal with these issues several solutions have been proposed. Bouchard [10] constructs a Delaunay triangulation to generate the initial mesh, followed by a mesh improvement strategy. The crack is obtained by nodal relaxation. Phongthonapanich et al. [11] use an adaptive Delaunay based mesh generator, where the shape and size of new elements are controlled by coefficients that define triangle creation or destruction, followed by a Laplacian smoothing. Miranda et al. [6] combine a quad-tree method for defining initial node position, a heuristic advancing front method for generating an initial mesh, and uses a mesh improvement technique for eliminating bad shape elements.

4.1. Lepp–Delaunay generation of the initial mesh

In this paper a simple and effective Lepp–Delaunay method, able to mesh any complex geometry with cracks by producing both good shape triangular elements in the whole mesh, and a smooth transition between small and large elements near crack tip and the rest of the geometry, is presented. This is performed in one

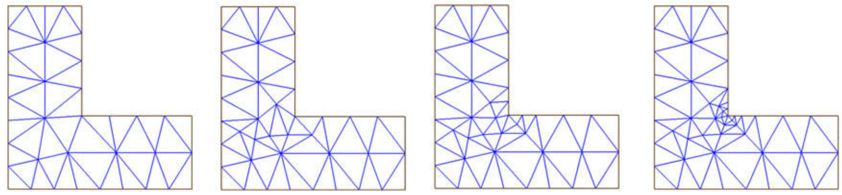


Fig. 3. Localized refinement near a corner over an existing mesh using Lepp–Delaunay terminal edge algorithm.

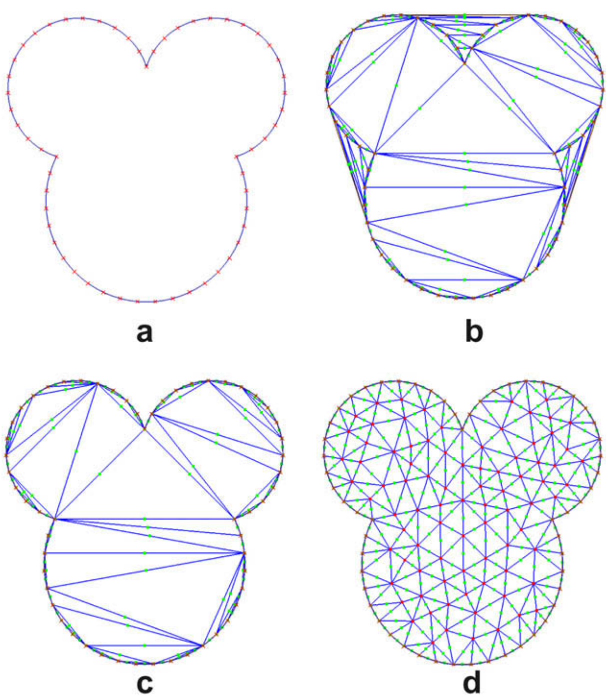


Fig. 4. Generation of the initial mesh. (a) A border discretization is introduced. (b) Nodes are Delaunay inserted producing an initial convex mesh. (c) Elements that are outside the geometry are eliminated. (d) Using Lepp–Delaunay algorithm, the triangulation is improved.

step, without requiring any pre-processing and/or post-processing steps.

The initial mesh is generated as follows:

- An initial discretization of the border is produced (Fig. 4a).
- Nodes are Delaunay inserted in the mesh which produces an initial convex triangulation that in general does not respect the geometry (Fig. 4b).
- External elements are eliminated, which produces a valid initial mesh (Fig. 4c).
- The triangulation is improved by using a Lepp–Delaunay algorithm until a user defined quality criteria is reached (Fig. 4d). A study on the geometrical and convergence properties of this method can be seen in [25].

4.2. Lepp–Delaunay technique for crack propagation

Since the refinement/improvement Lepp–Delaunay algorithms can be applied to any existing mesh, posterior refinements near the crack tip are performed by using the same Lepp based refinement technique. Once the initial mesh is ready, the next step consists in generating especial symmetrical elements in the crack tip, inserting and displacing nodes and then refining the mesh using the Lepp–Delaunay algorithm. This process generates a mesh with well shaped elements and smooth transition between small and large elements.

Fig. 5 illustrates the methodology for generating special elements in crack tip. First, the positions of external nodes associated to special elements are fixed (Fig. 5a). These are positioned at a distance r of the crack tip, where r is the length of the free edge of one of the crack elements. Then if there exists a node close to this position, the old node is moved there, otherwise, a new node is inserted (Fig. 5b). Since this process can generate bad shaped elements, the Lepp–Delaunay algorithm is used to improve the mesh quality near crack tip (Fig. 5c).

5. Crack increment

In this section the quasi-static propagation of a crack, and the calculation of the crack path is discussed. To this end, linear elastic problems are considered, but the methodology can be extended to more complex models such as elastic–plastic or dynamic fracture mechanics. Crack growth is computed discretly by using a finite element model for each new crack length step. Crack parameters and crack propagation direction are calculated as discussed in Section 2. Crack is modeled as a 1D entity formed by free surfaces that are coincident geometrically. For each crack increment, a node is inserted in the new position of the crack tip and a new free surface

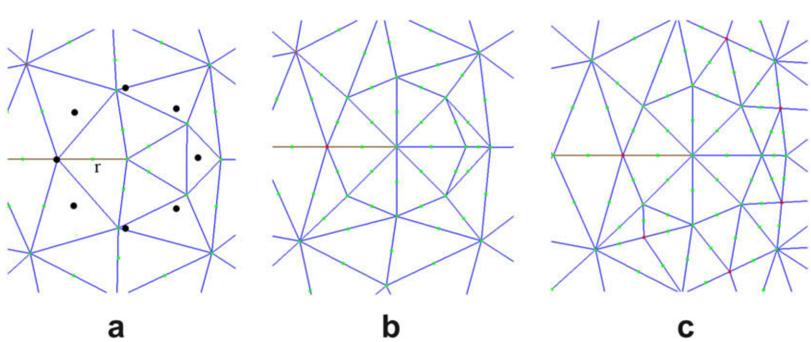


Fig. 5. Methodology for creating special elements in crack tip. (a) Node position is fixed for special elements. (b) If a previous node is close to these positions, thus is moved; if not, a new node is inserted. (c) Lepp–Delaunay algorithm is used in order to eliminate bad shape triangles generated due node displacement/insertion step.

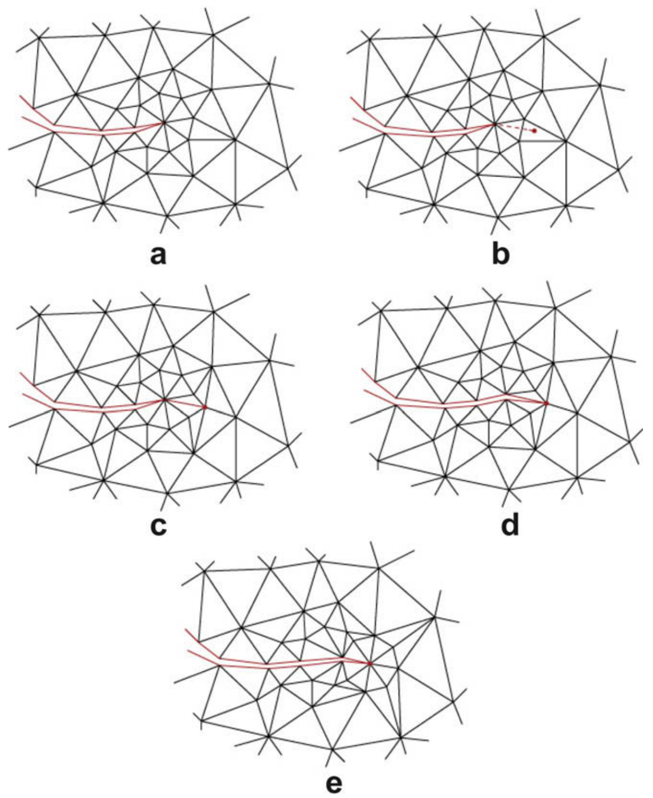


Fig. 6. Crack growth methodology.(a) Initial position of crack tip. (b) A new crack tip position is calculated. (c) A new node is inserted in the new position, and a new edge is generated between old and new crack tip. (d) Nodal relaxation is performed and a new free edge is obtained. (e) The special elements generation methodology is performed over the new crack tip.

is formed from the new edge that joins new and old crack tip, as shown in Fig. 6.

On the contrary to [6,8], elements near crack tip are not eliminated, but new special elements are generated by moving or inserting nodes near crack tip (Fig. 5). Then a Lepp–Delaunay improvement algorithm is used over those elements to generate a smooth transition between crack tip elements and larger elements, as shown in Fig. 7. Note that this is a very local work that leaves unaffected most of the previous mesh.

5.1. Bad shape triangle elimination

Since the generation of new elements can produce a pattern of bad shape elements due to the displacement of nodes, a bad shape triangle elimination methodology has been also implemented. This eliminates pair of triangles that have a common edge that is significantly smaller than other ones, as shown in Fig. 8.

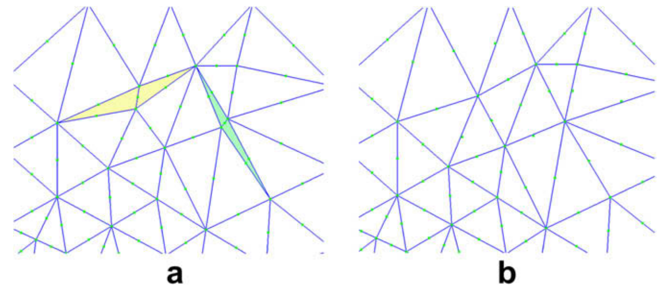


Fig. 8. (a) Bad shape elements can appear as pair of triangles that have a common edge that is significantly smaller than other sides. (b) Bad triangles are eliminated.

6. Results

In order to test the behavior and performance of the crack propagation method, several test cases have been solved and compared with previous numerical and experimental solutions. For this purpose a software with a graphical interface has been developed, based on a previous implementation of Lepp–Delaunay algorithms presented in [29]. All methods discussed in this paper are integrated in the software, including a user friendly interface which allows the generation of finite element models that can be solved by Abaqus. This software is able to read output files from Abaqus to perform Stress Intensify Factors computation and mesh modification to simulate crack propagation steps.

6.1. Center cracked beam with a hole

This case corresponds to a single notched beam with a hole, loaded in two points and supported in two points, as shown in Fig. 9. This corresponds to a plane strain problem which has been

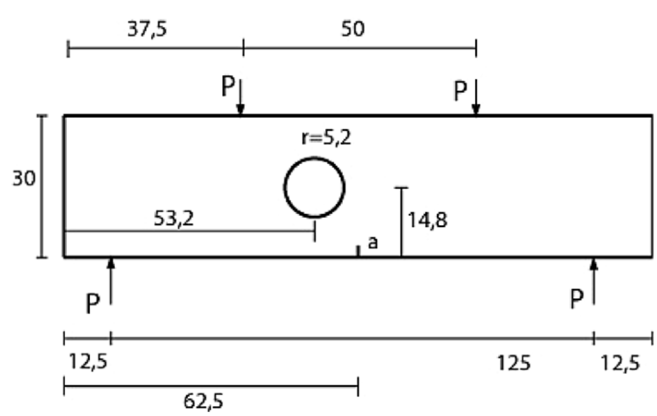


Fig. 9. Beam with a center crack, loaded in two points and supported in two points.

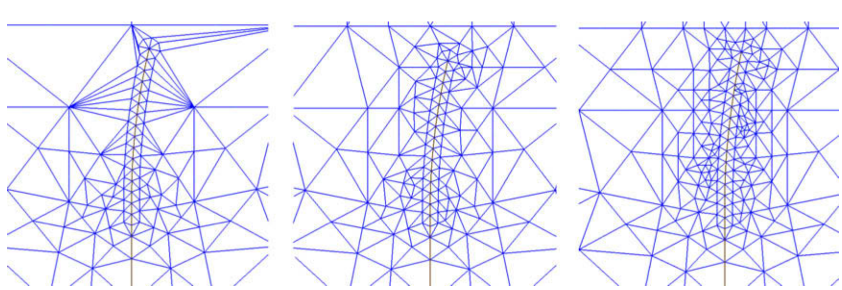


Fig. 7. (a) Crack remeshing without using Lepp–Delaunay algorithm for mesh improvement. (b) Application of Lepp–Delauany algorithm with a minimum tolerance of 15° for each propagation step. (c) Application of Lepp–Delaunay algorithm with a 25° angle tolerance.

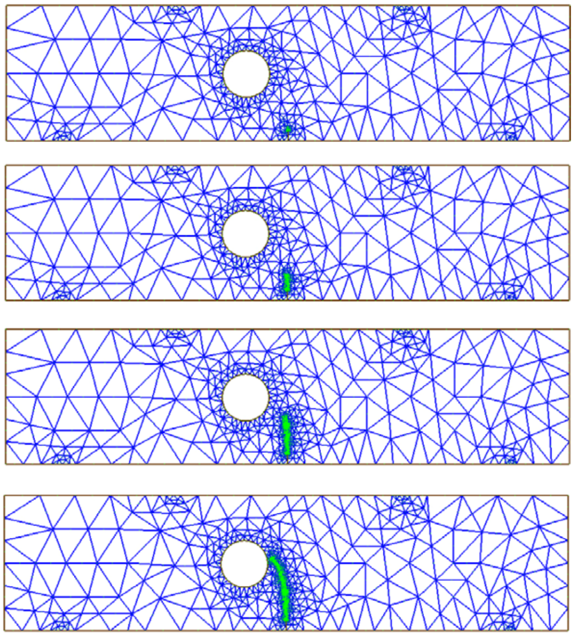


Fig. 10. From top to bottom, meshes associated to steps 0, 20, 50 and 92.

solved numerically in [6]. The physical parameters of the problem are $E = 205$ (GPa), $\nu = 0.3$, $P = 100$ (N) and $a = 2.5$ (mm). Results presented in this paper are compared with numerical results presented in [6]. In this case, a local Lepp–Delaunay algorithm has been applied for each propagation step. Triangles with minimal angle less than 15° are processed to be improved. The size of crack tip elements is $l = 0.625$ (mm) ($l/a = 1/4$). Bad shape triangle elimination methodology is also applied for each crack growth step.

This problem required 92 steps to be solved. Fig. 10 shows the meshes constructed for the time steps 0, 20, 50 and 92, whose associated number of nodes are 1278, 1738, 2518 and 3448, respectively. Fig. 11 shows a detail of the mesh for the step 92. Fig. 12 shows a comparison with the experimental results published in [6], where good agreement between numerical and experimental results can be seen.

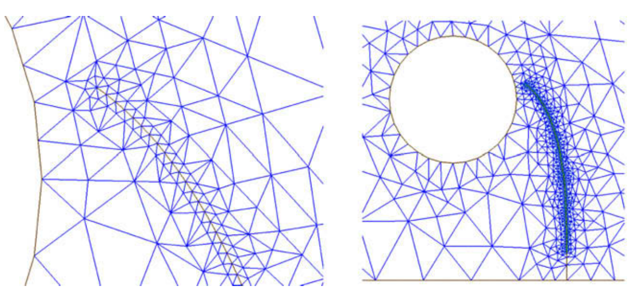


Fig. 11. Details of the mesh in step 92.

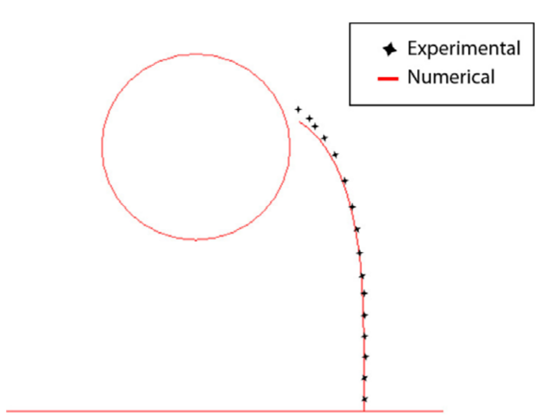
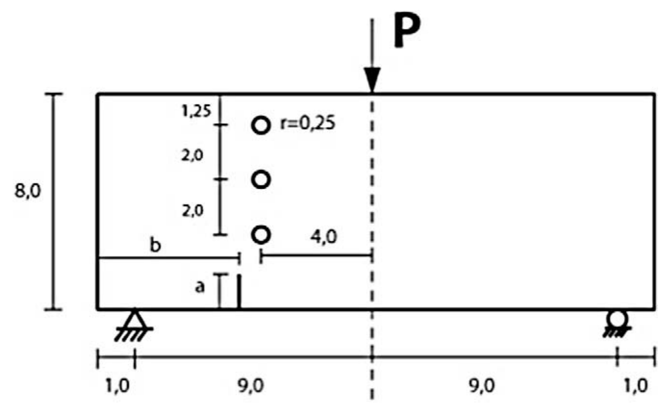


Fig. 12. Comparison of the crack paths between numerical results and experimental results.

6.2. Cracked beams with three holes

This example corresponds to a cracked beam supported in two points and with a load in the center as illustrated in Fig. 13. The beam has three holes that change the trajectory of the crack. This problem has experimental results for polymethylmethacrylate (PMMA) beams and has been used as a numerical test case in [8,11]. Two test cases (*I* and *II*) have been considered, for different values of a and b which produce very different crack paths. For



CASO	a	b
I	1,5	5,0
II	1,0	4,0

Fig. 13. Beam with a crack and a center load and supported in two points. The beam has three holes to create complex crack paths.

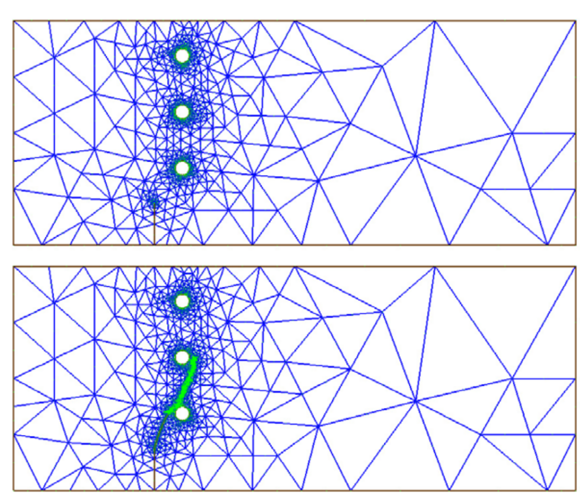


Fig. 14. Initial and final mesh after 110 steps for case *I* of a cracked beam with three holes.

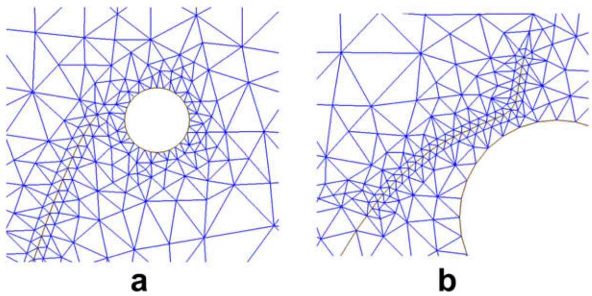


Fig. 15. Detail of the mesh after 15 steps for case *I* of a cracked beam with three holes.

both cases, the parameters of the problem are $P = 1$ (lbf) (4.45 (N)), $E = 29 \times 10^6$ (PSI) (199.95×10^6 (kPa)) and $\nu = 0.3$.

6.2.1. Case I

The Fig. 14 shows the initial and the final mesh after 110 propagation steps. The initial mesh has 2226 nodes and 1042 elements, while the final mesh has 4496 nodes and 2066 elements. For mesh modification, Lepp–Delaunay algorithm was applied with a minimum angle tolerance of 15°. At the first steps, the size of crack tips element are $l/a = 1/16$, where a is the initial length of the crack and l is the characteristic length of the crack tip element. After 15 steps, crack tip is near the first hole and a big error is obtained. This is because crack tip elements are the same order of magnitude of size of elements on the border of the hole. Considering this fact, the size of crack tip elements is reduced to $l/a = 1/64$ and the simulation continues (Fig. 15). This allows satisfactory crack path computing until the end. A detail of the mesh for the time steps 60 and 100 is shown in Fig. 16, where a smooth transition is obtained from crack tip elements to far elements, and from elements on the border of the crack faces to far elements. Finally, crack path and experimental results for case *I* are shown in Fig. 17.

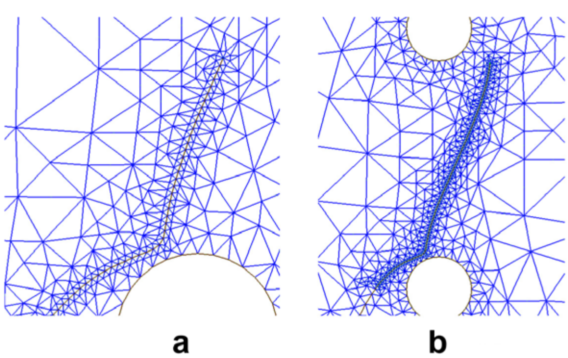


Fig. 16. Details of the mesh for case *I* of a cracked beam with three holes (a) for time step 60 and (b) for time step 100.

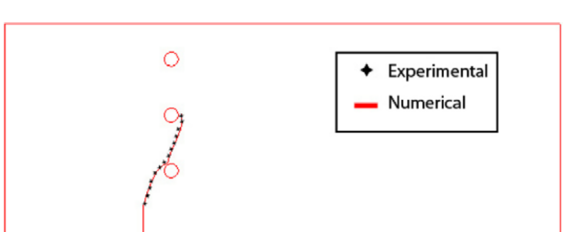


Fig. 17. Comparison of the crack paths between numerical results and experimental results for case *I* of a cracked beam with three holes.

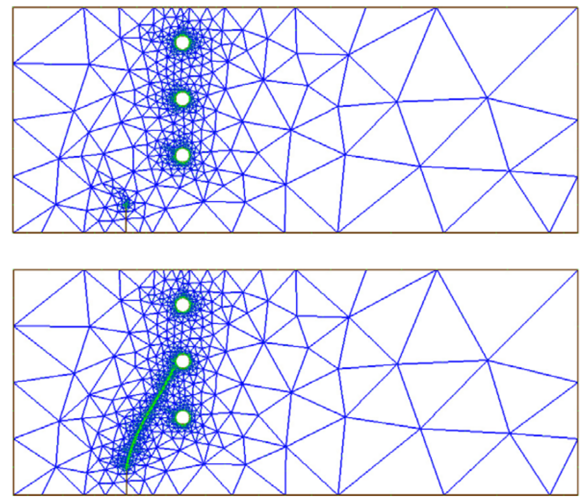


Fig. 18. Initial and final mesh after 72 steps for case *II* of a cracked beam with three holes.

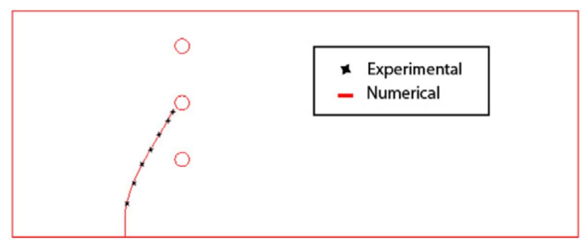


Fig. 19. Comparison of the crack paths between numerical results and experimental results for case *II* of a cracked beam with three holes.

6.2.2. Case II

Fig. 18 shows initial and final mesh after 72 steps of crack propagation. Initial mesh has 1950 nodes and 906 elements, while final mesh has 3332 nodes and 1534 elements. Results are similar to case *I* in terms of size transition and affected zone remeshing. Fig. 19 shows a comparison between numerical and experimental results for this problem.

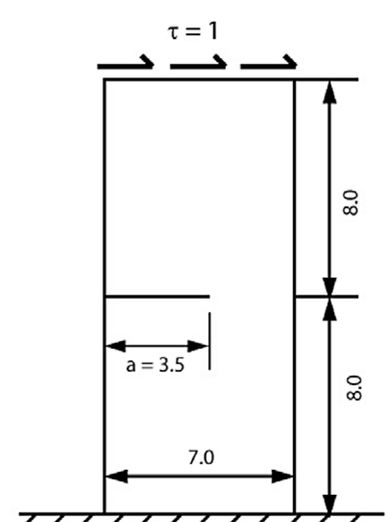


Fig. 20. Single notched plate, fixed at the bottom and constrained to far-field shear stress along the top edge.

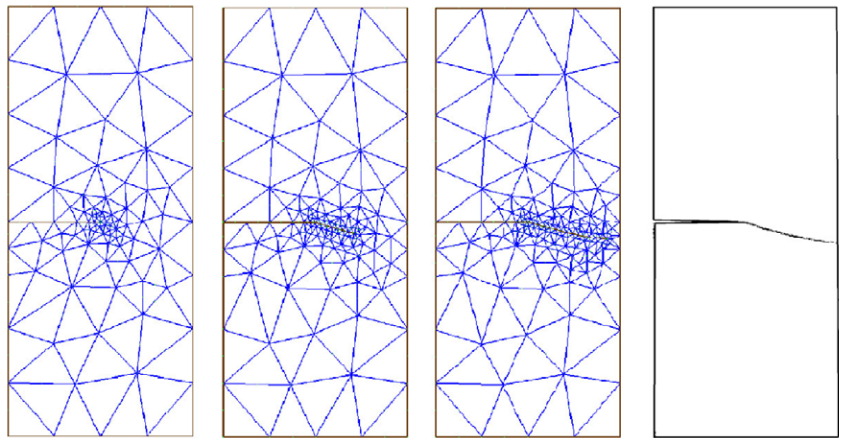


Fig. 21. From left to right, meshes associated to steps 0, 7 and 14 obtained with the implemented software, and final result obtained in reference [11].

6.3. Single edge cracked plate under mixed mode loading

This case corresponds to a single notched plate, fixed at the bottom and constrained to far-field shear stress along the top edge. Fig. 20 shows the geometry and initial conditions of the problem. The physical parameters are $\tau = 1$ unit, $E = 30 \times 10^6$ units and $\nu = 0.25$. This case was solved in [11] and is included here in order to demonstrate the robustness of the method and the range of problems, the developed software is able to deal with. The problem is solved in 14 steps. Fig. 21 shows the meshes constructed for the time steps 0, 7 and 14, whose associated number of nodes are 357, 468 and 618, respectively. Good agreement is obtained for the crack path with the results presented in [11].

7. Conclusions

A flexible and stable 2D crack propagation method based on Lepp–Delaunay mesh refinement/improvement algorithm was presented. This combines a finite element method for obtaining displacement values at the nodes with a crack advance technique which allows to obtain fracture parameters and crack propagation direction. The method presented uses a local mesh modification technique that selects and inserts new nodes, by using a Lepp–Delaunay algorithm, which in turn produces new elements with geometric quality analogous to those of the initial mesh. The algorithm can be used either to generate an initial mesh, or to refine/improve an existing mesh, without reconstructing the whole mesh, and without requiring mesh improvement post-processing. This characteristic is particularly useful for crack growth numerical methodologies, where the mesh needs to be modified in each simulation step, with meshing time consumption being relevant in global performance. The algorithm produces a new mesh by modifying the mesh of the previous step in a minimal time in comparison with alternative published methods. Meshing algorithm works locally, maintaining mesh element quality and generating a smooth transition between elements near crack tip and elements in the rest of the mesh. Three test problems with known numerical and/or experimental solutions were run in order to test the method, obtaining good results in terms of crack path prediction and mesh quality.

It is worth noting that the method presented has a big potential in crack propagation analysis for large and complex geometries since its scales very well because this implementation guarantees robust mesh generation and very little time consumption between propagation steps, independently of the size or complexity of the geometry, while alternative methods raise the time consumption and/or loose reliability in the mesh generation steps. The method can also be generalized to another fracture mechanics problems, including elastic plastic, dynamic fracture mechanics or fatigue crack growth. It is also desirable to generalize this method for adaptive fracture mechanics models, and to 3D problems, for which Lepp–Delaunay algorithms implementation are in progress.

Acknowledgement

We are grateful to the referees whose comments contributed to the improvement of this paper.

References

[1] Aliabadi MH. Boundary element formulations in fracture mechanics. Appl Mech Rev 1997;50:83–96.
[2] Belytschko T, Lu YY, Gu L. Element free Galerkin methods. Int J Numer Methods Eng 1994;37:229–56.
[3] Möes N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. Int J Numer Methods Eng 1999:131–50.
[4] Bouchard PO, Bay F, Chastel Y. Numerical modelling of crack propagation: automatic remeshing and comparison of different criteria. Comput Methods Appl Mech Eng 2003;192:3887–908.
[5] Réthoré J, Gravouil A, Combescure A. An energy-conserving scheme for dynamic crack growth using the eXtended finite element method. Int J Numer Methods Eng 2005;63:631–59.
[6] Miranda ACO, Meggiolaro MA, Castro JTP, Martha LF, Bittencourt TN. Fatigue life and crack predictions in generic 2D structural components. Eng Fract Mech 2003;70:1259–79.
[7] Khoei AR, Azadia H, Moslemia H. Modeling of crack propagation via an automatic adaptive mesh refinement based on modified superconvergent patch recovery technique. Eng Fract Mech 2008;75:2921–45.
[8] Bittencourt TN, Wawrzynek PA, Ingraffea AR. Quasi-automatic simulation of crack propagation for 2D LEFM problems. Eng Fract Mech 1996;55:321–34.
[9] Calvacante Neto JB, Wawrzynek PA, Carvalho TM, MArtha LF, Ingraffea AR. An algorithm for three-dimensional mesh generation for arbitrary regions with cracks. Eng Comput 2001;17:75–91.
[10] Bouchard PO, Bay F, Chastel Y, Tovena I. Crack propagation modelling using an advanced remeshing technique. Comput Methods Appl Mechan Eng 2000;189:723–42.
[11] Phongthanapanich S, Dechaumphai P. Adaptive Delaunay triangulation with object-oriented programming for crack propagation analysis. Finite Elem Anal Des 2004;40:1753–71.
[12] Meyer A, Rabold F, Scherzer M. Efficient finite element simulation of crack propagation using adaptive iterative solvers. Commun Numer Methods Eng 2006;22:93–108.
[13] Askes H, Sluys LJ, De Jong BBC. Remeshing techniques for r-adaptive and combined h/r-adaptive analysis with application to 2D/3D crack propagation. Int J Struct Eng Mech 2001;12:475–90.
[14] Shih CF, Asaro RJ. Elastic–plastic analysis of cracks on bimaterial interfaces: part I – small scale yielding. J Appl Mech 1988;55:299–316.
[15] Rybicki EF, Kanninen MF. Finite element calculation of stress intensity factors by a modified crack closure integral. Eng Fract Mech 1977;9:931–8.
[16] Parks DM. A stiffness derivative finite element technique for determination of crack tip stress intensity factors. Int J Fract 1974;10:487–502.
[17] Chan SK, Tuba IS, Wilson WK. On the finite element method in linear fracture mechanics. Eng Fract Mech 1970;2:1–17.

[18] Kanninen MF, Popelar CH. Advanced fracture mechanics. Oxford University Press; 1985.
[19] Barsoum RS. Triangular quarter-point elements as elastic and perfectly-plastic crack tip elements. Int J Numer Methods Eng 1977;11:85–98.
[20] Erdogan F, Sih GC. On the crack extension in plates under plane loading and transverse shear. J Basic Eng 1963;85:519–27.
[21] Hussain MA, Pu SL, Underwood JH. Strain energy release rate for a crack under combined mode I and mode II. Fract Anal 1974;560:2–28. ASTM STP.
[22] Sih GC. Strain-energy-density factor applied to mixed-mode fracture problems. Int J Fract Mech 1974;10:305–21.
[23] Rivara MC. New Longest-Edge algorithms for the refinement and/or improvement of unstructured triangulations. Int J Numer Methods Eng 1997;40:3313–24.
[24] Rivara MC. New mathematical tools and techniques for the refinement and/or improvement of unstructured triangulations. In: Proceedings 5th international meshing roundtable, Pittsburgh; 1996. p. 77–86.
[25] Rivara MC, Hitschfeld N, Simpson RB. Terminal edges Delaunay (small angle based) algorithm for the quality triangulation problem. Comput Aided Des 2001;33:263–77.
[26] Rivara MC, Palma M. New Lepp algorithms for quality polygon and volume triangulation: implementation issues and practical behavior. Trends Unstruct Mesh Generat 1997;220:1–8.
[27] Rivara MC, Calderón C. Lepp Terminal centroid method for quality triangulation: a study on a new algorithm. GMP 2008, LNCS 4975, Springer Verlag; 2008. p. 215–30.
[28] Rivara, MC, Calderon, C. Lepp terminal centroid method for quality triangulation. Comput Aided Des. doi:10.1016/j.cad.2008.11.004.
[29] Calderon C, Rivara MC. Herramienta computacional de apoyo a la investigacion de algoritmos de triangulaciones. In: Proceedings XII Jornadas Chilenas de Computacion, Encuentro Chileno de Computacion, Universidad de Tarapaca, Arica; 2004.