Finite Element Model of Fracture Formation on Growing Surfaces

Pavol Federl and Przemyslaw Prusinkiewicz

University of Calgary, Alberta, Canada

Abstract. We present a model of fracture formation on surfaces of bi-layered materials. The model makes it possible to synthesize patterns of fractures induced by growth or shrinkage of one layer with respect to another. We use the finite element methods (FEM) to obtain numerical solutions. This paper improves the standard FEM with techniques needed to efficiently capture growth and fractures.

1 Introduction and Background

We consider fracture pattern formation on differentially growing, bi-layered surfaces. The top layer, called the *material layer*, is assumed to grow slower than the bottom *background layer*. Through the attachment of the material layer to the background layer, such differential growth produces increasing stresses in the material layer. Eventually, the stresses exceed the material's *threshold stress*, which leads to formation of a fracture. As this process continues, a pattern of fractures develops. Here we present a method for simulating this pattern formation.

In our method, fracture mechanics [1] is combined with the framework of the finite element method (FEM) to form computer simulations that can predict whether and how a material will fail. The FEM is a numerical technique for solving partial differential equations [10], widely used in mechanical engineering to analyze stresses in materials under load [10]. Given some initial configuration of a structure, coupled with boundary conditions and a set of external forces, the FEM determines the shape of the deformed structure. The deformed shape represents the *equilibrium state*, since the sum of internal and external forces at any point in the structure is zero. Our method is most closely related to that of O'Brien and Hodgins [3], in that it treats fracture formation in the context of continuum mechanics and the finite element method. In contrast to their work, however, we are interested in patterns of fractures, rather than the breaking of brittle materials.

We consider formation of crack patterns in bark as an example of pattern formation due to *expansion* of one material layer with respect to another, and formation of crack patterns in mud as an example of pattern formation due to *shrinking* of one layer with respect to another. Tree bark consists of dead conductive tissue, phloem, which is expanded by the radial growth of cambium inside the trunk [6]. As a result of this expansion, the bark stretches until it reaches its limit of deformation and cracks. In our simulation we use a simplified,

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3037, pp. 138–145, 2004.

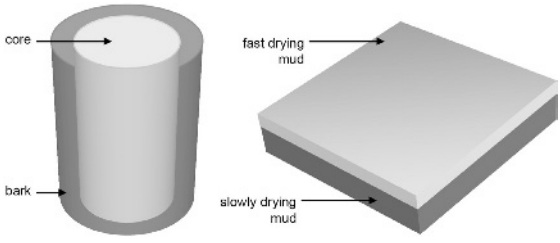

Fig. 1. The two-layered models of bark and drying mud.

two-layer model of a growing tree trunk (Fig. 1, left). The inside core grows radially and does not break, while the outer layer, which represents the bark, may break.

As water evaporates from mud, the mud shrinks. Since water evaporates faster from the layers closest to the surface, different layers shrink at different rates with respect to each other. This non-uniform shrinkage of the various layers leads to material stress and, consequently, to the formation of cracks. We model drying mud using two layers (Fig. 1, right). The background layer is assumed to be static, representing either mud that dries very slowly or the surface on which the drying mud rests. The material layer represents the drying mud and is attached to the background layer.

We use linear elastic fracture mechanics [1], and approximate the stress field near a crack tip using the theory of linear elasticity [8]. A fracture occurs where the *maximum principal stress* exceeds material's *threshold stress* (*maximum principal stress criterion* [8]). The direction of the newly formed fracture is perpendicular to the direction of this maximum principal stress. We also use the maximum principal stress criterion to establish the propagation direction of an existing fracture. We terminate the propagation of a fracture using the Griffith energy approach [1]. It states that a fracture propagates as long as the potential energy released by the fracture exceeds the energy required to form the fracture. An overview of our algorithm is given in Fig. 2.

2 Fracture Simulation Algorithm

Discretization. We model the material layer as a single layer of three dimensional 6-node wedge elements (prisms) (Fig. 3) [2]. The material layer in which the cracks are formed is attached to the background layer at *attachment points*, which are the bottom three nodes of each wedge element. The attachment points are randomly placed on the plane or a cylindrical surface, then repelled using a particle repelling algorithm [7] to obtain more uniform distribution. The resulting points are connected into a mesh using Delaunay triangulation.

Growth modeling. The growth of the background layer is modeled by adjusting the positions at which the wedge elements are attached to it. The trajectory of each attachment point is defined by its initial position and its velocity vector.

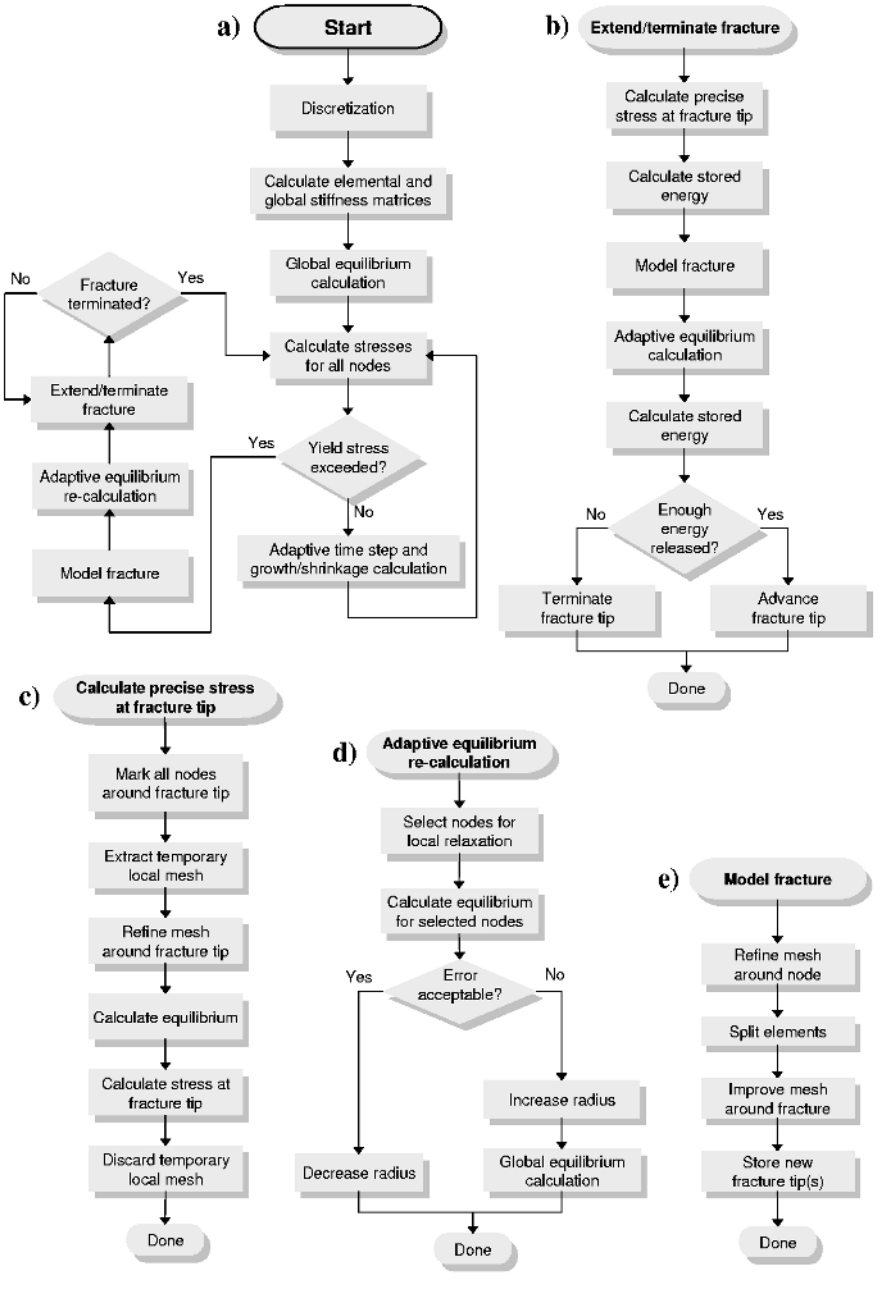

Fig. 2. Structure of our fracture simulation algorithm.

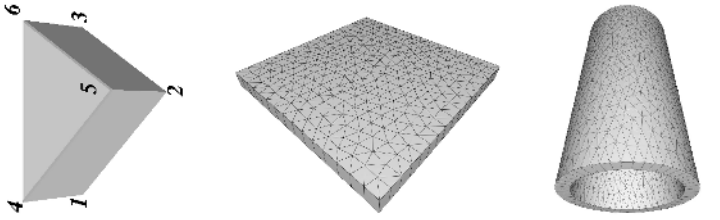

Fig. 3. The wedge element and the resulting representations of flat and cylindrical surfaces.

We consider both *isotropic* and *anisotropic* growth [6]. Shrinkage of the material layer is simulated by adjusting the reference shapes of the wedge elements.

Global stiffness matrix calculation. We calculate the equilibrium state of the mesh using the finite element method [10]. First, we calculate the elemental stiffness matrices K_e using 9-point Gaussian quadrature for each prism element, as described by Keeve et al. [2]. Next, we assemble the elemental stiffness matrices into the global stiffness matrix K_g, which represents the coefficients of a set of linear equations $K_g Q = F$. Here Q is the vector of nodal displacements and F is the vector of nodal forces.

Equilibrium calculation. At equilibrium, the total force acting on any free node is equal to zero. The calculation of the equilibrium is therefore performed by setting $F = 0$, imposing boundary conditions, and solving the resulting system of equations for Q. In our case, the boundary conditions consist of the known nodal displacements values of the fixed nodes, determined from the positions of the attachment points. We solve the resulting system of equations using the iterative conjugate gradient algorithm [4].

When a change is made to the geometry of the model, the equilibrium state of the model needs to be recalculated. Many of these changes, such as in fracture formation, mesh refinement, or node repositioning during mesh smoothing, are confined to small regions, and have negligible effect on more distant parts of the mesh. We take advantage of this locality by recalculating the equilibrium state adaptively, only in the regions of interest (*local relaxation*). These regions are detected by checking for large unbalanced nodal forces.

Modeling fracture behavior. Once the equilibrium state of the material layer is calculated, we compute the stress tensor at each node [11], and we use it to calculate the maximum principal stress s_1. If s_1 exceeds the threshold stress of the material, we mark the corresponding node n as a possible candidate for fracture initiation. In most cases there is only a single candidate. Having more than one candidate typically means that a too large a time step was used for simulating growth. We address this issue by advancing the simulation time with adaptive time step control (Fig. 2a).

Once a single fracture candidate node n has been identified, we extend the fracture at this node and adjust the finite element mesh accordingly. We use the

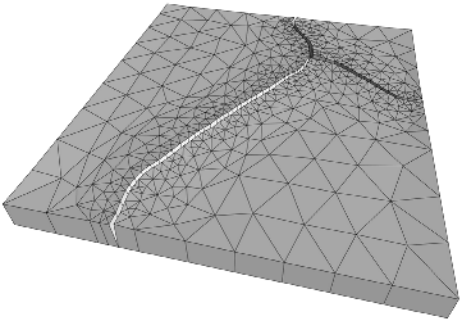

Fig. 4. The mesh is refined only around the fracture.

same procedure to both incorporate the onset of a new fracture and to propagate an existing fracture (Fig. 2e). The input to this procedure is the location of the fracture, specified by a fracture node n and the corresponding nodal stress tensor σ. The fracture plane p is determined from σ: its normal is the eigenvector of σ corresponding to the maximum principal stress s_1.

Modeling fracture extension. The first step consists of refining the elements sharing the fracture node n so that each element is smaller than a user-defined constant λ_{max}. The constant λ_{max} effectively denotes the maximum distance a fracture can extend before the nodal stress at its fracture tip is recalculated. Imposing the limit on the length of the fracture extension is important when the fractures turn rapidly.

We refine the elements with a version of the triangular mesh dynamic refinement algorithm proposed by Rivara and Inostroza [5]. This refinement step allows us to discretize the surface using a coarse global mesh, and subdivide it only where needed, leading to smaller memory requirements and faster simulations. An example of a mesh that has been dynamically refined around a fracture is shown in Fig. 4.

The next step is to create a new copy n' of the fracture node n. All elements that contain node n are then adjusted according to their locations with respect to the fracture plane p. The elements situated entirely on one side of the plane are assigned the original node n, while the elements on the other side are assigned the new copy n'. The remaining elements, sharing the node n, are split by the fracture plane. If a T-junction is formed by this process, the adjacent element is also subdivided to remove it.

When the fracture plane intersects an element close to one of its edges, a degenerate wedge may be formed as a result of splitting. The solution proposed by O'Brien and Hodgins [3] is not to allow degenerate elements to be created; this is accomplished by rotating the fracture plane by a small amount to align it with an edge in the mesh. This approach suffers from fracture directions being occasionally influenced by the geometry of the surface subdivision. We adopted a reverse approach: instead of snapping the fracture plane to a near parallel edge, we snap the edge to the fracture plane, as illustrated in Fig.5.

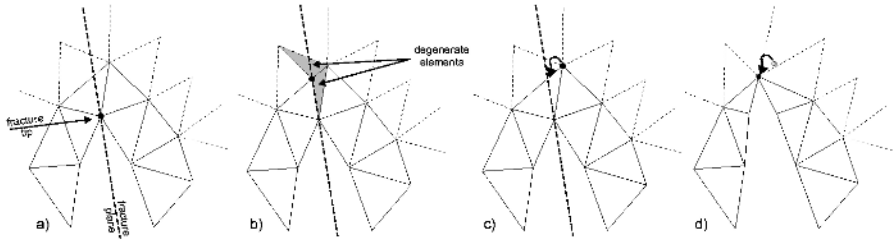

Fig. 5. Example of snapping a node to a fracture plane. a) The node and fracture plane are identified, b) simple node insertion can lead to degenerate elements, c) our approach is to snap the node to the fracture plane, d) the resulting mesh does not contain degenerate elements.

The accuracy of the nodal stress calculation depends highly on the shapes of the elements [10]. The closer the top faces of elements are to equilateral triangles, the more precise are the stress calculations. Unfortunately, even though the edge-snapping technique prevents formation of degenerate elements, the introduction of a fracture into the mesh can produce elements of sub-optimal shapes. To further improve the mesh around a fracture after it has been extended, we employ the angle smoothing algorithm developed by Zhou and Shimada [9]. Since a global application of mesh smoothing would require re-computation of all elemental stiffness matrices, we only apply the smoothing to the mesh nodes around the fracture tips.

Local multi-resolution calculation of nodal stress at crack tips. The elements around a fracture tip must be very small in order to calculate the stress at the fracture tip correctly. On the other hand, once the nodal stress has been evaluated, the need for such small elements disappears. To reconcile these requirements, we evaluate nodal stresses at fracture tips using a local multi-resolution method (Fig 2c). First, we extract a sub-model from the original model, consisting of the mesh in the neighborhood of the fracture tip. This sub-model is then refined around the fracture tip to a user-controlled level of detail with the algorithm of Rivara and Inostroza [5]. The equilibrium state of the refined mesh is calculated next; this is followed by the computation of the nodal stress at the fracture tip. The refined sub-model is then discarded. The end-result is the original mesh and a more accurate approximation of the stress at the fracture tip. This process is illustrated in Fig. 6.

3 Results and Discussion

Sample bark and mud patterns synthesized using the presented method are shown in illustrated in Figs. 7 and 8. The different patterns were obtained by varying simulation parameters, including the thickness of the material layer, rate of growth and shrinkage, Young's modulus, threshold stress of the material, fracture toughness, etc. The average size of the models used to generate these

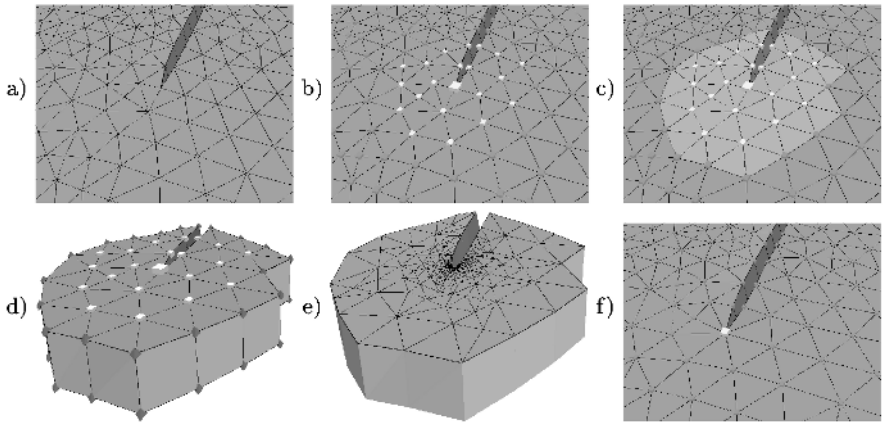

Fig. 6. Illustration of the local multi-resolution calculation of stresses and fracture propagation. a) View of a fracture before it is extended. b) Nodes close to the fracture tip are identified. c) All elements sharing the selected nodes are identified. d) A sub-mesh with the selected elements is created. The nodes on its boundary are treated as fixed. e) This mesh is refined and the stress at the fracture tip is computed with increased precision. f) The sub-model is discarded and the calculated stress at the fracture tip is used to extend the fracture.

Fig. 7. A variety of bark-like patterns generated by the proposed method.

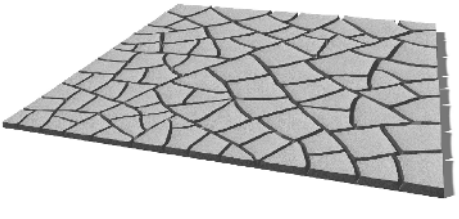

Fig. 8. Generated fracture pattern in dried mud.

patterns was between 60 and 150 thousand elements. The running times were of the order of few hours on a 1.4GHz Pentium IV computer.

We found that the largest performance improvement was achieved due to the dynamic subdivision of elements around the fractures. The local equilibrium (relaxation) calculation algorithm also improves the simulation efficiency. For example, the mud pattern in Fig. 8 was generated in approximately two hours using the local relaxation algorithm. The same pattern took almost eight hours to synthesize when the local relaxation was turned off. This large improvement in the simulation time is due to the fact that fractures reduce the global effects of localized changes.

In conclusion, this paper shows that the finite element method is a viable tool for modeling not only individual fractures, but also fracture patterns. The acceleration techniques presented in this paper, taken together, decrease the computation time an order of magnitude, compared to the non-accelerated method.

Acknowledgments. We thank Brendan Lane and Colin Smith for editorial help. The support of the Nationial Science and Engineering Research Council is gratefully acknowledged.

References

1. Anderson T. L. Fracture Mechanics: Fundamentals and Applications. CRC Press, Boca Raton, second edition, 1995.
2. Keeve E., Girod S., Pfeifle P, Girod B. Anatomy-Based Facial Tissue Modeling Using the Finite Element Method. Proceedings of Visualization'96, 1996.
3. O'Brien J. F., Hodgins J. K. Graphical Modeling and Animation of Brittle Fracture. Proceedings of ACM SIGGRAPH'99, 1999.
4. Press W. H., Teukolsky S. A., Wetterling W. T., Flannery B. P. Numerical recipes in C: the art of scientific computing. Second edition. Cambridge University Press.
5. Rivara M. and Inostroza P. Using Longest-side Bisection Techniques for the Automatic Refinement of Delaunay Triangulations. The 4th International Meshing Roundtable, Sandia National Laboratories, pp.335-346, October 1995.
6. Romberger J. A., Hejnowicz Z. and Hill J. F. Plant Structure: Function and Development. Springer-Verlag,1993.
7. Witkin A. P. and Heckbert P. A. Using particles to sample and control implicit surfaces. SIGGRAPH'94, pp. 269-277, July 1994.
8. Zhang L. C. Solid Mechanics for Engineers, Palgrave, 2001.
9. Zhou T. and Shimada K. An Angle-Based Approach to Two-Dimensional Mesh Smoothing. The 9th International Meshing Roundtable, pp.373-84, 2000.
10. Zienkiewicz O. C. and Taylor R. L. Finite element method: Volume 2 - Solid Mechanics. Butterworth Heinemann, London, 2000.
11. Zienkiewicz O. C. and Zhu J. Z. The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique. International Journal for Numerical Methods in Engineering, 33:1331-1364, 1992.