A Modular Approach Based on Graph Transformation to Simulate Tearing and Fractures on Various Mechanical Models

Fatma Ben Salah, Hakim Belhaouari, Agnès Arnould, Philippe Meseure

A Modular Approach Based on Graph Transformation to Simulate Tearing and Fractures on Various Mechanical Models

Fatma BEN SALAH, Hakim BELHAOUARI, Agnès ARNOULD and Philippe MESEURE
University of Poitiers
Laboratory XLIM/ASALI , UMR CNRS 7252,
86962, Futuroscope Poitiers, France
fatma.ben.salah, hakim.belhaouari, agnes.arnould, philippe.meseure @univ-poitiers.fr

ABSTRACT

This paper introduces an extension of a general framework that allows the simulation of various mechanical models (discrete or continuous ones, for different kinds of meshes, in any dimension). This framework relies on a topological model and a rule-based language, that performs sub-graph matching and, possibly, transformations. This extension allows topological modifications such as tearing and fractures for all the implemented physical models. A general process has been used to simulate fractures and tearing: the topological transformation is described using the provided rule-based language and its application is triggered when a selected criterion is verified. Several criteria are proposed, that depend upon the strain or stress generated by a single or a set of interactions. This method raises the question of the link between the location where a criterion is applied and the mesh elements involved in a modification. This question has motivated us to design new criteria which are closely related to the mesh modification. Using this method, a minimal number of mechanical data need to be updated after a transformation and any interaction relying on mesh features (vertex, edge, face, volume) that are suppressed or split can be automatically ignored.

Keywords

Physical simulation, tearing, fracture, criterion, topological model, graph recognition.

1 INTRODUCTION

Due to their complexity, tearing and fracture are ones of the most studied phenomena in computer graphics, especially in physically-based animation. They occur due to the stress undergone by an object when it deforms. Much work in the literature has studied these phenomena using physical models, as they produce more realistic and accurate results. These physical models can be discrete (e.g. mass/interaction) [NTB+91, HTK00, SWB00], or based on continuous mechanics and solved by a Finite Element Methods (FEM) [OBH02, OH99, MMD+01, KLB14]... In this paper, we consider tearing and fracture as similar phenomena, depending on the rigidity of the material of the simulated object.

To initiate the tearing/fracture in an object, a criterion is required. Several types of criteria have been defined in the literature. These criteria can be based on strain or deformation (for instance, when the length of a spring goes beyond a threshold [NTB+91]). Or criteria can be based on stress, that is forces applied by one or a set of interaction (for instance the separation tensor defined by [OH99]).

After selecting a criterion that initiates the tearing/fracture and, more precisely, defines its location, a topological transformation is applied. Many types of topological transformations have also been proposed in the literature. For example, splitting several faces/edges surrounding a vertex [SWB00], removing an element from the mesh [NTB+91], etc. These mesh modifications have an effect both on mass distribution and local interactions. Therefore, after a transformation, mechanical information has to be updated.

Previous work did not focus on the relation between the location of a selected criterion and the location of the topological modifications. However, it is seldom the same location, and the link between these two locations is not necessarily immediate. For instance, in [NTB+91], when a spring is too much stretched, the support edge is not removed alone, but also volumes around this edge.

In [BBAM17], a general mechanical framework using a topological structure and based on rules of graph transformation for physical simulation is defined. This paper proposes an extension of this framework that allows tearing/fracture while still preserving its genericity. More precisely, this extensions:

- Offers many types of criteria (proposed in other approaches) to initiate topological modifications applied in 2D and 3D to different physical models and

different types of elements as well as many types of topological modifications;

- Formalizes the link between the locations of a criterion and a topological modification by using a graph transformation-based approach. New criteria that are more closely related to topological transformation are also proposed;
- Allows topological modifications with minimal update of mechanical information.

The paper is organized as follows: in Section 2, some previous work related to fracture and tearing is briefly presented. Section 3 details the used topological model, the modeling approach used by the framework described in [BBAM17]. Section 4 discussed the usually used topological modifications and how they have been hosted in our framework. Section 5 presents the simulation of tearing/fraction using different criteria. Finally, some results and a discussion are given in Section 6.

2 PREVIOUS WORK

Topological modifications of physical models have been widely studied. Many approaches have simulated the cutting of deformable bodies, often for medical applications [WWD15]. However, cutting is a user-controlled phenomenon that does not require a criterion to trigger it. In practice, only the contact between the body to cut and an appropriate tool is sufficient to initiate cutting.

Although some approaches such as [LBC+14] have focused on tearing and/or fracture using meshless models, most methods actually rely on meshed objects. Contrary to cutting, tearing and fracture require a criterion to determine whether a crack should appear or not, the location of this crack and its direction. Thus, many types of criteria have been proposed in the literature. These criteria can be based on local strain or stress in an object, and can be subdivided into two categories, namely *Atomic criteria* and *Cumulative criteria*.

Atomic criteria focus on a single interaction and consider a force or a deformation threshold to trigger a topological modification. For instance, Norton et al. [NTB+91] use a mass/spring system (MSS) to simulate deformation and check if the length of springs exceeds a given value to decide where a fracture should be applied. The same criterion has been used by many following approaches: Hirota et al [HTK00] to simulate fracture on the surface of drying clay, Boux de Casson et al. [BL00] to simulate tearing of biological tissues represented by triangular meshes in 2D and tetrahedral meshes in 3D. It is also been used in [DKS+11] and [LBC+14].

Using cumulative criteria, a tearing/fracture is triggered if the sum of internal forces applied on a vertex or an element exceeds a threshold. Most methods that have used this type of criteria are based on continuous mechanics. The pioneer approach has been proposed by O'brien et al. for brittle materials [OH99] and ductile ones [OBH02]. Their criterion is based on a separation tensor computed for each vertex that triggers a fracture when one of its eigenvalues exceeds a given threshold. Recently, Koshier et al. [KLB14] used a similar criterion, but a tensor is computed for each vertex as the average of all stress tensors of its surrounding elements. Many other studies rely on a vertex tensor in 2D or 3D [IO09, WRK+10, PNdJO14, PO09].

All above-mentioned approaches use various topological transformations. Norton et al. [NTB+91] remove one or more elements. Faces or volumes are split in [SWB00] and [BL00], elements are cut and duplicated in [MBF04]. In [OH99, OBH02, KLB14], for a vertex to split, adjacent volumes are separated (by splitting a fan of faces) in 3D. In 2D, two faces must be separated by splitting their common edge. To facilitate this step, some studies use a predefined crack pattern [IO09, MCK13, PO09]...

Note that the link between the criterion location and the resulting topological transformation is not necessarily immediate. For instance, in [DKS+11], when the length of a spring exceeds a threshold, one of its two extremities is arbitrarily split. In [NTB+91], the same criterion leads to an element removal, even if all the other springs of the element do not exceed the deformation threshold. Since no solution to directly deal with the support edge of the spring has been proposed, some consecutive methods have chosen not to rely on a mesh [LBC+14]. Any criterion that should result in a vertex split, often requires to split a fan of faces in 3D and one or two edges in 2D. Finally, no approach focuses on the relation between the location of the fracture criterion and the location of the consequent topological modification.

After any topological modification, some mechanical properties (for instance, mass or stiffness) have to be updated. Using FEM, only the mass of vertices have to be updated, which can however be a tedious process, depending upon the method used to take inertia into account. Using MSS, this step can be considered as costly [FZDJ14, MDS10], because springs split into several elementary springs. To summarize, the application of a topological modification can be divided into three steps: first the choice of a criterion, second, a topological modification to apply and, last, the update of mechanical information.

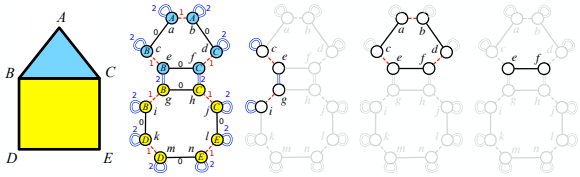
Some general frameworks that allow objects simulation using various physical models exist (Sofa is probably the most general one [Sof]). However, no general frameworks that allow the tearing and/or fracture

with many types of criteria and topological modifications have been proposed. The approach proposed in this paper relies on the general framework described in [BBAM17] which is based on rules of graph transformations applied on a topological model. This framework seems to be the most adapted to our needs due to the fact that it is multi-element, multi-physics and multi-dimension. Furthermore, all mechanical properties and forces are stored at convenient topological places. To generate a modular system for tearing and fracture, several types of criteria (including new ones) and topological modifications have been integrated in this framework.

3 MODELING AND SIMULATION

3.1 Topological Model

Much work have shown the benefits of using a topological model to simulate topological transformations of an object [MDS10, FZDJ14]. The model used in [BBAM17] is the generalized maps (G-maps) [DL14]. This topological model is based on the concept of *dart*, that can be considered as the extremity of an edge of a face of a volume.



(a) (b) (c) (d) (e)

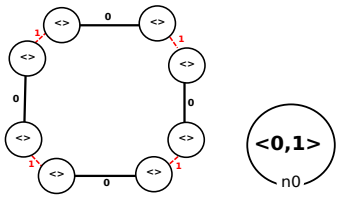
Figure 1: Decomposition of a 2D object and examples of orbits: (a) Object 2D, (b) G-map (connected component), (c) Vertex, (d) face and (e) half edge.

The representation of an object using a G-map is defined from its successive subdivisions into topological cells (volumes, faces, edges...). For example, the 2D object presented in Fig.1a can be decomposed into 2-dimensional cells (faces) connected by 2-links (blue double arcs). The faces are also split into 1-dimensional cells (edges) connected by 1-links (red arcs). In the same way, edges are decomposed into 0-dimensional cells (darts) connected by 0-links (black arcs). As represented in Fig. 1b, a G-map can be defined as a graph where nodes are darts and arcs labeled in $0,..,n$ are the adjacency relationships between topological cells (vertice, edge, face...).

These cells are defined by sub-graphs called *orbits*. Fig. 1 presents some examples of orbits corresponding to cells. Thus the sub-graph reachable from the dart e using 1- and 2-links in Fig. 1c represents the vertex B of Fig. 1a. It is noted $\langle 1,2 \rangle$(e). Similarly, the edge BC is the orbit $\langle 0,2 \rangle$(e), and the face ABC is represented by the orbit $\langle 0,1 \rangle$(e) (Fig. 1d). There exists also orbits which are not cells, for example the half-edge in Fig. 1e, or the connected component (Fig. 1b).

Depending on the targeted application, to model an object , much information (geometrical, mechanical, colorimetric...) must be added to the topological structure. These data are carried by all the darts and are called *embeddings*. Note that every embedding has its own data type and is attached to a particular orbit. For example in Fig. 1b, all the darts of the same vertex (A, B, C and D of Fig. 1a) are supplied with the same position information. Similarly, all darts of every face orbits $\langle 0,1 \rangle$ share the same color information (blue or yellow). Evenly, information can be associated to any type of orbits such as corner of face $\langle 1 \rangle$.

In this paper, an approach based on rules of graph transformation is used. These rules are created using a tool called Jerboa [BALB14], freely available at [Jer]. A rule in Jerboa is composed of two members and has the following form $L \rightarrow R$. The left member L contains the matched pattern that has to be recognized in the structure. The right member R corresponds to topological and/or embeddings modifications of this matched pattern. In fact, the application of a rule on a graph G consists in searching the sub-graph presented in the left part and replace it by the sub-graph presented in the right member of the rule. If the structure does not contain a matched pattern, the application of the corresponding rule fails.



(a) Rectangular face (b) Face

Figure 2: Examples of matched pattern

A matched pattern can be represented explicitly. In this case, all nodes have to be labeled by the dart orbit ($\langle\rangle$). An example of a rectangular face with all its darts is represented on Fig. 2a. A matched pattern can be also represented implicitly using darts labeled by orbits. For instance a face $\langle 0,1 \rangle$ is represented in Fig. 2b. This pattern can match not only rectangular faces but all possible types of faces (rectangular, triangular,...). Some examples of rules are presented in section 4.

3.2 Modeling

The framework described in [BBAM17] has been used. It is multi-dimensional (2D,3D), multi-element (allowing triangular, rectangular, hexahedral, etc. elements) and multi-physics (mass/spring, mass/tensor, finite element models). It represents objects by a 2D/3D mesh and stores as specific embeddings any mechanical data and forces needed to simulate them. Actually, it is

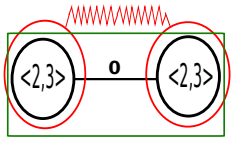
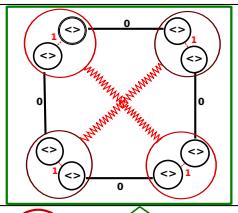
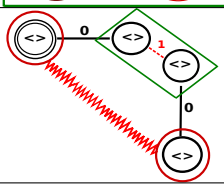
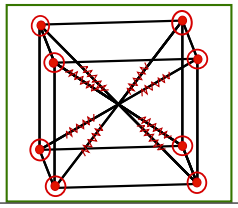
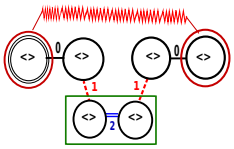
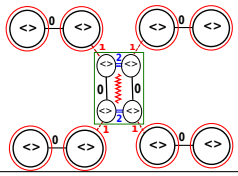
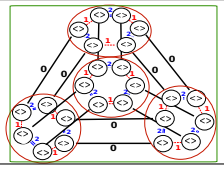
Interaction	*Matched pattern*	*Interac -tion orbits*	*Forces orbits*
Stretching spring		Half-edge: $\langle 0 \rangle$	Dart: $\langle\rangle$
Linear shear spring		Face: $\langle 0,1 \rangle$	Face corner: $\langle 1 \rangle$
Angular shear spring		Face corner: $\langle 1 \rangle$	Dart: $\langle\rangle$
3D shear spring		Volume: $\langle 0,1,2 \rangle$	volume corner: $\langle 1,2 \rangle$
Linear bending spring		Edge extrem-ity: $\langle 2 \rangle$	Dart: $\langle\rangle$
Angular bending spring		Edge: $\langle 0,2 \rangle$	Dart: $\langle\rangle$
Co-rotational		Volume: $\langle 0,1,2 \rangle$	Volume corner: $\langle 1,2 \rangle$

Table 1: Modeling of interactions

based on G-maps and rules of graph transformation presented in previous section. In this approach, the mass is distributed between particles that are placed on each vertex of the mesh. To compute forces applied by interactions, the same method is defined for all mechanical models and for all types of meshes. First, every data characterizing the interaction (stiffness, damping,...) is carried by an orbit. This orbit is related to the source/origin of the interaction and is represented by a green frame in Table 1 and more specifically in the third column. From this orbit, a matched pattern is defined. This sub-graph represents the left member of the rules used to compute forces. It allows the determination of the vertices involved in the forces computation (that contribute to the calculation and/or undergo forces). Finally, these forces are stored in sub-orbits of each involved vertex' orbit. They are represented in a red frame in Table 1 and explained in the last column.

The first lines of Table 1 presents MSS with the three types of springs defined by Provot [Pro95] (stretching, shear and the linear bending). Angular springs are also defined to control the angle between two adjacent faces [GHDS03]. In 3D, shear springs are also modeled. The last line in Table 1 presents how the FEM based on co-rotational [MG04] is modeled. Note that for continuous models, it is compulsory to tag a dart as the first dart to define the order of vertices and respect this order while exploiting the element stiffness matrix and its stress tensor.

3.3 Simulation

The simulation loop used in [BBAM17] to simulate objects is the same for all physical models. (see Fig. 3). The loops begins by computing the forces applied to each particle. This is done in three steps. First, by walking through the structure, any orbit that embeds properties of an interaction is found. Second, if this interaction's matched pattern is recognized, the involved vertices are identified. Third, the interaction rule is applied that is, the interaction is calculated and the obtained forces are embedded in the planned orbits. After this process, the algorithm computes the acceleration depending on Newton's second law. For this purpose, it walks through the structure and, for every vertex, collect forces stored in its sub-orbits($\langle 2 \rangle$,$\langle 1 \rangle$,$\langle 0 \rangle$,$\langle 1,2 \rangle$...). After that, it computes new velocities and positions of particles. Finally, it walks through the structure, evaluates the chosen criterion, and when the condition is satisfied, applies the corresponding topological modification.

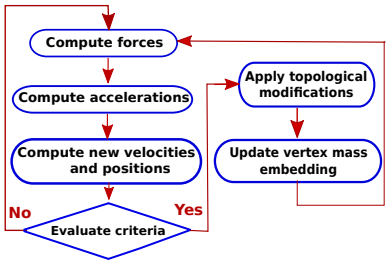


Figure 3: Fracture plan.

A special attention must be paid to the storage of particles' mass. In practice, the mass of a particle comes from faces/volumes surrounding the corresponding vertex. Therefore, an embedding called "corner mass" is stored in the orbit corner of face (2D) or volume(3D)

($\langle 1 \rangle$ or $\langle 1,2 \rangle$), to memorize the contribution of each face/volume surrounding a vertex. Moreover, to allow a fast computation of a particle behavior, its total mass must be known and is also stored in a dedicated embedding, in its corresponding vertex orbit. It is called "vertex mass" and defined as the sum of all masses stored in all its corner sub-orbits. In case of topological modification, this last embedding must be updated.

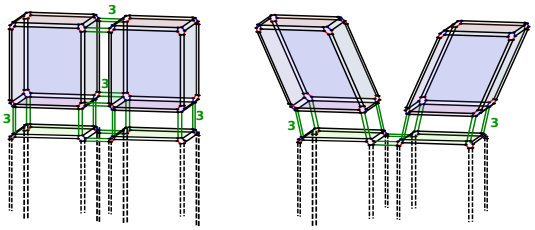


Figure 5: Face split to separate two adjacent volumes.

4 TOPOLOGICAL MODIFICATIONS FOR TEARING/FRACTURE

Topological modifications such as tearing/fracture can be simulated very easily using G-maps. More complex modifications can thereafter be built using a composition of separations of adjacent elements. For instance, it is possible to isolate an element and to remove it [MDS10]. It is also possible to separate several faces or volumes, to make a vertex split. In this section, we first focus on separation of elements, since it is a fundamental modification, before describing modifications based on composition of transformations.

4.1 Separation of Adjacent Elements

In 2D, two adjacent faces can be separated by splitting their common edge, more precisely by removing the 2-links that bind them as shown in Fig. 4. In a similar way, two adjacent volumes can be separated by splitting their common face, that is, removing the 3-links that bind them, as shown in Fig. 5. These link removals are elementary transformations in G-maps, that automatically deals with vertex splits [MDS10]. Note that, in 2D, removing 2-links should cancel the corresponding bending interactions (linear or angular). Using the rule-based approach, any suppression of topological links can automatically make any interaction be ignored, if the support orbit of the interaction embedding includes these links. Indeed, since the associated pattern no longer matches the local, unlinked, structure, the rule can not be applied anymore. The rule just has to reset all the corresponding forces.

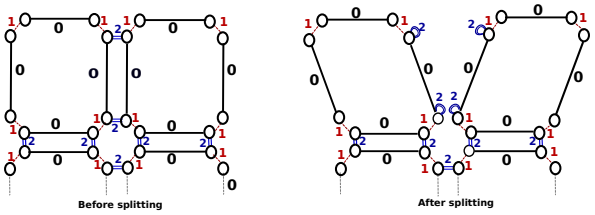


Figure 4: Edge split to separate two adjacent faces.

Shear springs are stored in orbits $\langle 0,1 \rangle$ or $\langle 1 \rangle$, and they are not influenced by the removing of 2 and 3-links. Neither do stretch springs stored in orbits $\langle 0 \rangle$. Concerning continuous mechanics, in $2D$, no embedding are supported by orbits with 2-links. In $3D$, orbits of embeddings do not include any 3-links. Finally, no embedding needs to be changed. Only *vertex mass* embedding of split vertices must be updated.

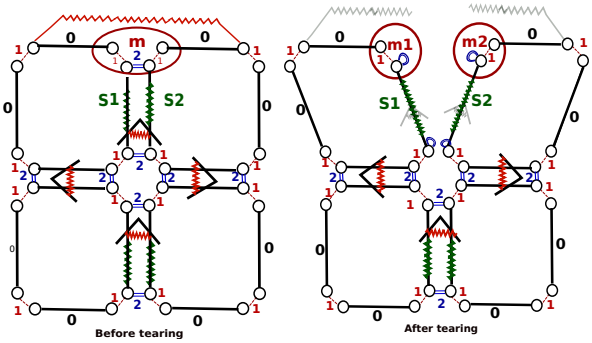


Figure 6: Impact of tearing/fracture on mass embedding and angular springs.

For instance, in Fig. 6, if adjacent faces are separated, 2-links are broken and the common edge is split. As a result, bending springs (angular or linear one) embedded in sub-orbits $\langle 2 \rangle$ or $\langle 0,2 \rangle$ of the split edge are ignored after the separation. More precisely, while identifying the sub-graph corresponding to the force computation, the pattern does not match, so forces can not be computed. This example also explains that the accumulation embedding *vertex mass* stored in vertices (orbits $\langle 1,2,3 \rangle$) has to be updated as it relies on 2-links that have been suppressed. In fact, as shown in Fig. 6, the *vertex mass* is the contribution of two *corner masses* provided by the two adjacent faces ($m_{vertex} = m1 + m2$). After the edge split, this vertex splits and each new vertex gets a new *vertex mass* value, (computed, here, using a single *corner mass*).

Rules for Separation of Elements

To split a face between two volumes, the rule presented in Fig. 7 has been created. In the left member of this rule, two faces are bound with 3-links. These faces are represented by their orbits $\langle 0,1 \rangle$. In the right member, the 3-links are broken to split the two faces. However this rule only copes with topology and does not handle embeddings (they are handled in a dedicated rule).

Another rule is presented in Fig. 8. It has been created to separate two adjancent faces by splitting an edge. In this rule, the pattern presented in the upper part of the figure matches explicitly the 2-links between

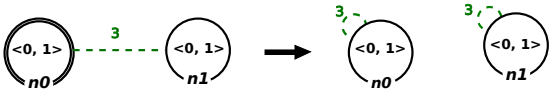


Figure 7: Rule to separate two adjacent volumes.

two adjacent faces. The lower part of the figure describes the topological modification which is the suppression of 2-links between faces. As linear bending springs are stored in orbits $\langle 2 \rangle$ and angular ones in orbits $\langle 0,2 \rangle$, they are removed when the 2-links are broken. This single rule updates simultaneously the embedding *mass vertex*, the topological modification and resets the forces applied by the bending springs. When a vertex splits, the other embeddings (position, velocity, etc.) are automatically duplicated so the new particle is taken into account in the forthcoming steps of the simulation.

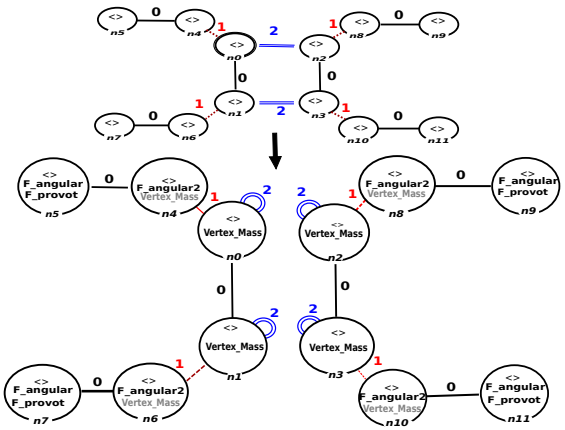


Figure 8: Rule to separate two adjacent faces.

4.2 Mesh Element Removal

To suppress an element (a volume in 3D and a face in 2D), the same process described in [MDS10] is used. This process consists first in the isolation of the element: in 2D, all its edges are 2-unlinked, in 3D, all its faces are 3-unlinked. This separation can be done using the rules described in Section 4.1. The second step consists in deleting the isolated element (using a rule with an empty right member).

4.3 Vertex Split

As proposed in [OH99], to split a vertex, a fan of faces surrounding this vertex can be split. In 2D, only two edges have to be split. A tearing/fracture plane must be computed based on the selected criterion. Then, the sign of all elements surrounding a vertex is determined by replacing the coordinates of the center of each element in the plane equation. The sign is stored in the orbit of every element as an embedding. All edges/faces placed between two elements with different signs have to be split using rules described in Section 4.1. This topological modification can be applied for any interaction model (MSS, FEM).

4.4 Edge Removal

If a criterion relies on an edge, it can require that this edge disappears. However, removing an edge from a mesh is a tricky topological modification. Indeed, this removal can lead to a type change of surrounding elements. For instance, the contraction of an edge in a rectangle mesh can induce a transformation of elements into triangles. Therefore, as a solution, it is better to use more general transformations involving the concerned edge that also control adjacent elements. For example, one or more elements adjacent to this edge can be deleted, as proposed by Norton et al. [NTB+91]. This modification is based on element removal described above (see Section 4.2). Vertex split can also be used [DKS+11]. Finally, whatever the chosen transformation, this last influences not only the used criterion's cell but also many adjacent mesh elements.

5 SIMULATION OF TEARING/FRACTURE

Tearing or fracture occurs when a given criterion is met. Different criteria exist and have been added to the framework. These criteria act as classical preconditions used by Jerboa that trigger topological modifications (see Section 4) when they are satisfied. As explained in Section 2, criteria can be divided into two categories: atomic and cumulative ones. Concerning cumulative criteria, additional rules must be created to compute the sum of needed stresses/forces.

5.1 Atomic Critera

Atomic criteria only relate to a single interaction. Criteria are based on strain or stress considerations. Note that stress-based criteria are particularly well adapted to our models, since, as described in section 3.2, all forces applied by all types of interactions are stored in sub-orbits of vertex orbits. Concerning linear (resp. angular) springs, strain or stress criteria are equivalent [BL00], that is, a length (resp. angle) or a force (resp. torque) threshold can be checked. If the chosen threshold is exceeded, the process consists in removing the spring. To remove a spring, the required topological transformation depends on its type. Stretch springs are placed on edges. As discussed in Section 4.4, volumes can be removed, as proposed in [NTB+91, HTK00, BL00]. If the same criterion is checked on a shear spring, the support element (volume in 3D or face in 2D) should be suppressed. If a bending (linear or angular) springs verifies the tearing/fracture criterion, the two concerned faces must be separated, as described in

Section 4.1. In a similar way, if continuous mechanics is used, eigenvalues of the stain or stress tensor of each element can be computed and, if a threshold is exceeded, the concerned element is suppressed.

However, to avoid the design of specific rules for all kinds of interaction, it is possible to use a more general approach, that can be applied both in 2D and 3D. Indeed, a criterion can be defined by checking if a force, whatever its origin, exerted on a vertex, exceeds a threshold. By considering a fracture plane perpendicular to this force, a vertex split transformation can be applied (Section 4.3). Note that, due to the action/reaction principle, the two extremities of a spring or a set of involved vertices for other interaction types are likely to be concerned by a threshold exceeding. In this case, it is possible to apply a vertex split on all the concerned vertices (to simulate multiple cracks) or on a single (arbitrarily chosen) vertex [DKS+11].

5.2 Cumulative Criteria

Cumulative criteria first compute the sum of all forces/stresses applied on every vertex. Using continuous mechanics, this method corresponds to the O'Brien et al.'s approach [OH99]. However, the simplified approach proposed in [KLB14] is used in our framework. The stress tensor (expressed here as a 6-vector in 3D) of each element i is computed as explained in [MDM+02]:

$$\sigma_{\mathbf{i}} = (\mathbf{C} * \mathbf{B_i})\mathbf{u} \tag{1}$$

With $\mathbf{C}$ the stress-strain matrix, $\mathbf{B_i}$ is the strain-displacement matrix (for more detail the reader must refer to [MDM+02]) and $\mathbf{u}$ is the displacement vector. A vertex tensor is computed by averaging the stress tensor of surrounding elements i:

$$\sigma_{\mathbf{vertex}} = \sum m_i \sigma_{\mathbf{i}} / \sum m_i \tag{2}$$

With m_i the mass of the element i and σ_i, the 3×3 stress tensor of this element. The eigenvalues of the vertex tensor are computed. If one of these values exceeds a threshold, the corresponding eigenvector is computed. Finally, the fracture plane is defined as perpendicular to this eigenvector. A vertex split is applied as described in Section 4.3. A similar approach can be used in 2D.

A more general cumulative criterion can be proposed, whatever the used deformation laws. Using a walk through all the forces stored in every vertex orbit (see Table 1), the sum of internal forces can be computed for each vertex. When the magnitude of such a force exceeds a threshold, a vertex split is triggered using a fracture plane perpendicular to this force. Note that this kind of criterion can only work in particular cases, for instance when internal forces must compensate a high external stress, or when a sudden deformation appears locally in the mesh (after the motion of one or more vertices). On the contrary, at rest, the internal forces can be null and no tearing/fracture appears, even if atomic interactions produce high magnitude forces.

5.3 Criteria dedicated To Mesh Elements Separation

As stated in [MDS10], separation of elements is the fundamental topological modification that is required to represent tearing or fractures, since it is the core on which the other transformations are based. However these transformations are not enough local and involve multiple elements not directly concerned by the used criterion. Separations of elements are surely desirable, but, unfortunately, no above-mentioned criterion results in a single application of such elementary modifications. In a similar way as Smith et al.[SWB00] who proposed a constraint-based criterion to split a face between two volumes, force-based solutions are investigated to trigger element separation. On other words, we want to find criteria that are applied on a cell and that trigger a modification on this same cell.

In 2D, faces are separated by splitting an edge, so this edge should support the fracture criterion. More precisely, the forces applied to the edge AB by all interactions supported by the adjacent faces are compared to check if they tend to make the edge split. First of all, the direction $\mathbf{n}$ along which the forces are compared (see Fig. 10) is computed as:

$$\mathbf{n} = \mathbf{AB} \times (\mathbf{n_1} + \mathbf{n_2}) \tag{3}$$

Let $\mathbf{f_A^{(i)}}$ and $\mathbf{f_B^{(i)}}$ the forces applied by each adjacent faces i on vertices A and B. These forces just require to walk through the orbits of A and B restricted to each face (orbit $\langle 1 \rangle$ in 2D, orbit $\langle 1,2 \rangle$ in 3D) and collect all calculated forces stored in sub-orbits. Let $\mathbf{f_i} = \mathbf{f_A^{(i)}} + \mathbf{f_B^{(i)}}$, the forces applied on the edge. A possible criterion compares the effect of these forces by computing:

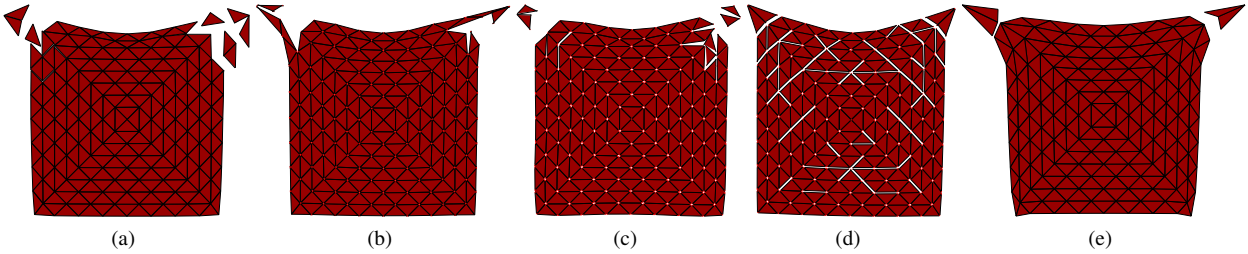
$$|(\mathbf{f_1} - \mathbf{f_2}) \cdot \mathbf{n}| \tag{4}$$

If the magnitude of this force is beyond a threshold, and if each force tends to contract the elements, then the common edge can split. This occurs when forces $\mathbf{f_1}$ and $\mathbf{f_2}$ are along opposite directions, and at least one of them has a high magnitude. This criterion is called "atomic edge criterion".

Let $\mathbf{f_A}$ and $\mathbf{f_B}$ the overall forces (coming from any element) applied on these vertices. Another criterion consists in computing the magnitude of the overall forces applied on the edge, projected on $\mathbf{n}$, and comparing it to a given threshold:

$$|(\mathbf{f_A} + \mathbf{f_B}) \cdot \mathbf{n}| \tag{5}$$

This kind of criteria is called "global edge criterion" and can also be extended as a face criterion in 3D. In this case, the normal of the face to split $\mathbf{n}$ is used to



(a) (b) (c) (d) (e)

Figure 9: Examples of tearing a 2D object with different criteria.

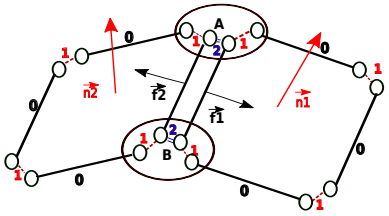


Figure 10: Computing internal forces of an edge.

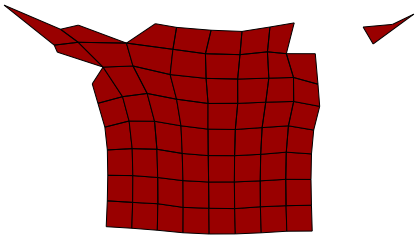


Figure 11: Edge split with criterion based on the difference between forces applied on an edge.

project the different faces and compute their contribution to the face split. As above, only forces stored in the volume orbits are first collected, projected on **n** and are compared to check if the common face splits. The overall forces applied to the face vertices can also be used to provide another criterion.

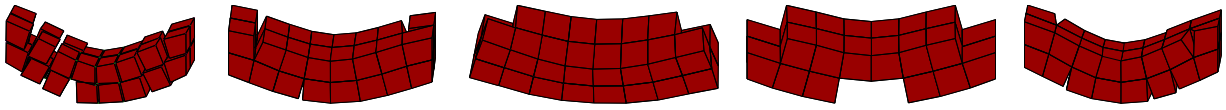
6 RESULTS AND DISCUSSION

We have implemented the criteria and topological modifications cited in the previous sections using the rule-based language. Thus, Fig. 9 presents the same object, a cloth modeled as a 2D triangular mesh, simulated with different types of criteria. Its two upper extremities are always fixed. In Fig. 9a, this cloth is modeled with co-rotational FEM and is torn by vertex split when the force applied by at least one of its adjacent face is beyond a threshold (atomic criterion). Fig. 9b presents the same cloth modeled as MSS, but when the force generated by a spring exceeds a threshold, its extremities split. Then, Fig. 9c presents this same system where a vertex splits wherever the sum of applied forces exceeds a threshold. In Fig. 9d, the same model is used, and an edge of the mesh can split where the *global edge criterion* is verified. Finally, Fig. 9e still presents the same MSS with edge splits where the *atomic edge criterion* is satisfied. As can be seen, different fracture criteria produce completely different results, where tearing is concentrated on the constraint areas or is disseminated in the overall body.

Similar simulations can be applied on rectangular or mixed meshes using the same criteria and topological modifications. An example of such a rectangular mesh is presented in Fig. 11. Fig. 12 presents some other results of fractures in 3D using mass/spring system and different criteria. An example of tearing of a liver (consisting of 597 elements) simulated using the co-rotational method is presented in Fig. 13.

The performance of our model is not the main concern of this paper. Indeed, Jerboa is a tool dedicated to prototyping and not interactive simulation, even if it allows the simulation of topological transformations on complex meshes. As a consequence, some of the discussed models have been ported to C++ to measure their efficiency. This specific implementation strictly respects all the principles described in this paper. In particular, when an interaction embedding is found, the corresponding graph is used to find how the interaction is computed and where forces apply. We chose to focus on a 2D cloth model consisting of 10×10 rectangular patches that can be further triangulated to produce a local triangular mesh. An example of simulation using this optimized implementation is presented in Fig. 14.

Some simulation times are summarized in Table 2 for different type of meshes, using Runge-Kutta 4 integration to allow a more stable simulation. No parallelization method has been used. Note that, thanks to the ruled-based approach, the implementation allows the simulation of heterogeneous meshes, with a little computation over-cost (below 1%) compared to meshes consisting exclusively of quads or triangles. Simulation have been measured on a 2.7GHz Intel I7 system (3740QM Processor). All simulation times are actually compatible with real time. Simulating with quads is faster because this model is subdivided in less faces. The mixed mesh triangulates half the quads of the initial mesh. It appears as a real compromise between quad and triangle meshes. The global edge criterion is faster than the atomic one, because this last requires to store atomic forces in the model instead of only accumulating them in each vertex, which can be a costly process.



(a) Atomic force criterion and vertex split

(b) Sum of forces criterion and vertex split

(c) Atomic forces criterion and volume suppression

(d) Atomic force criterion of shear springs and edge suppression by removing volumes

(e) Atomic force criterion applied on a face and face split

Figure 12: Examples of fracture of a 3D object with different criteria.

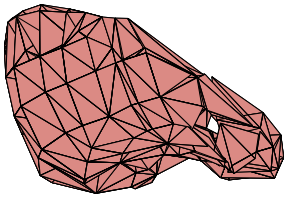


Figure 13: Elementary forces and vertex split (FEM).

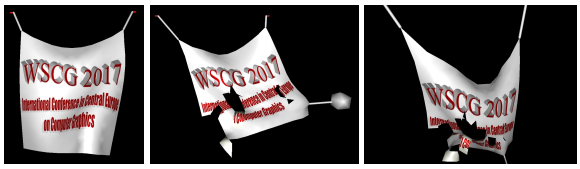


Figure 14: A cloth simulated in real-time (mass = 100g, stretching stiffness = 60 N/m, shear stiffness = 43 N/m, bending stiffness = 10 N/m, damping = 0.1, fracture threshold = 0.5 N), using a global edge criterion.

	Basic step	Global edge criterion	Atomic edge criterion
Quads	0.37	0.41	0.55
Triangles	0.78	0.95	1.3
Mix	0.61	0.72	0.97

Table 2: Simulation times (in ms) for a cloth depending upon the type of mesh. Basic step shows the simulation times without any tearing detection.

However, this criterion triggers tearing only on points of very high stress (constrained particles in the example), which can be a desired behavior.

7 CONCLUSION/PROSPECTS

This paper proposes an extension of the framework presented in [BBAM17] which relied on rules of graph transformations and a topological model to simulate various deformable bodies in 2D and 3D. This extension provides this framework with topological transformations such as tearing and fractures. The proposed approach is modular, since the user can select a criterion (or more) and a suitable mesh modification. This makes it an interesting tool to control tearing in an animation/simulation system. Many known types of criteria are implemented, some of them are specific to a given mechanical model and others are completely general. To produce mesh modifications when a criterion is met, several types of topological transformations are proposed. However, some of these modifications (vertex split, edge removal) tend to involve a lot of adjacent elements and are surely not enough local, so often require to use a high resolution mesh or subdivide the elements locally. To circumvent this problem, we propose to use more closely-related criteria and topological modifications, that is, criteria and topological modifications based on the same topological cell (face in 3D, edge in 2D). The mechanical information of the model is stored in an atomic way, therefore no update of interaction properties such as spring stiffness is needed after a topological change. The same way, any interaction that depends on the modified cells is automatically ignored thereafter. Only mass of particles must be updated, but this process consists in gathering elementary masses stored in the associated vertex' orbit. Simulations based on different scenarios show the effectiveness of our approach.

We have investigated its efficiency through a C++ implementation of some 2D models. The measured simulation times allow interactive and real-time simulations. However, it is possible to get even better simulation times by combining some interactions that appear as redundant: For instance, all stretch springs corresponding to the same edge can be cumulated in a single spring (that is, stiffness and damping values are added) and resulting forces computed once instead of being computed for each atomic spring. In 2D, only two springs are to be cumulated, but in 3D, more atomic springs are generally involved. Other redundancies exist, in particular in 3D (shear springs of 3-linked faces, linear provot springs placed between the same vertices, etc.). However, this approach implies that some cumulated interaction properties should be updated after any topological change, and that some atomic forces will no longer be available to contribute to a tearing/fracture criterion. In future work, we want to include these optimization solutions to our framework, whenever the selected criterion allows it, and enhance it with other complex types of topological transformations, for instance subdivisions of elements or local remeshing .

8 ACKNOWLEDGMENTS

This work has been partially funded by the European Erasmus Mundus - Al Idrisi II scholarship program. It also received fundings from the MIRES federation.

9 REFERENCES

[BALB14] Hakim Belhaouari, Agnès Arnould, Pascale LeGall, and Thomas Bellet. Jerboa: A graph transformation library for topology-based geometric modeling. In *ICGT*, pages 269–284, 2014.

[BBAM17] Fatma Ben Salah, Hakim Belhaouari, Agnès Arnould, and Philippe Meseure. A general physical-topological framework using rule-based language for physical simulation. In *VISIGRAPP*, 2017.

[BL00] François Boux de Casson and Christian Laugier. Simulating 2D tearing phenomena for interactive medical surgery simulators. In *Computer Animation*, pages 9–14, 2000.

[DKS+11] Emmanuelle Darles, Saman Kalantari, Xavier Skapin, Benoît Crespin, and Annie Luciani. Hybrid physical topological modeling of physical shapes transformations. In *CASA*, 2011.

[DL14] Guillaume Damiand and Pascal Lienhardt. *Combinatorial Maps: Efficient Data Structures for Computer Graphics and Image Processing*. A K Peter/CRC Press, 2014.

[FZDJ14] Elsa Fléchon, Florence Zara, Guillaume Damiand, and Fabrice Jaillet. A unified topological-physical model for adaptive refinement. In *VRIPHYS*, pages 39–48, 2014.

[GHDS03] Eitan Grinspun, Anil N Hirani, Mathieu Desbrun, and Peter Schroder. Discrete shells. In *Symp. on Computer Animation*, 2003.

[HTK00] Koichi Hirota, Yasuyuki Tanoue, and Toyohisa Kaneko. Simulation of three-dimensional cracks. *Visual Computer*, pages 371–378, 2000.

[IO09] Hayley N. Iben and James F. O'Brien. Generating surface crack patterns. *Graphical Models*, pages 198–208, 2009.

[Jer] http://xlim-sic.labo.univ-poitiers.fr/jerboa.

[KLB14] Dan Koschier, Sebastian Lipponer, and Jan Bender. Adaptive tetrahedral meshes for brittle fracture simulation. In *Symp. on Computer Animation*, pages 57–66, 2014.

[LBC+14] Joshua A. Levine, Adam W. Bargteil, Christopher Corsi, Jerry Tessendorf, and Robert Geist. A peridynamic perspective on spring-mass fracture. In *Symp. on Computer Animation*, pages 47–55, 2014.

[MBF04] Neil Molino, Zhaosheng Bao, and Ronald Fedkiw. A virtual node algorithm for changing mesh topology during simulation. *Trans. on Graphics*, pages 385–392, 2004.

[MCK13] Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Real time dynamic fracture with volumetric approximate convex decompositions. *Trans. on Graphics*, pages 115:1–115:10, 2013.

[MDM+02] Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. Stable real-time deformations. In *Symp. on Computer Animation*, pages 49–54, 2002.

[MDS10] Philippe Meseure, Emmanuelle Darles, and Xavier Skapin. Topology based physical simulation. In *VRIPHYS*, pages 1–10, 2010.

[MG04] Matthias Müller and Markus Gross. Interactive virtual materials. In *Graphics Interface*, pages 239–246, 2004.

[MMD+01] Matthias Muller, Leonard McMillan, Julie Dorsey, , and Robert Jagnow. Real time simulation of deformation and fracture of stiff materials. In *Works. on Animation and Simulation*, pages 113–124, 2001.

[NTB+91] Alan Norton, Greg Turk, Robert Bacon, John Gerth, and Paula Sweeney. Animation of fracture by physical modeling. *Visual Computer*, pages 210–219, 1991.

[OBH02] James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. Graphical modeling and animation of ductile fracture. In *SIGGRAPH*, pages 291–294, 2002.

[OH99] James F. O'Brien and Jessica K. Hodgins. Graphical modeling and animation of brittle fracture. In *SIGGRAPH*, pages 137–146, 1999.

[PNdJO14] Tobias Pfaff, Rahul Narain, Juan Miguel de Joya, and James F. O'Brien. Adaptive tearing and cracking of thin sheets. *Trans. on Graphics*, pages 110:1–110:9, 2014.

[PO09] Eric G. Parker and James F. O'Brien. Real-time deformation and fracture in a game environment. In *Symp. on Computer Animation*, pages 165–175, 2009.

[Pro95] Xavier Provot. Deformation constraints in a masss/pring model to describe rigid cloth behaviour. In *Graphics Interface*, pages 147–154, 1995.

[Sof] https://www.sofa-framework.org/.

[SWB00] Jeffrey Smith, Andrew P. Witkin, and David Baraff. Fast and controllable simulation of the shattering of brittle objects. In *Graphics Interface*, pages 27–34, 2000.

[WRK+10] Martin Wicke, Daniel Ritchie, Bryan Matthew Klingner, Sebastian Burke, Jonathan Richard Shewchuk, and James F. O'Brien. Dynamic local remeshing for elastoplastic simulation. *Trans. on Graphics*, pages 49:1–49:11, 2010.

[WWD15] Jun Wu, Rüdiger Westermann, and Christian Dick. A survey of physically based simulation of cuts in deformable bodies. *Computer Graphics Forum*, pages 161–187, 2015.