Virtual Cutting of Deformable Objects based on Efficient Topological Operations

Christoph Paulus, Lionel Untereiner, Hadrien Courtecuisse, Stephane Cotin, David Cazier

Noname manuscript No.
(will be inserted by the editor)

Virtual Cutting of Deformable Objects based on Efficient Topological Operations

Christoph J. Paulus[1,2,3] · **Lionel Untereiner**[1,2,3] · **Hadrien Courtecuisse**[2,3] · **Stéphane Cotin**[1] · **David Cazier**[2,3]

Abstract Virtual cutting of deformable objects is at the core of many applications in interactive simulation and especially in computational medicine. The ability to simulate surgical cuts, dissection, soft tissue tearing or micro-fractures, is essential for augmenting the capabilities of existing or future simulation systems. To support such features, we combine a new remeshing algorithm with a fast finite element approach. The proposed method is generic enough to support a large variety of applications. We show the benefits of our approach evaluating the impact of cuts on the number of nodes and the numerical quality of the mesh. These points are crucial to ensure accurate and stable real-time simulations.

1 Introduction

The role of three-dimensional models has attained major importance in many areas, ranging from scientific visualization to numerical simulation, biomechanical modeling or interactive sculpting. In each of these different applications, the ability to model the deformation of an object, and to apply topological changes, is gaining interest. Although such topological changes can take place in various contexts, applications in medicine raise important and specific requirements.

Interactive numerical simulations of surgical procedures, aimed at training, rehearsal or per-operative guidance, are now considered important avenues to improve patient care and reduce risks [13]. They require geometrical and mechanical models of the organs, and the ability to simulate interactions with various medical devices.

Such interactions involve deformations, cutting, or tearing of the modeled soft tissues. Being able to handle all these interactions in real-time is of major importance. Several conditions need to be met: the deformations should be computed accurately, computations must remain fast enough to allow interactivity, and topological changes need to cover a wide range of cases, yet not hinder computational efficiency.

Virtual cutting essentially involves two steps: first, the update of the geometrical and topological representation of the simulation domain; second, the numerical discretization of the governing equations and the simulation of the deformable body being cut.

These two steps are tightly linked since the numerical stability of the simulation is directly impacted by the quality of the mesh, at least in the context of finite element methods. For this reason, the virtual cutting operations should guarantee a minimal quality of the FEM mesh at any time of the simulation. Moreover, the computation time is directly impacted by the number of degrees of freedom in the discretized domain. Therefore, limiting the introduction of new elements or nodes during topological changes is another important requirement.

Besides these main challenges, the following features are desirable: the mesh should closely approximate the intended separation surface; the method should allow for partial cutting of an element and should allow for multiple cuts within a single element.

Several approaches have addressed these objectives focusing on a subset of the mentioned features. In the following, we give a short overview of other methods and outline our contributions.

[1] Inria Nancy Grand Est, Villers-les-Nancy 54600, France
[2] Université de Strasbourg, ICube Lab, Illkirch 67412, France
[3] CNRS, ICube Lab, Illkirch 67412, France

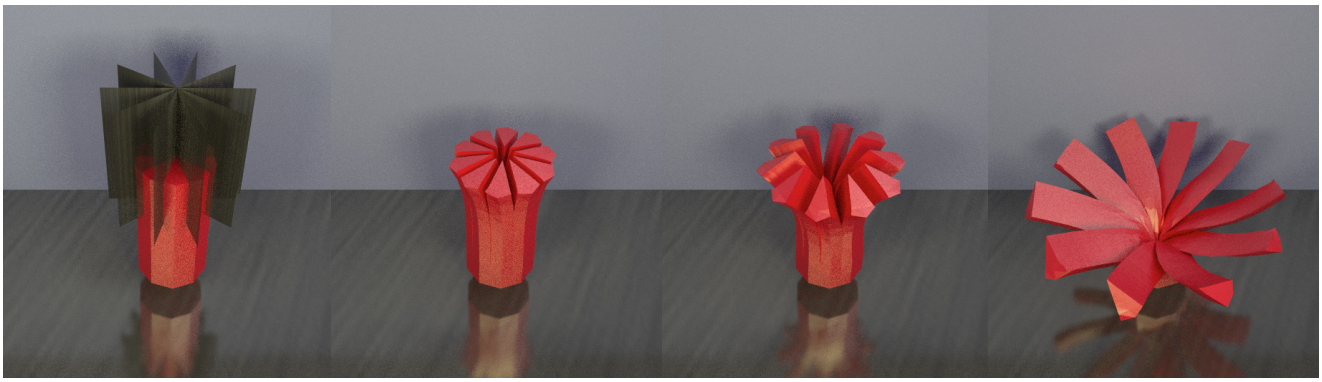


Fig. 1: Simultaneous cuts applied to a highly deformable model. The remeshing method reduces the number of new degrees of freedom while preserving the quality of the mesh, making it suitable for fast and robust simulations.

2 Related work

This section serves as a brief introduction to the relevant work in the fields of real-time (soft tissue) deformation and topological changes resulting from cutting or fracture. We will provide the reader with a few essential landmarks, and we refer to Wu et al [28] for a detailed review.

Cutting can be described as a controlled fracture process performed through a precisely directed path, exerted through sharp-edged devices or laser beams, while fracturing or tearing refers to the cracking or breaking of an object under the action of stress.

These two processes lead to the determination of a cutting or fracture path, requiring topological updates and computation of new data that will influence the physics of the object. In addition, both cutting and fracture can take place on rigid or deformable solids but some key challenges are emphasized in the context of deformations and surgical simulations. See [20] or [7] for detailed surveys in the field of soft tissue modeling and surgical simulations.

2.1 Simulation of deformable objects

The finite element method (FEM) has been introduced recently to the field of real-time soft body deformation. It has many benefits such as accuracy and robustness, but also shows some limitations: computation time and sensitivity to the mesh resolution and mesh quality are the most relevant for real-time simulations. The co-rotational FEM allows for geometric nonlinearities and realistically handle large displacements and rotations with minimal computational cost [10]. In the context of topological changes, real-time performances are obtained with specifically adapted preconditioners [7]. While several publications have addressed the problem of real-time computation with FEM, few address the mesh quality [30, 5].

To take into account the specific needs of surgical simulations, i.e. accuracy and real time performances, we model the behavior of the deformable objects with co-rotational FEM. The virtual cutting method that we present here, is particularly adapted to FEM meshes, but is general enough to be used with other simulation methods.

2.2 Remeshing approaches

There are several approaches to deal with topological changes with mesh-based methods. The challenge is to control the number of created nodes, and the quality of the inserted or updated elements. Failing to do so results in a slow down and in instabilities of the simulation due to ill-shaped elements. By now, proposed methods are challenged to solve this problem in a satisfactory manner, in particular in the case of partial, multiple and repeated cuts occurring in the same elements.

The work of [25] proposes a generalized framework to embed features like a separation surface inside a cell-complex. However, as the approach uses a surfacic mesh to represent the volumetric object, deformations calculated by the physically-based method like the FEM are not realistic enough for the medical context.

In [6], the authors proposed removing the elements impacted by the cut. This allows for real-time simulation of the cut since no additional degrees of freedom are introduced and no subdivision process is required. However, it may lead to a jagged cut surface for coarse meshes.

The snapping method presented in [18] moves the vertices of the original grid to the separation surface and disconnects the object at those vertices. The re-

sulting mesh approximates closely the cutting surface without adding a lot of degrees of freedom, but the method may lead to ill-shaped elements and contradicts the preservation of mass. Moreover, separations are restricted by the topology of the original mesh and partial cutting of an element is not possible.

The refinement or remeshing methods like the ones proposed in [19,17,11,2,14] replace a cut tetrahedron by several other tetrahedra, that have their nodes on the separation surface and are disconnected at the desired location. Some of these remeshing techniques allow for partial cuts, but usually introduce a lot more nodes. Most of these existing methods are either restricted to planar cuts or a further remeshing when a second cut appears, slowing down the simulation. In addition, separations close to the nodes may introduce ill-shaped elements, compromising the stability of the simulation.

A combination of snapping and refinement can alleviate some of the weaknesses of the methods [26]. However, since snapping algorithms do not allow for partially cut elements, partial separations have to rely on the existing remeshing techniques. Thus combined remeshing and snapping either do not allow for partial cuts or introduce again too many degrees of freedom for completely cut elements.

When used in combination with the FEM, remeshing operations need to build conformal meshes. In conformal meshes, the intersection of two elements is a sub-element of both elements - either a complete face, a complete edge or a vertex. Except from [2,14], most the methods proposed in the literature do not meet this requirement after the topological operations.

2.3 Multiresolution approaches

The composite finite element method (CFEM) [12] or [22] uses a coarse uniform grid for the simulation and a fine grid for the visualization and collision. The work introduced in [8] and improved by [27,29] proposes to connect elements in the fine grid by links, that are deleted if a separation takes place. An element of the coarse simulation grid is duplicated as soon as a separation of the complete fine grid inside of the coarse element takes place. The methods are very efficient and well-adapted to visually pleasing real time simulations. However, since a cut of a coarse element is based the complete separation of the fine grid inside the coarse element, partial cuts are not supported. Moreover, as the simulation takes place on the coarse grid, the visual accuracy is not backed up by a similar level of accuracy for the coarse simulation grid.

The approaches in [16] and [24] are similar: if a tetrahedron is separated, each separated part is simulated through a duplication of the original element. Both methods can subdivide one tetrahedron into many small components, but they are physically incorrect and do not handle partial cuts of the elements.

3 Contributions

The contribution of this paper is a remeshing method that introduces a separation surface inside a FEM mesh with two main benefits. First, the number of inserted nodes (vertices) and elements (tetrahedra) is kept as low as possible to maintain real time performances. Secondly, the quality of the generated mesh is controlled, i.e. the shape of the introduced elements do not hinder the numerical aspects of the FEM method. Our remeshing algorithm creates smart approximations of the cutting surface. It can deal with complex cutting scenarios such as crossing cut planes (Fig. 1) and is compatible with classical snapping approaches.

4 Handling topological changes

As cuts, cracks or tears occur in the simulation, the mesh that supports the FEM model has to be adapted so that the simulation takes account of the expected phenomena and produces what we call the *mesh separation*. The *separating surface* S may be defined as the trajectory of a cutting tool or computed by a fracture algorithm describing the occurrences of tearing or shearing in the material. The tetrahedral elements traversed by this surface have to be cut, refined or rearranged to reflect the new physical state.

The first step of our method is the sampling or detection of the separation surface at the level of the edges of the FEM mesh. For each edge e of the mesh, we compute or estimate a cutting or breaking point and the normal of the separation surface at this point. The surface S may also pass through some of the vertices of the mesh. This special case is addressed in section 4.3.

The second step consists in a local remeshing of the model. Several refinement methods have been proposed in the computer graphics field. Most of them insert too many vertices [1,21] or introduce non tetrahedral elements [23]. The idea we defend here consists in the building of a set of triangles τ_j that approximate the separation surface. These triangles emerge from the initial mesh after a series of splits and flips on the tetrahedral elements.

The remeshing algorithm we present is an extension of the $\sqrt{3}$-subdivision scheme introduced by [4]. This

approach soundly and efficiently handles partial cuts, the emergence and propagation of cracks and the existence of overlapping or crossing between separation surfaces.

4.1 Approximating the separation surface

An edge that crosses the separation surface indicates that incident volumes are cut by the surface S. To take account of that, we introduce vertices in the tetrahedra incident to crossed edges. The newly introduced vertices, denoted v_k, are positioned on the surface S. The next steps proceed with the insertion of edges between the v_k and finally the insertion of the triangles τ_j connecting the newly introduced vertices.

Inserting vertices: The first step consists in subdividing every tetrahedron that is crossed, even partially, by the separation surface S. We use the *1-4 split*, replacing the initial tetrahedron by four new tetrahedra sharing the inserted vertex v_k (see figure 2).

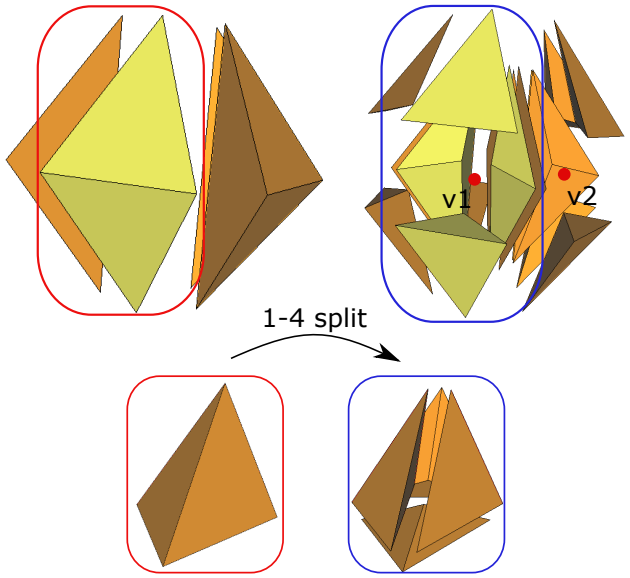


Fig. 2: A set of tetrahedra incident to an edge before (top left) and after (top right) the 1-4 split; (bottom) 1-4 split of a tetrahedron.

The 1-4 split allows for an arbitrary choice of the vertex position inside the volumes. The intersections of the edges with the separation surface are computed: If the separation surface crosses three or more edges e on a tetrahedron, we insert the new vertex at the barycenter of the intersections. If only one or two edge(s) of a tetrahedron are crossed, in the case of a partial cut, we use intersecting points between the edges of the tetrahedron and the extension of the separation surface to compute the barycenter. In the case of a further cut of the tetrahedron, the vertices will be moved to the updated barycenter following the same rules.

Inserting edges: The second step aims at creating edges between the inserted vertices. Let us consider the tetrahedra generated by the 1-4 splits around the initial edges e. They form a sequence of pairwise adjacent tetrahedra. Each pair of tetrahedra t_k, t_{k+1} share a face f, incident to e, and thus three vertices. Their fourth vertices are respectively v_k and v_{k+1}.

We perform a 2-3 flip around the face f, replacing two adjacent tetrahedra by three tetrahedra with the same vertices. The shared face is deleted, but the three new tetrahedra share the edge $\{v_k, v_{k+1}\}$. This way, the faces initially incident to e are replaced by a sequence of edges. These edges form a closed polygon lying on the surface S (see figure 3).

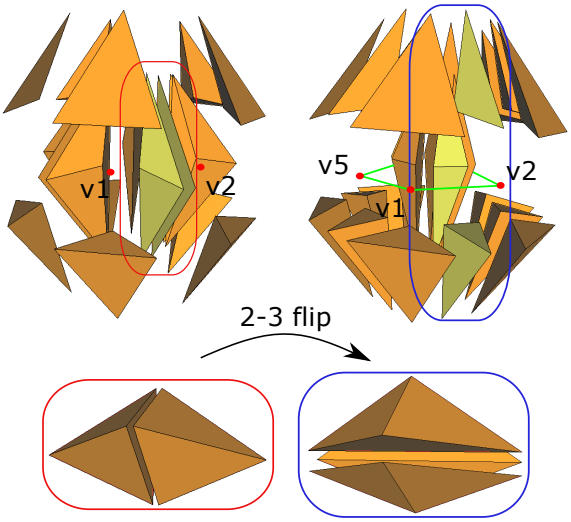


Fig. 3: A set of tetrahedra before (top left) and after (top right) the 2-3 flips. Flipping the faces incident to e creates a closed polygon $\{v_1, \ldots, v_n\}$ (draw in green); (bottom) 2-3 flip of two tetrahedra.

In order to perform the 2-3 flip, the connection between $\{v_k, v_{k+1}\}$ has to be completely inside the two neighboring tetrahedra. The vertices $\{v_k, v_{k+1}\}$ inserted by the 1-4 splits both depend on the intersection of the separation surface with the tetrahedras' edges. This choice helps to ensure that the 2-3 flip can be performed.

Inserting triangles: Each crossed edge e is now surrounded by a set of n tetrahedra that contain the ver-

tices of the crossed edge and two points of the polygon $\{v_1, \ldots, v_n\}$. The last step builds a set of $(n-2)$ triangles τ_j by triangulating the polygon $\{v_1, \ldots, v_n\}$.

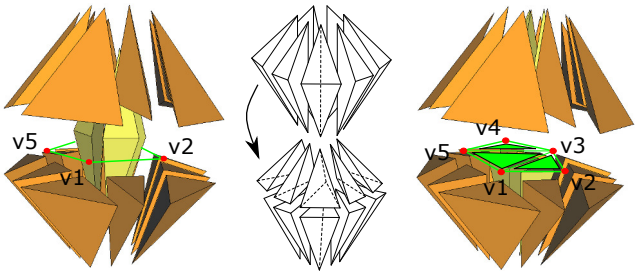


Fig. 4: To be cut edge surrounded by tetrahedra in yellow (left); general flip (middle); triangulation of the separation surface (right), showing the tetrahedra below the cut

As the vertices $\{v_k\}$ are placed to sample the separation surface, the triangles τ_j approximate the separation surface by construction. The n tetrahedra surrounding the edge are replaced by $2(n-2)$ tetrahedra defined by the three vertices of each τ_j and by one of the vertices of e (figure 4). In the literature, this operation is called a *general flip* or an *edge removal*.

4.2 Consideration of the boundary

When the separation surface crosses the boundary of the simulated object, a specific refinement of the boundary tetrahedra is needed to build this *cut line*. The additional refinement combines the previous tetrahedral remeshing with an extended $\sqrt{3}$-refinement of the boundary triangles.

Remeshing the boundary surface: In this paragraph, we describe the refinement of triangles of the boundary surface, ignoring the tetrahedra behind them (figure 5). The edges that are crossed by the cut line are first selected. New vertices are inserted on the cut line in the adjacent triangles that are split into three. Then the selected edges, except the boundary edges, are flipped to link the new vertices. The flipped edges define a polygonal line that smartly approximates the cut line. The boundary edges are finally split at their intersections with the cut line.

Remeshing the boundary tetrahedra: The refinement of boundary tetrahedra follows this scheme, but uses volumetric operators replacing the operators of the two dimensional case. Similar to the case for edges inside the object, a crossed edge induces a 1-4 split. Then

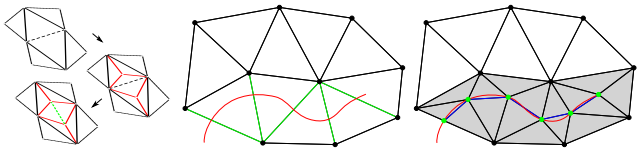


Fig. 5: An outline of the $\sqrt{3}$-refinement on two triangles (left); the edges crossed by the cut line are selected (middle); the adjacent triangles are refined to approximate the curve (right). The extremities of the cut line are obtained with the insertion of vertices on the boundary edges.

we deal separately with the remeshing of the boundary tetrahedra: The 1-3 split of triangles is replaced by a 1-3 split extended to tetrahedra shown in figure 6 (left). We obtain a polygon of $\{v_k\}$ with vertices on the separation, but in contrast to the inner edges, that polygon is opened, between the points inserted by the 1-3 split.

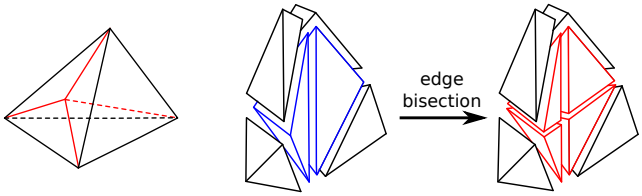


Fig. 6: 1-3 split of a tetrahedron (left): a vertex is inserted in the boundary triangle; bisection of a boundary edge: the incident tetrahedra are split (right)

Since more than two tetrahedra may be incident to a boundary edge, the replacement of the edge flip on the boundary faces is not straightforward. We use the general edge flip illustrated in figure 4 (left) to replace the n tetrahedra incident to the boundary edge with $2(n-2)$ tetrahedra. This operator inserts an edge that closes the polygon between the vertices inserted into the two boundary triangles as shown in figure 7 (middle top) and approximates the separation surface by a set of triangles τ_j.

The general flip removes the initial boundary edge, which may locally smooth the surface. If this edge belongs to an important feature of the mesh this could be undesirable. In that case, we replace the general flip by the bisection of the boundary edge illustrated on figure 6 (right). The bisection introduces a new vertex v at the intersection of the edge with the separation surface. The new vertex v is used to close the polygon. Again, we end up with tetrahedra, that approximate the separation surface with a set of triangles τ_j. This case is illustrated in figure 7 (middle bottom) and (right).

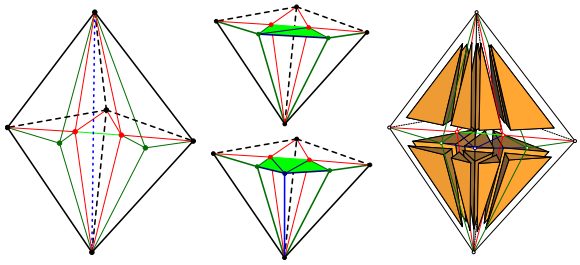


Fig. 7: Remeshing around a boundary edge (*dotted blue line*): 14 splits are applied to all tetrahedra (new edges are plot as *red lines*), followed by 13 splits on the boundary tetrahedra and 2-3 flips for the inner tetrahedra (new edges are plot as *green lines*) (left); then, the boundary edge is replaced by a general flip (middle top) or split to maintain a desired feature (middle bottom); the final remeshing on the boundary of the object after a split of the edge (right)

4.3 Disconnection of the mesh

At this point, the separation surface S is approximated with inserted vertices, edges and triangles built from the set of crossed edges. This approach is tailored to the case where S crosses the edges far away from their extremities. When the surface S passes near vertices of the FEM mesh, it may be desirable to move these vertices to the surface or *snap* them instead of inserting unnecessary elements and vertices. The presented remeshing approach is compatible with classical snapping strategies, like the one presented by [18]. Their combination is presented hereafter, before the disconnection step is discussed.

Snapping to the separation surface: Intersections of the separation surface S with an edge e either yield a remeshing or result in a movement of an existing vertex to the separation surface. In order to decide which alternative is chosen, we introduce a parameter ϵ.

If an intersection occurs, we obtain the position p of the intersection and we save the normal n of the separation surface S at the intersection. Then we calculate the distance of the vertices of the edge e to the plane through the point p with the normal n. We compare the distance to the threshold ϵ multiplied by a mesh parameter – in our case the average length of the edges in the rest position of the mesh. Finally, a vertex will be snapped or moved to p, if its distance is smaller than the distance of the other vertex on the edge and is below the value mentioned above. We will discuss the choice of the threshold in the section on the results.

Unsewing over the separation surface: To allow for a fine management of the topological relations linking cells (vertices, edges, faces, volumes) in the mesh, we use a specific volumetric data structure based on combinatorial maps. An efficient implementation of this model is proposed in the CGoGN library [15]. It provides tools to manipulate the topology of meshes and an attribute manager for the different cells. A simple example for the usage of an attribute is the position of a vertex.

An important property of combinatorial maps is that they explicitly represent the volumetric binding between the two faces of adjacent tetrahedra. In order to separate our mesh at the separation surface S we unsew that link as soon as a face f is known to be on the separation surface S. To track this information during the remeshing, we use markers in the way described in the following paragraph.

Each vertex introduced on the separation surface (by the previously presented operators) is marked. The snapped vertices are marked the same way. As soon as the three vertices of a face are marked, its volumetric binding is deleted. This operations results in a hole between the two elements sharing this face, but, at this stage, their vertices are still connected.

Finally, as soon as a cycle of adjacent faces, incident to a same vertex, are disconnected, the vertex is separated into two vertices triggering an update of the FEM part. This last condition is automatically checked by the CGoGN library.

4.4 Handling successive cuts

After carrying out the topological changes introduced in the preceding subsections, we obtain a volumetric mesh naturally supporting an occurrence of new separations. This allows our method to separate initial and newly inserted edges several times.

However, each additional separation of an edge reduces the edge length, which can result in ill-shaped elements or edges with greatly differing lengths. Our algorithm prevents this negative impact using the threshold ϵ introduced in the last subsection: as soon as the edge length decreases beyond the threshold multiplied by the average length of the edges, the snapping of vertices is performed, avoiding a further subdivision and additional computational cost.

Beyond that, our algorithm can be used iteratively to incorporate a separation surface into a volumetric mesh with higher detail: This can be achieved by using the operations mentioned beforehand repetitively – but inserting the nodes of the 1-4 split at the barycenter of the tetrahedron – in the elements that intersect with the separation surface. Finally, at the highest level of refinement, the vertices of the 1-4 split are inserted on the

separation surface, allowing a detailed representation of the separation surface inside the volumetric mesh. Our approach is specifically interesting in the context of iterative refinement, since a refinement around an edge only has an impact on the tetrahedra around the edge, but not on their neighbors – an advantage that has also been presented in [14]. However the nodes inserted by the iterative refinement that are not on the separation surface go along with an additional computational cost, that can endanger the real-time aspect of our approach. Therefore, we forgo using the iterative approach in the context of this work.

5 Results

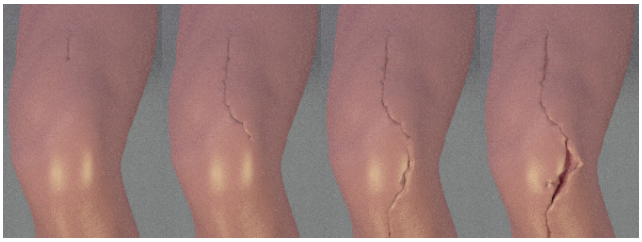
We implemented different examples, based on the proposed method, using the SOFA [9] open source framework. We perform interactive cuts on soft tissues models: for example on a knee in figure 8a or on a liver in figure 8b . In addition, we show that a combination of cutting and breaking of a cylinder can be realized using our method, see figure 8c.

The quality of the remeshing – regarding real-time FEM simulation – can be measured with the number of nodes and the *condition number* of the system that has to be solved. Thus, we use these two measures to demonstrate the benefits of our method.

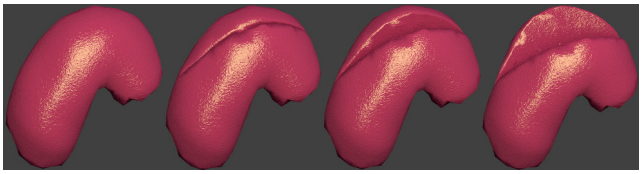
We choose to compare our approach with the approach followed by Bielser and al. [2,3] based on the 1:17 subdivision of the tetrahedra traversed by the separation surface. Introduced in the framework of mass-spring systems, this approach has the advantage of being compatible with the FEM based simulations as the built meshes are conformal. It supports the same range of cutting scenarios than our approach.

Moreover we compare our approach to the recent work of Koschier [14] which aims at enhancing the granularity of a mesh to generate detailed fractures without a limitation of the inserted nodes. The beginning of one refinement step of this method (1-4 splits followed by 2-3 flips) is similar to our approach and has its origin in [4]. However, the last steps differ: The work of Koschier inserts two nodes on each cut edge, where our approach avoids a node insertion and limits the number of nodes. Since this is an important property to maintain real-time performance, it will be considered in the following.

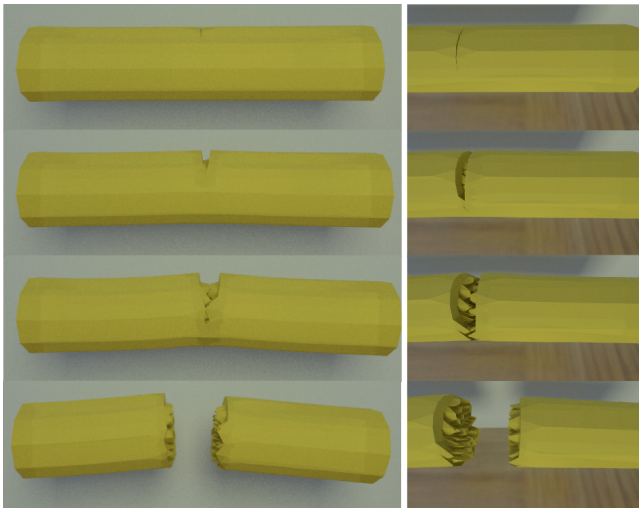
Hereafter, we give a theoretical analysis of the number of nodes introduced by Bielser's, Koschier's and our remeshing methods. Then, we present experimental results showing the evolution of the numerical quality of the meshes during a simulation considering a cut on a deformable beam.



(a) Cut in knee surgery, with the opening of the wound



(b) Advancement of a cut in liver surgery



(c) A cut that results in a propagating crack

Fig. 8: Virtual cuts produced with our method. **a** Cut in knee surgery, with the opening of the wound. **b** Advancement of a cut in liver surgery. **c** A cut that results in a propagating crack

5.1 Theoretical analysis

The remeshing impacts the FEM simulation in two ways: a local stiffness matrix must be computed for each modified element and solving the resulting linear system depends (at best) linearly on the number of nodes in the new mesh. In the following we evaluate these quantities for our remeshing method and for the methods proposed by Bielser and Koschier.

Let us consider a tetrahedral mesh and a cut surface that traverses n_T adjacent tetrahedra. For that, let n_V be the number of added nodes and $n_{T'}$ the number of tetrahedra that replace the initial n_T tetrahedra during the remeshing process. To simplify the calculations, we

assume that an average of 5 tetrahedra are adjacent to each edge and that the cutting plane intersects all tetrahedra at three (case 1) or four edges (case 2). The reality lies between these two cases.

Let us evaluate n_E the number of cut edges and n_F the number of cut faces. In the case 1, 3 edges are cut for each tetrahedron. It follows that $n_E = \frac{3}{5} \times n_T$ and $n_F = \frac{3}{2} \times n_T$. In the case 2, 4 edges are cut for each tetrahedron resulting in $n_E = \frac{4}{5} \times n_T$ and $n_F = \frac{4}{2} \times n_T$.

In the approach followed by Bielser, a node is inserted on the middle of each cut edge and face. Thus, in the case 1, $n_V = n_E + n_F = \frac{21}{10} \times n_T$ and in the case 2, $n_V = n_E + n_F = \frac{28}{10} \times n_T$. In one refinement step of the method presented by Koschier, one vertex is inserted for each cut tetrahedron and two vertices on each cut edge. We obtain $n_V = n_T + 2n_E = \frac{11}{5} \times n_T$ for case 1 and $n_V = n_T + 2n_E = \frac{13}{5} \times n_T$ for case 2. With our method, in either case, only one vertex by tetrahedron is inserted, during the $1-4$ split, and $n_V = n_T$.

The number of tetrahedra introduced by the $1:17$ subdivision is, $n_{T'} = 10 \times n_T$, in the case 1, and $n_{T'} = 16 \times n_T$, in the case 2. On average $n_{T'} \approx 13 \times n_T$. In the Koschier method the number of tetrahadra is $n_{T'} \approx 14 \times n_T$.

With our method, in the case 1, the n_T cut tetrahedra become $4 \times n_T$ tetrahedra after the 1-4 split, $n_T + 3 \times n_T \times \frac{3}{2}$ after the 2-3 flip, and $n_T + 3 \times n_T \times \frac{3}{2} \times \frac{2 \times 5 - 4}{5}$ after the last general flip. Thus, $n'_T = 6.4 \times n_T$ after these operations. In the case 2, the n_T cut tetrahedra become $4 \times n_T$ tetrahedra after the 1-4 split, $4 \times n_T \times \frac{3}{2}$ after the flip 2-3, and $4 \times n_T \times \frac{3}{2} \times \frac{2 \times 5 - 4}{5}$ after the last step. Thus, $n'_T = 7.2 \times n_T$ after these operations.

Our method is far more efficient as it introduces ≈ 2.5 time less nodes and ≈ 2 time less tetrahedra than Bielser or Koschier. The theoretical analysis implies that the proposed algorithm is a better match for real-time simulations. The experimental results confirm this observation.

5.2 Experimental results

This section shows the results we obtained in experiments performed to evaluate and compare the proposed method with the approach proposed by Bielser. In this scenario a deformable beam is progressively cut. The beam we consider initially has 371 tetrahedral elements and 131 nodes. The cut advances step by step, separating the beam in two pieces (see figure 9). We chose a cut plane close to the initial nodes in order to challenge the methods that we compare.

To make a sound statement about our method, we combined the approach of Bielser with snapping. The

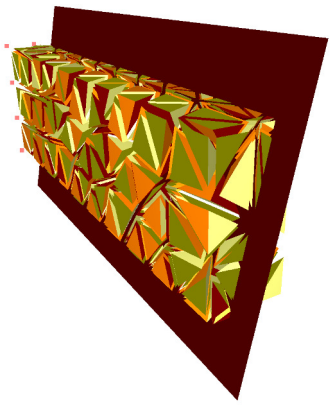


Fig. 9: Cutting a deformable beam

snapping of nodes is performed, based on the threshold ϵ. We evaluated the two remeshing methods with different values for ϵ. The results are plotted in figures 10 and 11. In these figures, the x-axis is the simulation step and is directly related to the number of tetrahedra that have been cut by the moving plane.

Figure 10 shows the evolution of the total number of nodes in the mesh. As expected, our method introduces less vertices than Bielser, even when the snapping is enforced.

Figure 11 show the evolution of the condition number of the systems. The condition number is a measure to evaluate the numerical quality of the systems. A higher condition number results in more iterations to solve a numerical system, negatively impacting the performance of the simulation. The figure shows that the decreasing quality of the elements' shape introduced along the separation surface is intimately involved in the evolution of the condition number.

With the Bielser method, the condition number increases and with that the numerical quality deteriorates quickly. An increase of the snapping threshold ϵ delays that deterioration, but the final performance remains poor. The benefits of our method are clearly visible here – the numerical quality of the mesh is much better controlled. The snapping clearly improves the performance, as it introduces less nodes and the condition number remains lower. But even if the number of nodes has been doubled, the simulation still run with real-time performance. The example demonstrates that our approach allows the system to stay well-conditioned even in case of large cuts.

Figure 12, shown the built separation surface with both remeshing methods. We can see the source of the difficulties: with the Bielser method, there is a greater difference between the smallest and the greatest size of triangles on the surface of the cut. With our new

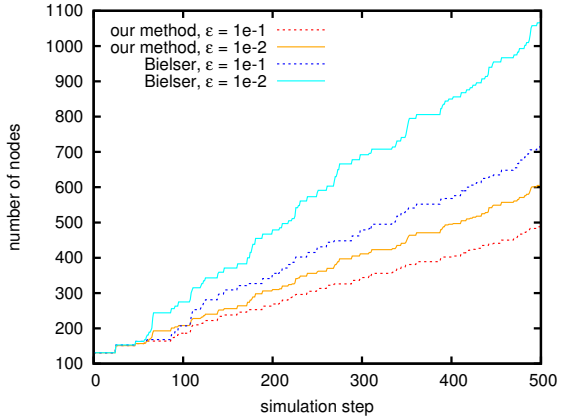


Fig. 10: Evolution of the number of nodes

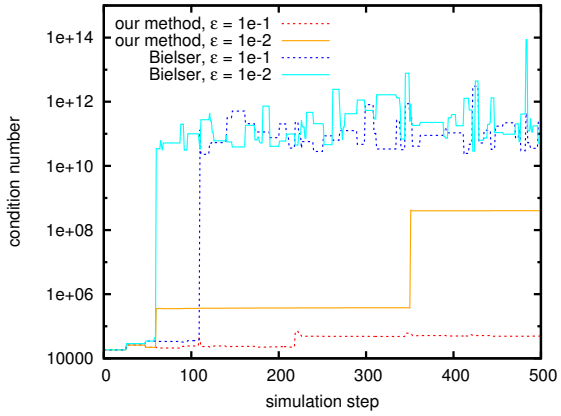


Fig. 11: Evolution of the condition number

remeshing algorithm, the size of the triangles – and thus the inner tetrahedra – are globally more homogenous.

6 Conclusion

In this paper we have addressed the challenge of efficient cutting simulation in the context of real-time soft tissue modeling. Our approach relies on a new remeshing algorithm, that introduces well-shaped elements that produce well-conditioned system matrices. During the cutting process, the number of added degrees of freedom is well controlled. For cuts close to nodes, we improve the performance and stability of our remeshing method with the snapping of vertices. The combination of the ideas has been implemented on top of a real-time finite element method. The implementation keeps performances of the simulation over time. The remeshing method has the ability to create an adaptive approximation of the cutting or tearing surface, which gives the ability to tune between the flexibility and speed of the remeshing.

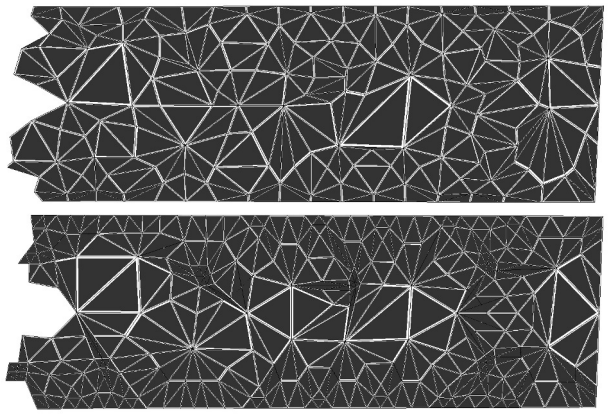


Fig. 12: The separation surface: (top) with our remeshing method; (bottom) with Bielser approach

The position of points on the separation surface (and in its neighborhood) could be improved. Points are currently inserted and placed on the separation surface using geometric constraints. For improved numerical stability and efficiency, we plan to use a fast relaxation process in order to optimize locally the shape of the elements.

As it has been mentioned in section 4.4, our algorithm has very interesting properties for an iterative or multiscale refinement of a mesh in order to incorporate the separation surface into the volumetric grid. It could be very interesting to evaluate these properties in a wider scale and to allow for real-time performance in the context with the iterative or multiscale approach.

References

1. Bey, J.: Tetrahedral grid refinement. Computing **55**(4), 355–378 (1995)
2. Bielser, D., Glardon, P., Teschner, M., Gross, M.H.: A state machine for real-time cutting of tetrahedral meshes. Graphical Models **66**(6), 398–417 (2004)
3. Bielser, D., Maiwald, V.A., Gross, M.H.: Interactive cuts through 3-dimensional soft tissue. Computer Graphics Forum **18**(3), 31–38 (1999)
4. Burkhart, D., Hamann, B., Umlauf, G.: Adaptive and feature-preserving subdivision for high-quality tetrahedral meshes. Computer Graphics Forum **29**(1), 117–127 (2010)
5. Chentanez, N., Alterovitz, R., Ritchie, D., Cho, L., Hauser, K.K., Goldberg, K., Shewchuk, J.R., O'Brien, J.F.: Interactive simulation of surgical needle insertion and steering. ACM Trans. Graph. **28**(3), 88:1–88:10 (2009)
6. Cotin, S., Delingette, H., Ayache, N.: A hybrid elastic model for real-time cutting, deformations, and force feedback for surgery training and simulation. The Visual Computer **16**(8), 437–452 (2000)
7. Courtecuisse, H., Allard, J., Kerfriden, P., Bordas, S.P.A., , Cotin, S., Duriez, C.: Real-time simulation of

contact and cutting of heterogeneous soft-tissues. Medical Image Analysis **18**(2), 394–410 (2014)
8. Dick, C., Georgii, J., Westermann, R.: A hexahedral multigrid approach for simulating cuts in deformable objects. Visualization and Computer Graphics, IEEE Transactions on **17**(11), 1663–1675 (2011)
9. Faure, F., Duriez, C., Delingette, H., Allard, J., Gilles, B., Marchesseau, S., Talbot, H., Courtecuisse, H., Bousquet, G., Peterlik, I., et al.: Sofa: A multi-model framework for interactive physical simulation. In: Soft Tissue Biomechanical Modeling for Computer Assisted Surgery, pp. 283–321. Springer (2012)
10. Felippa, C., Haugen, B.: A unified formulation of small-strain corotational finite elements: I. theory. Computer Methods in Applied Mechanics and Engineering **194**(21), 2285–2335 (2005)
11. Ganovelli, F., Cignoni, P., Montani, C., Scopigno, R.: A multiresolution model for soft objects supporting interactive cuts and lacerations. Computer Graphics Forum **19**(3), 271–281 (2000)
12. Hackbusch, W., Sauter, S.: Composite finite elements for the approximation of pdes on domains with complicated micro-structures. Numerische Mathematik **75**(4), 447–472 (1997)
13. Kohn, L.T., Corrigan, J.M., Donaldson, M.S., et al.: To err is human: building a safer health system, vol. 627. National Academies Press (2000)
14. Koschier, D., Lipponer, S., Bender, J.: Adaptive tetrahedral meshes for brittle fracture simulation. In: Proceedings of the 2014 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. Eurographics Association (2014)
15. Kraemer, P., Untereiner, L., Jund, T., Thery, S., Cazier, D.: CGoGN: N-dimensional meshes with combinatorial maps. In: 22nd International Meshing Roundtable, pp. 485–503. Springer International Publishing (2013)
16. Molino, N., Bao, Z., Fedkiw, R.: A virtual node algorithm for changing mesh topology during simulation. In: ACM SIGGRAPH 2005 Courses, SIGGRAPH '05. ACM, New York, NY, USA (2005)
17. Mor, A., Kanade, T.: Modifying soft tissue models: Progressive cutting with minimal new element creation. In: S. Delp, A. DiGoia, B. Jaramaz (eds.) Medical Image Computing and Computer-Assisted Intervention MICCAI 2000, *Lecture Notes in Computer Science*, vol. 1935, pp. 598–607. Springer Berlin Heidelberg (2000)
18. Nienhuys, H.W., Frank van der Stappen, A.: A surgery simulation supporting cuts and finite element deformation. In: W. Niessen, M. Viergever (eds.) Medical Image Computing and Computer-Assisted Intervention (MICCAI) 2001, *Lecture Notes in Computer Science*, vol. 2208, pp. 145–152. Springer Berlin Heidelberg (2001)
19. O'Brien, J.F., Hodgins, J.K.: Graphical modeling and animation of brittle fracture. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, pp. 137–146. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (1999)
20. Payan, Y.: Soft tissue biomechanical modeling for computer assisted surgery, vol. 11. Springer (2012)
21. Rivara, M.C.: Local modification of meshes for adaptive and/or multigrid finite-element methods. Journal of Computational and Applied Mathematics **36**(1), 79–89 (1991). Special Issue on Adaptive Methods
22. Sauter, S., Warnke, R.: Composite finite elements for elliptic boundary value problems with discontinuous coefficients. Computing **77**(1), 29–55 (2006)
23. Schaefer, S., Hakenberg, J.P., Warren, J.: Smooth subdivision of tetrahedral meshes. In: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on geometry processing, SGP '04, pp. 147–154. ACM, New York, NY, USA (2004)
24. Sifakis, E., Der, K.G., Fedkiw, R.: Arbitrary cutting of deformable tetrahedralized objects. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '07, pp. 73–80. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2007). URL http://dl.acm.org/citation.cfm?id=1272690.1272701
25. Stanculescu, L., Chaine, R., Cani, M.P., Singh, K.: Sculpting multi-dimensional nested structures. Computers & Graphics **37**(6), 753–763 (2013)
26. Steinemann, D., Harders, M., Gross, M., Szekely, G.: Hybrid cutting of deformable solids. In: Virtual Reality Conference, 2006, pp. 35–42 (2006)
27. Wu, J., Dick, C., Westermann, R.: Interactive high-resolution boundary surfaces for deformable bodies with changing topology. In: Proceedings of 8th Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS) 2011, pp. 29–38 (2011)
28. Wu, J., Westermann, R., Dick, C.: Physically-based simulation of cuts in deformable bodies: A survey. In: Eurographics 2014 State-of-the-Art Report, pp. 1–19. Eurographics Association, Strasbourg, France (2014)
29. Wu, J., Westermann, R., Dick, C.: Real-time haptic cutting of high resolution soft tissues. Studies in Health Technology and Informatics (Proc. Medicine Meets Virtual Reality 2014) **196**, 469–475 (2014). Published by IOS Press
30. Zhang, Y., Hughes, T., Bajaj, C.: An automatic 3d mesh generation method for domains with multiple materials. Computer methods in applied mechanics and engineering **199**(5), 405–415 (2010)