4th Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007)
J. Dingliana, F. Ganovelli (Editors)

Constraint Sets for Topology-changing Finite Element Models

Marc Gissler Markus Becker Matthias Teschner

University of Freiburg

Abstract

We propose constraint sets as an efficient data structure for topology-changing deformable tetrahedral meshes. Using constraint sets, data structure updates in case of topology changes are simple and efficient. The consistency of the geometric representation is maintained and elasto-mechanical properties of the object are preserved. In combination with a Finite Element model for elasto-plastic objects and a geometric constraint approach, constraint sets are applied to simulate the merging and breaking of conforming and non-conforming tetrahedral meshes. Experiments illustrate the efficiency of the data structure in interactive applications and its versatility.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism: Animation

Keywords: Physically-based simulation, deformable modeling, topology changes, FEM, geometric constraints

1. Introduction

Tetrahedral meshes are a convenient representation for deformable objects. However, if topological changes such as cutting [SDF07, SHGS06, BGTG04, GCMS00, CDA00, FDA02, NvdS01], fracturing [OH99, MTG04], or merging [BWHT07] are incorporated into a dynamic simulation, complex data structure updates have to be performed in order to maintain a consistent tetrahedral mesh and to preserve the elasto-mechanical properties. Although the requirements of such an update are well-defined, the actual implementation can be error-prone. Further, the update can cause differing computing times per simulation step which can be disturbing in interactive applications.

In this paper, we propose an efficient data structure for topology-changing tetrahedral meshes. Instead of processing mass points, a mass point is replaced by a constraint set that comprises a distinctive mass portion for each adjacent tetrahedron. Using constraint sets, topology changes of conforming meshes are simplified to inserting or deleting mass portions from a constraint set. Meshes resulting from these simple operations are always consistent and elasto-mechanical properties are preserved without any additional effort. In addition to conforming meshes, the data structure can also handle non-conforming meshes by inserting additional constraints into a set of mass portions. Constraint sets are independent of the underlying deformation model and the constraint method for non-conforming meshes. Although constraint sets cause memory overhead, the consistent and property-preserving data structure update in case of topological changes is very simple and fast. The risk of implementation errors and differences in computing times per simulation step are minimized.

Throughout the paper, we consider constraint sets in combination with a linear co-rotational Finite Element approach for deformable objects [MG04, HS04]. For non-conforming meshes, constraint sets are combined with a geometric constraint approach [GBT06]. After the discussion of related research and the detailed explanation of the data structure and its implementation, we illustrate the properties and the capabilities of constraint sets. We show that mass points can be replaced by constraint sets without changing the elasto-mechanical properties or the dynamic behavior of objects. Further, we investigate the memory consumption and the computing costs related to topology changes. The versatility of the proposed data structure is illustrated by breaking and merging conforming and non-conforming tetrahedral meshes.

2. Related Work

In 1987, Terzopoulos et al. [TPBF87] introduced the physically-based modeling of deformable solids to the graphics community. They used a finite difference scheme to discretize the underlying partial differential equations. In [TF88b, TF88a], the model was extended to handle plasticity and topological changes due to fracture. Starting with early finite element works of [GTT89] and [CZ92], the finite element method has constantly been refined. Recent advances for the finite element method include adaptive sampling of [DDCB01] and the stiffness warping approach of [MDM*02]. [ITF04] introduced a modified constitutive model to handle inverted finite elements. Plastic deformation has also been dealt with by [MG04], who adapted the model of [OBH02] to the corotated linear FE model.

In terms of topological changes, various methods for splitting meshes have been proposed. E. g., [MBF04, SDF07] duplicate elements and assigns partial volumes to them, thereby avoiding ill-shaped primitives. [OH99] generates new tetrahedral elements by splitting existing ones. He duplicates nodes where fractures originate and splits the simulation mesh based on fracture planes. In [NvdS01], a data structure for cutting tetrahedral meshes on existing faces based on simplicial complexes is proposed. In contrast to our method this approach relies on both a simplex model for the topological changes as well as an conventional object representation which has to be updated as well. Additionally this model does not allow for non-conforming meshes.

Various data structures for tetrahedral meshes have been proposed in the past that are mostly optimized with respect to storage costs or topological navigation. For a thorough discussion see [FH05].

Our data structure is well-suited for topological changes when the splitting is performed on existing edges. In this case, we do not need to duplicate structures and the consistency of the mesh is easily and efficiently preserved. If elements need to be duplicated as in [MBF04], they can easily be integrated into our data structure.

Further, non-conforming boundaries between merging meshes can easily be incorporated into our data structure using constraint handling as proposed e. g. in [GBT06] and [SSIF07]. Both approaches insert virtual points in order to be able to handle T-junctions. We use this concept especially when merging non-conforming meshes.

Previous approaches for constrained boundary handling at interfaces have used Lagrange multipliers [QB95] in the context of non-conforming FE meshes or exceptional points [LC06] for T-junctions in 2D Lagrangian hydrodynamics. Lagrange multipliers however, introduce additional degrees of freedom in the system. Expectional points on the other hand have their velocity simply enslaved to neighboring regular points.

Only a few authors have dealt with the merging of meshes in dynamic simulations. [BWHT07] presents a merging approach in the context of visco-plastic flow. As soon as two or more objects are in persistent contact and undergo large deformation they merge and the objects are re-meshed at the same time. In contrast to [BWHT07], we do not re-mesh merging objects. Although this restricts the range of deformation, the computational overhead for merging and separating of meshes is negligible using our data structure. Similar to fracture and merge is the addition and removal of material in virtual clay simulations, e.g. [CA06, DC04]. The clay surface is defined as the iso-surface of a scalar field stored in a 3D grid. To our knowledge, however, there exists no virtual clay simulation that uses tetrahedral meshes.

A number of publications have addressed mesh-free methods for fracturing and merging [PKA*05] or splitting [SOG06], Clavet et al. [CBP05] introduced a particle system with springs being dynamically added or removed to connect particles and thereby simulating visco-elastic materials. However, as the proposed data structure is based on tetrahedral meshes, we have concentrated on mesh-based approaches.

3. Constraint Sets

In this section, we describe the data structure. First, the concept of constraint sets is introduced for conforming meshes. Aspects related to force computations, object dynamics, topology changes, and computational overhead are discussed. Extensions for non-conforming meshes are described followed by implementation details.

3.1. Conforming meshes

Constraint sets can be used with objects that are discretized into mass points with a defined topology represented with basic primitives. In such a scenario, each mass point is replaced with a constraint set that is comprised of n mass portions that represent the mass of the n adjacent object primitives (see Fig. 1). The overall mass of a constraint set equals the mass of the replaced point. Basically, a mass point is divided into a set of mass points that are constrained to each other. In terms of dynamic simulations, the processing of all relevant attributes is transferred from the original mass point to the constraint set (see Fig. 2). Merging of object primitives now conforms to merging mass portions from different constraint sets into one set. Splitting of primitives is realized by dividing mass portions from one constraint set into different sets.

In the case of tetrahedral meshes, the number of mass portions per constraint set is given by the number of tetrahedrons adjacent to a mass point. The mass of an element of a constraint set is given by the respective fracture of mass of the respective tetrahedron. Thus, the accumulated mass in a constraint set equals the mass of a point that is initialized using a density approach. For a single tetrahedron, each constraint set contains one mass portion that corresponds to the replaced mass point.

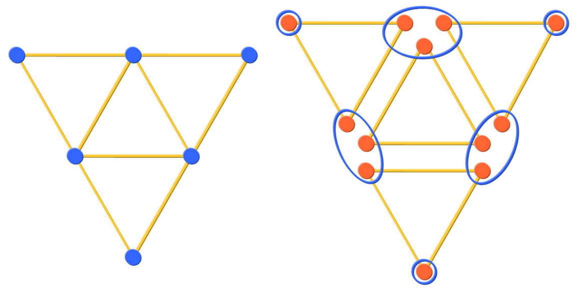


Figure 1: *A set of mass points with a defined topology represented with basic primitives i.e. triangles (left). All mass points are replaced by constraint sets that contain a number of mass portions (right). The number of mass portions in a constraint set depends on the number of adjacent primitives.*

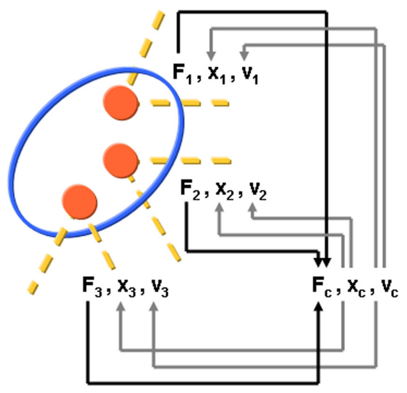


Figure 2: *Forces F_i computed at the mass portions are accumulated in a constraint set. After a topology change, position* $\mathbf{x}_c$ *and velocity* $\mathbf{v}_c$ *are communicated to all mass portions in the constraint set.*

3.2. Dynamics

To compute internal forces, we employ a linear, co-rotational Finite Element approach for elasto-plastic objects [MG04, HS04, OH99]. The computed forces are entirely based on the spatial configuration of a tetrahedron. Since masses are not considered, the accumulated internal force in a constraint set equals the internal force that would have been accumulated at the replaced mass point. External forces are computed per constraint set. Thus, the resulting forces at a constraint set and at the replaced mass point are equal. These characteristics guarantee that constraint sets do not influence the elasto-mechanical properties of an object and its dynamic behavior.

Constraint sets can easily be combined with deformation models that are independent of the actual mass of affected mass portions, i. e. FEM and mass-spring. If mass-dependent forces are computed for an object primitive, the overall mass of a constraint set has to be considered instead of the respective mass portion in order to guarantee unaltered forces compared to the original object representation without constraint sets.

The numerical integration processes constraint sets instead of mass portions. Therefore, the overall mass and the overall force of a constraint set is considered. Integrated properties of a constraint set, such as position and velocity, are assigned to all mass portions. Since the overall mass and force of a constraint set equals mass and force of the replaced mass point, constraint sets do not influence the dynamics of an object. In our applications, the Verlet integration is used.

3.3. Topological changes

Updates of data structures in case of topological changes are now reduced to inserting mass portions into constraint sets in case of merging or deleting mass portions from constraint sets in case of breaking. This is particularly interesting for the merging and breaking of objects with different densities. In this case, the explicit update of masses is avoided by processing the respective mass portions within a constraint set.

3.4. Computational aspects

Constraint sets are basically abstract containers for mass portions that carry all information. Thus, the required memory for constraint sets is given by the number of mass portions which is given by the number of primitives. In case of a tetrahedral mesh, four mass portions are required per tetrahedron. This number is fixed and independent of the topology of an object. The topology just defines the number of constraint sets and the number of mass portions per set. The memory requirement is linear in the number of primitives.

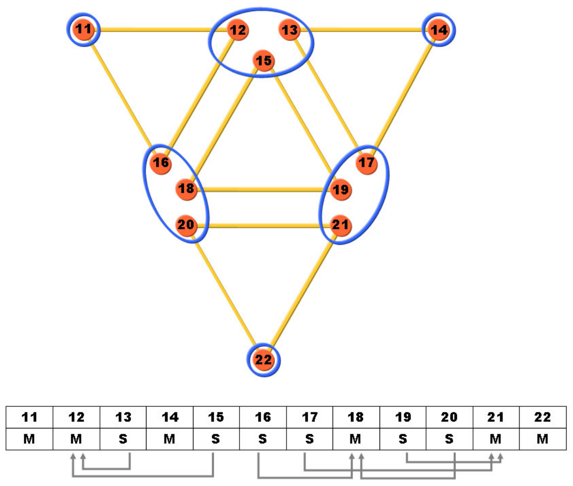
Although there is some memory overhead for constraint sets, their overhead in terms of computational costs is negligible. It does not make any difference whether to accumulate forces into a constraint set or into the replaced mass point. Forces are calculated exactly the same way in both scenarios. The numerical integration is performed once for each constraint set that replaces an original mass point. Thus, the same number of elements is integrated in both scenarios. The only overhead in the constraint set data structure is the propagation of the integration result (position and velocity) to all mass portions of a constraint in case of a topology change.

3.5. Non-conforming meshes

The handling of non-conforming meshes is an important issue in the simulation of merging objects. In order to allow for non-conforming meshes, additional constraints can be attached to a constraint set. In our application, point-to-line, point-to-face, and point-to-tetrahedron constraints can be considered within a constraint set. The approach is inspired by [GBT06] and constrains a constraint set position to a position on a line, face, or tetrahedron.

3.6. Implementation

Containers for constraint sets might constitute memory overhead with redundant information on forces, positions, and velocities, accumulated from or to be transferred to their mass portions. Further, constraint set containers have to be generated or deleted dynamically in case of topological changes. As an alternative to an explicit container representation, constraint sets can simply be realized by adding a state and a pointer to the original mass point data structure. Exactly one arbitrary mass portion within a constraint set is defined to be a master. All other mass portions are slaves and have a pointer to their master. Now, a constraint set either consists of a master or of a master with a number of slaves (see Fig. 3). This concept enables the communication



11	12	13	14	15	16	17	18	19	20	21	22
M	M	S	M	S	S	S	M	S	S	M	M

Figure 3: *All mass portions of an object are stored in an array. One arbitrary mass portion is declared master of a constraint set. All other mass portions of the respective constraint set store a pointer to its master in the array.*

of forces, positions, and velocities between a master and its slaves and the numerical integration only considers masters. With the master-slave concept, dynamic data structures are avoided since the number of mass portions is constant and independent of any topology change. If two constraint sets are merged, one master is turned into a slave and all slaves point to the remaining master. If a constraint set is split, the group of mass portions containing the master does not have to be processed. In the second group, a master is defined and the pointers of the slaves are updated accordingly.

4. Results

In order to illustrate the properties and the performance of the proposed data structure, we discuss four scenarios. All experiments have been performed on an Intel Core 2 PC, 2.13 GHz with 2.0 GB of memory.

In the first scenario, two block-shaped tetrahedral meshes with identical geometry are attached to a wall. The cuboid on the right-hand side is represented with mass points. For the cuboid on the left-hand side, constraint sets have been employed. Fig. 4 shows three frames of a dynamic simulation with varying elasto-mechanical properties. The images show the identical dynamic behavior of both objects. This illustrates that the elasto-mechanical properties of the FE model are preserved if mass points are replaced by constraint sets.

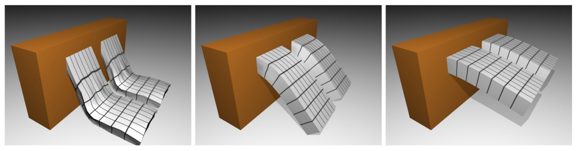


Figure 4: *Two cuboids with identical geometry are simulated using constraint sets (left) and mass points (right). In the three images, the Youngt's Modulus is varied from* $5kN/m^2$ *(left),* $500kN/m^2$ *(middle), to* $5000kN/m^2$ *(right). The Poisson ratio is always 0.3.*

Each cuboid consists of 2400 tetrahedrons. Masses are represented with 748 nodes and with 748 constraint sets with 9600 mass portions, respectively. The computation of the internal forces takes 21 ms for each cuboid. The integration takes 0.08 ms for the nodes and 0.13 ms for the constraint sets. The propagation of velocities and positions from all 748 masters to their 8852 slaves takes 1.1 ms. The propagation, however, is only required in case of a topology change and is only performed for a small number of affected constraint sets. These measurements illustrate that the computational overhead of constraint sets is negligible.

In a second scenario, the application of constraints to fracturing objects is shown (see Fig. 5). The teddy consists of 4300 tetrahedrons, 1k constraint sets, 17k mass portions. Forces and numerical integration are computed in 40 ms. If a connection within the tetrahedral mesh breaks down, constraint sets are updated by re-organizing the respective mass portions. The computational overhead for the data structure update including propagation is 0.005 ms. This means that in case the object fractures into single tetrahedrons, the computational costs for the update would be 5 ms. As the number of splits is in general much lower than the number of nodes, this overhead is negligible compared to the computation of internal forces and numerical integration. The propagation of velocities and positions is only performed for a small number of affected constraint sets. This results in constant computing times for each time step throughout the simulation.

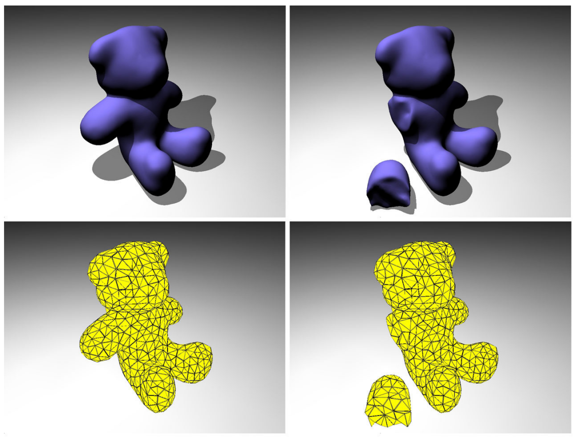


Figure 5: *A teddy model is fractured. The object is represented with a tetrahedral mesh and a high-resolution surface mesh for visualization purposes. In case of topological changes, the data structure is efficiently updated by reassigning mass portions to the respective constraint sets. The computational overhead in order to maintain a consistent mesh is negligible.*

In a third scenario, the processing of non-conforming meshes is demonstrated (see Fig. 6). Two deformable objects are glued to each other. The dynamic merging process is governed by a spatial subdivision approach that detects collisions and self-collisions [THM*03]. Detected collisions result in geometric constraints that are added to a constraint set. The scenario consists of 2088 tetrahedrons, 780 constraint sets, and 8352 mass portions. The computation including FE forces and integration takes 18 ms. Again, there is no significant overhead for updating the constraint sets in case of a topology change.

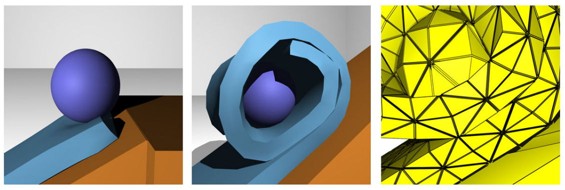


Figure 6: *Objects are glued to each other. Connections are realized with geometric constraints that can be added to a constraint set. Collisions and self-collisions are handled.*

The last scenario illustrates the versatility of the data structure. Two objects are merged and split arbitrarily (see Fig. 7). The scenario consists of 2160 tetrahedrons, 686 constraint sets, and 8640 mass portions. The computation time is 19 ms, the overhead for data structure updates is negligible. As can be seen in Fig. 7, constraint sets enable a very flexible processing of combinations of merge and split operations.

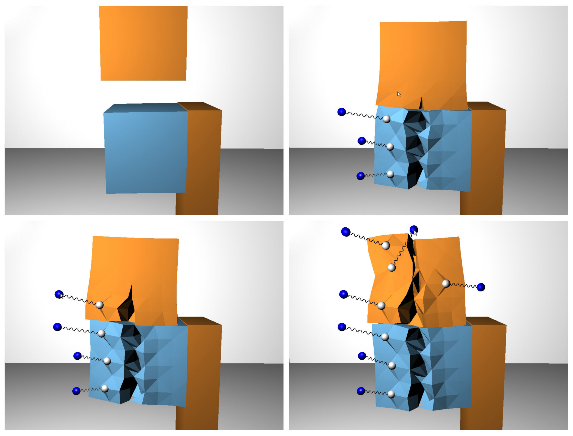


Figure 7: *Two objects are merged and split. Constraint sets allow for a very simple and efficient combination of arbitrary split and merge operations.*

5. Conclusion

We have proposed constraint sets as an efficient data structure for topology-changing tetrahedral meshes. Mass points are replaced by constraint sets in order to accelerate and simplify data structure updates for merge and split operations. Processed meshes are guaranteed to be consistent and elastomechanical properties are preserved without any additional effort. Complex combinations of merge and split operations for tetrahedral meshes can efficiently be processed. The data structure handles conforming and non-conforming meshes. It is independent of the underlying deformation model. We have shown, that the computational overhead for the consistent and property-preserving data structure update in case of topological changes is negligible compared to the computation of FE forces and the numerical integration. Even for the rare case of an object fracturing into single tetrahedrons the costs for the data structure update are only about 10% of the force and integration costs. Due to its ease, the risk of implementation errors and differences in computing times per simulation step are minimized.

References

[BGTG04] BIELSER D., GLARDON P., TESCHNER M., GROSS M.: A state machine for real-time cutting of tetrahedral meshes. *Graph. Models 66*, 6 (2004), 398–417.

[BWHT07] BARGTEIL A. W., WOJTAN C., HODGINS J. K., TURK G.: A finite element method for animating large viscoplastic flow. *ACM Trans. Graph. 26*, 3 (2007).

[CA06] CANI M.-P., ANGELIDIS A.: Towards virtual clay. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses* (2006), ACM Press, pp. 67–83.

[CBP05] CLAVET S., BEAUDOIN P., POULIN P.: Particle-based viscoelastic fluid simulation. In *SCA '05: Proc. of the 2005 ACM*

SIGGRAPH/Eurographics Symposium on Computer animation (2005), ACM Press, pp. 219–228.

[CDA00] COTIN S., DELINGETTE H., AYACHE N.: A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. *The Visual Computer 16*, 8 (2000), 437–452.

[CZ92] CHEN D. T., ZELTZER D.: Pump it up: computer animation of a biomechanically based model of muscle using the finite element method. *SIGGRAPH Comput. Graph. 26*, 2 (1992), 89–98.

[DC04] DEWAELE G., CANI M.-P.: Virtual clay for direct hand manipulation. In *Eurographics (short papers)* (2004).

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic real-time deformations using space & time adaptive sampling. In *SIGGRAPH '01: Proc. of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM Press, pp. 31–36.

[FDA02] FOREST C., DELINGETTE H., AYACHE N.: Cutting simulation of manifold volumetric meshes. In *MICCAI '02: Proc. of the 5th Int. Conf. on Medical Image Computing and Computer-Assisted Intervention-Part II* (2002), Springer, pp. 235–244.

[FH05] FLORIANI L. D., HUI A.: Shape representations based on simplicial and cell complexes. *Eurographics STAR Report* (2005), 63–87.

[GBT06] GISSLER M., BECKER M., TESCHNER M.: Local constraint methods for deformable objects. In *Proc. of VriPhys* (2006), pp. 25–32.

[GCMS00] GANOVELLI F., CIGNONI P., MONTANI C., SCOPIGNO R.: Enabling cuts on multiresolution representation. In *Computer Graphics International* (2000), pp. 183–191.

[GTT89] GOURRET J.-P., THALMANN N. M., THALMANN D.: Simulation of object and human skin formations in a grasping task. In *SIGGRAPH '89: Proc. of the 16th annual conference on Computer graphics and interactive techniques* (1989), ACM Press, pp. 21–30.

[HS04] HAUTH M., STRASSER W.: Corotational simulation of deformable solids. In *Proc. of WSCG* (2004), pp. 137–145.

[ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *SCA '04: Proc. of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2004), Eurographics Association, pp. 131–140.

[LC06] LOUBÈRE R., CARAMANA E. J.: The force/work differencing of exceptional points in the discrete, compatible formulation of lagrangian hydrodynamics. *J. Comput. Phys. 216*, 1 (2006), 1–18.

[MBF04] MOLINO N., BAO Z., FEDKIW R.: A virtual node algorithm for changing mesh topology during simulation. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (2004), ACM Press, pp. 385–392.

[MDM*02] MÜLLER M., DORSEY J., MCMILLAN L., JAGNOW R., CUTLER B.: Stable real-time deformations. In *SCA '02: Proc. of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2002), ACM Press, pp. 49–54.

[MG04] MÜLLER M., GROSS M.: Interactive virtual materials. *Proc. of the 2004 conference on Graphics interface* (2004), 239–246.

[MTG04] MÜLLER M., TESCHNER M., GROSS M.: Physically-based simulation of objects represented by surface meshes. In *CGI '04: Proc. of the Computer Graphics International (CGI'04)* (2004), IEEE Computer Society, pp. 26–33.

[NvdS01] NIENHUYS H.-W., VAN DER STAPPEN A. F.: A surgery simulation supporting cuts and finite element deformation. In *Proc. of Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (2001), Springer, pp. 145–152.

[OBH02] O'BRIEN J. F., BARGTEIL A. W., HODGINS J. K.: Graphical modeling and animation of ductile fracture. In *SIGGRAPH '02: Proc. of the 29th annual conference on Computer graphics and interactive techniques* (2002), ACM Press, pp. 291–294.

[OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *SIGGRAPH '99: Proc. of the 26th annual conference on Computer graphics and interactive techniques* (1999), ACM Press/Addison-Wesley Publishing Co., pp. 137–146.

[PKA*05] PAULY M., KEISER R., ADAMS B., DUTRÉ; P., GROSS M., GUIBAS L. J.: Meshless animation of fracturing solids. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (2005), ACM Press, pp. 957–964.

[QB95] QUIROZ L., BECKERS P.: Non-conforming mesh gluing in the finite element method. *International Journal for Numerical Methods in Engineering 38*, 13 (1995), 2134–2165.

[SDF07] SIFAKIS E., DER K. G., FEDKIW R.: Arbitrary cutting of deformable tetrahedralized objects. In *SCA '07: Proc. of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), Eurographics Association, pp. 73–80.

[SHGS06] STEINEMANN D., HARDERS M., GROSS M., SZEKELY G.: Hybrid cutting of deformable solids. In *VR '06: Proc. of the IEEE Virtual Reality Conference* (2006), IEEE Computer Society, pp. 35–42.

[SOG06] STEINEMANN D., OTADUY M. A., GROSS M.: Fast arbitrary splitting of deforming objects. In *SCA '06: Proc. of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2006), Eurographics Association, pp. 63–72.

[SSIF07] SIFAKIS E., SHINAR T., IRVING G., FEDKIW R.: Hybrid simulation of deformable solids. In *SCA '07: Proc. of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), Eurographics Association, pp. 81–90.

[TF88a] TERZOPOULOS D., FLEISCHER K.: Deformable models. *The Visual Computer 4*, 6 (1988), 306–331.

[TF88b] TERZOPOULOS D., FLEISCHER K.: Modeling inelastic deformation: viscolelasticity, plasticity, fracture. *Proc. of the 15th annual conference on Computer graphics and interactive techniques* (1988), 269–278.

[THM*03] TESCHNER M., HEIDELBERGER B., MUELLER M., POMERANETS D., GROSS M.: Optimized spatial hashing for collision detection of deformable objects. In *Proc. of VMV* (2003), pp. 47–54.

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. *ACM SIGGRAPH Computer Graphics 21*, 4 (1987), 205–214.