

Deformation Modelling for a Physics-Based Perception System



Antoine Petit, Vincenzo Lippiello, and Bruno Siciliano

Abstract Manipulation tasks require capturing the deformations undergone by the manipulated object. In order to handle various sorts and magnitudes of deformations, physical deformation models have been employed based on FEM. By resorting to linear elasticity with a volumetric formulation, these models simulate large strains while handling volumetric effects. For perception, such modelling provides physically consistent regularization to registered deformations, especially over areas unobserved by the sensing system. Besides elasticity modelling, fracture models are investigated to cope with potential cracks or tears of the manipulated objects as well as collision models to deal with multiple interacting objects or interactions with the environment. Also, these models have proven to be computationally efficient and are thus suitable for dynamic manipulation. This chapter is based on the works presented in [1, 2].

1 Brief Introduction

Unlike vision-based tracking problems with rigid objects, for which a certain maturity has been reached, perception for non-rigid objects is still a challenging problem. It has aroused much interest in recent years in computer vision, computer graphics [3, 4], and robotics community [5]. Many potential applications would indeed be targeted in augmented reality, medical imaging, and robotic manipulation by handling a vast variety of objects: tissues, paper, rubber, viscous fluids, cables, food, organs, and so on. For instance, in the RoDyMan project, a demonstration scenario is the

A. Petit (✉)

Therenga, 74F rue de Paris, 35000 Rennes, France

e-mail: antoine.a.petit@inria.fr

V. Lippiello · B. Siciliano

Department of Electrical Engineering and Information Technology, PRISMA Lab, CREATE Consortium & University of Naples Federico II, Via Claudio 21, 80125 Naples, Italy

e-mail: vincenzo.lippiello@unina.it

B. Siciliano

e-mail: bruno.siciliano@unina.it

© Springer Nature Switzerland AG 2022

B. Siciliano and F. Ruggiero (eds.), *Robot Dynamic Manipulation*,

Springer Tracts in Advanced Robotics 144,

https://doi.org/10.1007/978-3-030-93290-9_1

Table 1 Main symbols used in this chapter

Definition	Symbol
Position of a generic point of the element e in the world frame $\mathcal{W}$	$\mathbf{x} \in \mathbb{R}^3$
Deformation field	$\mathbf{u}_e(\mathbf{x}) \in \mathbb{R}^3$
Initial coordinates of the four vertices of the element e in the world frame $\mathcal{W}$	$\mathbf{x}_{e,0} \in \mathbb{R}^{12}$
Coordinates of the four vertices of the element e in the world frame $\mathcal{W}$	$\mathbf{x}_e \in \mathbb{R}^{12}$
Cauchy's linear symmetric strain tensor using Voigt notation	$\boldsymbol{\epsilon}_e \in \mathbb{R}^6$
Infinitesimal stress tensor using Voigt notation	$\boldsymbol{\sigma}_e \in \mathbb{R}^6$
Internal elastic forces on the four vertices of the element e	$\mathbf{f}_e \in \mathbb{R}^{12}$
Stiffness matrix of the element e	$\mathbf{K}_e \in \mathbb{R}^{12 \times 12}$
Block-diagonal matrix containing four copies of the rotation matrix in $SO(3)$ corresponding to the rotational component of the deformations of the element e	$\mathbf{R}_e \in \mathbb{R}^{12 \times 12}$

humanoid dual-arm/hand manipulation of the pizza dough, which requires complex dynamic manipulation tasks of a deformable object. Research has increasingly focused on robots involved in food manipulation tasks in recent years [6–9]. The perception system aims to provide the robot controller accurate, robust, and real-time sensing of the manipulated deformable object. With respect to rigid objects, dealing with deformations poses several additional challenges, such as modelling the considered material properties and fitting this model with the vision and/or range data. This registration problem also involves critical real-time concerns, which are primarily required for robotic dynamic manipulation. Although numerous studies have proposed efficient real-time techniques to handle 3D surfaces (paper, clothes) which undergo isometric or slightly elastic deformations, a large open field remains when considering more significant elastic deformations. Thus, the aim is to propose a real-time tracking system able to handle elastic objects, potentially textureless, by tracking large deformations and fast rigid motions, using visual and range data provided by an RGB-D sensor, as described in the next chapter. Our approach involves physical modelling of the considered object to cope with deformations by relying on a FEM. The considerable progresses recently made within the computer graphics and medical simulation domains have enabled real-time performance for processing such models.

In this chapter, we first introduce in Sect. 2 the various techniques to model deformations, and also topological events such as tearing, fractures, splitting, or merging. These numerical methods have been widely studied in the domains of computer graphics [3, 10] and medical simulation [11].

Based on this background, in Sects. 3, 4, and 5 we describe in more details the elastic, fracture, and collision models we have employed for the RoDyMan project to address, in a physics-based manner, the perception task when these behaviours are encountered.

2 Background on Deformation Models

Modelling deformations can be handled through different approaches: mesh-based heuristic or physically realistic approaches, and mesh-free methods such as particle systems.

2.1 *Mesh-Based Approaches*

For mesh-based heuristic approaches, potentially relying on simple heuristical physical modeling, can be distinguished free-form geometrical mesh models, such as Bezier curves, deformable splines, active contours [12, 13], or NURBS. These representations implicitly take into account elastic or stretching constraints in an energy function to minimize. Other models are explicitly based on approximate physical properties. In this category, a famous and widespread model is the planar or volumetric mass-spring model. The mesh for this model is made of discrete points or nodes assigned with masses and which are linked by springs, using triangular shapes [14], T-shapes [15] for planar models, or spread over cubic elements for volumetric models [10, 16]. In order to simulate the deformations of the modeled object, because of external forces and internal forces between the nodes, the displacement of each node is computed by applying Newton's second law, solving the resulting differential equations, usually using Euler or Runge-Kutta integration methods. The simplicity and low computational complexity of surface spring-mass models have proven their efficiency on planar deformable objects showing elastic properties. However, when dealing with volumetric objects, the surface mass-spring model often fails to maintain a constant volume. The deformations do not propagate very fast, limiting the approach to small elastic deformations. With volumetric mass-spring models, a slightly better volumetric behavior, although with still slow propagation of deformations, can be obtained at the cost of significantly higher computational costs. Linked volume models, which are less employed, can also be mentioned. Instead, the FEM provides a realistic physical model by relying on continuous mechanics. In order to compute the deformation of the object, the basic idea is to solve Navier-Stokes or Euler PDEs, which continuously describe the deformations of the object and consist in balancing internal forces, resulting from viscosity and elastic properties of the material, and external forces, such as gravity or collisions. Then, solving these PDEs can be addressed by the FEM. It relies on finding numerical solutions of the equations by first discretizing the deformable object into a mesh made of elements,

usually tetrahedrons, and performing numerical explicit or implicit integration over this mesh. A significant limitation of FEM approaches such as [17, 18] is their computational complexity since they involve solving non-linear equations involving the displacements and the forces at the nodes of the elements. Approximations of these equations are thus inevitable for real-time computations and simulations. In the simplest case, a static linear relation between forces and displacement can be derived by means of the stiffness matrix, which last involves the elastic properties of the considered element by depending on two parameters, namely, the Young's modulus and the Poisson's ratio. The methods solving these equations can then be operated in real-time [19]. However, they are still restricted to 100 relatively small elastic deformations. This drawback has been circumvented with co-rotational approaches [20, 21], or by using more complex tensors to model the stiffness, such as the non-linear St. Venant-Kirchhoff strain tensor [22], handling some hyper-elastic deformations, but with a more considerable computational burden. Other approaches [23] have been able to cope with more general hyper-elastic materials, potentially in real-time. More recently, [24] proposed the MJED, an algorithm relying on a total Lagrangian FEM to handle, in real-time, materials showing hyper-elasticity, porosity, and viscosity behaviours.

2.2 *Mesh-Free Approaches*

We can introduce a different line of approaches which does not consider a geometrical mesh to model the object but instead a system of particles involving more or less realistic physical properties. With particle-based methods, which have been primarily developed in the computer graphics community [3], the object is represented by a set of 3D points storing attributes such as position, velocity, physical properties, and so on. In an initial description [25], referred to as coupled spatial particle system, the particles are dynamically evolving in a stochastic manner. Their attributes are potentially related to the behavior of other particles, with static or dynamic interactions, to model topological changes, for instance. This simple model has enabled us to simulate fuzzy and inelastic objects such as clouds, water, and so on. Another particle-based approach, referred to as SPH [26], employs a more continuous method by interpolating particular quantities defined at discrete locations on the whole space, smoothing properties between particles. This approach has been widely used to simulate fluid, based on Navier-Stokes equations, the various internal forces resulting from pressure, viscosity, or surface tension being computed with smoothing functions. With respect to mesh (or grid)-based techniques such as FEM, SPH has several advantages by simplifying the fluid equations since mass conservation is guaranteed. The involved internal forces are computed from weighted contributions from other particles instead of solving systems of equations for FEM. Due to its higher simplicity and flexibility, SPH particle systems enable to model highly viscous, viscoelastic, or even plastic materials, as well as topological changes such as splitting or merging events [27–30]. However, compared to mesh-based

approaches, SPH approaches have problems when maintaining the incompressibility of the material, and they require a large number of particles (with respect to the number of vertices for the mesh) to perform simulations with equivalent resolution. Nevertheless, real-time simulations can be performed in the approaches proposed in [27–29]. A challenge for particle-based methods is to recover the surface of the object. It can be done explicitly, through the surface reconstruction between the particles [27, 28], or implicitly, based on level sets [30], or using both explicit and implicit representations [29]. The level sets implicitly define the surface by signed values for points in the volume. These values generally correspond to a distance (positive when inside the fluid, negative otherwise) to the explicit surface. Level sets are especially suited to handle severe topological changes.

2.3 Hybrid Approaches

Coupling both mesh and particle-based methods combines the advantages of both ones. An example is the particle level set method [31, 32] which introduces particles in the vicinity of the mesh’s surface. Such a coupling enables the accurate simulation of highly viscoelastic and plastic objects, as well as fluids with topological changes.

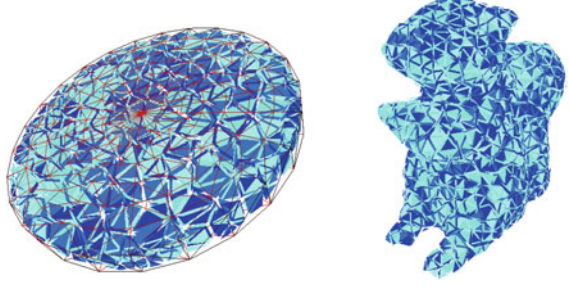
3 FEM Elastic Model

Among the different physical modelling methods mentioned above, we have adopted a mesh-based FEM model for the perception system of the RoDyMan platform. Besides modelling potentially large elastic volumetric deformations, FEM models can act as reliable regularizers for registration on noisy vision data, which mesh-free methods might not guarantee. They can also now achieve real-time performances, and implementation is eased by the availability of open-source simulators embedding such models. These models assume the prior knowledge of a consistent mesh (which could be automatically reconstructed offline) and the material properties (through the Young’s modulus and the Poisson’s ratio) of the observed and manipulated object.

3.1 Finite Element Modelling

As introduced above, the FEM provides a realistic physical model by relying on continuum mechanics instead of finite differences for mass-spring systems, for instance. For a detailed description, the reader can refer to [33]. It consists of tessellating the deformable object into a mesh made of elements. Here, we rely on a volumetric linear FEM approach with tetrahedral elements (see Fig. 1), which are preferred over other topologies such as triangles or hexahedrons, due to their modeling accuracy

Fig. 1 Examples of tetrahedral meshes



(with respect to triangles, for instance), while being computationally efficient and convenient for meshing volumes with topologies of any complexity (with respect to hexahedrons).

The deformation field $\mathbf{u}_e$ over an element e is then approximated as a continuous interpolation of the displacement vectors $\hat{\mathbf{u}}_e$ of its four vertices, through the matrix $\mathbf{N}_e(\mathbf{x}) \in \mathbb{R}^{3 \times 12}$, which contains the polynomial basis functions of the element

$$\mathbf{u}_e(\mathbf{x}) = \mathbf{N}_e(\mathbf{x})\hat{\mathbf{u}}_e, \quad (1)$$

where $\hat{\mathbf{u}}_e = \mathbf{x}_e - \mathbf{x}_{e,0}$.

3.2 Linear Elasticity

In order to model deformations and elasticity, we resort to the infinitesimal strain theory [33] to compute the Cauchy's linear strain tensor $\boldsymbol{\epsilon}_e$ within the tetrahedron. It can be linearly expressed with respect to $\hat{\mathbf{u}}_e$ as

$$\boldsymbol{\epsilon}_e = \mathbf{L}_e \hat{\mathbf{u}}_e, \quad (2)$$

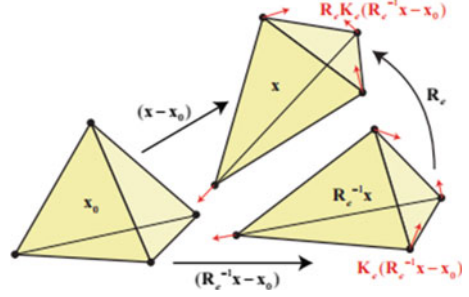
with $\mathbf{L}_e \in \mathbb{R}^{6 \times 12}$ a constant matrix. In order to relate the stress with the strain, we then rely on Hooke's law linear elasticity theory for a continuous isotropic material, which leads us to write the infinitesimal stress tensor $\boldsymbol{\sigma}_e$ in the element e as

$$\boldsymbol{\sigma}_e = \mathbf{C}_e \boldsymbol{\epsilon}_e, \quad (3)$$

where $\mathbf{C}_e \in \mathbb{R}^{6 \times 6}$ symmetric matrix depending on two elastic parameters of the material, namely, the Young's modulus $E \in \mathbb{R}$ and the Poisson's ratio $\nu \in \mathbb{R}$. Folding (2) into (3) yields

$$\boldsymbol{\sigma}_e = \mathbf{C}_e \mathbf{L}_e \hat{\mathbf{u}}_e. \quad (4)$$

Fig. 2 The corotational approach



The strain energy in e can be computed as a function of σ_e , ϵ_e , and the volume $V_e \in \mathbb{R}$ of the element. It can then be derived to determine the internal elastic forces $\mathbf{f}_e$ exerted on the four vertices of e . Based on (2) and (4), $\mathbf{f}_e$ can be linearly related to the displacements $\hat{\mathbf{u}}_e$ of the vertices as

$$\mathbf{f}_e = \mathbf{K}_e \hat{\mathbf{u}}_e = V_e \mathbf{L}_e^T \sigma_e, \quad (5)$$

with $\mathbf{K}_e = V_e \mathbf{L}_e^T \mathbf{C}_e \mathbf{L}_e$, and $V_e \in \mathbb{R}$.

3.3 The Corotational Approach

The model using an infinitesimal approximation of the strain tensor, giving a constant $\mathbf{K}_e$ linearising the elastic forces, is insensitive to linear transformations. However, it is inaccurate when modeling large rotations of the elements causing, for instance, unexpected growth of volume. A workaround consists of the corotational approach proposed in [21, 34, 35], used for registration purposes in [36], which is a good compromise between the ability to model large elastic deformations and computational efficiency. The displacement of an element can be decomposed into a rigid rotation $\mathbf{R}_e$ and a pure deformation. As suggested in [35], we use a polar decomposition on the transformation matrix between the current and initial state of the vertices to extract $\mathbf{R}_e$. Then, the stiffness matrix $\mathbf{K}_e$ can be warped with respect to this rotation to accommodate rotation transformations, giving

$$\mathbf{f}_e = \mathbf{R}_e \mathbf{K}_e \hat{\mathbf{u}}_e^r = \mathbf{R}_e \mathbf{K}_e (\mathbf{R}_e^{-1} \mathbf{x}_e - \mathbf{x}_{e,0}), \quad (6)$$

where $\hat{\mathbf{u}}_e^r = \mathbf{R}_e^{-1} \mathbf{x}_e - \mathbf{x}_{e,0}$, with $\mathbf{R}_e^{-1} \mathbf{x}_e$ the back rotated deformed coordinates of the vertices of e to an unrotated frame. Notice that the forces $\mathbf{K}_e \hat{\mathbf{u}}_e^r$ are rotated to the current deformed element through $\mathbf{R}_e$. Figure 2 illustrates the principle of such a corotational approach.

In this way, the overall forces on the whole mesh can be summed to zero, while computational efficiency is ensured since $\mathbf{K}_e$ can be computed in advance, in contrast to non-linear FEM approaches.

4 Fracture Model

During deformable object manipulation, excessive stress may result in fractures of the object. Therefore, regardless of whether it might be a desirable effect, detecting and capturing such a topological event becomes essential for proper manipulation.

In the computer graphics community, various systems modeling fractures were proposed. The detection of fractures, or cracks, in 3D object simulation has been initially achieved using mass-spring systems [37] by removing springs whose lengths exceed a certain threshold. Based on the FEM, a physically consistent method using the linear elastic fracture theory [38] was proposed in [18] through the computation, for some given strains, of tensile forces undergone by the nodes. A fracture is then propagated from each detected fracturable node (or crack tip) by subdividing the neighboring tetrahedra according to a fracture plane. Similarly, a more computationally efficient solution was suggested in [21, 39] by resorting the fracture being propagated along element boundaries to element-wise stresses. Although improvements have been achieved from these widespread approaches, for both fracture detection and propagation [40], we propose here an approach based on [18]. One reason is the ability to cope appropriately with the provided noisy point cloud data continuously. Another reason is to preserve real-time performance.

4.1 Fracture Detection

Detecting fractures in the mesh is performed similarly to [18], by decomposing the internal forces exerted on the nodes into tensile and compressive forces, with the difference that [18] relies on a standard linear FEM method. For this purpose, the stress tensor undergone by a tetrahedron is separated into tensile and compressive components. Here, using a corotational approach, we compute σ_e as

$$\sigma_e = \mathbf{C}_e \mathbf{L}_e (\mathbf{R}_e^{-1} \mathbf{x}_e - \mathbf{x}_{e,0}). \quad (7)$$

Then, the three eigenvalues, $v_i(\sigma_e) \in \mathbb{R}$, and eigenvectors, $\mathbf{n}_i(\sigma_e) \in \mathbb{R}^3$, associated to σ_e are computed. Positive eigenvalues correspond to tensile stresses, whereas negative eigenvalues correspond to compressive stresses. As in [18], tensile and compressive tensors can be then defined as

$$\boldsymbol{\sigma}_e^+ = \sum_{i=1}^3 \max(0, v_i(\boldsymbol{\sigma}_e)) \mathbf{M}(\mathbf{n}_i(\boldsymbol{\sigma}_e)), \quad (8)$$

$$\boldsymbol{\sigma}_e^- = \sum_{i=1}^3 \min(0, v_i(\boldsymbol{\sigma}_e)) \mathbf{M}(\mathbf{n}_i(\boldsymbol{\sigma}_e)), \quad (9)$$

with

$$\mathbf{M}(\mathbf{a}) = \begin{cases} \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|} & \text{if } \mathbf{a} \neq \mathbf{0}_3, \\ \mathbf{O}_3 & \text{if } \mathbf{a} = \mathbf{0}_3, \end{cases} \quad (10)$$

for any vector $\mathbf{a} \in \mathbb{R}^3$.

Based on these tensile and compressive stresses and using (5), the internal forces exerted on the vertices of e can be decomposed as

$$\mathbf{f}_e^+ = V_e \mathbf{L}_e^T \boldsymbol{\sigma}_e^+, \quad (11)$$

$$\mathbf{f}_e^- = V_e \mathbf{L}_e^T \boldsymbol{\sigma}_e^-. \quad (12)$$

Considering a mesh with $n_X > 0$ vertices, it is possible to define the set X of all these vertices, $\mathbf{x}_j \in \mathbb{R}^3$, with $j = 1, \dots, n_X$. Then, as in [18], the separation tensor, $\boldsymbol{\zeta}_j \in \mathbb{R}^6$ can be computed for each node $\mathbf{x}_j$ in the mesh using the sets $\{\mathbf{f}_{e,j}^+ \mid j \in e\}_e$ and $\{\mathbf{f}_{e,j}^- \mid j \in e\}_e$ of tensile and compressive forces, respectively, exerted by the elements $\{e \mid j \in e\}$ attached to the node

$$\boldsymbol{\zeta}_j^- = \frac{1}{2} (\sum_e \mathbf{M}(\mathbf{f}_{e,j}^+) - \mathbf{M}(\mathbf{f}_j^+) - \sum_e (\mathbf{f}_{e,j}^-) - \mathbf{M}(\mathbf{f}_j^-)), \quad (13)$$

with

$$\mathbf{f}_j^+ = \sum_e \mathbf{f}_{e,j}^+ \quad \text{and} \quad \mathbf{f}_j^- = \sum_e \mathbf{f}_{e,j}^-. \quad (14)$$

The tensor $\boldsymbol{\zeta}_j$ enables to evaluate deformation imbalance between tensile and compressive forces, while being invariant to imbalance resulting in rigid motions. A fracture at node $\mathbf{x}_j$ is then detected if the largest positive eigenvalue $v_j^+ \in \mathbb{R}$ of $\boldsymbol{\zeta}_j$ is above a certain threshold corresponding to the toughness of the material. The eigenvector $\mathbf{n}_j^+ \in \mathbb{R}^3$, corresponding to v_j^+ , defines the normal to the fracture plane at node $\mathbf{x}_j$.

4.2 Fracture Propagation and Remeshing

The mesh around the considered node shall be modified to propagate the fracture from the detected fracturable nodes in the direction of the fracture plane. The elements attached to the node and intersected by the fracture plane are subdivided according to the plane's orientation [18]. Other neighboring elements also need to be modified to maintain the consistency of the mesh. However, such a re-meshing process, which is proper and suitable for simulations purposes since it manages to maintain the fracture orientation, is computationally costly and tends to generate a continuously growing number of elements, which is not desirable to keep real-time performance. Besides, for registration concerns, by changing locally and drastically the resolution of the mesh, matching the re-meshed regions with the acquired point cloud (which has a constant and homogeneous resolution while being noisy) may lead to heterogeneous or unstable behaviors. Another approach, suggested in [21, 39, 41], consists of propagating the fracture along the boundaries of the neighbouring elements. Although it enables to keep the resolution constant, this method tends to produce jumbled fracture patterns and artefacts, which is also not desirable in our case when coping with the noisy point cloud data.

We have instead opted here for a solution preserving the mesh homogeneity and performing a neat fracture propagation (see Fig. 3). As in [18], the fracturable node is first replicated with the same positions. Then, within the set of elements attached to the original fracturable node, the nodes which are not intersected by the given fracture plane are determined. They are thus re-assigned to the original fracturable node or the replicated one, based on which side of the plane they are located. This procedure appears coarse, affects the volume of the model, and tends to degenerate the mesh. However, it remains acceptable if we consider objects undergoing simple fracture events, so that element losses in the mesh can be neglected.

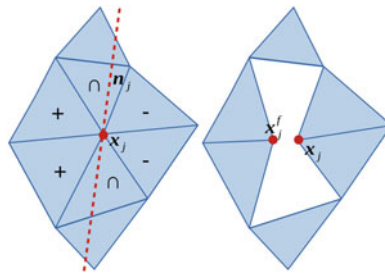
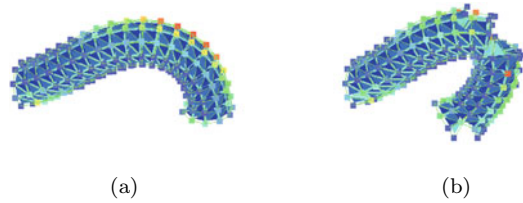


Fig. 3 Re-meshing procedure, here represented in the 2D case with triangles for simplicity. The fracturable node x_j is replicated, initially, with the same position providing a new vertex x_j^f for the mesh (it is here translated from x_j for clarity). The intersected elements are removed. The elements on the positive side of the fracture plane (determined by its normal n_j) assigned to x_j and the others remain attached to x_j

Fig. 4 Example of a fracture occurring on a cylindrical beam object



4.3 Comments on the Results

An example of a beam-like object being bent and fractured according to our fracture model is shown in Fig. 4. In Fig. 4a, we observe the fracture detection process with the value for each vertex of the fracture measure, i.e., the largest eigenvalue of the separation tensor, which is mapped to colours. In Fig. 4b, we can notice the fracture propagation step once fractures are detected.

5 Interaction Model

When manipulating multiple objects, objects interacting with their environments, or the manipulation tools, it becomes necessary to sense the potential contacts and the resulting deformations reliably. Here, we propose to use interaction models to handle these phenomena.

Indeed, modelling the interactions between the considered objects constitutes a central issue to allow a joint registration of the deformations. We propose here a state-of-the-art model consisting of classical collision detection and collision response strategies. Collision being performed on surface meshes, we rigidly map each object's collision triangular surface mesh on their corresponding volumetric tetrahedral mesh.

5.1 Collision Detection

Collisions can be handled by employing different approaches to determine the intersections and the contacts between the interacting objects. Reviews of collision detection methods can be found in [42, 43]. Intersections are computed by performing successive distance tests based on a hierarchical set of bounding volumes around the polyhedral triangular collision meshes.

Here, we employ AABB hierarchies around the considered objects. First, as a broad phase, intersections between the coarsest AABBs are found. Then, as a narrow phase, we descend in the AABB hierarchies of the objects in a coarse-to-fine manner. Within the finest AABBs, a search for intersections between pairs of geometric primitives is performed. The primitives are triangles in our case. The computation of



Fig. 5 Illustration of the collision detection algorithm with the hierarchies of bounding volumes (left), and the computed pairs of collision points (middle, right), tested on the SOFA framework

intersections is based on a threshold on the distances between the primitives (triangle-to-triangle), and it returns corresponding pairs of collision (or contact) points on the colliding primitives, as illustrated in Fig. 5.

5.2 Collision Response

In order to model the contact forces between two colliding deformable objects i and j , we use a simple penalty-based method. For two detected collision points on the collision surfaces of the interacting objects, barycentric mappings on the tetrahedral meshes are carried out to determine two corresponding points $\mathbf{x}_i \in \mathbb{R}^3$ and $\mathbf{x}_j \in \mathbb{R}^3$, in both volumetric models. Then a penetration distance $\delta > 0$ can be defined as $\delta = d - D$, with $d = \|\mathbf{x}_i - \mathbf{x}_j\|$ and $D > 0$ a proximity threshold. As a collision normal $\mathbf{n}_{i,j}$, we choose $\mathbf{n}_{i,j} = \frac{\mathbf{x}_i - \mathbf{x}_j}{d}$.

A spring-like repulsive penalty force $\mathbf{f}_{j/i}^c(\mathbf{x}_i)$, exerted by $\mathbf{x}_j$ on $\mathbf{x}_i$, can then be computed as

$$\mathbf{f}_{j/i}^c(\mathbf{x}_i) = -k\delta\mathbf{n}_{i,j}, \quad (15)$$

with $k > 0$ a stiffness factor.

6 Discussion and Conclusion

This chapter introduced some background on physical simulation with FEM to regularize observed deformations in a perception system. Despite more complex and accurate models exist, relying on continuum mechanics and elasticity theory for deformation modelling offers some physical realism and the ability to deal with various magnitudes of deformations, with volumetric effects, which is not guaranteed by standard geometric and surface-based approaches. It requires only the knowledge

of a couple of parameters (Young modulus and Poisson ratio) setting the mechanical properties of the involved object. However, the physical realism of the approach shall be mitigated since capturing the deformations in the considered perception system would consist in balancing internal forces from the FEM computations and external forces resulting from observations through the vision sensors. These latter forces have no relation with the object's real forces: the knowledge of the actual material properties does not have much significance. Deformation modelling acts as a regularizer for the observed deformations and shall not be considered as an accurate model for other purposes such as control or planning. The use of simple fracture and collision models also provides tools to handle topological changes and multiple interacting objects. The efficiency of the solvers and the implementation, through the SOFA platform, make these models suitable for real-time perception.

References

1. A. Petit, S. Cotin, V. Lippiello, B. Siciliano, Capturing deformations of interacting non-rigid objects using RGB-D data, in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 491–497, Madrid, E (2018)
2. A. Petit, V. Lippiello, B. Siciliano. Tracking fractures of deformable objects in real-time with an RGB-D sensor, in *2015 International Conference on 3D Vision*, pp. 632–639, Lyon, F (2015)
3. A. Nealen, M. Müller, R. Keiser, E. Boxerman, M. Carlson, Physically based deformable models in computer graphics. *Comput. Gr. Forum* **25**(4), 809–836 (2006)
4. J. Bender, M. Müller, M.A. Otaduy, M. Teschner, M. Macklin, A survey on position-based simulation methods in computer graphics. *Comput. Gr. Forum* **33**(6), 228–251 (2014)
5. C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D.V. Dimarogonas, D. Kragic, Dual arm manipulation: A survey. *Robot. Auton. Syst.* **60**(10), 1340–1353 (2012)
6. M. Tenorth, M. Beetz, A unified representation for reasoning about robot actions, processes, and their effects on objects, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1351–1358 (2012)
7. M. Bollini, S. Tellex, T. Thompson, N. Roy, D. Rus, Interpreting and executing recipes with a cooking robot, in *Experimental Robotics* (Springer, 2013), pp. 481–495
8. P. Kormushev, S. Calinon, D.G. Caldwell. Robot motor skill coordination with EM-based reinforcement learning, in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3232–3237 (2010)
9. M.C. Gemici, A. Saxena, Learning haptic representation for manipulating deformable food objects, in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 638–645 (2014)
10. S.F.F. Gibson, B. Mirtich, *A survey of deformable modeling in computer graphics* (Technical report, Citeseer, 1997)
11. U. Meier, O. López, C. Monserrat, M.C. Juan, M. Alcaniz, Real-time deformable models for surgery simulation: A survey. *Comput. Methods Prog. Biomed.* **77**(3), 183–197 (2005)
12. D. Terzopoulos, J. Platt, A. Barr, K. Fleischer, Elastically deformable models. *ACM Siggraph Comput. Gr.* **21**(4), 205–214 (1987)
13. D. Terzopoulos, A. Witkin, M. Kass, Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artif. Intell.* **36**(1), 91–123 (1988)
14. H. Delingette, G. Subsol, S. Cotin, J. Pignon, Craniofacial surgery simulation testbed. *Vis. Biomed. Comput.* **1994**, 607–618 (1994)
15. E. Keeve, S. Girod, B. Girod, Craniofacial surgery simulation, in *Visualization in Biomedical Computing 1996* (Springer, 1996), pp. 541–546

16. Y. Chen, Q.-H. Zhu, A. Kaufman, S. Muraki, Physically-based animation of volumetric objects, in *Computer Animation '98*, pp. 154–160 (1998)
17. G. DeBunne, M. Desbrun, M.-P. Cani, A.H. Barr. Dynamic real-time deformations using space & time adaptive sampling, in *28th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 31–36 (2001)
18. J.F. O'Brien, J.K. Hodgins, Graphical modeling and animation of brittle fracture, in *26th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 137–146 (1999)
19. S. Burion, F. Conti, A. Petrovskaya, C. Baur, O. Khatib, Identifying physical properties of deformable objects by using particle filters, in *2008 IEEE International Conference on Robotics and Automation*, pp. 1112–1117 (2008)
20. C.A. Felippa, B. Haugen, A unified formulation of small-strain corotational finite elements: I. Theory. *Comput. Methods Appl. Mech. Eng.* **194**(21), 2285–2335 (2005)
21. M. Müller, M. Gross, Interactive virtual materials. In *Graphics. Interface* **2004**, 239–246 (2004)
22. G. Picinbono, H. Delingette, N. Ayache, Real-time large displacement elasticity for surgery simulation: Non-linear tensor-mass model, in *Medical Image Computing and Computer-Assisted Intervention*, pp. 643–652 (2000)
23. J. Teran, S. Blemker, V. Hing, R. Fedkiw. Finite volume methods for the simulation of skeletal muscle, in *2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 68–74 (2003)
24. S. Marchesseau, T. Heimann, S. Chatelin, R. Willinger, H. Delingette, Fast porous visco-hyperelastic soft tissue model for surgery simulation: Application to liver surgery. *Prog. Biophys. Mol. Biol.* **103**(2), 185–196 (2010)
25. W.T. Reeves, Particle systems—a technique for modeling a class of fuzzy objects. *ACM SIGGRAPH Comput. Gr.* **17**(3), 359–375 (1983)
26. J.J. Monaghan, Smoothed particle hydrodynamics. *Ann. Rev. Astron. Astrophys.* **30**(1), 543–574 (1992)
27. S. Clavet, P. Beaudoin, P. Poulin, Particle-based viscoelastic fluid simulation, in *2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 219–228 (2005)
28. M. Müller, D. Charypar, M. Gross, Particle-based fluid simulation for interactive applications, in *ACM Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 154–159 (2003)
29. M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. In *2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 141–151, 2004
30. S. Premžoe, T. Tasdizen, J. Bigler, A. Lefohn, R.T. Whitaker, Particle-based simulation of fluids. *Computer Graphics Forum* **22**(3), 401–410 (2003)
31. D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing. *J. Comput. Phys.* **183**(1), 83–116 (2002)
32. T.G. Goktekin, A.W. Bargteil, J.F. O'Brien, A method for animating viscoelastic fluids. *ACM Transactions on Graphics* **23**(3), 463–468 (2004)
33. R.D. Cook. *Finite Element Modeling for Stress Analysis*. Wiley, 1994
34. O. Eitzmuß, M. Keckeisen, and W. Straßer. A fast finite element solution for cloth modelling. In *11th Pacific Conference on Computer Graphics and Applications*, pages 244–251, 2003
35. M. Nesme, Y. Payan, and F. Faure. Efficient, physically plausible finite elements. In *Eurographics*, 2005
36. N. Haouchine, J. Dequidt, I. Peterlik, E. Kerrien, M.-O. Berger, and S. Cotin. Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery. In *2013 IEEE International Symposium on Mixed and Augmented Reality*, pages 199–208, 2013
37. A. Norton, G. Turk, B. Bacon, J. Gerth, P. Sweeney, Animation of fracture by physical modeling. *Vis. Comput.* **7**(4), 210–219 (1991)
38. T.L. Anderson. *Fracture Mechanics: Fundamentals and Applications*. CRC Press, 2005
39. M. Müller, L. McMillan, J. Dorsey, and R. Jagnow. *Real-time Simulation of Deformation and Fracture of Stiff Materials*. Springer, 2001

40. L. Glondou, M. Marchal, G. Dumont, Real-time simulation of brittle fracture using modal analysis. *IEEE Trans. Visual Comput. Graphics* **19**(2), 201–209 (2013)
41. J. Smith, A. Witkin, D. Baraff, Fast and controllable simulation of the shattering of brittle objects. *Computer Graphics Forum* **20**(2), 81–91 (2001)
42. P. Jiménez, F. Thomas, C. Torras, 3D collision detection: A survey. *Computers & Graphics* **25**(2), 269–285 (2001)
43. M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision detection for deformable objects. *Computer Graphics Forum*, 24:61–81, 2005