Efficient Suturing of Deformable Models

Georges Younes, Julien abi-Nahed, and George Turkiyyah

Abstract Suturing is a fundamental operation in surgical procedures. Simulation of suturing involves the simulation of needle–tissue and thread–tissue interaction, as well as contact between sutured boundaries. In this work, robust methods are proposed for an efficient finite element-based suture simulation. Contact conditions are modeled using complementarity relations. By exploiting adjacency relationships inherent to simplicial meshes and decomposing rigid body motion into sliding and bulk-deforming components, needle driving can be simulated in real time without introducing any new mesh elements. In addition, a computationally efficient formulation of thread pulling is presented. These techniques are implemented and tested in a prototype system which allows for interactive simulation of suturing with high resolution models.

1 Introduction

Suturing is a fundamental operation in surgical procedures. It is an ever-present task in procedures ranging from the closing of lacerations and stitching of wounds in open surgery, to anastomosis and closing of incisions after tissue removal in laparoscopic and robotic-assisted procedures. The techniques for planning suturing

G. Younes (✉)
Department of Computer Science, American University of Beirut, Beirut, Lebanon

Qatar Robotic Surgery Centre, Qatar Science & Technology Park, Qatar Foundation, Doha, Qatar

J. abi-Nahed
Qatar Robotic Surgery Centre, Qatar Science & Technology Park, Qatar Foundation, Doha, Qatar

G. Turkiyyah
Department of Computer Science, American University of Beirut, Beirut, Lebanon

tasks and the manual dexterity involved in executing them are essential skills that student surgeons and practitioners develop through training and practice. In the context of laparoscopic and robotic-assisted surgery, with their constrained workspace due to instrument kinematics, this practice often has a steep learning curve, and it takes a substantial amount of training to perfect the necessary skills. The practice generally takes place on suturing boards, phantom models, or live animals (w/o on), but there has been significant interest in the use of simulation-based training tools because of the flexibility in the training scenarios that the simulators can provide, their convenience, and their ultimate cost-effectiveness.

The development of surgical training simulators and procedure-rehearsal systems requires reliable models for suture simulation that can pass the tests of face and content validity. The importance of this problem has led to a number of efforts for modeling various aspects of the suturing task. In particular, finite element models for needle insertion and steering inside soft tissue have been described in [1, 2]. In these works, node repositioning, additions, and local remeshing are performed in a two-dimensional or three-dimensional volumetric finite element mesh to support the compatible sliding and sticking movements of a needle inside the mesh. Duriez et al. [3] describe a flexible needle insertion model that does not remesh locally but instead uses complementarity constraints to tie points on the needle to the elements of the tissue mesh. This strategy has obvious computational savings and allows realistic biopsy and brachytherapy simulations to be performed. Another set of research efforts [4, 5] has focused on experiments intended to produce realistic in vivo and in vitro values for friction forces, puncture thresholds, and other parameters needed in validated simulations. Contact involving rigid tools and deformable models have been studied in physically based surgical simulations and related haptic environments. Laycock and Day [6] survey some early models that have been proposed for generating contact forces. More recent works have successfully used models involving linear complementarity relations between gaps and reaction forces [7, 8] to handle contact. Methods for collision detection for deformable models have been described in [9, 10].

In the context of suturing simulations, an early work [11] used constraints to model suture closing in a finite element model. A model that uses a spline endowed with a continuum dynamic model and sliding constraints was proposed in [12]. Wittek et al. [13] simulate needle insertion into the brain with high fidelity and precision using a predictive nonlinear biomechanical tissue model that accounts for large deformations, but it is not clear if it can produce haptic rates of interaction for realistic models. Guébert et al. [7] propose a promising simulation based on complementarity constraint modeling of all interactions between surgical threads and the embedding soft tissue. Choi et al. [14] describe a suturing system using the physics available in a commodity physics engine including line springs and chained linear segments. Mass-spring chains have also been proposed for thread modeling. Despite these and related efforts however, there is not yet a satisfactory suturing simulation model that has sufficient realism and interactive response rate, suitable for training medical students and related professionals.

The difficulties of modeling of suturing operations come from a number of factors. An inextensible thread, essentially massless, with almost negligible bending stiffness is driven by a rigid tool in a highly deformable anatomically complex, heterogeneous, anisotropic soft tissue environment. The topology of the environment is continuously changing because of the needle movement and the general contact conditions between the deforming tissue being sutured as well as between tools, threads, and tissue. These simulations have to be performed in a limited computational budget to allow for interactive use operations that support real-time visual feedback and haptic interaction.

The main contribution of this chapter is a computationally effective model that allows threads to be inserted into neighboring soft tissue volumes and pulled at two points to close an opening in real time and in a mechanically plausible fashion. The model introduces efficient geometric operations to allow needle and thread to traverse multiple boundaries and builds the appropriate constraints to use in a finite element model. We take advantage of the rigid nature of the needle to decompose driving motion into sliding and bulk-deforming components and handle them separately. We also introduce a new thread pulling constraint model and demonstrate it on a realistic example of an anastomosis procedure involving a fine mesh with multiple puncture points.

The rest of this chapter is organized as follows. In Sect. 2 we describe how three-dimensional inter-boundary contact is handled in the context of a finite element model. In Sect. 3, we describe how we model needle driving and puncturing of multiple tissue boundaries. Section 4 presents a thread pulling model, suitable for single stitches as well as running sutures, to close openings between two boundaries. Section 5 shows results and concludes.

2 Interaction Modeling and Contact Handling

In this chapter, deformable models are simulated using a quasi-static finite element model. At every step of the simulation $\mathbf{Kx} = \mathbf{f}$ is solved, where $\mathbf{x}$ is a vector with the geometric positions of the n_0 nodes, $\mathbf{K}$ is a $3n_0 \times 3n_0$ (stiffness) matrix, and $\mathbf{f}$ contains the external forces.

The solution $\mathbf{x}$ can be constrained with the aid of algebraic constraints of the form $\mathbf{C}_i\mathbf{x} = \mathbf{b}_i$. The constraints linearly couple a subset of $\mathbf{x}$ and are solved by augmenting the original system:

$$\begin{bmatrix} \mathbf{K} & \mathbf{C}^t \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{X} \\ \lambda \end{Bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{b} \end{bmatrix} \tag{1}$$

where $\mathbf{C} = [\mathbf{C}_0; \mathbf{C}_1; \ldots; \mathbf{C}_m]$ and $\mathbf{b} = [\mathbf{b}_0; \mathbf{b}_1; \ldots; \mathbf{b}_m]$. This system can be solved efficiently by precomputing the Cholesky decomposition of $\mathbf{K}$ at the start of the simulation and reusing it with backward/forward passes and different right-hand sides.

This constraint framework allows us to model complex interactions between the different objects that are being simulated, for example, a vertex v can be fixed at position $\mathbf{v}_0$ by adding a constraint of the form: $g(v) = \mathbf{v}_0$, where $g(s)$ returns the geometric positions of the vertices of a simplex s. In addition, Vertex–

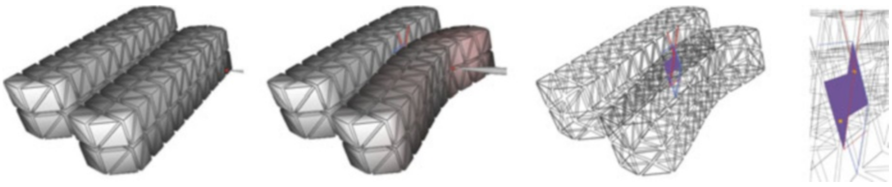


Fig. 1 A sequence of interaction demonstrating the handling of contact. The contact complex is zoomed in the rightmost figure (VF in *orange and purple* EE in *red and blue*)

Face (VF) and Edge–Edge (EE) contact collisions can be handled by properly imposed constraints to prevent interpenetration. A VF contact between a point with barycentric coordinates γ in face f and a vertex v is represented using $g(f) \cdot \gamma = g(v)$ and an EE collision between two points with barycentric coordinates α and β in edges e_0 and e_1, respectively, is represented using $g(e_0) \cdot \alpha = g(e_1) \cdot \beta$. Figure 1 shows an example of contact handling with this formulation.

These constraints are activated and deactivated dynamically based on their Lagrange multipliers. For example, for a VF constraint, the associated Lagrange multipliers represent the force being exerted by the vertex on the face. When the normal component of the multiplier is negative, the corresponding constraint is deactivated. In order to prevent sticking of the two boundaries when they get into contact, the tangential components of the constraints are released to allow tangential movement of the two boundaries with respect to each other using $\hat{\mathbf{n}}_f \cdot g(f) \cdot \gamma = \hat{\mathbf{n}}_f \cdot g(v)$, where $\hat{\mathbf{n}}_f$ is the face normal. A similar equation is used for EE constraints. Note that the coefficients have to be updated at every step because the normal vectors are changing.

3 Needle Driving

In this section we describe the needle–tissue interaction model. The most common types of needles are handled, namely straight and circular. The end goal is to efficiently and robustly simulate the driving of a needle into soft tissue. The different steps of our approach are depicted in Fig. 3.

3.1 Motion Decomposition

Let the positions and orientations of the needle at two successive time steps be denoted by $\mathbf{R}^{[k]}$ and $\mathbf{R}^{[k+1]}$, respectively, and $\mathbf{T}$ be the transformation between them: $\mathbf{R}^{[k+1]} = \mathbf{T}\mathbf{R}^{[k]}$. It is convenient to decompose the motion $\mathbf{T}$ into two components: a sliding component where the needle moves along its natural axis with every point moving tangentially along the axis and a normal component where the needle translates and rotates to reach its final position.

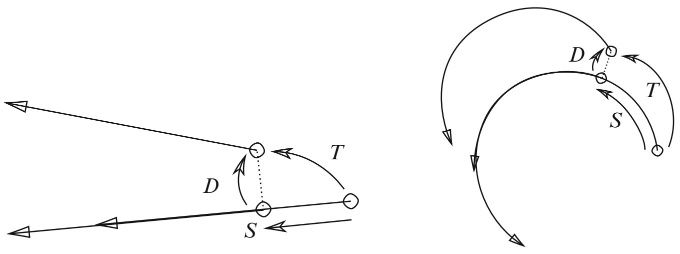


Fig. 2 Decomposition of inter-frame motion into sliding and sticking components

$$\mathbf{R}^{[k+1]} = \mathbf{T}\mathbf{R}^{[k]} = \underbrace{\mathbf{D}}_{\text{sticking}}\underbrace{\mathbf{S}}_{\text{sliding}}\mathbf{R}^{[k]} \tag{2}$$

These two components are used in updating the displacements and forces in the system in two fractional steps. This decomposition is primarily motivated by the different types of tool–tissue mechanical behavior that take place when the needle displaces in the embedding tissue.

When the needle is sliding in the tissue, it is essentially gliding unimpeded except by the (typically small) dynamic friction forces along its path. The tissue deformation induced is small since it is only caused by the frictional forces tangential to the path. On the other hand, when the needle is sticking to the tissue, its motion causes significant bulk deformation of the embedding tissue. Larger forces are induced because of the strain energy build up in the tissue and these forces are needed to maintain the deformed shape.

Both types of motion take place simultaneously during interaction and it is convenient computationally to divide an update step into a sliding step with minimal (or no) friction and tissue deformation, and a step where the needle sticks to the tissue and deforms it in its motion. While the motion takes place, the haptic forces felt by the user are the resultants of the stresses on the needle due to both the sliding and non-sliding steps. If at any point the needle's motion is halted, the forces felt by the user will be only the resultants of the needle contact stresses due to the non-sliding component. This is essentially a viscous model of the frictional phenomenon.

Note that both factors in the decomposition include rotational as well as translational components. Moreover, the **DS** decomposition of the transformation matrix **T** is not unique. Our decomposition strategy is to define the sliding component to be the transformation that causes the needle to move the closest to its final position (Fig. 2), followed by the transformation that causes bulk deformations.

3.2 Needle Tracking

In what follows, we present a strategy for incrementally tracking the cells intersected by the needle to be able to generate the constraints representing sticking motion later on. The strategy relies on using a list of *history* cells and incrementally updating it

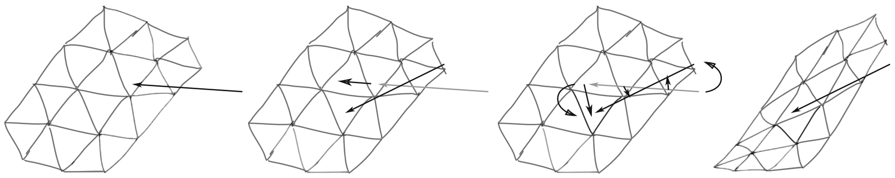


Fig. 3 Example of tracking a straight needle inside a two-dimensional mesh that depicts the general idea behind our proposed approach. Inter-frame needle motion is decomposed into two separate components: sliding (non-deforming) and sticking (deforming), then the needle is slid and its position in the mesh is updated. The new needle position is coupled to the midpoints of the previous cell intersections and the system is solved and updated

using the sliding components and the simplicial adjacency operators provided by the supporting data structure.

The sliding component ($\sigma^{[k]}$), from the inter-frame motion of the needle at step k, is used to incrementally track the position of the needle inside the mesh as it slides. A history of the cells' indices that have been intersected by the needle is saved in a list (Λ) along with the indices of the intersected faces and the barycentric coordinates of their respective intersection points—they are used in imposing needle–tissue interaction constraints. The needle points that correspond to these interaction points are also saved. The entries of Λ are tagged as either active (the needle intersects the corresponding cell) or inactive (there is no intersection).

The first element of Λ always references the cell that contained the tip at the previous step and is used as a starting point when searching for the new location of the tip. The tip is displaced by $\sigma^{[k]}$ and the curve joining its previous and current (slid) position is tested for intersection with the current cell's faces. If there is no intersection, the tip remained in the same cell and only its barycentric coordinates are updated, otherwise, the tip has crossed a face. If the adjacent cell (that shares the same intersected face) is already present in the list, the tip has moved *backward* and the current entry is deleted from Λ, otherwise, the tip is moving *forward* and the adjacent cell is added to Λ along with the needle–tissue interaction points. This process is repeated using the intersected adjacent cell, if any, until the tip is located (inside or outside the mesh). Whenever the tip lies outside the mesh, its path is tested for intersection with the mesh's boundary; this allows for multiple boundary crossing.

After locating the tip inside or outside the mesh using its new position, Λ is traversed backwards to update the parametric coordinates of the needle points and to mark cells that are no longer being intersected by the needle. This fully tracks the position of the needle inside the mesh.

For the sticking motion, we compute for every point of the needle, $s(\alpha)$, that is associated with an active cell, its sticking component ($\tau^{[k]}(\alpha)$) by subtracting the sliding component from $s(\alpha)$'s inter-frame motion. The vertices are driven into their new positions by imposing constraints of the following form:

$$g(t) \cdot \xi = s(\alpha) + \tau^{[k]}(\alpha), \tag{3}$$

where ξ are the barycentric coordinates of the midpoints of the needle segment inside one of Λ's cells with tetrahedron index t. The Lagrange multipliers associated with these constraints are used to compute the resultant forces.

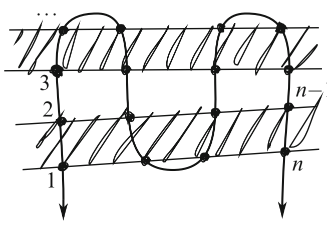
4 Thread Pulling

The needle path inside the body serves to define the topological path in the mesh. The mesh elements crossed by the needle, and the puncture points introduced while entering and exiting the mesh, define an ordered data structure that is used to model the thread–tissue interaction while the thread is pulled to close an opening. Consider, for example, the thread in Fig. 4 that shows a running suture with entry and exit points numbered sequentially from 1 to n. Under displacement control, the user is pulling on the pieces of the thread at points 1 and n. For simplicity of the presentation, we consider here the case of a frictionless interaction between the thread and tissue. The induced tissue deformation and internal stresses are due to the tensile forces in the thread and the contact forces from the closing boundaries.

The pull of the thread is modeled by a constraint equation that constrains the length of the thread portion that is outside the tissue,

$$\sum_{j=1}^{n/2-1} \left(\mathbf{x}_{2j}^{[k+1]} - \mathbf{x}_{2j+1}^{[k+1]}\right) \cdot \hat{\mathbf{d}}_{2j,2j+1}^{[k]} = \sum_{j=1}^{n/2-1} \ell_{2j,2j+1}^{[k]} - \Delta\ell^{[k+1]}, \tag{4}$$

where $\mathbf{x}^{[k]}$ denotes the positions of tissue points at puncture locations at the kth time step. The sum on the left-hand side is over the segments that join successive exit and entry points along the thread. Every term in the sum represents the length of such a segment as described below. $\ell^{[k]}$ is the length of the corresponding portion of the engaged straight thread at time step k and $\Delta\ell^{[k+1]}$ is the incremental pull enacted by the user at time step $k+1$. This constraint presumes and is active only



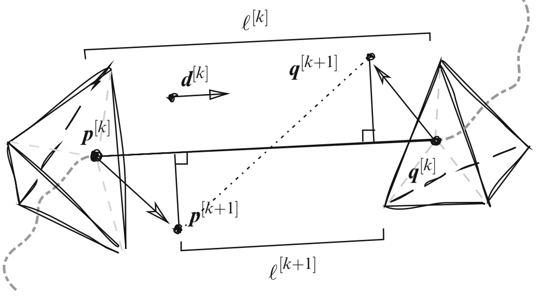


Fig. 4 Sequence of puncture points in a running suture (*left*) and linearization of pulling distance constraint (*right*)

when the thread is fully engaged and has no slack. As long as the right-hand side is less than the actual length of the thread between the two end points, pulling causes pure sliding until the thread is fully engaged. The scalar Lagrange multiplier of this constraint corresponds to the tensile force in the thread as it is pulled. This force may be rendered haptically as needed. Another way of expressing this pull constraint that makes its complementarity clearer is:

$$\sum_{j=1}^{n/2-1} (\mathbf{x}_{2j}^{[k+1]} - \mathbf{x}_{2j+1}^{[k+1]}) \cdot \hat{\mathbf{d}}_{2j,2j+1}^{[k]} + (\mathbf{r}_0 - \mathbf{x}_1^{[k]}) \cdot \hat{\mathbf{d}}_{0,1}^{[k]} + (\mathbf{x}_n^{[k]} - \mathbf{r}_{n+1}) \cdot \hat{\mathbf{d}}_{n,n+1}^{[k]} \leq \ell^{[0]}. \quad (5)$$

$\mathbf{r}_0$ and $\mathbf{r}_{n+1}$ are thread points at locations before first entry (point 1) and after last exit (point n) of the suture. These points are under direct user positional control. $\ell^{[0]}$ is the actual thread length outside the tissue between points 0 and $n+1$. When the left-hand side of (5) is satisfied with strict inequality, the thread has slack and the corresponding Lagrange multiplier is zero. Only when it is satisfied with equality, the thread gets engaged and applies forces to deform the tissue and bring the sutured boundaries together. As the suture closes, additional contact constraints are introduced as described in Sect. 2.

For efficient computations with the pull constraint of (4), we linearize it by measuring the length of every segment as the projection on its direction at the previous step. The justification for this linearization may be explained with reference to Fig. 4. Let $\mathbf{p}$ and $\mathbf{q}$ be two sequential puncture points in the volumetric mesh and $\hat{\mathbf{d}} = (\mathbf{q} - \mathbf{p})/\|\mathbf{q} - \mathbf{p}\|$ be the direction of the line joining $\mathbf{p}$ and $\mathbf{q}$. When $\mathbf{p}$ and $\mathbf{q}$ are pulled close together, the length of the line segment, $\ell = \|\mathbf{q} - \mathbf{p}\|$, joining the two punctures points decreases. At every iteration $\hat{\mathbf{d}}^{[k+1]} \cdot (\mathbf{q}^{[k+1]} - \mathbf{p}^{[k+1]})$ is linearized as $\hat{\mathbf{d}}^{[k]} \cdot (\mathbf{q}^{[k+1]} - \mathbf{p}^{[k+1]})$. The computation of $\hat{\mathbf{d}}$ is also suitably stabilized as the distance between $\mathbf{p}$ and $\mathbf{q}$ becomes small to avoid the near-zero denominator.

5 Results and Conclusion

Figure 5 shows snapshots of an anastomosis procedure using our prototype simulator. The first row shows the initiation of the procedure with the needle puncturing (and deforming) the two vessels at four puncture points. In the second and third rows, running sutures are simulated using a variant number of punctures. Contact constraints prevent the two vessels from interpenetrating and allow the closure of the separating gap. Notice the high principal stress in the regions near the suture.

In order to avoid the small time steps required for the stability of explicit methods [15], we solve the system of constrained equilibrium equations at every step. Our efficient incremental solution strategies, which we will report on in future publications, allow us to achieve haptic rates for medium-sized models with 1,650 nodes and 5,853 linear tetrahedrons. Each step in these simulations takes an average of 30 ms to compute and render using a single 3.3 GHz core, resulting in a refresh rate of 33 FPS on average.

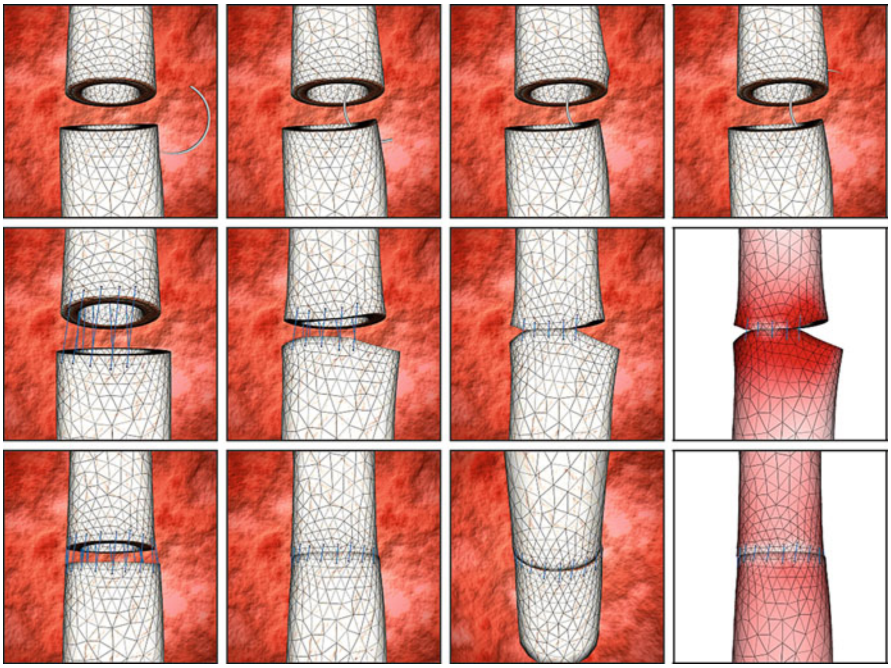


Fig. 5 Needle driving using a $1/2$-circle round body needle (top) and thread pulling with 20 (middle) and 50 (bottom) puncture points. The vessels have different diameters and are discretized with 1,650 nodes and 5,853 tetrahedrons

We believe the techniques presented here form a robust basis for simple and efficient suturing simulation and are currently being used as building blocks for a full-fledged surgical simulator.

Acknowledgments We thank the Lebanese National Council for Scientific Research (LNCSR) for providing support for this project.

References

1. Chentanez, N., Alterovitz, R., Ritchie, D., et al.: Interactive simulation of surgical needle insertion and steering. ACM Trans. Graph. **28**(1), 88:1–10 (2009)
2. DiMaio, S., Salcudean, S.: Needle steering and motion planning in soft tissues. IEEE Trans. Biomed. Eng. **52**(6), 965–974 (2005)
3. Duriez, C., Guébert, C., Marchal, M., et al.: Interactive simulation of flexible needle insertions based on constraint models. Lect. Note Comput. Sci. **5762**, 291–299 (2009)
4. O'Leary, MD., Simone, C., Washio, T et al.: Robotic needle insertion: effects of friction and needle geometry (2003). doi:10.1109/ROBOT.2003.1241851
5. Rosen, J., Brown, J., De, S., et al.: Biomechanical properties of abdominal organs in vivo and postmortem under compression loads. J. Biomech. Eng. **130**(2), 021020 (2008)

6. Laycock, S., Day, A.: A survey of haptic rendering techniques. Comput. Graph. Forum **56**(1), 50–65 (2007)
7. Guébert, C., Duriez, C., Cotin, S., et al.: Suturing simulation based on complementarity constraints. Proc. SCA (Poster) (2009)
8. Otaduy, M.A., Tamstorf, R., Steinemann, D., et al.: Implicit contact handling for deformable objects. Comput. Graph. Forum **28**(2), 559–568 (2009)
9. Tang, M., Manocha, D., Tong, R.: Fast continuous collision detection using deforming non-penetration filters (2010). doi:10.1145/1730804.1730806
10. Teschner, M., Kimmerle, S., Heidelberger, B., et al.: Collision detection for deformable objects. Comput. Graph. Forum **24**(1), 61–81 (2005)
11. Berkley, J., Turkiyyah, G., Berg, D., et al.: Real-time finite element modeling for surgery simulation: an application to virtual suturing. IEEE Trans. Vis. Comput. Graph. **10**(3), 314–325 (2004)
12. Lenoir, J., Meseure, P., Grisoni, L.: A suture model for surgical simulation. Lect. Notes Comput. Sci. **3078**, 105–113 (2004)
13. Wittek, A., Dutta-Roy, T., Taylor, Z., et al.: Subject-specific non-linear biomechanical model of needle insertion into brain. Comput. Meth. Biomech. Biomed. Eng. **11**(2), 135–146 (2008)
14. Choi, K.S., Chan, S.H., Pang, W.M.: Virtual suturing simulation based on commodity physics engine for medical learning. J. Med. Syst. **36**(3), 1781–1793 (2012)
15. Miller, K., Joldes, G., Lance, D., et al.: Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. Comm. Numer. Meth. Eng. **23**(2), 121–134 (2007)