Surface Contact and Reaction Force Models for Laparoscopic Simulation

Clément Forest, Hervé Delingette, Nicholas Ayache

Epidaure Research Project
INRIA Sophia-Antipolis, 2004 route des Lucioles
06902 Sophia-Antipolis, France

Abstract. In surgery simulation, most existing methods assume that the contact between a virtual instrument and a soft tissue model occur at a single point. However, there is a gross approximation when simulating laparoscopic procedures since the instrument shaft is used in several surgical tasks. In this paper, we propose a new algorithm for modeling the collision response of a soft tissue when interacting with a volumetric virtual instrument involving both the shaft and the tip of the instrument. The proposed method generates visually coherent mesh deformations and plausible force-feedback in a real-time surgical simulator even when the mesh geometry is irregular.

1 Introduction

The purpose of a surgical simulator is essentially to provide a computerized system suitable for the training of young residents. This system can be decomposed into two components: a user interface and a simulation engine. The nature of the user interface is clearly important because a large part of the training consists in acquiring gesture skills. As an example, in the context of a laparoscopic simulator, the length of surgical instruments (nearly 30 cm) and their specific motion must be carefully modeled. Indeed, because the motion of those instruments are restricted to pass through a fixed point, surgeons often use the shaft of their instruments to gently push soft tissue away without causing any bleeding. For instance, this type of gesture is used quite extensively to manipulate the liver in cholecysectomy procedures.

In most surgical simulators, when considering the collision response of soft tissues with a virtual instrument, that interaction is assumed to occur at a single point [1]: only the possible contact with the instrument tip is considered. Of course this assumption greatly simplifies the collision response algorithm and consequently decreases the computation time. However, it also worsens the realism of the simulation and, most

importantly, it significantly limits the set of gestures that can be learned by medical residents.

In [2, 3], Ho *et al.* propose an haptic rendering technique that can model the contact between a line segment and an object. However, this approach is only suitable for convex objects or for objects that can be divided into a limited number of convex components. More recently, Picinbono *et al.* [4] have introduced an algorithm that can handle the collision between laparoscopic instruments and soft tissue meshes which consists in the projection of vertices on an average plane. Unfortunately, that approach is only valid when the surface of the soft tissue model is smooth. Therefore, it cannot be used during the resection of soft tissue when the surface can become quite irregular.

In this paper, we describe a surface contact and reaction force model that is suitable for the simulation of surgical instruments interacting with soft tissue for laparoscopic simulation. Note that we do not focus on the action of specific instruments like a bipolar cautery device but solely on the mechanical contact caused by the shaft or tip of instruments. Those models can be applied on any type of triangulated surface whether it is smooth or not.

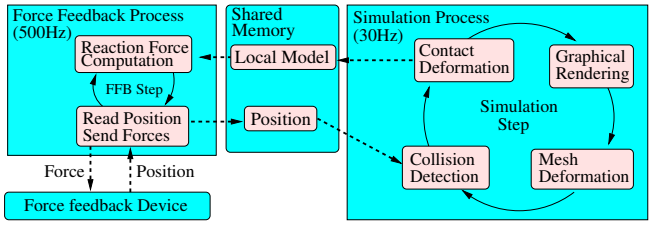


Fig. 1. Architecture of the Epidaure Laparoscopic Simulator

The contact and reaction force models are presented in the context of the hepatic surgery simulator developed in the Epidaure Project at INRIA Sophia-Antipolis. The simulated procedure in this platform is the resection of one or several anatomical segments of the liver with an ultrasonic-based device called a cavitron [5]. The system includes three force-feedback devices that serve as input devices for two instruments and one endoscope. Those devices are currently Laparoscopic Impulse Engines (LIE) from Immersion Corp. In figure 1, a sketch of the simulation engine is shown. In this paper, we only partially describe the *contact deformation* and the *reaction force processing* algorithms. The former computes the displacement of vertices entailed by the collision between an instrument and the liver while the latter computes the reaction forces that are felt by the user when manipulating the surgical instruments. The description

of other algorithms falls outside the scope of this paper. In a nutshell, the collision detection between each virtual instrument and each soft tissue model is based on graphics acceleration [6] due to the cylindrical geometry of those instruments. The tetrahedral mesh of the liver is deformed according to the *tensor-mass* algorithm [7], based on linear elasticity and finite-elements modeling. Finally, topological modifications use a removal tetrahedra method [5] which maintains the manifold property of the mesh.

2 Contact Deformation Algorithm

2.1 Introduction

In physically-based simulation, there are two common methods for simulating the contact between objects. The former one is called the *constraint method* [8]: whenever a collision is detected at the given time step, the exact time of the collision is determined, then the position or shape of those objects are updated and the simulation resumes at the time of collision. Because it implies moving back in time, this approach is widely used in computer animation but is hard to adapt to real-time applications.

The second method for simulating the contact between objects is the *penalty method* [9] and it consists in adding a force proportional to the inter-penetration distance as to push both objects apart. This approach is simple to implement and is in general used in real-time simulation or when the objects geometry is complex, as in clothes simulation [10]. However, the choice of the optimal amplitude of the reaction force is difficult to estimate and it does not truly prevent the collision between those objects. This last limitation has been reduced for force feedback applications through the *god-object* method [11]. This approach was devised to model the contact between a point and a rigid surface and was later extended by Ho [2] to include the contact with a line segment and a rigid object. Its main idea is to maintain simultaneously two positions of the probe, one being its *ideal* position (always located outside or on the object surface) the other being its *virtual* corresponds to the actual position of the external device.

In this paper, we are modeling the contact of instruments with soft tissue and not rigid objects. Since a collision can be caused by the mesh deformation, the proxy method cannot be used directly. Indeed, it might be impossible to determine a non colliding ideal position for the probe. Furthermore, our algorithm copes with cylindrical instruments (not only line segments) and with general soft tissue surfaces, convex or non convex, smooth or non smooth. It proceeds in five steps (see Figure 2):

1. Definition of a reference frame;
2. Determining colliding edges;
3. Preventing edge collision;
4. Moving triangles away from the tip of the instrument;
5. Moving edges and vertices outside of the instrument volume.

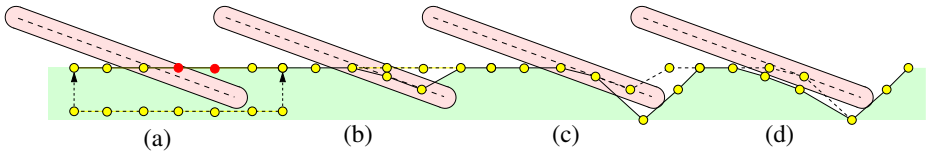


Fig. 2. The main steps of the algorithm seen in the instrument frame. (a) Determining the colliding edges. (b) Preventing the edge collision. (c) Moving triangles out of the instrument tip. (d) Moving edges and vertices out of the instrument volume.

2.2 Definition of a reference frame

In the remainder, each virtual instrument, independently of its nature, is represented as the dilatation of radius r of a line segment of length l. Let Z and P be the instrument main axis direction and tip position at the current time step and Z_{old}, P_{old} be its direction and tip position at the previous time step. During the last time step, the instrument and the mesh representing the soft tissue may have moved. To simplify the analysis of the collision, we propose to consider the relative displacement of the mesh with respect to the instrument. Because all surgical instruments are supposed to be rigid objects, we need to determine the rigid transformation $\mathcal{R}$ that transforms the instrument position from its previous configuration $\{Z_{old}, P_{old}\}$ into its current configuration $\{Z, P\}$. This transformation is simply determined by writing the 3 equalities of equation 1.

$$\left.\begin{array}{rl} \mathcal{R}(Z_{old}) = Z & (a) \\ \mathcal{R}(P_{old}) = P & (b) \\ \mathcal{R}(Z_{old} \wedge Z) = Z_{old} \wedge Z & (c) \end{array}\right\} \quad (1)$$

The relative displacement between a virtual instrument and a vertex A, moving from position $A^{t-\Delta t}$ to position A^t can then be estimated in the reference frame of the instrument at its current state $\{Z, P\}$. In this frame, point $A^{t-\Delta t}$ is transformed into $\mathcal{R}(A^{t-\Delta t})$. To simplify the analysis, we will consider that in this frame, the trajectory linking $\mathcal{R}(A^{t-\Delta t})$ and A^t is a straight line. This assumption is justified by the relatively small speed at which a surgical instrument is moved compared to the frequency (nearly $30Hz$) of the trajectory analysis. We propose to further simplify

notations by writing $A = \mathcal{R}(A^{t-\Delta t})$ and $\Delta A = A^t - \mathcal{R}(A^{t-\Delta t})$ such that $A + \Delta A = A^t$ (dropping the temporal exponent).

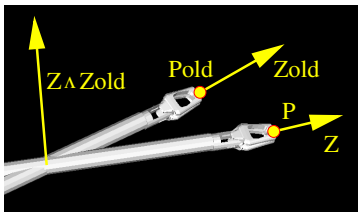


Fig. 3. The instrument rigid transformation

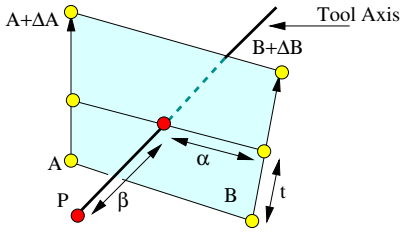


Fig. 4. Detection of the intersecting edges

2.3 Determining colliding edges

Let $E = \{A, B\}$ be an edge of the soft tissue mesh. We want to find whether this edge has intersected the axis during the last time step. Note that the computation of the intersection between two edges, one static and one moving has been proposed by Schömer [12] but we believe that the solution below is more efficient to implement. Following the assumptions on the linearity of the points trajectory in the reference frame, edge E collided with the virtual instrument if there exists (t, α, β) in $[0,1]\times[0,1]\times -]\infty, 0]$ such that:

$$A + t.\Delta A + \alpha(B + t.\Delta B - A - t.\Delta A) = P + \beta Z \tag{2}$$

The variable t represents the instant of collision, and α and β are the relative positions of the point of collision with respect to the edge and the instrument axis. To find possible solutions, we first eliminate parameters α and β by taking the dot product of equation 2 with vector $(B+t.\Delta B - A - t.\Delta A) \wedge Z$. This leads to a second degree equation of variable t:

$$\begin{aligned} t^2.(\Delta A.((\Delta B - \Delta A)) \wedge Z) + t.(\Delta A.((B - A) \wedge Z) \\ -(P - A).(\Delta B - \Delta A) \wedge Z)) = (P - A).((B - A) \wedge Z) \end{aligned} \tag{3}$$

Lets t_1 and t_2 be the roots of that equation. The corresponding values of α and β can be computed easily by taking the vectorial product of the equation (2) with the vector Z and with the vector $(B+t_i.\Delta B - A - \Delta B)$ respectively:

$$\alpha_i = \frac{\| (P - A - t_i.\Delta A) \wedge Z \|}{\| (B + t_i.\Delta B - A - t_i.\Delta A) \wedge Z \|} \tag{4}$$

$$\beta_i = \frac{\| (A + t_i \Delta A - P) \wedge (B + t_i \Delta B - A - t_i \Delta A) \|}{\| Z \wedge (B + \Delta B - A - t_i \Delta A) \|} \tag{5}$$

If exactly one of the two triplets (t_i, α_i, β_i) corresponds to an intersection (ie. is inside the set $[0,1] \times [0,1] \times [0,l]$), then we consider that edge E "has crossed the instrument axis" and is called a *colliding edge*. Otherwise, we consider that no collisions between the edge and instrument have occurred. Also, if the instrument has collided the mesh and bounced back during the previous time step, this collision will not be taken into account.

Finding all colliding edges could be very computationally intensive, if all edges were tested. Instead, we take advantage of the list of triangles that is outputed by the collision detection algorithm. Those triangles are intersected by the virtual instrument in its current position $\{Z, P\}$ and therefore have colliding edges. From those edges, we use a marching algorithm that searches for colliding edges from one triangle to the next, towards the tip of the instrument, until no additional colliding edge is found.

Using collision detection to find colliding edges may not be reliable if the instrument entirely crosses the mesh in one time step. Again, in the context of surgery simulation, given the speed of the tip of the instrument and the typical shape of the liver, this should not occur if the main process runs at nearly $30Hz$.

2.4 Preventing edge collision

The second stage of our contact processing algorithm consists in moving vertices in order to prevent all edges from crossing the instrument axis. One could simply stop the movement of a vertex relatively to the instrument as soon as one of its adjacent edges intersects the axis of the instrument. Unfortunately, for some configurations, this method does not prevent edges from crossing the axis (see Figure 5).

For each vertex adjacent to a colliding edge we assign a *confidence interval* $[t_{min}, t_{max}]$ such that when the vertex position is moved within this interval, we can guaranty that none of its adjacent edges cross the instrument axis. We first initialize that interval to $[0,1]$ for all vertices. Then we successively consider every edge $E = (A, B)$ that crosses the axis at instant $t_c \in [0,1]$. Let ΔA be the displacement of vertex A during the last time step *in the reference frame*. If the motion of vertex A moves edge E *closer* to the axis, then the new confidence interval for this vertex is set to $[t_{min}, t_{max}] \cap [0, t_c]$; otherwise it is set to $[t_{min}, t_{max}] \cap [t_c, 1]$. To find whether the vertex motion will make edge E become closer or further

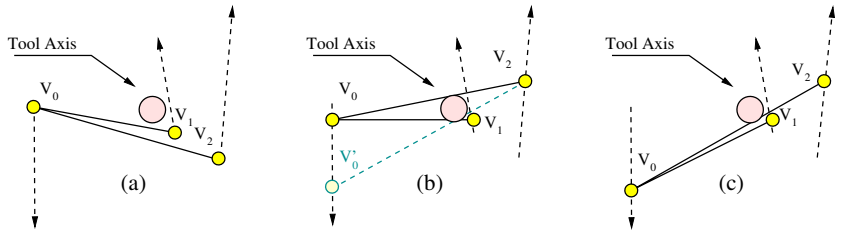


Fig. 5. (a) Initial position of two edges sharing a vertices in the reference frame. Arrows indicate the displacements from the previous to the current configuration. (b) Axis intersections are tested for each edge independently and vertices are stopped as soon as an adjacent edge intersect the axis: V_0 and V_1 are stopped first followed by V_2, but this does not prevent the edge V_0V_2 from crossing the axis. (c) Vertices are stopped using our method described below.

away from the axis, we just look at the sign of following expression:

$$(\Delta A \wedge (Z \wedge AB)).(AP \wedge (Z \wedge AB)) \tag{6}$$

When processing an edge, the confidence interval of one vertex may become empty. In this case, the vertex position is set to its position at t_{min} if the vertex bring the edge towards the axis and to its position at t_{max} otherwise. With that vertex set to a fixed position, we compute the new instant $t_c^{'}$ of the collision with that particular edge and we update the confidence interval for its other adjacent vertex; in practice, that confidence interval appears to be never empty.

Once a confidence interval is assigned for all the vertices we move them to the position corresponding to the center of their confidence interval.

2.5 Moving triangles at the tip of the instrument

At the end of the previous stage, all colliding edges have been removed. However, if the tip of the instrument was previously colliding with the mesh, that collision should still occur. The next step consists in moving the colliding triangle outside the volume of the instrument. Lets P_{prox} be the intersection point between the mesh and the axis. If they do not intersect, P_{prox} is set to the mesh point closest to the tip of the instrument. We compute the normal vector n_{prox} at P_{prox} with the following formula:

$$n_{prox} = \alpha_0 n_0 + \alpha_1 n_1 + \alpha_2 n_2 \tag{7}$$

where α_0, α_1 and α_2 are the barycentric coordinates of P_{prox} in his triangle and where n_0, n_1 and n_2 are the normals at the triangle vertices. We

propose to project the triangle containing P_{prox} on a plane $\mathcal{P}$ along the axis Z. The plane $\mathcal{P}$ is orthogonal to n_{prox} and is slightly moved away at a distance r from the tip of the instrument.

2.6 Moving edges and vertices outside of the instrument volume

In the last stage of the algorithm, edges and vertices located in the neighborhood of the instrument are moved outside the cylindrical volume of the instrument. First, vertices are moved away in a direction orthogonal to the axis Z and then edges are eventually pushed away from the cylindrical volume by computing the closest distance from that edge to the main axis of the instrument.

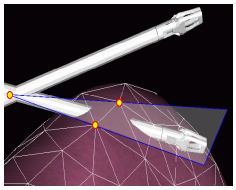
3 Reaction Forces Computation

In the literature, there are three main algorithms for computing reacting forces. First, reacting force can be precomputed before the simulation and extrapolated in real time [13]. They can also be computed according to the current mechanical model [1] or be estimated in a pure geometric manner based on inter-penetration distance. We chose to provide a stable and efficient solution by combining the last two approaches. In order to deal with the difference in update rate between the mechanical model (30 Hz) and the reaction force model (at least 300 Hz for a stable haptic feedback with soft tissues [14]), we use a method based on the *buffer model* [15]. A separate reaction force loop transmits the positions of the input devices and receives a simplified local representation of the soft tissue model suitable for an efficient force computation. Transitions between two successive local models are made progressively in order to smooth irregularities.

Our local model consists of two planes. The former represents the neighborhood of the tip of the instrument and is defined as the plane P having normal n_{prox} (see section 2.5). The latter represents the contact with the shaft of the instrument and contains the instrument main axis and is orthogonal to the average cross product of the instrument axis with the direction of all neighboring edges. For each of those two planes, a force F_i is computed which is proportional to the penetration depth of the tip in the half-space delimited by the corresponding plane. Therefore, if n_i and A_i are respectively the normal and a point of the i-th plane, the force F_i is computed in the following way:

$$F_i = \lambda_i * (P - A_i).n_i \tag{8}$$

In this equation λ_i is the "apparent stiffness" of the material. This value is proportional to the average Young Modulus of the tetrahedra located in the neighborhood of the instrument (since our soft tissue model is based on non-homogeneous linear elastic materials). The average Young Modulus is sent in the local model and allows to feel the difference between a soft and a stiff part of the model.



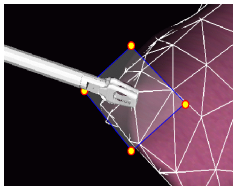


Fig. 6. The two planes composing the local model sent to the reaction force separate process.

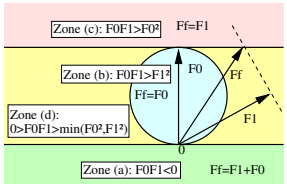


Fig. 7. The value of F_f given F_0 and according F_1

To obtain a force which is smooth over time, the final force F_f is not the sum of the two forces. Indeed, there are many configurations for which the collision between the instrument and the mesh may randomly include or not the tip of the instrument: this could generate a discontinuous force thus leading to vibrations. Therefore, we propose to combine then is such that if the two forces are pointing in the same direction, then the resulting force is the force vector of maximum intensity. If the two forces are orthogonal or parallel in opposite direction, then the resulting force is the sum of the two forces vectors (see also Figure 7).

$$\left.\begin{array}{ll} \text{if } F_0.F_1 \leq 0, \quad F_f = F_0 + F_1 & (a) \\ \text{if } \|F_0\|^2 \leq F_0.F_1, F_f = F_0 & (b) \\ \text{if } \|F_1\|^2 \leq F_0.F_1, F_f = F_1 & (c) \\ \text{else } F_f = \frac{F_0*(\|F_1\|^2*(F_0.F_1-\|F_0\|^2))+F_1*(\|F_0\|^2*(F_0.F_1-\|F_1\|^2))}{(F_0.F_1)^2-\|F_0\|.\|F_1\|} & (d) \end{array}\right\} \quad (9)$$

Once computed, that force is split into axial and orthogonal components. Those components are sent to the force feedback device once translated into axial force and torques directives.

4 Conclusion

We have presented a method to simulate the contact between a soft tissue triangular mesh and a volumetric virtual surgical instrument including its shaft and its tip. It provides a robust way to constraint the mesh to

keep outside the cylindrical volume of the instrument and to generate coherent and stable reaction forces. Its main limitation may lay in the computation of the reaction forces that is mostly based on geometric reasoning. In most circumstances, the computed force would differ from the purely mechanical forces computed from the finite element method.

References

1. Mendoza, C., Sundaraj, K., Laugier, C.: Faithfull Force Feedback in Medical Simulators. In: International Symposium in Experimental Robotics. Volume VIII: Experimental Robotics of Tracts in Advanced Robotics. Springer, Italy (2002)
2. Ho, C.H., Basdogan, C., Srinivasan, M.: Ray_based haptic rendering: Force and torque interactions between a line probe and 3d objects in virtual environments. International Journal of Robotics Research **19** (2000) 668–683
3. Basdogan, C., Ho, C.H., Srinivasan, M.A.: Virtual environments for medical training: Graphical and haptic simulation of laparoscopic common bile duct exploration. IEEE/ASME Transactions on Mechatronics **6** (2001) 269–285
4. Picinbono, G., *et alt.*: Improving realism of a surgery simulator: linear anisotropic elasticity, complex interactions and force extrapolation. Journal of Visualisation and Computer Animation **13** (2001) 147–167
5. Forest, C., Delingette, H., Ayache, N.: Cutting simulation of manifold volumetric meshes. In Dohi, T., Kikinis, R., eds.: Medical Image Computing and Computer-Assisted Intervention (MICCAI'02). Volume 2489 of LNCS., Tokyo, Springer (2002) 235–244
6. Lombardo, J.C., Cani, M., Neyret, F.: Real-time collision detection for virtual surgery. In: Computer Animation, Geneva Switzerland (1999)
7. Cotin, S., Delingette, H., Ayache, N.: Real-time elastic deformations of soft tissues for surgery simulation. IEEE Transactions On Visualization and Computer Graphics **5** (1999) 62–73
8. Witkin, A., Baraff, D., Kass, M.: An introduction to physically based modeling (1994) SIGGRAPH'94 Course Notes, Course No. 32.
9. Deguet, A., Joukhadar, A., Laugier, C.: Models and algorithms for the collision of rigid and deformable bodies. In: Robotics: the algorithmic perspective. AKPeters (1998) 327–338
10. Bridson, R., Fedkiw, R., Anderson, J.: Robust treatment of collisions, contact and friction for cloth animation. In: 29th annual conference on Computer graphics and interactive techniques, ACM Press (2002) 594–603
11. Zilles, C.B., Salisbury, J.K.: A constraint-based god-object method for haptic display. In: International Conference on Intelligent Robots and Systems. Volume 3., Pittsburgh, Pennsylvania (1995) 146–151
12. Schömer, E., Christian, T.: Efficient collision detection for moving polyhedra. In: Proc. of the Eleventh Annual Symp. on Computational Geometry. (1995) 51–60
13. Mahvash, M., Hayward, V.: Haptic simulation of a tool in contact with a nonlinear deformable body. In: Surgical Simulation and Soft Tissue Deformation. Volume 2673 (lncs)., Juan-les-pins, France (2003) 311–320
14. Ellis, R.E., Ismaeil, O.M., Lipsett, M.: Design and evaluation of a high-performance haptic interface. Robotica **14** (1997) 321–327
15. Balaniuk, R.: Using fast local modeling to buffer haptic data. In: Proceeding of the 4th PhantoM User Group Workshop (PUG'99), Cambridge (1999) 7–11