Interactive Contacts Resolution Using Smooth Surface Representation

Jérémie Dequidt[1], Julien Lenoir[2], and Stéphane Cotin[1,3]

[1] SimGroup, CIMIT, Cambridge, USA
dequidt@lifl.fr
[2] Alcove Project, LIFL/INRIA Futurs, Lille, France
lenoir@lifl.fr
[3] Harvard Medical School, Boston, USA
cotin.stephane@mgh.harvard.edu

Abstract. Accurately describing interactions between medical devices and anatomical structures, or between anatomical structures themselves, is an essential step towards the adoption of computer-based medical simulation as an alternative to traditional training methods. However, while substantial work has been done in the area of real-time soft tissue modeling, little has been done to study the problem of contacts occurring during tissue manipulation. In this paper we introduce a new method for correctly handling complex contacts between various combination of rigid and deformable objects. Our approach verifies Signorini's law by combining Lagrange multipliers and the status method to solve unilateral constraints. Our method handles both concave and convex surfaces by using a displacement subdivision strategy, and the proposed algorithm allows interactive computation times even in very constrained situations. We demonstrate the efficiency of our approach in the context of interventional radiology, with the navigation of catheters and guidewires in tortuous vessels and with the deployment of coils to treat aneurysms.

1 Introduction

Real-time soft tissue modeling has been the focus of a majority of publications in the field of medical simulation [1,2,3,4]. This can be explained by the importance of tissue-tool interactions in the overall realism of a simulation, but also by the complexity of the problem. Accurately modeling the deformation of an anatomical structure during tissue manipulation is a very difficult task, in particular when non-linear stress-strain relationships are required while at the same time maintaining real-time computation [2,4]. However, even complex models cannot correctly describe soft tissue deformations unless the contacts occurring during tissue manipulation are correctly determined. Very often, interactions are limited to a single point of contact [5] or at least a very localized area of contact, that would correspond for instance to grasping the tissue or probing it. There are many medical procedures, however, where such limited interactions are not sufficient. For instance, in surgery, tissue palpation requires much more complex

N. Ayache, S. Ourselin, A. Maeder (Eds.): MICCAI 2007, Part II, LNCS 4792, pp. 850–857, 2007.

interactions, and in laparoscopic surgery, many procedures require a combination of sliding and grasping that cannot be modeled by usual approaches. A very illustrative example is found in interventional radiology, with catheter navigation or coil deployment. Such procedures involve inserting or deploying flexible devices in very tight spaces, thus leading to a large number of contacts combined with sliding conditions. Finally, besides contacts occurring during tissue-instrument interactions, it is typical in surgical procedures that organs slide and collide against other anatomical structures. Taking into account that type of contacts would certainly have a positive impact on the realism of the simulation.

Modeling contacts involves not only detecting the occurrence of a contact but also computing the involved structures' collision response. While the problem of collision detection has been often addressed in Computer Graphics and to some extent in medical simulation, the issue of modeling the collision response has mostly been addressed in the fields of Mechanical Engineering and Robotics. However, when taking into account the particular constraints inherent to real-time simulation and deformable structures, little has been done. Among the most relevant work, Kry *et al.* [6] propose a technique well suited for evolving contacts on a smooth surface, incorporating both slip and no-slip friction. Their method is very fast but only handles simple contacts between rigid surfaces, which need to be described as a parametrization. Garcia *et al.* [7] introduce a fast algorithm based on fuzzy logic. Using kinetic and geometric information of a surgical tool interacting with an organ, they compute the new position of the colliding vertices using simple rules. However this method is limited to collisions between a rigid object and a deformable one and the collision response, based on a projection technique, does not take into account the physics of the objects in contact. The method we previously introduced in [8] handles contacts by computing the local compliance at the point of contact and by describing it in the contact space using the Delassus operator. Collision detection is performed using a proximity measure with the triangulation of the surface. The problem of multiple contacts is then solved using an iterative solver, in particular a Gauss-Seidel algorithm. Although efficient, this approach has a non-optimal rate of convergence that lead to either non plausible behavior or increased computation times.

In this paper, we present a real-time algorithm based on Lagrange multipliers to handle multiple complex contact situations between combinations of rigid and deformable objects. After introducing some basic notions of contact mechanics, we present in section 2.2 a method based on Lagrange multipliers to design a fast, robust and generic algorithm to handle contacts between hundreds of colliding degrees of freedom. We then describe in section 2.3 a fast collision detection method based on implicit surface representation, and we conclude with a series of results in the context of interventional radiology.

2 Contact Modeling

Modeling contacts is a well known topic in Computational Mechanics and has recently been an active research field in Computer Graphics [9,10,11]. In this

section we give an overview of the mechanics of contacts, in particular Signorini's law and its linearization in the contact space, formulated as the Delassus operator. Then we present our approach based on the use of Lagrange multipliers and a displacement subdivision strategy.

2.1 Mechanics of Contacts

Definition. A contact is an unilateral constraint applied on a specific point $\boldsymbol{P}$ (the contact point): $g(\boldsymbol{P}) \geq 0$. A mechanical system usually defines a set of degrees of freedom (DOFs) $\boldsymbol{x}$ characterizing its physical state. Given this notation, a contact g_i links several DOFs via a linear relationship $g_i(\boldsymbol{x}) = \sum_j h_{ij} x_j$. By extension, a set of contacts $g_i(\boldsymbol{x}) \geq 0$ noted $g(\boldsymbol{x})$ are linked to a set of DOFs via a matrix $g(\boldsymbol{x}) = H\boldsymbol{x}$ where each line of H expresses a contact relatively to the DOFs.

Signorini's Law. The conditions of contact are given by the Signorini's law, for each point $\boldsymbol{P}$ of the contact area, as: $0 \leq \delta^n_{\boldsymbol{P}} \perp f^n_{\boldsymbol{P}} \geq 0$ where $\delta^n_{\boldsymbol{P}}$ is the interpenetration distance evaluated at $\boldsymbol{P}$ (shortest euclidian distance to the other object's surface) and $f^n_{\boldsymbol{P}}$ the amplitude of the normal force needed to solve the contact. In the case of frictional sliding, a tangential component $f^t_{\boldsymbol{P}}$ is introduced, leading to a contact force $f_{\boldsymbol{P}} = f^n_{\boldsymbol{P}} + f^t_{\boldsymbol{P}}$. From a mathematical stand point, this law translates the orthogonality between the contact force and the interpenetration distance $\delta^n_{\boldsymbol{P}} f^n_{\boldsymbol{P}} = 0$. This means that either an interpenetration occurs, requiring a non-null normal force to bring back the contact, or that the constraint is not violated because the distance to the surface is non-null, therefore no force is required to correct the position.

Delassus Operator. When dealing with simulations, the motion of the objects is discretized into a series of time steps. Some of the external forces are known at the beginning of a time step (gravity, user-specified forces, etc.) while others only appear during the time step, and depend on the current state of the mechanical system. This is the case of the contact forces. Such forces are called *implicit*, while the known ones are called *explicit*. Dealing with implicit forces leads in general to solving a non-linear problem. If the deformations are linear (or linearized during the time step), a way of dealing with both *implicit* and *explicit* forces is to split the computation in two steps. First we compute a configuration called *free motion*, noted $\boldsymbol{x^f}$, in which we take into account only the explicit forces, not the contacts. Second, a collision detection is performed and a corrective motion $\boldsymbol{x^c}$ is computed. The correction $\boldsymbol{x^c}$ is such that the final position $\boldsymbol{x} = \boldsymbol{x^f} + \boldsymbol{x^c}$ verifies the unilateral constraints. Separating explicit and implicit forces independently is a consequence of the superposition principle, and therefore can only be applied if the equations of motion are linearized. After computing $\boldsymbol{x^f}$ we can evaluate the actual contact violation $\delta^{nfree}_{\boldsymbol{P}}$ and solve the contact problem $\delta^n_{\boldsymbol{P}} = \left(HCH^T\right) f^n_{\boldsymbol{P}} + \delta^{nfree}_{\boldsymbol{P}}$ where C is the compliance matrix of the mechanical

system, and the matrix HCH^T expresses the contact's coupling in the contact space[1] This operator is well known in mechanics as the *Delassus* operator, and the previous equation is called a linear complementary problem (LCP). A LCP can be solved in different ways, using a Lemke or Gauss-Seidel technique for instance. Once the contact problem is solved, we obtain the contact force $f_{\boldsymbol{P}}^n$. Since $f_{\boldsymbol{P}}^n$ is defined in the contact space, it has to be transformed back to the DOFs space before being applied to $\boldsymbol{P}$. We write $\boldsymbol{f} = H^T f_{\boldsymbol{P}}^n$, and the solution $\boldsymbol{x}$ verifying the constraints is then determined by $\boldsymbol{x} = \boldsymbol{x^f} + (CH^T) f_{\boldsymbol{P}}^n$.

2.2 Solving Contacts with Lagrange Multipliers

Lagrange multipliers is a well known mathematical method to define bilateral constraints, although there exist a few references of work using Lagrange multipliers to solve unilateral constraints, such as [12] for instance. In this section we describe our contact modeling algorithm and show the equivalence between the definition of a contact in classical mechanics (Signorini's law and Delassus operator) and the use of Lagrange multipliers.

Solving Unilateral Contacts. If we assume that several objects are in contact (these objects can be deformable, rigid or even inert) then we can define a mechanical system representing this set of objects. Whether it is static or dynamic, the stiffness or mass matrix of the system will have a similar structure, i.e. a block diagonal matrix where each block is the stiffness or mass matrix of an object within the mechanical system. Without lack of generality, lets assume the system is static, and that its stiffness matrix is $\boldsymbol{K}$. In the absence of contacts, each block of $\boldsymbol{K}$ is independent of the other ones. When contacts are detected, we introduce Lagrange multipliers in the system, thus creating a dependency between certain DOFs. Then, if we take into account the decomposition $\boldsymbol{x} = \boldsymbol{x^f} + \boldsymbol{x^c}$, the contact problem can be described as:

$$\left\{\begin{array}{rcl} K\boldsymbol{x^f} & = & \boldsymbol{f} \\ K\boldsymbol{x^c} & = & H^T\boldsymbol{\lambda} \\ H(\boldsymbol{x^c} + \boldsymbol{x^f}) & = & \boldsymbol{\delta} \end{array}\right. \Leftrightarrow \left\{\begin{array}{rcl} K\boldsymbol{x^f} & = & \boldsymbol{f} \\ \left(HK^{-1}H^T\right)\boldsymbol{\lambda} & = & \boldsymbol{\delta} - H\boldsymbol{x^f} \\ \boldsymbol{x^c} & = & K^{-1}H^T\boldsymbol{\lambda} \end{array}\right.$$

Two steps are required to solve these equations. First we compute the free motion from the explicit forces ($\boldsymbol{x^f} = K^{-1}\boldsymbol{f}$). Given $\boldsymbol{x^f}$, we perform a collision detection that allows us to evaluate H, and therefore $\boldsymbol{\delta}$. We then solve $\left(HK^{-1}H^T\right)\boldsymbol{\lambda} = \boldsymbol{\delta} - H\boldsymbol{x^f}$ and obtain $\boldsymbol{\lambda}$. Since we are dealing with unilateral constraints, not all constraints are necessarily needed to enforce the inequality condition on $\boldsymbol{x}$. Redundant constraints (for which the corresponding value in $\boldsymbol{\lambda}$ is negative) are then deactivated. This is the so called *status method*. At this point, we can evaluate the corrective motion $\boldsymbol{x^c} = K^{-1}H^T\boldsymbol{\lambda}$ and compute the new position $\boldsymbol{x} = \boldsymbol{x^f} + \boldsymbol{x^c}$. However, this new configuration does not necessarily meet the

[1] In the case of a static system, we have $C = K^{-1}$. For a dynamic system, the previous equations involve the acceleration $\ddot{\boldsymbol{x}}$, thus requiring a time integration to determine $\boldsymbol{x}$. Using an Euler implicit integration scheme, this leads to $C = \left(\frac{M}{\Delta t^2} + \frac{D}{\Delta t} + K\right)^{-1}$.

initial constraints since we use a linear approximation of the local shape at the point of contact. As a consequence, an iterative scheme is introduced, during which a collision detection is performed on the new configuration $\boldsymbol{x}$ to check if some contacts are still violated (lines 18 to 22 in the algorithm below). If it is the case, a new evaluation of H is performed, and a new value of $\boldsymbol{x^c}$ is computed. This is repeated until all current contacts are solved, and in most cases, less than 10 iterations and required. Since K^{-1} does not need to be recomputed, if the collision detection is handled efficiently (see section 2.3), these iterations lead to a limited overhead. At this point, all contacts initially detected are solved. However, when solving these contacts, it is likely that new ones will appear. This is typical of any collision response algorithm. In our case we solve all contacts within a given time step, rather than the next time step. This explains the main loop (lines 7 to 22) in the algorithm below. Checking for new contacts within the same time step adds a computational overhead but ensures a more consistent (and contact free) configuration at the beginning of the following time step.

```
1  Solve Kx^f = f
2  contact = () , done = true
3  for i ← 1 to n do
4      if DetectCollision(P_i(x^f)) then
5          contact+ = (H_i, δ_i)
6          done = false
7  while !done do
8      repeat
9          Solve (HK^{-1}H^T).λ = δ − Hx^f
10         // I) Constraints deactivation using status method
11         done = true
12         if ∃ i | (λ_i < 0) then
13             Remove from contact : contact_j | (λ_j = min(λ_i))
14             done = false
15     until done;
16     x^c = (K^{-1}H^T) λ
17     // II) Constraints activation
18     done = true
19     for i ← 1 to n do
20         if DetectCollision(P_i(x^f + x^c) ) then
21             contact+ = (H_i, δ_i)
22             done = false
```

Equivalence with the Mechanics of Contacts. One can note that $HK^{-1}H^T$ represents the contacts coupling, which is exactly the meaning of the Delassus operator defined in section (2.1) with $K^{-1} \equiv C$ the compliance matrix of the mechanical system. Moreover, the Lagrange multipliers $\boldsymbol{\lambda}$ give the force in the contact space, which is equivalent to $f^n_{\boldsymbol{P}}$ in the Delassus operator approach. This means the corrective motion computed using Lagrange multipliers is identical to the one derived from the Delassus operator, i.e. $\boldsymbol{x^c} = K^{-1}H^T\boldsymbol{\lambda} = CH^T f^n_{\boldsymbol{P}}$. The

equivalence between Signorini's law / Delassus operator and our approach based on Lagrange multipliers / status method is very important as it shows that the contacts are modeled accurately with our approach.

2.3 Collision Detection

Implicit Surface Modeling. For organic shapes (i.e. shapes that do not exhibit sharp features) a fast collision detection can be performed by using an implicit description of the surface, rather than a triangulation. The surface is then described using a combination of geometrical primitives and a convolution filter. The primitives can be either points, segments, triangles or other simple shapes. The convolution filter h is defined as a function from $\mathbb{R}^3 \longrightarrow \mathbb{R}^+$ with a finite support or fast decay to 0 and the resulting surface is $f(P) = h(P) \otimes s = iso$, where iso is an isosurface value and $f(P)$, with the potential at P a point in $\mathbb{R}^3$. In addition, pathologies such as tumors or aneurysms can be modeled by locally modifying the potential field.

Collision Detection. Given a function g defined as $g = f - iso$ and a point P at two different time steps t and $t+1$, the collision detection consists in finding where $[P_t, P_{t+1}]$ intersects the surface f. This is equivalent to finding the first root i_0 of g on the interval $[P_t, P_{t+1}]$. This is achieved using a modified version of the Newton-Raphson algorithm. From i_0 and $-\nabla g(i_0)$ the surface gradient at i_0, we can compute a linear approximation of the surface. This approximation defines the tangent plane $-\nabla g(i_0) \times P = \|i_0\|$ at i_0 which parameters are used by our contact algorithm. Since this tangent plane is only a valid approximation of the surface around i_0, the correction $\boldsymbol{x^c}$ might not be on the actual surface (see section 2.2). Therefore we need to update the tangent plane based on the corrected position $\boldsymbol{x} = \boldsymbol{x^f} + \boldsymbol{x^c}$. This is done by estimating the gradient ∇g at $\boldsymbol{x}$, which provides the direction to reach the surface with a minimal distance. This gives us a new point $i_k = x + \nu \nabla g$, where ν is a scalar value. This new point and its normal are used as an updated linear approximation of the surface.

From a mathematical point of view, such updates of the contact point and its normal is close to the secant method algorithm (because the gradient is evaluated using finite differences) to find the root of a function. Indeed, in convex cases, the distance between x^c and the surface decreases through the successive iterations and finally lead to a point of the surface. Such method is proved to converge in convex cases and we use a dedicated strategy to solve the concave cases (see Section).

3 Results

We have applied our method to a complex simulation: the navigation of a catheter and guidewire inside a reconstructed vascular network to perform a

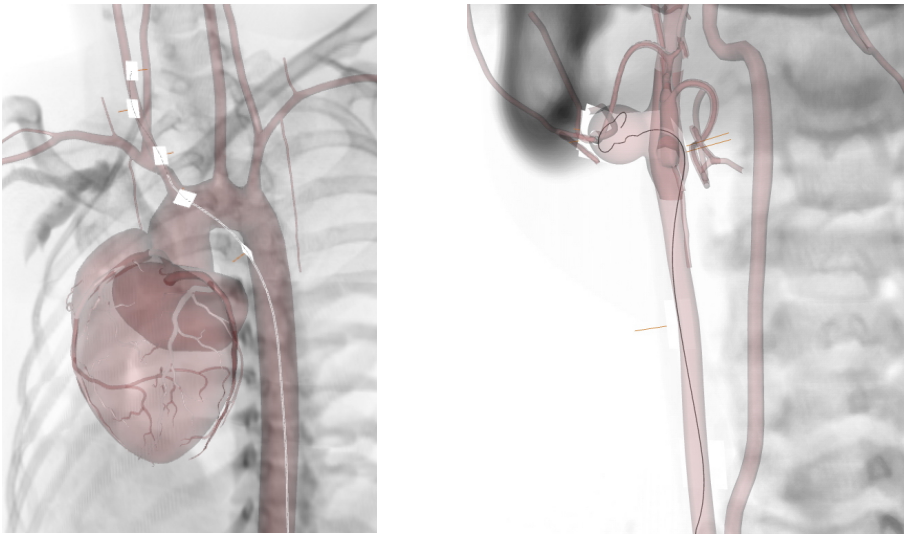

Fig. 1. Simulation of catheter and guidewire navigation from the aorta to the common carotid artery (left). Simulation of coil deployment inside an aneurysm (right). Contact locations and the corresponding contact planes (in white) are shown.

virtual angiography and the deployment of a coil inside an aneurysm to perform an embolization. The catheter, guidewire and coil models consist of a series of non-linear deformable beam elements. Each device is composed from 100 to 200 beam elements while the vascular network is constituted of more than $4,000$ vessels and undergoes periodic deformations due to both cardiac and respiratory motions. When entering the cerebrovascular system, where the diameter of the vessels is very small, the catheter or guidewire are constantly colliding and sliding along the vessels wall (see Figure 1). Similarly, when deployed within an aneurysm, the coil becomes highly constrained, and proper contact modeling becomes of prime importance to guarantee a correct behavior during the simulation (see Figure 1).

We have performed a series of simulations on a Dual Core processor machine with 2 GB of memory and obtained real-time computation rates (25 Hz). These timings include the computation *and* inversion of the system stiffness matrix K at each time step, as well as collision detection and collision response. Since the contacts are solved in the contact space, the size of the system is the number c of contact (defining n as the number of DOFs, $c \leq n$ and usually $c \ll n$). It is also important to mention that, in order to enforce the convergence and stability of the contact algorithm, we use a subdivision strategy where each time step is subdivided into a variable number of sub-steps. The initial time step is subdivided if not all contacts have been solved after N iterations of the main loop of the algorithm (see section 2.2). This subdivision strategy allows us to solve complex contact configurations and to handle concave cases has a succession of convex cases.

4 Conclusion

In this paper we have proposed an efficient method for solving complex contacts between various types of physics-based objects, in particular deformable structures. The proposed algorithm is accurate since the exact forces required to solve the contacts are computed and coupling between contacts is taken into account. Computational efficiency is achieved by solving contacts in the contact space to reduce the size of the system of equations. The approach uses an iterative scheme to solve *all* contacts at each time step and a subdivision strategy insures the robustness of the algorithm even in the case of complex contact configurations.

References

1. Cotin, S., Delingette, H., Ayache, N.: Real-time elastic deformations of soft tissues for surgery simulation. IEEE Transactions on Visualization and Computer Graphics 5(1), 62–73 (1999)
2. Picinbono, G., Delingette, H., Ayache, N.: Real-time large displacement elasticity for surgery simulation: Non-linear tensor-mass model. In: Delp, S.L., DiGoia, A.M., Jaramaz, B. (eds.) MICCAI 2000. LNCS, vol. 1935, pp. 643–652. Springer, Heidelberg (2000)
3. Mosegaard, J., Herborg, P., Sørensen, T.: A gpu accelerated spring-mass system for surgical simulation. In: MMVR. Proceedings of the 13th Medicine Meets Virtual Reality conference, pp. 342–348 (2005)
4. Miller, K., amd Dane Lance, G.J., Wittek, A.: Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. Communications in Numerical Methods in Engineering 23(2), 121–134 (2007)
5. Chou, W., Wang, T.: Human-computer interactive simulation for the training of minimally invasive neurosurgery. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1110–1115. IEEE Computer Society Press, Los Alamitos (2003)
6. Kry, P.G., Pai, D.K.: Continuous contact simulation for smooth surfaces. ACM Transactions on Graphics 22(1), 106–129 (2003)
7. Garcia-Perez, V., Munoz-Moreno, E., de Luis-Garcia, R., Alberola-Lopez, C.: A 3d collision handling algorithm for surgery simulation based on feedback fuzzy logic. In: International Conference on Information Technology in Biomedicine (2006)
8. Cotin, S., Duriez, C., Lenoir, J., Neumann, P., Dawson, S.: New approaches to catheter navigation for interventional radiology simulation. In: Duncan, J.S., Gerig, G. (eds.) MICCAI 2005. LNCS, vol. 3750, pp. 534–542. Springer, Heidelberg (2005)
9. Renouf, M., Acary, V.: Comparison and coupling of algorithms for collisions, contact and friction in rigid multi-body simulations. In: Proceedings of ECCM - Solids, Structures and Coupled Problems in Engineering, Lisbon, Portugal (2006)
10. Le Garrec, J., Andriot, C., Merlhiot, X., Bidaud, P.: Virtual grasping of deformable objects with exact contact friction in real time. In: Int. Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision, pp. 87–92 (2006)
11. Duriez, C., Andriot, C., Kheddar, A.: Signorini's contact model for deformable objects in haptic simulations. In: International Conference on Intelligent Robots and Systems, IROS, IEEE/RSJ, pp. 3232–3237 (2004)
12. Lenoir, J., Fonteneau, S.: Mixing deformable and rigid-body mechanics simulation. In: Computer Graphics International, Hersonissos, Crete - Greece, pp. 327–334 (2004)