A Framework for Modelling Thin Objects: from Meshing to Shell Finite Element Method

No Author Given

No Institute Given

Abstract.

1 Introduction

Human body is composed of various deformable anatomical structures. Realistic modelling of organ deformations is a challenging research field that opens the door to new clinical applications including: medical training and rehearsal systems, patient-specific planning of surgical procedure and per-operative guidance based on simulation. In all these cases the clinician needs fast update of the deformation model to obtain a real-time display of the computed deformations. To realise this kind of interactive systems, one faces the trade-off problem between accuracy of the model and the need of fast computations.

Another key challenge of real-time human soft-tissue modelling is the variousness of the mechanical behaviours. It seems unrealistic to target a unique model for all tissues even if there are obvious advantages to make maximum use of optimised generic models. Yet most of previous work focus on volume models that are able to capture the behavior of solid organs like liver [] or brain tissues [] **CD: TODO: biblio**. In contrast, this paper seeks to propose a solution for simulating, in real-time, deformation of thin anatomical structures, whose volume is negligible compared to their surface area. Examples of such anatomical structures include hollow structures, like the wall of blood vessels or the colon but also the envelope of solid organs, like the capsule of Glisson.

In the following of this paper, we present a framework for realistic and real-time simulation of thin organs. Our method, presented in section 2 is based on shell theory and computes the internal forces using Finite Element Modelling (FEM) on curved triangles. The design of the elements relies on a polynomial interpolation that is also used to capture the interactions with other models, including solid models. We also present an automatic process to obtain meshes from image based reconstruction in an attempt to optimise the computation using fewer elements (see section 3). The benefits of our approach (meshing of a curved surface, fast computation and possible interactions with solid models) are illustrated using various examples presented in section 4.

2 A physical model for thin structures

2.1 Related work

Rather than resorting to shell theory which involves the most complex formulations in continuum mechanics, previous works have often relied on discrete formulations. Early approaches only considered in-plane deformation, and often relied on mass-spring models. More recent works have considered adding bending through angular springs. For instance Wang et al. [1] successfully used a network of linear and angular springs to describe bending and twisting of catheters and guidewires in an interventional radiology simulator. **CD: C'est pas des shells si?** Yet, such models are limited in their ability to describe certain behaviour, as they do not rely on continuum mechanics. Another limitation of such models is the difficulty to derive spring stiffness (in particular for angular springs) from elastic properties (Young's modulus and Poisson's ratio). For these reasons, other approaches have been proposed.

Continuum mechanics provides many formulations able to accurately describe stresses occurring within thin objects. Most of them fall into one of the following two categories: plate theory or shell theory. Those theories have been a subject of interest in the mechanical community for decades. The difference between these two kinds of structures is very well explained by Liu et al. [2] and can be summarized by the fact that plate bending elements can only carry transversal loads while shells can undergo more complex deformations. Considering the wide range of shapes and solicitations that can be applied to hollow organs, a thin plate model would not correctly capture their deformation and therefore a shell formulation was retained for this work.

2.2 Co-rotational triangular shell model

A complete description and validation of our co-rotational triangular shell finite element model is available in one of our previous publication [3]. Therefore we will only remind briefly the key points. We improved and extended a plate model first introduced by Przemieniecki [4] to a co-rotational formulation. Corotational approaches offer a good trade-off between computational efficiency and accuracy by allowing small deformations but large displacements. Once combined with an in-plane membrane formulation we obtain an accurate, yet computationally efficient, shell finite element method featuring both membrane and bending energies.

Triangular elastic membrane The element stiffness matrix $\mathbf{K}_e$ can be computed as follows:

$$\mathbf{K}_e = \int_v \mathbf{J}\boldsymbol{\chi}\mathbf{J}^T dV \tag{1}$$

where $\mathbf{J}$ is the strain-displacement matrix and $\boldsymbol{\chi}$ embodies the material's behaviour. The stiffness matrix in the global frame is eventually obtained using the rotation matrix of the element: $\mathbf{K} = \mathbf{R}^T \mathbf{K}_e \mathbf{R}$ where $\mathbf{R}$ describes the rotation of the (triangular) element with respect to its initial configuration.

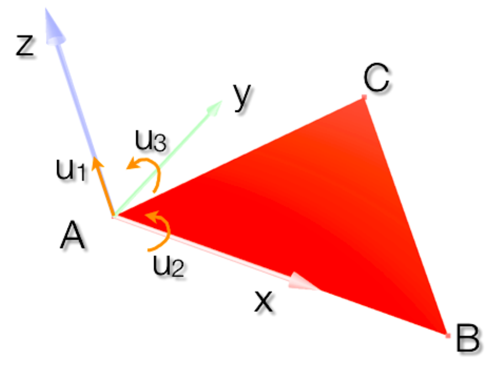

Fig. 1. The different degrees of freedom u of a triangular thin plate in bending are illustrated above.

Triangular plate bending To calculate the stiffness matrix for the transverse deflections and rotations shown on figure 1, we start with the assumed deflection u_z of the form

$$u_z = c_1 + c_2 x + c_3 y + c_4 x^2 + c_5 xy + c_6 y^2 + c_7 x^3 + c_8 xy^2 + c_9 y^3 \tag{2}$$

where $c_1, \ldots, c_9$ are constants. Using a third-degree polynomial expression allows us to reach a greater precision for both the computation of the bending energy and the interpolation within the surface of the shell. Let us define the vector $\mathbf{u} = \{u_1 u_2 \ldots u_9\}$ of the displacements and slopes at the three corners of the triangular plate using the following notations:

$$u_1 = (u_z)_{x_1,y_1} \qquad u_2 = \left(\frac{\partial u_z}{\partial y}\right)_{x_1,y_1} \qquad u_3 = -\left(\frac{\partial u_z}{\partial x}\right)_{x_1,y_1} \tag{3}$$

and so on for the two other vertices. We then can derive a matrix $\mathbf{C}$ such as $\mathbf{u} = \mathbf{Cc}$ where $\mathbf{c} = \{c_1 c_2 \ldots c_9\}$.

We can calculate the strains from the flat-plate theory using:

$$e_{xx} = -z\frac{\partial^2 u_z}{\partial x^2} \qquad e_{yy} = -z\frac{\partial^2 u_z}{\partial y^2} \qquad e_{xy} = -2z\frac{\partial^2 u_z}{\partial x \partial y} \tag{4}$$

Symbolically this may be expressed as $\mathbf{e} = \mathbf{Dc}$ where $\mathbf{D}$ derives from equations 2 and 4. Noting that $\mathbf{c} = \mathbf{C}^{-1}\mathbf{u}$, we have:

$$\mathbf{e} = \mathbf{DC}^{-1}\mathbf{u} = \mathbf{bu} \tag{5}$$

where the strain-displacement matrix $\mathbf{b} = \mathbf{DC}^{-1}$. The stiffness matrix $\mathbf{K}_e$ for an element is then classically obtained from

$$\mathbf{K}_e = \int_v \mathbf{b}^T \boldsymbol{\chi} \mathbf{b} dV \tag{6}$$

where $\boldsymbol{\chi}$ is the material matrix. The stiffness matrix in the global frame is eventually obtained using the rotation matrix of the element as in the membrane case.

Mechanical interactions with the curved surface of shells But the practical interest of modelling complex behaviours such as bending and twisting would remain fairly low for medical simulation if contacts were not handled properly. In our case the difficulty comes from different sources. First the collision detection must be carried out with the curved surface of shell elements as opposed to the classic detection on plane triangles. Then forces applied to a given triangle need to be distributed between linear forces and torques onto its three vertices. As we will see, the same polynomial interpolation function 2 chosen to compute the bending energy in our FEM formulation is also used to capture the interactions between the curved surface and other objects.

In order to detect the collision with the bent surface, we have chosen the subdivision approach. Therefore a finer mesh is created by adding vertices on the surface of each element. We first sample the flat surface of each element by recursively dividing each triangle into four smaller ones and the deflection of each new vertex is computed using (2) and the displacements and slopes at the three vertices of the triangular element. This process of subdivision allows us to render each shell as a curved triangle (Fig. 2) and detect any collision with the curved surface of the shell using any of the classic collision detection algorithms working on flat triangles.

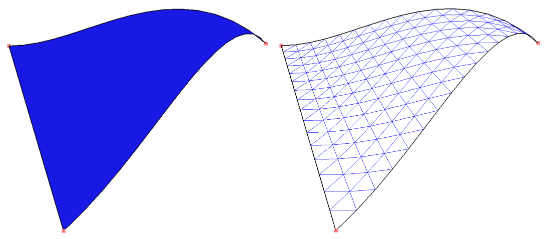

Fig. 2. The triangle formed by the three vertices of the shell has been recursively subdivided 3 times and the deflection of each new vertex was computed according to the same deflection function used in our shell finite element formulation. Sampling the actual surface of the shell allows rendering and more accurate collision detection.

Once a collision has been detected, it must be processed by distributing the linear force received on the bent surface between the three vertices of the triangle.

First the linear part of the force is simply transmitted on each node using the barycentric coordinates of the contact point's projection onto the triangle.

The main difficulty is to convert the normal component of the force applied to the bent surface into a torque at each of the three nodes. Our approach is the following: during force computation, we use the angle measured at each node to compute the local deflection of each subvertex within the triangle. Differentiating the formulation twice yields a relation between the torque applied at each node and the generated force in bending. We therefore need to invert the latter formulation to convert a bending force into torques at each vertex. We start by retrieving the normal component of the applied force vector F_{z}. We project the application point of the force into the triangle's plane and compute its local coordinates (x, y). We create the polynom $P = F_{\mathrm{z}}(1 \;\; x \;\; y \;\; x^2 \;\; xy \;\; y^2 \;\; x^3 \;\; xy^2 \;\; y^3)^T$. The moments at each vertex are then obtained with $\Omega = (\mathbf{C}^{-1})^T P$. Thus we are able to transmit any force coming from elaborate interactions with the curved surface of shells to the mechanical vertices used in our FEM formulation.

3 Meshing a curved surface

3.1 Related work

Another key component of finite element methods is the mesh of the simulated object being used. While a sufficient number of elements is needed to accurately describe the geometry, too many of them will yield a computationally expensive simulation and solving the system in real-time may no longer be possible. Literature about volumic mesh generation algorithms is fairly dense. However, there are only a few that are concerned about the generation of meshes over curved surfaces. One of the most widely used techniques for the creation of surface meshes is the plane to surface transformation method [5], mesh is first generated on a two-dimensional domain and then mapped onto the surface. If this method gives reasonably good meshes on smooth surfaces, the results are usually quite poor with more complex curved surfaces. The finite elements may be generated directly on the curved surfaces based on the advancing front technique [6,7]. The main issue with this approach is that an analytical description of the geometry is needed, which is not the case in medical simulation. Another method consists of using an a priori error estimator to build an adaptive mesh generation [8]. However, the tolerance of this indicator should be chosen depending on the desired accuracy of the finite element solution, and therefore requires some knowledge about the problem in order to choose an effective tolerance. In [9] the authors start from an existing triangular mesh created with a CAD software and refine and smooth the mesh based on element quality and surface curvature. While all those methods allow their authors to get satisfactory meshes over curved surfaces according to their needs, they all make use of flat elements to mesh geometries and do not generate actual shells.

3.2 A mechanically-driven method to mesh curved surfaces with shells

Indeed, one main difference between shell and plate elements is that the rest shape of the former can describe a curved surface. We propose to take this crucial feature into account to optimise the meshing process. While many flat triangles are often required to describe highly curved surfaces, fewer triangular shell elements are needed to describe the given geometry with the same precision since they can be naturally curved. In the following we assume that we have a high resolution triangular mesh obtained from a binary segmented image of the organ we want to simulate (via a Marching Cube algorithm for instance). Our goal is to create a mesh featuring the optimal number of shell elements while staying as close as possible to our targeted geometry.

Therefore we need to insure that the distance between the surface of our shell-based mesh and the targeted high resolution mesh will be minimal. An efficient technique for measuring the error between two surfaces is the Hausdorff distance [10,11]. As a reminder the Hausdorff distance between two meshes is the maximum between the two so-called one-sided Hausdorff distances:

$$d_{\mathrm{H}}(X,Y) = \max\left\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\right\} \quad . \tag{7}$$

where $d()$ is the Euclidian distance between two points. The same technique of subdivision used for rendering allows us to sample the actual surface described by the shells to compute the Hausdorff distance with the targeted high resolution mesh.

The first step in the process of generating a shell-based mesh is an aggressive simplification of the targeted high resolution mesh by using the quadric edge collapse decimation technique implemented in Meshlab [12]. The algorithm tries as much as possible to preserve mesh boundaries and generate high quality triangular elements. We then apply a heuristic method derived from the work of Saupin et. al [13] with tetrahedral meshes based on very simple geometrical rules. For each node of the coarse mesh, we find the three closest triangles on the high resolution mesh and we move the node to the barycenter of the three centres of mass of those triangles. This technique locally smooths the surface of the mesh while converging towards the desired high resolution mesh. At each iteration of this algorithm we measure the error with the target using the Hausdorff distance and the process is stopped when the required precision has been reached. A simple example is shown Fig. 3 and 4 to illustrate the method.

4 Results

4.1 Meshing of anatomical structures

This approach has been applied to mesh more complex anatomical structures with curved shell elements. In each case the error is expressed as a percentage of the diagonal of the object's bounding box.

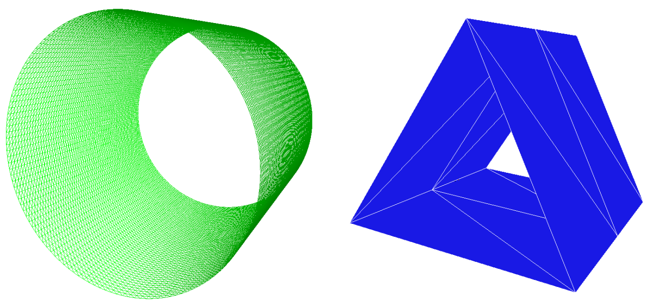

Fig. 3. Illustration of our method on a simple example. The target is the high resolution cylinder mesh (left) and the start point is a very coarse mesh approximating the shape of a cylinder (right).

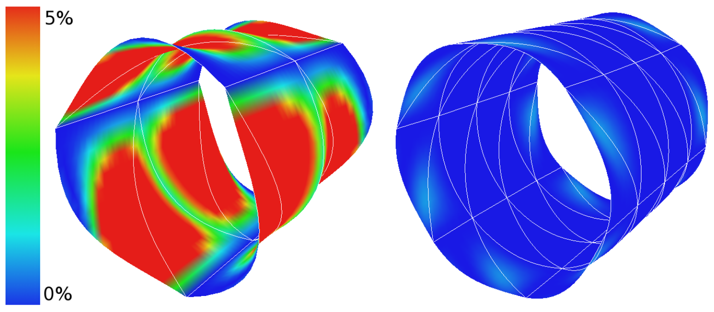

Fig. 4. (Left) One-sided Hausdorff distance colour map of the coarse mesh rendered with shells. (Right) One-sided Hausdorff distance colour map after only one iteration of our algorithm. We can notice that differences with the target mesh have been subtantially reduced.

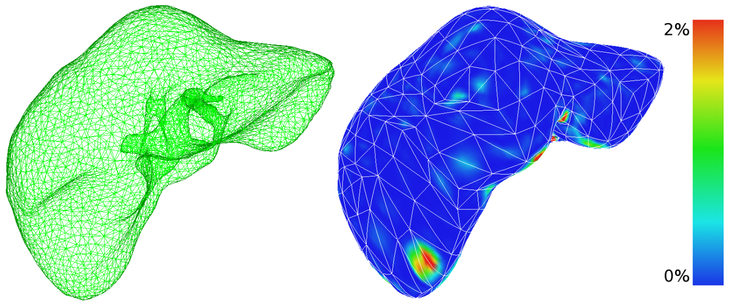

Fig. 5. Left: the targeteted high resolution liver mesh (8,000 triangles). Right: the one-sided Hausdorff distance error map after applying only one iteration of our algorithm to the coarse mesh (1,200 shells)

.

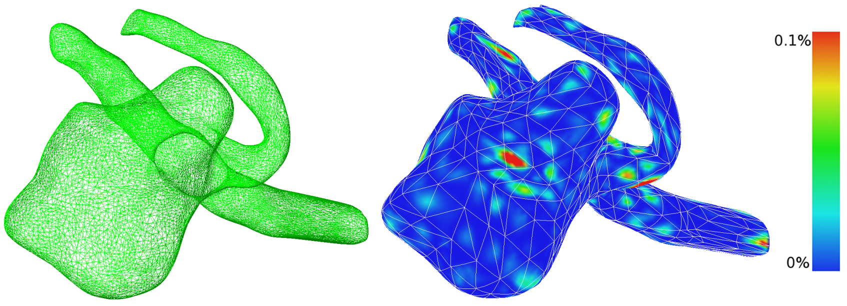

Fig. 6. Left: the targeteted high resolution anevrism mesh (28,368 triangles). Right: the one-sided Hausdorff distance error map after applying only one iteration of our algorithm to the coarse mesh (1,160 shells)

.

4.2 Deformations

Computation times Anevrism: 7.5 FPS with a timestep of 0.02s so we would need to be about 7 times faster to be real-time (easily doable on GPU)

Coupling between tetrahedra and shells for advanced modelling Structures in human body can be either solid (brain, liver, prostrate etc.) or hollow (blood vessels, colon, stomach etc.). However knowing how to model the two kind of structures is not sufficient to reach a high degree of accuracy in medical simulation. Indeed real life situations are more complex. As an example, the external surface of the liver is covered by a single layer of cells called Glisson's capsule. Its interaction with the liver plays an important role into the overall structure's mechanical behaviour. Therefore considering the interaction between solid and hollow objects is as crucial as modelling the two structures separately.

An example of medical procedure to illustrate this point even further is angioplasty. Angioplasty is the technique of mechanically widening a narrowed or obstructed blood vessel, typically as a result of atherosclerosis. An empty and collapsed balloon on a guide wire, known as a balloon catheter, is passed into the narrowed locations and then inflated to a fixed size using water pressures. The balloon crushes the fatty deposits, so opening up the blood vessel to improved flow. As a proof of concept we tried to simulate an angioplasty (Fig. 7). The blood vessel is modelled using the shell FEM formulation described in this paper and the fatty deposits are simulated with a tetrahedral FEM method and are fixed to the interior wall of the blood vessel. When the balloon inflates it crushes the deposits and the deposits then apply a pressure onto the curved surfaces of shells modelling the interior wall. The forces are then distributed onto the mechanical nodes of the blood vessel mesh as detailled in section 2.2, which widens the blood vessel as expected.

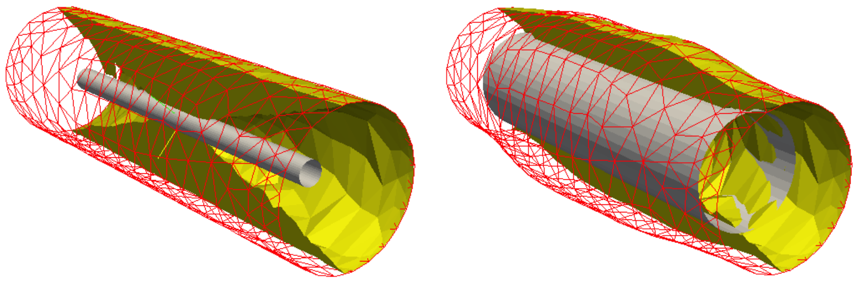

Fig. 7. Simulation of an angioplasty procedure. (Left) The collapsed balloon was inserted into the blood vessel simulated with our shell FEM formulation. The fatty deposits (in yellow) are modelled with a tetrahedral FEM. (Right) Upon inflation the balloon is crushing the fatty deposits, which is applying a pressure onto the interior wall and widening the blood vessel.

5 Conclusion

Acknowledgments

References

1. Wang, F., Duratti, L., Samur, E., Spaelter, U., Bleuler, H.: A Computer-Based Real-Time Simulation of Interventional Radiology. In: The 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE-EMBS). (2007) 1742–1745
2. Liu, G., Quek, S.: Finite Element Method: A Practical Course. Butterworth (2003)
3.
4. Przemieniecki, J.: Theory of matrix structural analysis. McGraw-Hill (1985)
5. Zienkiewicz, O.C., Phillips, D.V.: An automatic mesh generation scheme for plane and curved surfaces by isoparametric co-ordinates. International Journal for Numerical Methods in Engineering **3**(4) (1971) 519–528
6. Lo, S.: A new mesh generation scheme for arbitrary planar domains. International Journal for Numerical Methods in Engineering **21**(8) (1985) 1403–1426
7. Lau, T.S., Lo, S.H.: Finite element mesh generation over analytical curved surfaces. Computers & Structures **59**(2) (1996) 301–309
8. Baumann, M., Schweizerhof, K.: Adaptive mesh generation on arbitrarily curved shell structures. Computers & Structures **64**(1–4) (1997) 209–220
9. Béchet, E., Cuilliere, J.C., Trochu, F.: Generation of a finite element MESH from stereolithography (STL) files. Computer-Aided Design **34**(1) (2002) 1–17
10. Klein, R., Liebich, G., Straßer, W.: Mesh reduction with error control. In: Visualization 96. ACM. (1996) 311–318
11. Cignoni, P., Rocchini, C., Scopigno, R.: Metro: Measuring error on simplified surfaces. Computer Graphics Forum **17**(2) (1998) 167–174
12. CNR, V.C.L.I.: Meshlab http://meshlab.sourceforge.net/.
13. Saupin, G., Duriez, C., Grisoni., L.: Embedded multigrid approach for real-time volumetric deformation. In: nternational Symposium on Visual Computing (ISVC 2007). (2007) 149–159