Saturday 26th September, 2015

Towards surgeon-authored VR training: the scene-development cycle

Saleh DINDAR, [a] Thien NGUYEN [a] and Jörg PETERS [a,1]
[a] *University of Florida*

Abstract. Enabling surgeon-educators to themselves create virtual reality (VR) training units promises greater variety, specialization and relevance of the units. This paper describes a software bridge that semi-automates the scene-generation cycle, a key bottleneck in authoring, modeling and developing VR units. Augmenting an open-source modeling environment with physical behavior attachment and collision specifications yields single-click testing of the full force-feedback enabled anatomical scene.

Keywords. surgeon-author, interactive, simulation, force feedback, haptic, collision model, geometry model, mechanical model

1. Introduction

Virtual reality simulators allow trainees to practice decision-making and execution prior to entering the OR [3]. Due to their interface, laparoscopic techniques lend themselves to such simulation using force-feedback devices [5]. However, creating VR training units is neither cheap nor fast since the necessary back-and-forth between engineers, computer scientists and medical experts can take months or years. Even then, the result is not easily adjustable to reflect uncommon or specialized scenarios or to allow a surgeon-educator to implement a variation in technique.

The goal of the Toolkit for Illustrations of Procedures in Surgery (TIPS) [8] is to provide surgeon-educators with an environment so they themselves can create and customize training units – at a cost and effort comparable to writing a journal article in a word-processing environment. One component of such units focuses on instruction and base knowledge transfer. This paper focuses on the other component: hands-on, interactive force-feedback simulation of the critical steps of the surgical procedure.

Central to a robust and realistic 3D surgical environment are interactions between the geometry models, collision models and physical models. These interactions are often under-appreciated since they do not neatly fit into the publishable categories of numerical techniques, geometry representation or collision testing and response. Yet, incompletely defined models, collisions at initialization, models too complex for interactive execution, incorrect linking of visual mechanical and collision model, etc. can each cause simulations to fail even when the constituent components are state-of-the-art and individually stable.

[1] Corresponding Author: Dept CISE, U Florida, jorg@cise.ufl.edu.

This paper reports on a bridging software that provides the ‘glue’ to efficiently turn a database of mechanical, collision and visual models into a working TIPS simulation. The bridging software consists of:

- augmentation of the opensource modeling environment *Blender* [2] to support specification of behavior, attachment and collision properties;
- a plug-in that automatically transfers the *Blender*-set specifications into input consumed by the opensource simulation environment *SOFA* [1] for immediate testing of the scene.

Figure 1. Computer simulated TIPS environment: *left* surgery simulation; *right* instructions, questionnaires, and video. Two low-cost ($2K/piece) six degrees-of-freedom haptic devices provide physical feedback.

2. Methods and Materials

2.1. Software Packages

The Toolkit for Illustrations of Procedures in Surgery (TIPS) is a low-cost computer-based environment enabling expert surgeons to create and publish laparoscopic surgery training units. The units include interactive force-feedback practice of critical steps of the surgical procedure in a 3D virtual environment. To author a new unit the surgeon-educator lists tasks and safety cautions in a fixed format specifying 3D anatomy and actions. From a database of models, this specification defines a scene initialization that the author can fine-tune. Figure 1 shows an adrenalectomy unit created as a proof-of-concept. Over the past two years, 23 junior surgeons (residents and fellows) and five experienced surgeons tested this unit (and judged it both effective and superior to physical props, one-on-one teaching, medical atlases, or video recordings). The team that developed this prototype disbanded in 2012 when, despite being ranked in the top 5 percentile, US budget politics caused a funding gap. Supported by NIBIB, a new team was formed in 2014, combining expertise in software development, numerical computing, computer graphics and laparoscopic surgery. TIPS was restructured to rely on European open-source software in the form of *Blender* and *SOFA*[1].

Blender [2] is a professional geometric modeling, rendering and animation software distributed by the Blender Foundation under the GNU GPL license. Several animated movie shorts have been created using *Blender* as the main modeling and rendering environment. The new bridging software in TIPS uses *Blender* exclusively for geometry modification, starting from basic anatomy created offline; and it extends the graphical user interface to add annotations that define the simulation behavior of the scene.

SOFA [4] is a collection of numerical, geometric and visual routines for developing simulation codes. In particular, it offers support for soft-tissue manipulation. *SOFA* is distributed by INRIA under a GNU LGPL license. Porting, via plug-ins, only the most popular custom-built solutions, such as fatty tissue, from the 2012 version, the new TIPS simulation now relies almost entirely on *SOFA* routines. Our modular approach differs from a commercial effort [7] that aims at pulling into *Blender* the *SOFA* developer level including numerical routine selection and parameter fine-tuning. Since this deep integration is complex and requires careful and constant maintenance, we think our opensource, federated approach is more useful for surgeon-authoring.

2.2. Levels of Scene Generation

Layers of abstraction have to cushion the educator's specification of the surgical steps from the details of the physical simulation and visual presentation. A highly responsive scenario re-generation cycle is needed at the three levels of authoring and author support:

- at the *developer level*, where numerical simulation routines are selected or adjusted, and parameter ranges are determined;
- at the *artist level*, where models are combined into scenes of compatible geometry, collision and physics models and default parameters are initialized; and
- at the *surgeon-author level*, where variants of the scenario are explored based on the artists' packaging and individual scenarios are customized by fine tuning geometry and behavioral parameters.

2.3. Shortcomings of the Pre-existing Work-flow

To make concrete the challenge of generating and re-generating geometry suitable for physical simulation, we describe and discuss the work-flow natively supported by *SOFA*. For illustration, we choose simulation of *Nissen Fundoplication* (NF). In NF, the upper part of the stomach is wrapped around the lower end of the food pipe (esophagus) and then stitched in place to prevent reflux. NF includes mobilisation of organs by cauterizing connective layers, tearing of tissue for access, interactive soft-tissue deformation and sliding contacts when the top (fundus) of the stomach is pulled through an opening between esophagus and diaphragm. Geometric models of the stomach, the spleen, the esophagus, the diaphragm and the enclosing peritoneum exist in the TIPS database but need to be customized before serving as visual, behavioral and collision models. In particular, the model of the stomach needs to support stretching, stitching as well as collision and sliding along the esophagus. When manipulated as a surface, collapse and self-penetration make clear that a volumetric, collision-enabled stomach model is required: each time the surface model is changed, a 3D meshing software needs to be invoked to add a model to the database. Then, using *SOFA*'s *Modeler* interface (Figure 2), the pre-authored anatomy is loaded. Often information, such as color and global position and orientation, needs to be (re-)entered due to the transfer from the modeling to the *SOFA* environment. To simulate the connective tissue between stomach and spleen, one has to find, by trial and error, a list of index pairs that connect vertices of the stomach to the spleen triangulation. A similar search is necessary to attach the peritoneum to organs. The *Modeler* interface then facilitates associating behavioral routines with each of the organs (stomach, spleen and diaphragm) and to type in the behavior parameters such as stiffness, friction, solver iterations, contact distance, etc. (see Figure 2).

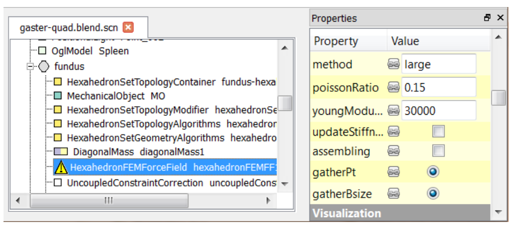

Figure 2. Existing non-geometric *Modeler* interface of *SOFA*.

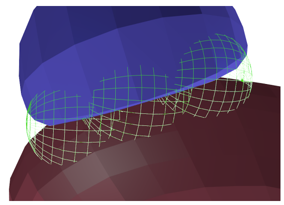

Figure 3. 'Glue spheres' to connect anatomy in the new *Blender* interface.

While very useful for development and testing of new *SOFA* routines and functionalities on simple test data, adjusting complex scene geometry with complex behaviors, such as NF, is time-consuming and error-prone within the *Modeler* interface. Moreover, the modeling artist who puts together the simulation scenario needs to be well-versed in the options and choices of *SOFA* in order for objects to interact properly and the simulation to remain stable. For, even though the *Modeler* lists all the options, the only documentation of the *SOFA* options is the C++ class construction in the *SOFA* code where the parameters are defined with brief comments. In some cases, one has to read *SOFA* source code to understand how parameters influence the simulation. The alternative, to manipulate *SOFA* XML scene files directly in a text editor, has the additional disadvantage that not even the parameters of all the XML tags are documented and that XML tags of a *SOFA* scene often depend on one another. For example, the LCP (linear complimentary problem) solver requires UncoupleConstraintSolver or PrecomputedConstraintSolver for every Mechanical Object. Clearly, such detective work should be left to the developer and not to the modeling artist or surgeon-author.

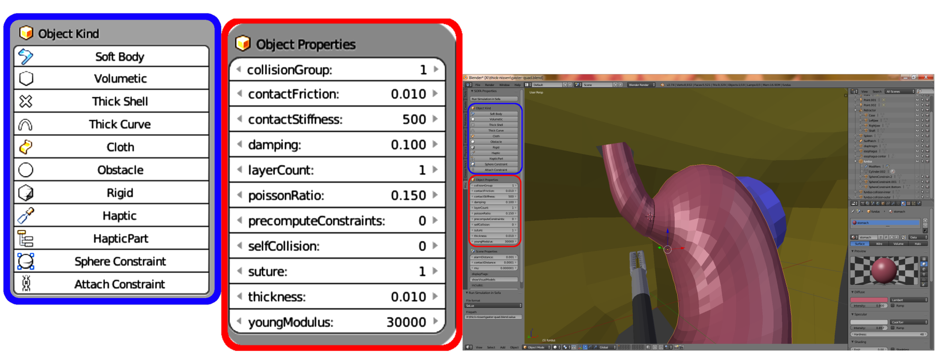

Figure 4. *left, enlarged* New behavior manipulation user interface for TIPS authoring within *Blender*. *right* A subset of *Blender*'s native geometry and visual manipulation interface. texture.

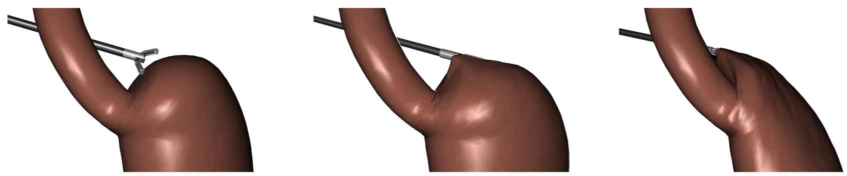

Figure 5. Instant *SOFA* simulation (3 snap shots): a *Blender*-defined Nissen Fundoplication scene.

3. Result: A Simplified Work-Flow.

To simplify and speed up the generation of scenes, and to enable modeling with limited knowledge of *SOFA*'s C++ routines, we created a *Blender* plug-in that outputs scenes as XML files and augmented the *Blender* interface to be able to specify behavioral templates (see Figure 4, left). For example, a single click turns the surface model of the stomach into a *thick shell* (that we determined is the best representation of the stomach for simulating NF; see Figure 5). Another menu entry attaches the label *obstacle* the diaphragm to initialize collision detection. To create volumetric fatty tissue that can be torn, *Blender*'s data structures were augmented to include tetrahedral soft objects generated by meshing tools such as *QTetraMesher* [6]. Placing a sphere-like *glue sphere* into the scene as in Figure 3 allows linking two anatomical structures, such as the stomach and the spleen: the vertices of one structure inside the glue sphere are linked to vertices of the other inside the glue sphere. The connections calculated by the plug-in are output as XML tags understood by *SOFA*.

To summarize, we implemented the following extensions of *Blender*:

- augmentation of the user interface to annotate geometric models (organs, vessels, etc.) with tissue behavior and simulation parameters (Figure 4) for numerical soft-tissue simulation code (co-rotational finite elements, intersection testing, etc.)
- data structures for importing/exporting/storing tetrahedral meshes.
- geometry processing to automatically link objects (see Figure 3).
- automatic extraction of extended *Blender* data and export to XML format consumable by *SOFA* for single-click simulation start.

Importantly, thanks to the extensibility of *Blender*, all information of a scene can be saved in a *Blender*-readable file. This allows saving all work; and it simplifies archiving and sharing of simulation scenes, one of the goals of the TIPS project.

At a higher level, our plug-in provides a set of tested templates for different behavioral models that successfully execute in *SOFA*. Numerical and collision detection routines for classes of anatomy are pre-selected by the developer when augmenting the Python-based bridging software. This leaves the artist free to focus on assembling partial scenes with default behavior. The surgeon-author may then still adjust position details, and specific behavior such as stiffness and stretching of soft tissue.

4. Discussion

Initializing and adjusting shape, partition and choice of elements (volumetric, surface, shells) and their behavior are critical and time-consuming tasks when creating simulations for a VR training unit. The TIPS bridging software is presently able to transfer basic anatomic geometry, convert during transfer surfaces to volumetric representations using *QTetraMesher*[6] or our own thick-shell construction, set tissue properties and leverage an intuitive *Blender* user interface to create constraints (including non-intersection and attachment) between anatomic entities. This has tremendously sped up, from hours to minutes, the scene-creation cycle for TIPS for developers, artists and authors. Developers can now create and immediately debug geometry-dependent physical behavior and save results in a database. Artists can package such combinations into default scenes. Leveraging a growing TIPS data base of anatomic models with associated ranges of simulation solvers and parameters, artists of widely differing ability have successfully set up laparoscopy scenes. Crucially, surgeon-authors can now fine tune pre-packaged scenes and repeatedly adjust them by themselves and without delay to give it the correct feel and emphasis. Since *SOFA* provides a remarkable range of routines and choices, we were able to add TIPS few *SOFA* extensions (of haptics, visuals, tearing/dissecting, and suturing) via plug-ins, i.e. without having to alter *SOFA* core code. This reduces version-dependency.

5. Conclusion

To date, junior surgeons are exposed to only fraction of the full spectrum of laparoscopic procedures and scenarios in the OR. Engaging surgeons as authors promises greater variety, specialization and relevance of laparoscopic VR training. Automatic scene-generation is crucial to enable author-support by artists and developers and to present surgeon-authors with a high-level interface and yet allow them to customize anatomic layout and fine tune physical behaviour. Our bridging software is an important step in this direction.

Acknowledgements The work was supported by NIH grant R01 EB018625 (Grace Peng). We thank Martin Sarov, Garima Singh and Erick Birkholz for testing the pipeline, George Sarosi for expert input on NF and Stéfane Cotin, Christian Duriez, Jeremie Dequidt, Bruno Carrez of INRIA's *SOFA* team for hosting the third author in summer 2015.

References

[1] J. Allard, S. Cotin, F. Faure, P.-J. Bensoussan, F. Poyer, C. Duriez, H. Delingette, and L. Grisoni. Sofa an open source framework for medical simulation. In *Medicine Meets Virtual Reality (MMVR'15)*, Long Beach, USA, February 2007.

[2] Blender Foundation. Blender. http://blender.org/, 2015.

[3] A.G. Gallagher, E.M. Ritter, H. Champion, G. Higgins, M.P. Fried, G. Moses, C.D. Smith, and R.M. Satava. Virtual reality simulation for the operating room: proficiency-based training as a paradigm shift in surgical skills training. *Annals of Surgery*, 241(2):364–372, 2005.

[4] INRIA. Simulation open framework architecture. http://www.sofa-framework.org, 2015.

[5] L. Palanivelu K.S. Gurusamy, R. Aggarwal and B.R. Davidson. Virtual reality training for surgical trainees in laparoscopic surgery. *The Cochrane Database of Systematic Reviews*, 1, 2009.

[6] Dennis Lübke. QTetraMesher. http://qtm.dennis2society.de, 2013.

[7] Digital Trainers. Blender SOFA. http://www.digital-trainers.com/blender-sofa-en/, 2015.

[8] Y-I Yeo, S. Dindar, G. Sarosi, and J. Peters. Enabling surgeons to create simulation-based teaching modules. In *Medicine Meets Virtual Reality (MMVR) Feb 8-12 2011, Long Beach,CA*, pages 1–6. Studies in Health Technology and Informatics (SHTI), IOS Press, Amsterdam, 2011.