In *Proceedings of Medicine Meets Virtual Reality XII (MMVR 2004)*, Newport Beach, CA, January 14-17, 2004, pp.46-48

GiPSi: An Open Source/Open Architecture Software Development Framework for Surgical Simulation

M. Cenk Çavuşoğlu (1), Tolga G. Göktekin (2),
Frank Tendick (3), and Shankar Sastry (2)
(1) Dept. of Electrical Eng. and Computer Sci., Case Western Reserve University
(2) Dept. of Electrical Eng. and Computer Sci., University of California, Berkeley
(3) Dept. of Surgery, University of California, San Francisco

Abstract. In this paper we propose an open source/open architecture framework for developing organ level surgical simulations. Our goal is to facilitate shared development of reusable models, to accommodate heterogeneous models of computation, and to provide a framework for interfacing multiple heterogeneous models. The framework provides an intuitive API for interfacing models with spatial relationships. It is specifically designed to be independent of the specifics of the modeling methods used and therefore facilitates seamless integration of heterogeneous models and processes. Furthermore, each model has separate geometries for visualization, simulation, and interfacing, allowing the modeler choose the most natural geometric representation for each case.

1 Introduction

The current state of the field of medical simulation is characterized by scattered research projects using a variety of models that are neither inter-operable nor independently verifiable models. An open source, open architecture software development model provides an attractive framework to address : *i)* interfacing of models from multiple research groups, *ii)* validation of quantitative biological simulations, *iii)* inter-connectibility of the software modules, and *iv)* open source or proprietary third party development of additional models, model data, and computation modules. The motivation behind this study is our prior experience in surgical training simulators and physically based modeling [4].

2 Methods and Tools

In this paper we propose GiPSi (General Interactive Physical Simulation Interface), an open source/open architecture framework for developing surgical simulations such as interactive surgical training and planning systems. GiPSi focuses on providing support for heterogeneous models of computation and defined APIs for interfacing these heterogeneous physical processes. In addition, I/O interfaces for visualization and haptics for real-time interactive applications have been provided. The implementation of the framework is done using C++ and it is platform independent.

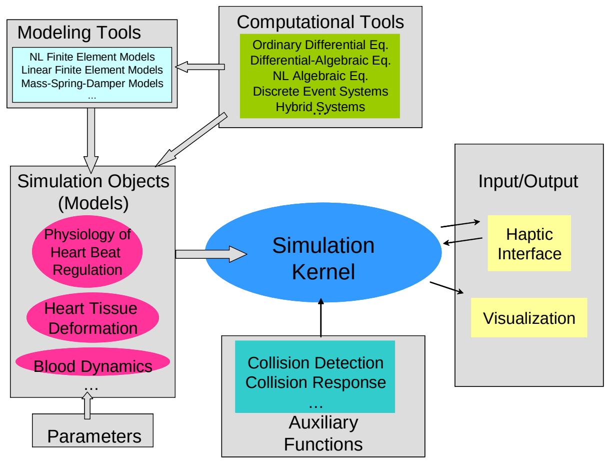

Figure 1: The system architecture of GiPSi

The overall system architecture of GiPSi is shown in Fig. 1. The models of physical processes are represented as Simulation Objects. Each simulation object can be derived from a specific computational model contained in Modeling Tools such as finite elements, lumped elements, etc. The Computational Tools, such as ODE solvers, provide a library of numerical methods for low level computation of the object's dynamics. The objects are created and maintained by the Simulation Kernel which arbitrates their communication to other objects and components of the system. One such component is the I/O subsystem which provides basic user input provided through the haptic interface tools and basic output through visualization tools. There are also Auxiliary Functions that provide application dependent support to the system such as collision detection and collision response tools.

Simulation and Interfacing APIs The majority of the models in organ level simulations involve solving multiple time varying PDEs that are defined over spatial domains and are coupled via boundary conditions. Our goal is to design a flexible API that facilitates the shared development and reuse of models based on these PDEs. Therefore the focus of our effort is to provide: *i)* a common geometric representation of the domain, *ii)* a library of tools for solving these PDEs, *iii)* a standard API for coupling them.

The first step in solving a continuous PDE is to discretize the spatial domain it is defined on. Therefore, every object must contain a proper geometry that describes its discretized domain, called the *Domain Geometry*. The definition of this geometry is flexible enough to accommodate the traditional mesh based methods as well as point based (mesh free) formulations. GiPSi defines a set of geometries that can be used as a domain including but not limited to polygonal surface and polyhedral volume meshes. In our current implementation we provide geometries for triangular and tetrahedral meshes. Second, a method for solving a PDE should be employed. Basic general purpose objects that implement these methods are provided as Modeling Tools, e.g. there is a general customizable FEM object that implements the basics of the finite element method. So far we implemented objects for FEM and MSD methods. GiPSi provides a library of numerical analysis tools in the Computational Tools that can be used to solve these discretized equations.

The simulation API needs to provide the standard means to interface multiple objects. In the models mentioned above, the basic coupling of two objects are defined via the boundary conditions between them. Therefore, we need to provide an API to facilitate the passing of boundary conditions between different models. First, we need a common definition of the boundary, i.e. each object needs to have a specific *Boundary Geometry*. In our current implementation, we chose triangular surfaces as our standard boundary geometry. Even though the type of the boundary geometry is fixed for every object, the values that can be set at the boundary and their semantics are up to the modeler and should be documented.

Use of boundary conditions is not the only interfacing scheme for objects. Coupling can occur through a commonly occupied volume rather than a shared boundary. A more general information passing is provided by a simple Get/Set scheme. The values that can be get and set by other objects and their semantics are again left to the modeler.

3 Results and Conclusion

We propose an open source/open architecture framework for developing organ level surgical simulations that facilitates shared development and reuse of models. This framework provides an intuitive API for interfacing models with spatial relationships. In addition, it is independent of the specifics of modeling methods and thus facilitates seamless integration of heterogeneous models and processes. Furthermore, each model has separate geometries for visualization, simulation and interfacing. This lets the modeler choose the most natural geometric representation for each. I/O interfaces for visualization and haptics for real-time interactive applications have also been provided.

GiPSi is designed to be general and independent of the specifics of the implemented modeling methods, unlike earlier dynamic modeling frameworks [3, 2]. This allows GiPSi to seamlessly integrate heterogeneous models and processes, and enforce time dependent spatial relationships among them, which is not possible with the earlier frameworks [1]. GiPSi has been tested on a specific test-bed application: the construction of a heart model for simulation of heart surgery.

4 Acknowledgements

This research was supported in part by NSF under grants CISE IIS-0222743, CDA-9726362 and BCS-9980122, EIA-0329811 and US Air Force Research Laboratory under grant F30602-01-2-0588. We also would like to thank Xunlei Wu for his valuable discussions and feedback.

References

[1] S. Cotin, D. W. Shaffer, D. A. Meglan, M. P. Ottensmeyer, P. S. Berry, and S. L. Dawson. CAML: A general framework for the development of medical simulations. In *Proceedings of SPIE Vol. 4037: Battlefield Biomedical Technologies II*, 2000.

[2] A. Joukhadar and C. Laugier. Dynamic simulation: Model, basic algorithms, and optimization. In J.-P. Laumond and M. Overmars, editors, *Algorithms For Robotic Motion and Manipulation*, pages 419–434. A.K. Peters Publisher, 1997.

[3] K. Montgomery, C. Bruyns, J. Brown, S. Sorkin, F. Mazzella, G. Thonier, A. Tellier, B. Lerman, and A. C. Menon. Spring: A general framework for collaborative, real-time surgical simulation. In J. Westwood et al., editor, *Medicine Meets Virtual Reality (MMVR 2002)*, Amsterdam, 2002. IOS Press.

[4] F. Tendick, M. Downes, T. Goktekin, M. C. Çavuşoğlu, D. Feygin, X. Wu, R. Eyal, M. Hegarty, and L. W. Way. A virtual environment testbed for training laparoscopic surgical skills. *Presence*, 9(3):236–255, June 2000.