Book Title
Book Editors
IOS Press, 2003
In Proceedings of Medicine Meets Virtual Reality XIV
(MMVR'06), Long Beach, CA, January 24 - 27, 2006.

Evaluation Methods of a Middleware for Networked Surgical Simulations

Qingbo Cai [1], Vincenzo Liberatore, M. Cenk Çavuşoğlu and Youngjin Yoo
Case Western Reserve University

Abstract. Distributed surgical virtual environments are desirable since they substantially extend the accessibility of computational resources by network communication. However, network conditions critically affects the quality of a networked surgical simulation in terms of bandwidth limit, delays, and packet losses, etc. A solution to this problem is to introduce a middleware between the simulation application and the network so that it can take actions to enhance the user-perceived simulation performance. To comprehensively assess the effectiveness of such a middleware, we propose several evaluation methods in this paper, i.e., semi-automatic evaluation, middleware overhead measurement, and usability test.

Keywords. Surgical simulation, Virtual environment, Network, Middleware, Usability test

1. Introduction

Distributed surgical virtual environments are desirable since they substantially extend the accessibility of computational resources by network communication. However, network conditions critically affects the quality (fidelity and realism) of a networked surgical simulation in terms of bandwidth, delays, packet losses, etc. A solution to this problem is to introduce an intermediary (middleware) between the surgical simulation application and the network so that it can take actions to remediate for the lack of network *QoS* (Quality of Service) [1].

Systematic evaluation methods are needed to comprehensively assess the effectiveness of such a middleware, i.e., to measure the performance enhancement the middleware brings and the overheads it incurs. In this paper, we propose three methods to achieve this: semi-automatic evaluation, overhead measurement, and usability test.

2. Evaluation methods

2.1. A semi-automatic evaluation framework

In order to evaluate a middleware's performance on different network configurations, we can execute a standard task suite remotely and measure the task performance with an

[1]Correspondence to: Vincenzo Liberatore or M. Cenk Çavuşoğlu, Department of Electrical Engineering and Computer Science, Case Western Reserve University, 10900 Euclid Avenue, Cleveland, OH 44106, U.S.A. Tel.: +1 216 368 4088; E-mail: vxl11@case.edu or cavusoglu@case.edu.

automatic procedure. Specifically, we can implement an automatic controller to execute each task. For example, we may build an automatic controller, which moves the scalpel using compliant control to perform the Fitts' task [2,3] and uses the Fitts' index of performance as performance metrics, to observe the controller performance under various network conditions.

A semi-automatic evaluation framework consists of a benchmark suite of representative surgical tasks and the infrastructure for network emulations of a middleware. The benchmark tasks should be i) simple so that they can be programmed with control techniques of general knowledge; and ii) representative of a human user's simple operations. One example is the standard Fitts' task. Alternative benchmark task can be general tasks performed in typical surgery, including tasks that require tracking of simple trajectories using surgical tools and maintaining constant contact force between the surgical tool and tissue, with appropriate performance metrics such as task completion time and error in trajectory tracking, and the amount of simulated damage that a controller inflicts under poor network conditions for constant force application tasks.

2.2. Overhead measurement

An ideal middleware should have minimal latency and throughput overhead, and moderate computational requirement. Unfortunately, this is not always the case. The overhead imposed by a middleware can be evaluated by comparing the packet delay and throughput of a middleware with point-to-point plain socket communication. To assess the computational requirements and running time of the individual modules within a middleware or the middleware as a whole, a middleware should include instrumentation software that can be selectively turned on or off at run-time.

2.3. Usability test

The ultimate goal of the middleware is to enhance the user-perceived performance of a networked surgical simulation. Consequently, user experiences should also be included in a comprehensive evaluation. From the middleware perspective of view, our concerns are the issues related to the network influences on user experiences, such as the responsiveness of haptic devices, the stability of haptic feedback, the smoothness of remote visualization, etc.

A usability test of virtual environments (VE) differs from traditional one in that it introduces factors such as presence, physiological effects, locating and manipulating objects, etc. Although intensive research has been done in VE and usability, few examples exist to apply usability in VE. An overview of usability evaluation in virtual environments is given in [4]. A popular and effective technique to collect qualitative data during usability tests is the "think-aloud" protocol [5], where a participant vocalizes his feelings and thoughts regarding the interfaces while performing a task. Comparative evaluation may also be conducted by asking subjects to operate the benchmark tasks on several programs representing different configurations of the middleware. Then, a statistical analysis is performed to validate the techniques applied in the middleware. There are several factors that can affect the result of a usability test, for example, the subjects' prior experience with VE and surgical tasks, the model of learning (individual or pair work), the learning curve, etc. Therefore, we should consider the influences of these factors

during the usability evaluation planning and analyzing so that the test can carry more meaningful results.

3. Conclusions and future work

In this paper, we described the methods to evaluate a middleware for networked surgical simulations. In our ongoing project, we will develop a middleware *GiPSiNet* [6] to extend *GiPSi* (General Interactive Physical Simulation Interface) [7,8], our open source/open architecture framework for developing surgical simulations, to a network environment. We will apply the evaluation methods presented to assess the effectiveness of GiPSiNet.

Acknowledgements

The authors acknowledge Technology Opportunities Program (TOP) of Department of Commerce, NASA NNC05CB20C, NSF CCR-039910, NSF IIS-0222743, NSF EIA-0329811, NSF CNS-0423253, and the Virtual Worlds Laboratory at Case Western Reserve University.

References

[1] J. Smed, T. Kaukoranta, H. Hakonen: Aspects of networking in multiplayer computer games, *Proceedings of the International Conference on Applications and Development of Computer Games in the 21st Century* 2001, 74–81.

[2] P. M. Fitts: The information capacity of the human motor system in controlling the amplitude of movement, *Journal of Experimental Psychology* **47**, 1954, 381–391.

[3] P. M. Fitts, J. R. Peterson: Information capacity of discrete motor responses, *Journal of Experimental Psychology* **67**, 1964, 103–113.

[4] D. A. Bowman, J. L. Gabbard, D. Hix: A Survey of Usability Evaluation in Virtual Environments: Classification and Comparison of Methods", *Presence: Teleoperators and Virtual Environments* **11**(4), 2002, 404–424.

[5] D. Hix, H. R. Hartson: *Developing user interfaces: Ensuring usability through product and process*, John Wisley and Sons, 1993.

[6] V. Liberatore, M. C. Cavusoglu, Q. Cai: GiPSiNet: An Open Source/Open Architecture Network Middleware for Surgical Simulations, *Medicine Meets Virtual Reality 14 (MMVR 2006)*, 2006.

[7] M. C. Cavusoglu, T. G. Goktekin, F. Tendick, S. S. Sastry: GiPSi: An Open Source/Open Architecture Software Development Framework for Surgical Simulation, *Medicine Meets Virtual Reality XII (MMVR 2004)*, 2004, 46–48.

[8] M. C. Cavusoglu, T. G. Goktekin, F. Tendick: GiPSi: A Framework for Open Source/Open Architecture Software Development for Organ Level Surgical Simulation, *IEEE Transactions on Information Technology in Biomedicine*, 2005 (In Press).