Touch-Hapsys: Deliverable 6.7
Part 4

Aurélien Pocheville and Abderrahmane Kheddar

September 24, 2004

1 Introduction

In order to be able prototype mutimodal integration schemes and models, the LSC developed a generic framework called I-Touch that serves also the rigid bodies prototyping demonstrator of the deliverable D7.5 and D7.6. This part will describe the components of the I-Touch.

2 The I-Touch framework - Prototype version

2.1 Design of the I-Touch framework

The I-Touch framework is designed to be modular from its core to its interfaces. Although it is still in the development process, it already allows plug-ins (static linking in program) of different behaviors models and different collision detection algorithms. The framework architecture is given in the Figure 1.

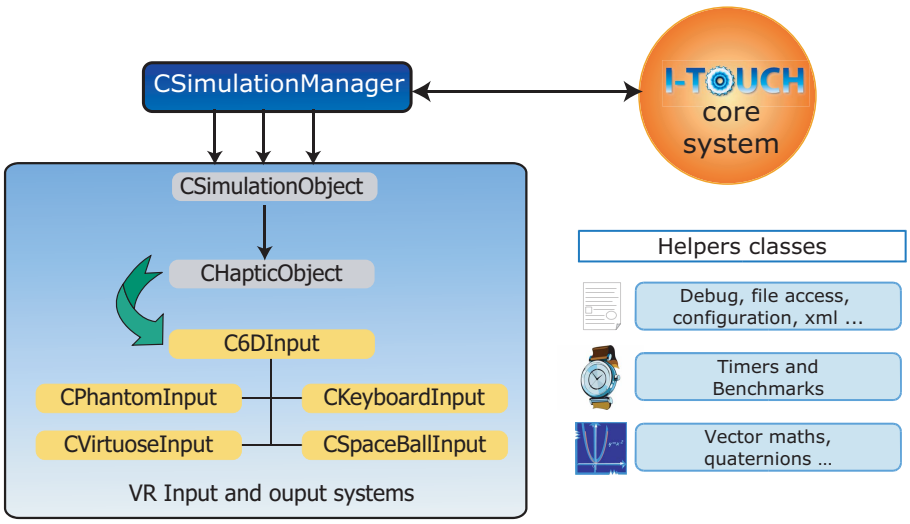

Figure 1: I-Touch framework architecture.

As it can be noticed, haptic interfacing, among other rendering capabilities, takes an important place. This particular focus meets the Touch-HapSys objectives in better understand haptics and its relation with the other modalities.

From a pure technical point-of-view, the framework is divided in three main modules; each of them is further subdivided in as many submodules as needed. The I-Touch framework is completely object-oriented. This allows easy part replacement and improvements. It is implemented

in a pure standard C++, and apart from the driver libraries, does not use platform dependent code. It can be easily ported to Linux or MacOsX, however it is designed to be best suited to MS Windows OS. Of course, this object-orientation make also easy the sub classing of some part of the framework, in order to create completely new applications. The fact we can devise new application "on demand" is of vital importance, for dedicated psychophysical investigations.

2.2 The core system

The core system is responsible for handling the operating system, the configuration, and the basic functionalities of a physically-based simulation. It provides a basic scene graph for managing the various objects that composes the virtual scene. This core system can accept many simulation algorithms along with different input methods. Classical mathematical methods [1] [2] and structures are also provided, for the easy prototyping/evaluation of new/existing algorithms. The core system also provides a very flexible configuration in XML, which parametrizes any aspect of the simulation, thus allowing the creation of test-beds rather easily.

In addition to configuration tools, a file format for holding together "geometrical" object properties has been devised. This format is open and flexible; moreover, additional data can be included and be ignored if not necessary to the simulation, even if it is an unknow data. An importer and exporter have been written for 3DSMax[1], along with C++ and C# libraries for loading efficiently theses files. This exporter can, for now, export geometry, normal, binormal and texture information, however, it does not save the relationship between texture information and texture map (this link has to be recreated in configuration files, however, the result is of course exactly the same).

This data format, exporter, importer and C++ libraries have been conjointly developed by Aurlien Pocheville and Mathieu Brunel. Details concerning the implementation can be found in [3].

2.3 The input and output system

While the input system needs to be flexible and needs to manage many different inputs, the output system should ensure high fidelity rendering along with adequate refresh rates according to the addressed modality/output.

2.4 The simulation system

The simulation is composed of the simulation manager, and a set of simulation virtual objects. The simulation system uses the core system for standard interaction with the computer and the user, and input/output system for multi-modal interaction. Collision detection algorithms are part of the simulation system. The simulation system, like any other part of the framework, can be replaced or modified rather easily.

3 Multimodal integration

One of the most challenging issues of I-Touch is multimodal integration. Visual, auditive and haptic senses have different refresh rates: from as low as 30Hz for visual interaction, up to as high as 10kHz for the 3D sound one. Integrating each of these modalities is not a trivial task. Others tentatives tried through parallelization of the computation on different computers. Here, we decided to push the limits on focusing in using one computer, but this unveils some problems as exposed later.

[1] `http://www.discreet.com/3dsmax/`.

3.1 Simulation engine flexibility

The fact that the simulation engine is completely flexible and modular allows the integration of different behaviors models with the same multimodal rendering. However, the simulation engine has to provide some information to the output routines. For example, for the real-time 3D sound rendering, contact information (and changes in contact through time) are required. The immediate benefit of this is that we can benchmark how well does a simulation engine behaves with multimodal rendering. For example, bounce models have difficulties in rendering contact information with sound, while they provide excellent rendering of bounce sounds.

3.2 Sound integration

3D positional audio enhances the immersion of the operator in the simulation. We have two methods for rendering 3D sound: real-time rendering, and semi-real-time rendering.

The real-time rendering uses information directly provided by the simulation, such as changes in the friction map to produce sound. It also uses object properties such as resonance frequencies to computes contact sounds. While this is the correct method for producing friction and bounce sound, it suffers from several drawbacks. First of all, it is very computational-time consuming, and, in a system composed by only one processor, it can become the bottleneck of the simulation (and take the place of the collision detection!). Maybe relocating the sound computations could solve this problem. The other fact is that the sounds generated are, for now, less "realistic" than the ones produced by the second approach.

The semi-real-time sound rendering approach uses off-line recorded sounds of different materials in contact. These different sound are stored in a database according to some material properties. They are used by the simulation as they are and the only amplitude and/or frequency modulation (pitch, volume...) are processed.

3.3 Visual integration

Relatively to the sound and the haptic rendering, the visual one is the easiest. We can use the same geometry as the one used for physics calculations, or a higher level, smoother one for better rendering. Objects are linked to rendering information, such as geometry, material and alpha information, and pixel and vertex shaders. This allows almost any rendering of the objects, from standard Gouraud-shaded plastic look, to advanced Phong-shaded semi-reflecting materials with bump mapping. Dynamic lighting is also supported. The visual rendering is completely configuration controlled, so there is a great flexibility in the rendering process.

3.4 Haptic integration

Our approach differs from the previous ones. Indeed we are conceptually considering that *the haptic devices (interfaces) interact with the simulation and not the reverse i.e. the haptic device does not drive the simulation*. Obviously, haptic devices can induce a change in the course of the simulation but they cannot compromise its integrity. To be more clear, the simulation does not take as granted what is needed from the input device and, in extreme cases, these particular inputs are ignored. In fact, these enhances considerably the stability of the interaction. For example, when the operator actions are toward violating a given non-penetration constraints, they are not considered integrally (as is the case for classical computer haptics methods).

The fact that haptic integration is not considered as a special rendering allows new synergies between rendering to be investigated. One example of this is the recently implemented *haptic bump*. As for visual bump mapping, we can simulate rough haptic surfaces through haptic bump[2].

We tried two different approaches: height based forces, and normal based forces. The basic principle is the same: the force computed by the simulation engine is slightly modulated by a term, which depends either on the height or the normal. In our actual implementation, haptic

[2]A specific device is being developed to render surface tactile and roughness informations [4]

bump does only work with one contact point, but we are working on extension to multiple points. As far as " bump sensation" is concerned, the normal based force give superior results. The bump map used for haptic bump is exactly the same as the one used in the visual bump, thus the two modalities match perfectly and the rendering is coherent.

3.5 Putting them together

Many synchronizing algorithms do exist, but for now, we did not experimented them, because of the fact that the very bottleneck of the simulation if the creation of physical data. Computation of different contacts, and then contact forces, takes the most of the cpu time. Consequently, the rendering processes are sufficiently quick to render the information without desynchronization. The data is given to renderers in this order (immediately one after the other): haptics, sound and visual.

In the future it would interesting to have scalable algorithms that would be totally disconnected for the physical simulation, we will then need synchronization algorithms. This would prevent that the new bottleneck (let's say, sound rendering) to harm the other displays. However, it will also mean that one of the modalities will have degraded performance, we will need to investigate how far we go in degradation in one of the modalities without ruining interaction.

4 Evaluation tools

The testing of research projects is made easy with I-Touch, however such a testing requires to analyze data from the simulation. In I-Touch, every simulation variable can be "tagged" from recording, this allows after-run simulation analysis through tools.

For example, FPS data, or time taken to compute a frame (much more speaking than FPS in regard to performance) evolution can be viewed easily. Also, the debugging facilities and text functions in I-Touch make it easy to dump data. However, we want to go further, and new real-time tools are in development. Such tools will render in real time evolution of variables, in numerous manners (time graphs, bars, standard text, etc.). Specific analysis tools to simulation should also be included, such as automatic reporting of number of contacts, physics calculation time, time spent in rendering or in other tasks, etc. This will create a complete and easy-to-use evaluation tool, in order to accelerate further the development of test cases. This step is very important in the relationship that this project can have with psycho-physical and others experiments.

5 Conclusion and future work

We have shown that we have made a prototype of software framework that allows the creation of many test cases. They all have in common experiments with operation, in a multimodal way. The haptic rendering is not considered with special concerns, thus facilitating the evaluation of haptic impact in a multimodal context.

This framework has unveiled new requirements in collision detection, we are in the middle of new algorithm to handle contact (and not collision) topology. Interesting breakthrough in the modeling of contact have been made by Yann Chenu in this regard [5].

We have succeeded in creating sample applications that use the flexibility of the I-Touch framework. However, integration with real psycho-physical tests is yet to be done. Further work is oriented toward refining I-Touch through its multimodal component to serve also as a psychophysics evaluation tool. Progressively our aim is to evolve it to a complete piece of software that can serve haptic research. In the annex, a copy of a recent published paper concerning I-Touch describes more implementation details.

References

[1] L. Dorst and S. Mann, "Geometric algebra: A computational framework for geometrical aplications," *IEEE Computer Graphics and Applications*, 2002.

[2] J. C. Hart, G. K. Francis, and L. H. Kauffman, "Visualizing quaternion rotation," *ACM Transactions on Graphics*, vol. 13, no. 3, pp. 256–276, 1994.

[3] M. Brunel, "Rendu multimodal haute fidelit, rapport de fin d'tude," 2004.

[4] A. Drif, J. Citérin, and A. Kheddar, "A multilevel haptic display design," in *IEEE/RSJ International Conference on Intelligent Robotic Systems (IROS)*, 2004.

[5] Y. Chenu, "Algorithmes de dtection du contact, topologie du contact, rapport de dea," 2004.

[6] C. Lennerz, E. Schömer, and T. Warken, "A framework for collision detection and response," in *11th European Simulation Symposium, ESS'99*, 1999, pp. 309–314.

[7] J. Sauer and E. Schömer, "A constraint-based approach to rigid body dynamics for virtual reality applications," *Proc. ACM Symposium on Virtual Reality Software and Technology*, 1998.

[8] K. T. McDonnell, "Dynamic subdivision-based solid modeling," 2000.

[9] D. E. Stewart, *Time-stepping methods and the mathematics of rigid body dynamics.* Birkhuser, 2000, ch. 9.

[10] K. G. Murty, *Linear Complementary Linear And Nonlinear Programming.* Internet Edition, 1997.

[11] A. Lécuyer, "Contribution l'étude des retours haptique et pseudo-haptique et de leur impact sur les simulations d'opérations de montage/démontage en aréonautique," Ph.D. dissertation, Université Paris XI, 2001.

[12] S. Redon, "Algorithmes de simulation dynamique interactive d'objets rigides," Ph.D. dissertation, Université d'Evry, 2002.

[13] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," *ACM SIGGRAPH*, 2001.

[14] P. J. Berkelman, R. L. Hollis, and D. Baraff, "Interaction with a realtime dynamic environment simulation using a magnetic levitation haptic interface device," *IEEE International Conference on Robotics and Automation*, pp. 3261 – 3266, 1999.

[15] D. Baraff, "Fast contact force computation for nonpenetrating rigid bodies, siggraph," *ACM SIGGRAPH*, 1994.

[16] R. Barzel and A. H. Barr, "A modeling system based on dynamic constraints," *Computer Graphics*, vol. 22, 1988.

[17] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, *Six Degree-of-Freedom Haptic Display of Polygonal Models.* T. Ertl and B. Hamann and A. Varshney, 2000, pp. 139–146.

[18] D. Baraff and A. Witkin, "Dynamic simulation of non-penetrating flexible bodies," *Computer Graphics*, vol. 26, no. 2, pp. 303–308, 1992.

[19] J. Schmidt and H. Niemann, "Using Quaternions for Parametrizing 3–D Rotations in Unconstrained Nonlinear Optimization," in *Vision, Modeling, and Visualization 2001*, T. Ertl, B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, Eds. Stuttgart, Germany: AKA/IOS Press, Berlin, Amsterdam, 2001, pp. 399–406.

[20] A. Lécuyer, S. Coquillart, A. Kheddar, P. Richard, and P. Coiffet, "Pseudo-haptic feedback: can isometric input devices simulate force feedback?" in *IEEE International Conference on Virtual Reality*, New Brunswick, 2000, pp. 83–90.

[21] G. V. Popescu, "The Rutgers haptic library," in *IEEE International Conference on Virtual Reality, Haptics Workshop*, New Brunswick, 2000.

[22] G. Bouyer, "Rendu haptique et sonore 3d en prototypage virtuel," 2002.

[23] P. Meseure, A. Kheddar, and F. Faure, "Détection des collisions et calcul de la réponse," Action Spécifique DdC du CNRS, Tech. Rep., 2003.

[24] M. C. Lin and S. Gottschalk, "Collision detection between geometric models: a survey," in *IMA Conference on Mathematics of Surfaces*, vol. 1, San Diego (CA), May 1998, pp. 602–608.

[25] D. A. Lawrence, "Stability and transparency in bilateral teleoperation," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 624–637, 1993.