

Final Project Report
Summer Scholarship

Haptic Devices

Submitted To

**South Australia Partnership for Advanced
Computing (SAPAC)**

December 2006 – February 2007

Compiled By:

Ashley Fonseca
ashley.fonseca@flinders.edu.au

Acknowledgements

This project has benefited from the generous contributions of time of many individuals. I would like in particular to thank my supervisor A/Prof Karen Reynolds and Greg Ruthenbeck for their time and support of my work. I would never have completed my work without their assistance.

Introduction

Haptic means sense of touch. A haptic interface is a device which allows a user to interact with a computer by receiving tactile feed back. This feedback is achieved by applying a degree of opposing force to the user along the x, y, and z axes. These devices can be used by, medical practicing, people with disabilities or people who learn best through tactile or kinesthetic experiences (<http://www.utoronto.ca>).

Various haptic interfaces for medical simulation may prove especially useful for training of minimally invasive procedures (laparoscopy/interventional radiology) and remote surgery using teleoperators. In the future, expert surgeons may work from a central workstation, performing operations in various locations, with machine setup and patient preparation performed by local nursing staff. Rather than traveling to an operating room, the surgeon instead becomes a telepresence. A particular advantage of this type of work is that the surgeon can perform many more operations of a similar type, and with less fatigue. It is well documented that a surgeon who performs more procedures of a given kind will have statistically better outcomes for his patients. (Wikipedia)

Review of Problems and Opportunities

This research was undertaken to learn and solve some of the problems faced by using Haptic Devices. Initially I spent some time understanding the working of haptic device. I have found that the major problem was the refreshment frame rate i.e. frames per second (fps) rate, which generated problem of judder. The first challenge was to increase the fps rate, so the graphical application using haptics, runs smoothly.

To deal with this issue, first I have to understand the haptic device operation and the way it is used. Then create the dll file to access the haptic device from any application at very high speed. This dll file should have most of the functions that is used for haptic operations. It should include both the SensAble haptic APIs viz. HDAPI and HLAPI.

Later on graphical application demonstrating haptic is to be created. These applications well demonstrate the capability of the haptic device. Ogre library can be use to develop this application in short time, as they are easy to learn and implement.

The SensAble Technologies PHANTOM® product line of haptic devices makes it possible for users to touch and manipulate virtual objects. PHANTOM Omni device was used thru out this research. This device has six degree of freedom. The SensAble OpenHaptics toolkit was used to operate this device. This toolkit enables us to add haptics and true 3D navigation to a broad range of applications, from design to games and entertainment to simulation and visualization. This toolkit includes the Haptic Device API (HDAPI), the Haptic Library API (HLAPI), utilities, PHANTOM Device Drivers (PDD), and source code examples.

Additional Background

The funding for this project was awarded by the South Australia Partnership for Advanced Computing (SAPAC) in form of summer scholarship. This project was fully done at the lab of engineering building of Flinders University under the supervision of A/Prof Karen Reynolds.

Tools Developed

Initially I studied some of the open source code demo application to understand the working of the application and to know their problems. It was found that there were actually many layers of interfaces between application and haptic device, which slows down the refreshment rate. So the main problem was the refreshment rate, which needs to be improved. This can be done by providing direct interface between the application and haptics.

The primary goal emphasized was the development of dll file, which well act a driver between haptic device and application. To achieve this target, better understanding of haptic device and its operation need to known. To do this Haptic Studio application was develop.

Haptic Studio

Haptic Studio application was intended to help to understand the working of the haptic device and its operation. This application demonstrates the Position, Velocity, Acceleration and Force in graph format as well as in

numbers. It also allow changing the time for the update rate, shows target update rate and Actual frame update rate that is currently achieved. This application was develop is C#. When the Y coordinate of Phantom goes below zero, then the force is applied in the same proportion.

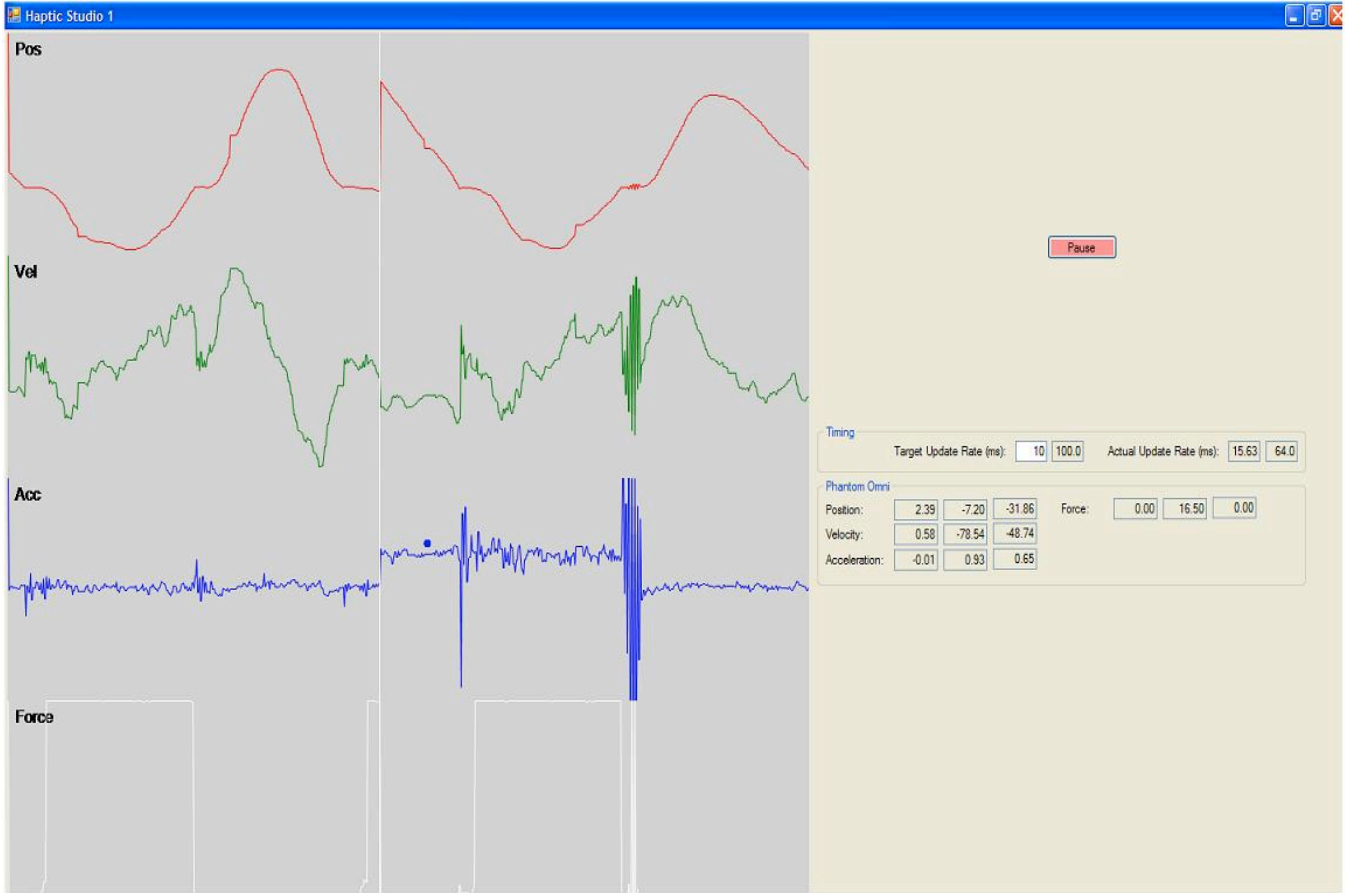


Figure 1: Haptic Studio Application Screen Shot

Here on top graph position of Y co-ordinate of haptic is shown in red. Then Velocity is shown in second graph in green. Then acceleration is calculated based on velocity and shown in third graph in blue. And the last graph represents the current force of the Phantom.

In the right hand panel, there is numerical representation of the same data with some extra information. The time frame represents the target update rate that we are looking for, which can be changed and actual update rate of frame. The X, Y, and Z coordinate of position, velocity, acceleration and force is displayed. There also option to pause and resume the application.

It was found that the actual frame rate per second was only 64. Even though target frame rate was changed, maximum speed could be

achieved was 64 Hz. This makes clear that the judder problem of haptics was due to refreshment rate (low frame rate).

HDWrapper DLL Project

HDWrapper is a DLL project, in which most of the functions required by haptics are included. This project uses SensAble's OpenHaptics Toolkit v2.0. The namespace HDN is used and there are two classes viz HDFuncs and HLFuncs containing the functions for Haptic Device API (HDAPI) and Haptic Library API (HLAPI) respectively.

The main purpose of this project is to increase the frame rate per second (speed) of haptic devices. In Haptic Studio project it was found that the current driver class that accesses the haptics was not more than 64 frames per second (fps). To run the haptics smoothly on graphical applications, fps need to be increased.

HDAPI is the low level API in which programmer gets direct access for rendering the device, whereas HLAPI provides OpenGL functionality, whereby it allows high-level haptic rendering. HLAPI is high level API which provides higher level control to haptics than HDAPI.

To use the dll file from the DynamicLibrary solution, each time we need to give entry point while importing function. For example:

```
[DllImport("HDWrapper.dll", EntryPoint = "?Initialise@HDFuncs@HDN@@@SAIM@Z")]  
public static extern int Initialise(float rampRate);
```

To see how to get entrypoints of the function read the documentation of this project. This entry points changes, if dll project is modified and rebuilt. That means each time the dll project file is modified, the entrypoints has to be changed in the application project. That takes lot of time and is not the efficient way of doing things. So this dll project can structure / modify in such a way, that while importing its function, we don't have to give the entrypoints.

This dll project has got namespace HDN and two classes HDFuncs and HLFuncs. To get read of entrypoints we have to remove this namespace and classes. The prototype of function in file HDWrapper.h should be declared directly in file (not in namespace and classes). Similarly the function declaration in file HDWrapper.cpp should not be in any namespace.

Then the function has to be called by standard call by keyword `__stdcall` and by removing static from static class.

In current prototype, function is define as
`static __declspec(dllexport) HHD Initialise(float rampRate);`

instead it should be defined as
`__declspec(dllexport) HHD __stdcall Initialise(float rampRate);`

Similarly HDWrapper.cpp change the function declaration.

Build this project and update your applications dll with this new dll. Then call the function as shown below.

```
[DllImport("HDWrapper.dll")]  
public static extern int Initialise(float rampRate);
```

Note that there is no entry point use in DLLImport function.

Why namespace and classes are used in this project ?

This dll project uses namespace and classes. One of the reasons for that is using namespace allows us to use the function name which is used already in other source files. In addition, functions are not just floating anywhere in the project. There are in well structure namespace.

Classes are used to store any value return by the function. For example, we can save the handle number in class variable and can make use of it later on.

HapticWrapperConsole

To check the performance of the HDWrapper dll, HapticWrapperConsole application was developed in C#. This application use HDFuncs class functions from HDWrapper dll.

This application displays the position, force, time taken per frame, start time of frame and end time of frame on console. Force is set when Y coordinate goes below zero, and then force is increase half time of Y coordinate position. Also it has made sure that force never goes above maximum force of the device.

```
C:\Documents and Settings\Ashley\My Documents\Visual Studio 2005\Projects\My exe\Hapti...
start time: 2/14/2007 10:49:35 AM
end time: 2/14/2007 10:49:37 AM

pos: 45.0627098083496, 135.09895324707, -17.2841091156006
force: 0,0,0
time difference in milliseconds: 2
start time: 2/14/2007 10:49:37 AM
end time: 2/14/2007 10:49:39 AM

pos: -78.2534866333008, -77.5109939575195, -11.915111541748
force: 0,38.7554969787598,0
time difference in milliseconds: 2
start time: 2/14/2007 10:49:39 AM
end time: 2/14/2007 10:49:41 AM

pos: 18.6799011230469, -0.300712287425995, 35.7116508483887
force: 0,0.150356143712997,0
time difference in milliseconds: 2
start time: 2/14/2007 10:49:41 AM
end time: 2/14/2007 10:49:43 AM
```

Figure 2: Snap Shot of HapticWrapperConsole Application

The resulting speed of this application was 2 milliseconds per frame i.e. 500 Mhz / sec. This speed was obviously more than any other haptic application that we have developed. This application proves that new dll file actually improves the performance of the haptic device.

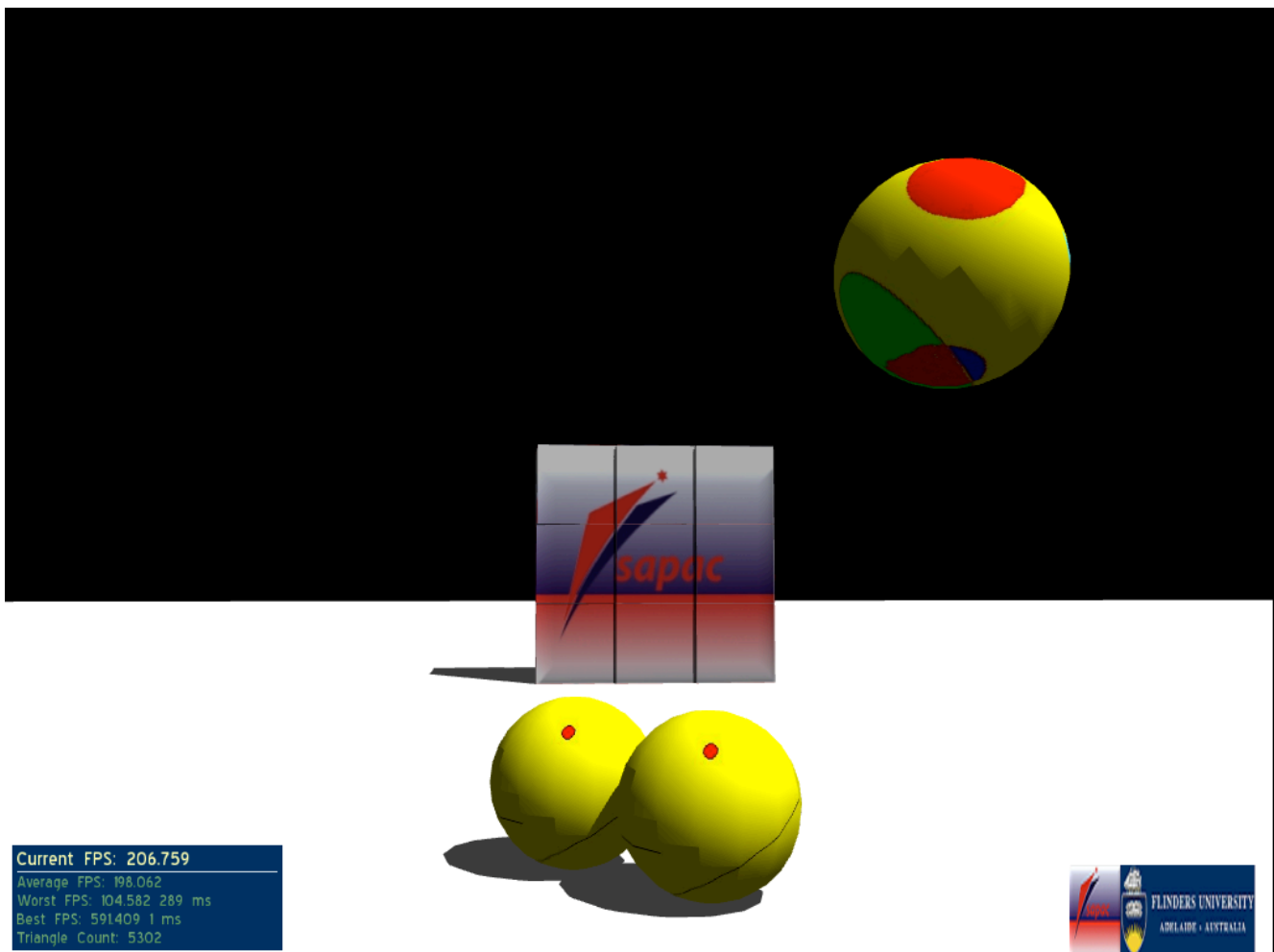
HelloSphere Demo

HDWrapper dll project was tested on HelloSphere Demo application of SensAble. The original source code was in C, which was converted in C++ by Visual Studio 8. Then, all the references of SensAble dll was removed and HDWrapper.dll file was copied in this application folder. The new reference to all the required functions were added from HLFuncls classes, which is HLAPI.

This application uses HLAPI shape functions to create the shape and to add the rendering and haptic effect on it. This experiment was also successful as graphical implementation of HLAPI give positive result. This sphere application is running smoothly on new HDWrapper dll file.

Haptic Playground Demo

This application was developed on OGRE libraries. OGRE is packed with features to make development life easier, whether we're making games, architectural visualisation, simulations, or anything else which requires a top-notch 3D rendering solution. This application demonstrates the capabilities of the haptic device. It uses OGRE libraries with OpenGL and DirectX support.



It has got nine boxes and three balls that can be played by Phantom devices. It has got force and gravity effects which make it feel real. There is spring force used to throw the balls on the boxes. To pick any object

first, haptic has to touch that object and then press stylus button. To reinitialise the all the object press 'l'.

This application uses different layers of interface. Lowest layer is OGRE libraries for OpenGL and DirectX above that is OGRE library for haptic devices, then above it comes the Physics library and then on top level this application is build. This application needs OGRE dll and lib files.

Conclusion

In this scholarship, I had working experience in 3D programming and its implementation using haptic device. I got to learn difference types of matrix transformation for the orientation. I am quite familiar with haptic device and its uses in medical simulation.