Creating a Multimodal 3D Virtual Environment

Johannes Pystynen

University of Tampere
School of Information Sciences
Interactive Technology
M.Sc. Thesis
Supervisor: Roope Raisamo
30.12.2011
Multimodal virtual environments can add value to teaching and learning in school contexts although touch based multimodal virtual environments have rarely been used to support the studying of natural sciences. This thesis describes the techniques and tools used in developing an application framework for multimodal virtual environments. Three multimodal applications were designed and implemented: the Density and the Leverage applications, both of which were meant to help in physics and chemistry studies, as well as a 3D Construction application that can help to perceive three-dimensional objects and spatial perception. To find out the effectiveness of the applications a study was conducted in a school context in which the applications were used by the teachers and pupils in Ylöjärvi elementary school in Finland. The results from the study were generally positive on utilizing the multimodal applications in school context. With the applications it was possible to concretize often abstract and complex physical and geometrical phenomena.

Keywords: Multimodal virtual environment, haptics, graphics, force feedback.
Content
1. Introduction..............................................................................................................1
2. The History of Virtual Environments.................................................................4
   2.1. Computer Graphics.........................................................................................4
   2.1.1. 3D Graphics and Applications.................................................................5
   2.2. Haptics............................................................................................................6
       2.2.1. History of Active Haptic Feedback .......................................................6
       2.2.2. Haptics in Multimodal Virtual Environments........................................6
3. The Technology Behind Multimodal Virtual Environments..............................8
   3.1. Three-dimensional Computer Graphics.......................................................8
       3.1.1. OpenGL..................................................................................................8
       3.1.2. High Level Shading Languages............................................................8
       3.1.3. Planar Reflections..................................................................................9
       3.1.4. Shadowing............................................................................................11
       3.1.4.1 Shadow Mapping..................................................................................11
       3.1.5. Texture Mapping..................................................................................12
       3.1.5.1 Normal Mapping..................................................................................12
       3.1.5.2 Advanced Bump Mapping....................................................................13
   3.2. Computer Generated Haptics.........................................................................14
       3.2.1. Haptic Rendering..................................................................................14
       3.2.1.1 A Constraint-based God-object Rendering Algorithm .......................14
       3.2.1.2 The “Ruspini” Rendering Algorithm..................................................15
   3.2.2. Force Feedback Haptic Devices...............................................................16
   3.3. Algorithms and Techniques............................................................................18
       3.3.1. Constructive Solid Geometry (CSG)......................................................18
       3.3.2. Marching Cubes....................................................................................19
   3.4. Libraries and Tools.......................................................................................19
       3.4.1. LibSDL..................................................................................................19
       3.4.2. Haptic Application Programming Interfaces..........................................20
       3.4.3. Physics Simulation in Virtual Environment..............................................20
       3.4.4. XML-File Support..................................................................................20
       3.4.5. Auditory Feedback in Virtual Environment.............................................21
4. Implemented Multimodal Applications............................................................22
   4.1. The Application Framework and the Tools...................................................23
   4.2. The Density Application................................................................................24
   4.3. The Leverage Application..............................................................................27
1. Introduction

Computer-generated environments can help to simulate real-life and imaginary phenomena. As of today, virtual environments are primarily based on visual feedback though we study our world with different senses. Multimodal interaction with different inputs and outputs can augment the feel of a virtual environment. A multimodal virtual environment (MVE) is a computer generated virtual environment that utilizes more than one modality. There are different modalities that have been of interest. Especially computer graphics, which are graphics created with computer software and hardware, play a big role in virtual environments. Three-dimensional computer graphics, are achieved by presenting a three-dimensional image on a two-dimensional raster display. In more detail, that means that geometric data is presented mathematically in three dimensions, and with three-dimensional rendering pipeline the objects are transformed to present the scene in two dimensions the way that it can be displayed on a raster display [Puhakka, 2008; Watt, 2000].

Furthermore, there are many modalities that can enhance virtual environments. One of these is haptics. The word haptics refers to the utilization of the sense of touch and comes from a Greek term meaning 'able to lay hold of' [Klatzky and Lederman, 1987]. Haptic feedback can be active or passive. Active haptics is a way to present haptics actively, meaning that the haptic device generates forces that do not only resist movement but create movement. The actual force feedback, that is felt by the user, is computationally generated with a process called haptic rendering [Ruspini et al., 1997]. In multimodal virtual environments force feedback haptics is the common haptic utilization method.

In recent years, virtual environments have been of growing interest in teaching. This is a result of the realization that computer based virtual environments can add value to teaching and learning. Multimodal virtual environments (MVEs) can improve the teaching experience even more by augmenting different modalities to support the most commonly used visual aspect. In addition to the obvious visual feedback, touch plays an important part in our life and, therefore, it is one of the most suitable modalities to be used in virtual environments. Furthermore, speech interfaces have been used in MVEs to enable visually impaired people to become involved.

However, there have been few researches on utilizing MVEs in educational context. Wiebe et al. studied a simulation on the principles of levers in a cross-modal setting [Wiebe et al., 2009]. In their study, haptic feedback was added to their lever-study environment. Hamza-Lup et al. have developed a novel E-learning system that incorporates a multimodal haptic simulator and studied its facilitation in a school context [Hamza-Lup and Adams, 2009]. In addition, children with disabilities have been of interest, especially in the area of utilizing haptics with visually impaired users and enabling with it their learning [Saarinen et al., 2006; Tanhua-Piirainen et al., 2008]. Furthermore, Calle Sjöström has studied MVEs and has presented some guidelines for their
creation [Sjöström, 2002]. In short, multimodal virtual environments have not been widely accepted to class rooms and homes and there are not many robust multimodal application programming interfaces (APIs) available. In addition to education, there are many areas where multimodality can be used in virtual environments to add value to the experience. These areas include medicine, gaming, and robotics.

Multimodal virtual environments have been created with different tools for different purposes. There are few well-known multimodal application programming interfaces available that utilize haptics and graphics. These APIs include H3DAPI [SenseGraphics, 2011], CHAI3D [Force Dimension, 2011], and OpenHaptics [Sensible, 2011] that are mainly used in research.

This thesis relies heavily on the vast amount of work done in the field of computer graphics [Puhakka, 2008; Watt, 2000] and haptic APIs [Kadlecek, 2011; SenseGraphics, 2011; Force Dimension, 2011; Sensable, 2011]. This thesis describes the process and techniques used in implementing an application framework for three-dimensional haptic device controlled virtual environments. When building such an application many things have to be taken into consideration. What is the focus group, and should the applications add value to different level of users? Should there be support for visual, auditory, and haptic feedback?

There are three applications that were created and are presented in this thesis. Two of the applications, the Density and the Leverage, have been composed by the author and researchers at the University of Tampere with ideas from teachers of Ylöjärvi elementary school in Finland. The 3D Geometric Construction application has been brainstormed by the author, Erika Tanhua-Piirainen, and Roope Raisamo at the University of Tampere, and implemented by the author.

The multimodal application framework has been designed and implemented by the author and all the program code has been written by the author except the open source libraries and tools that are used. The application framework is the same in all of the three applications. Small modifications in the framework have been done for each application.

The created applications have been used in teaching in Ylöjärvi elementary school in Finland. In addition, their usage has been studied, by Tanhua-Piirainen et al., and published in their award-winning conference article “Haptic Applications as Physics Teaching Tools” [Tanhua-Piirainen et al., 2010]. The findings by Tanhua-Piirainen et al. were generally positive and supportive for using multimodal virtual environments in lessons in addition to conventional physical tools. Even though the study was qualitative, the general feedback by the teachers and the pupils strongly recommend that the applications enhance the learning experience and makes it more fun.

This thesis is divided into seven chapters. The second chapter presents a brief history on multimodal virtual environment related areas, three-dimensional graphics and haptics. The history section gives an aspect to the development of these areas, and how the technology has advanced to that point where it is currently possible to have real-time MVEs.
The third chapter goes deeper into the technologies that are needed to create MVE applications. Graphics libraries and important visual enhancement techniques are explained. In addition, haptic rendering methods are explained and compared to substantiate the chosen haptics methods for the applications.

In chapter four, the actual implemented applications are explained in detail; the user interfaces and use-cases are presented verbally with images.

Chapter five evaluates the applications and uses the published study [Tanhua-Piirinen et al., 2010] by Tanhua-Piirinen et al. as a reference to bring forth the pros and cons of the applications.

The sixth chapter is the discussion chapter, that recaps the previous chapters and summarises the need for these applications.

The last chapter, chapter seven, concludes the study and gives an idea where to go from here. What were the pros and cons of the implemented applications? Could these applications be further developed easily, what could be done better and differently?
2. The History of Virtual Environments
Multimodal Virtual Environments (MVEs) bring together different modalities. Usually these modalities include visual and haptic rendering. Additionally, speech interaction can be present. Neither computer graphics nor haptics have developed to their present stage over night. It has taken many iterations and research to get where we are now.

2.1. Computer Graphics
The history of computer graphics is vague due to it being a relatively young area of science and its applications are even younger. In addition, the term "computer graphics", proposed by a Boeing designer William Fetter [Puhakka, 2008], was established in the 1960s. Sketchpad [Sutherland, 1963], created by Ivan Sutherland in 1963 is widely regarded as the starting point for computer graphics and graphical applications [Machover, 1978].

It can be said that the first actual computer graphics applications were implemented during the 1950s in the United States. The applications included among others, the SAGE air-defense command and control system [Puhakka, 2008; Machover, 1978]. In SAGE, and other graphical systems of that day, the image was represented to users by vector graphics. With SAGE, users were able to select information from the user interface, displayed on a CRT, by pointing at the appropriate target with a light pen.

In the early 1960s, IBM designed the first commercial computer-aided design program called DAC-1. In the same era, the first graphical computer game Spacewar was implemented. In 1962, Pierre Bezier introduced and patented the Bezier curves and Bezier surfaces, although they had first been developed by Paul de Casteljau in 1959, using de Casteljau's algorithm [de Casteljau, 1962]. Bezier curves are widely used in computer graphics and related fields to model smooth curves that can be controlled and manipulated with control points.

As stated above, the basis for today's graphical user interface comes from the Sketchpad – a program written in 1963 by Ivan Sutherland, and which at that time was thought to be revolutionary. The Sketchpad system broadened human-computer interaction by enabling communication with line drawings instead of commonly used typed statements.

In the 1960s and at the beginning of the 1970s, hidden surface removal was one of the important areas researched in three-dimensional graphics [Puhakka, 2008]. One of the first techniques for hidden surface removal was the Z-buffer technique discovered by Edwin Catmull. He described this technique in his doctoral thesis [Catmull, 1974], published in 1974 although the idea was also discovered by others in the same year. The Z-buffer technique is still widely used in real-time applications and it is essential in rendering when deciding the actual pixel to be drawn in 3D graphics.
Vector displays were replaced by pixel based raster displays in the 1970s. This was a result of a drop in RAM prices. With RAM, frame buffer was now available at reasonable prices. This further ignited the development of computer graphics and made it more broadly available.

During the 1970s lighting was an important part of the research in computer graphics. Gouraud shading [Gouraud, 1971], where a curved impression is achieved by interpolating the color from the edges of triangles, was introduced by Henri Gouraud. In addition, in 1975, Bui Tuong Phong introduced an advance to Gouraud shading with specular lighting in his doctoral thesis [Phong, 1973]. Jim Blinn has also had a big influence on computer graphics with his bump mapping and environment mapping techniques that are widely used in 3D applications, such as games. Nowadays, one of the most popular lighting techniques is Blinn-Phong [Blinn, 1977] shading, which creates smooth and specular lighting for objects.

In the 1980s, computer graphics was widely adopted in the manufacturing industries and the first AutoCAD program was made available for the PC.

The leading developments in computer graphics are mostly presented at the SIGGRAPH conference, the most important conference in the field of computer graphics. It has been held annually since 1974 and is convened by the ACM SIGGRAPH (Special Interest Group in Graphics and Interactive Techniques) organization [ACM SIGGRAPH, 2011].

2.1.1. 3D Graphics and Applications
Computer graphics is all around us in several different fields. It helps in industrial design, hospitals, and every day life. 3D graphics are also commonly used in the entertainment businesses. Nowadays it is hard to find an action motion picture without computer generated image enhancements. Even some full length motion pictures have been acted in front of blue screens.

Computer games were popularized in the 1980s. The development of computer graphics has been closely related to the development of video games. Without 3D video games it is hard to imagine where computer graphics might be today. At first, the games were two dimensional and it took until 1993, when Id Software's Doom [Id Software, 1993] was introduced, to start the 3D development. This opened doors for graphics vendors to sell graphics acceleration cards to consumers. Development has been fast and the acceleration cards are taking on more and more computational responsibilities with every generation. Even the newest smartphones ship with 3D capable graphics chips. In addition, general-purpose computing on graphics processing units (GPGPU) has grown popular in recent years.
2.2. Haptics

“Haptic technology does for the sense of touch what computer graphics does for vision”

[Robles-De-La-Torre, 2009]

We feel and examine objects and their properties everyday with touch. By feeling the objects, we learn from the object its weight, elasticity, shape, and texture. When we have prior knowledge of an object through touch, we can combine those properties with visual properties and have a more complete knowledge of the object. With computer graphics, it is easy to combine learned properties with objects that we have seen and touched before. If we are visually examining a new object for the first time we cannot fully identify and understand the physical appearance of it by not knowing all of the objects properties. By combining haptic feedback with computer graphics we can study the properties of the object more broadly.

2.2.1. History of Active Haptic Feedback

Force feedback is a mechanical stimulation that can be used to assist in controlling virtual objects or to give users more realistic feedback to simulate the real world. Active haptic feedback has been used in the industry where massive vehicles or control systems have to be dealt with and controlled. One of those areas is aircraft that need to give feedback to pilots through control systems. Many simulators and robot control systems use haptic feedback to get some feedback to the controller of the devices.

Medicine is an area where haptic feedback has been adopted by applications that train users by mimicking for example the tissue feedback of a real life organ. In addition, haptic feedback could enhance the teleoperation of minimally invasive surgical robots [Okamura, 2009] or could enhance the remote operating of robotics.

In addition, haptic feedback is commonly used in arcade racing games. Sega's arcade game Moto-Cross [Sega, 1976] (rebranded as “Fonz”) was the first game to use haptic feedback. Furthermore, force feedback was introduced to racing games in 1983 with TX-1 arcade racing game by Tatsumi [Tatsumi, 1983]. Today, all the popular game consoles offer force feedback game controllers. From 2007 onwards, consumer gamers can use a three dimensional force feedback device named “Novint Falcon” [Novint Technologies, 2007] that is available for the PC.

2.2.2. Haptics in Multimodal Virtual Environments

Multimodal Virtual Environments (MVEs) have been developed from the beginning of the 21st century. Usually, the sense of sight has been utilized, but additionally haptics has been used to fulfil the sense of touch in virtual environments. Haptics offer a new way to interact for sighted
people, but it enables virtual environments for blind people as well. In addition, speech feedback is commonly used in MVEs, at least when dealing with visually impaired people.

Research has been done on MVEs with haptics, in medical applications [Okamura, 2009]. In addition, visually impaired people has been a focus group in some studies [Saarinen et al., 2006; Tanhua-Piirinen et al., 2008]. Calle Sjöstöm has studied multimodal virtual environments and presents some guidelines for non-visual haptic interaction in his doctoral thesis [Sjöstöm, 2002]. There has been some new research studying the possibilities to integrate MVEs into the school context by Tanhua-Piirinen et. al [Tanhua-Piirinen et al., 2010] and by Wiebe et al [Wiebe et al., 2009]. Especially haptics with force feedback has been considered to be a possible addition to virtual environments in the teaching of natural sciences. Hamza-Lup et al. have developed a novel E-learning system that incorporates a multimodal haptic simulator [Hamza-Lup and Adams, 2009]. The simulator was meant for a school context to facilitate students' understanding of difficult concepts, such as in physics. They have designed and implemented a novel visuo-haptic simulation called “Haptic Environments for K-16” (HaptEK16) for teaching physics concepts. The system was developed using the Extensible 3D modeling language and the SenseGraphics H3D API [SenseGraphics, 2011].
3. The Technology Behind Multimodal Virtual Environments

A multimodal virtual environment consists of various technologies that are seamlessly put together. It has two or more modalities for input and/or feedback. Commonly there is visual feedback that is enhanced or taken further with haptic or auditory feedback. You could say that a multimodal application is stitched together from different pieces. How seamlessly the parts are fit together can impact heavily on the performance and the usability. This chapter introduces some key multimodal virtual environment features that are used in the applications presented in the next chapter.

3.1. Three-dimensional Computer Graphics

We feel and see the world in three dimensions. Even though the trend seems to be bringing a three-dimensional visual experience even to home displays, we still have raster displays that present a three-dimensional image in two dimensions usually with perspective projection. The output image, that is seen on the display is constructed from a three-dimensional scene with transformations.

3.1.1. OpenGL

OpenGL (Open Graphics Library) [OpenGL, 2011], is an open and platform independent graphics standard that was made available in 1992. It was built on the basis of SGI's IRIS graphics library and an industry-wide consortium was set up to maintain the new standard. It is a standard specification defining a cross-language, cross-platform API for writing applications that produce 3D and 2D computer graphics. The OpenGL interface has more than 250 different function calls that can be used to produce various graphics from simple primitive drawing to more complex 3D scenes. OpenGL is based on the C programming language, and has a state machine style approach. Though, in recent years the development has been made to transfer the API to more of an object based system. At the time of writing, the newest version of the standard is 4.2. [OpenGL, 2011]

OpenGL has serious competition from the Direct3D rendering API [Microsoft, 2011]. The Direct3D is Microsoft's creation, and it is mainly used on Microsoft products such as Microsoft Windows and the Xbox, whereas OpenGL as an open source platform is available for a wide variety of platforms. Both platforms are implemented in the display driver, and the graphics card vendors are usually producing the drivers for best performance.

There are still many differences in the APIs, but in the recent years the APIs have been heading to the same direction and are offering somewhat the same functionalities.

3.1.2. High Level Shading Languages

From OpenGL 2.1 onwards shaders has been the way to go. OpenGL's high level shading language is called OpenGL Shading Language (GLSL) [OpenGL, 2011]. Prior to the introduction of shaders, video hardware was mainly programmed with a fixed function pipeline. The instructions to
graphics hardware were sent as is, and no changes were possible in the pipeline after that. Post processing effects and more modifiability in the pipeline was desired and the answer was programmable shaders. The shaders enable programmability throughout the graphics pipeline and make possible wide variety of graphical effects. Besides OpenGL, Microsoft's DirectX has a similar shading language called High Level Shading Language (HLSL) [Microsoft, 2011].

The vertex shader and the pixel shader were the first programmable shaders, but DX10 and OpenGL 3.2 versions introduced the possibility, with supported hardware, to program geometry shaders.

The shaders are programmed with a C-style language. The input in vertex shader is a vertex that goes through transformations, and the output is a transformed vertex. Geometry shader takes as an input one primitive and as output puts out the same primitive or creates new primitives and outputs all of them. Pixel shader, also known as fragment shader, computes colour value for a pixel with different attributes.

Prior to high level shading languages, there was an assembly style shading language with what it was possible to implement shader programs. Because high-level shading languages are easier and faster to use, the shaders are usually not programmed by assembly style any more.

3.1.3. Planar Reflections

Reflection in computer graphics is used to mimic the behavior of reflective surfaces like mirrors, glass, and water. The most common surfaces that are made reflective in real-time computer graphics are planes. The technique here is to draw the objects that are seen from the mirror-like surface to the position that they would been seen from the mirror-like surface in the right angle. This means that the actual objects will be rendered two times, in the right position for the viewer and in the position seen through the reflector.

The technique often used is render to texture meaning that the reflected view is rendered to a texture and the texture is used in some surface. In Figure 1, the reflected view is rendered to a plane. The size of the texture influences notably the pleasantness of the reflective surface. If the texture, that has the reflection, is small, there will be visible aliasing (see Figure 1). With bigger resolution the outcome looks more real and pleasant to viewer (see Figure 2).
Figure 1. Reflective surface generated with small 512x512 target resolution.

Figure 2. Reflective surface generated with 1536x1536 target resolution.
3.1.4. Shadowing

Shadows enhance greatly the three-dimensional experience in virtual reality. The virtual scenes become much more intelligible when shadows are used, because shadows present a big part on our view of the world on defining objects and distances.

3.1.4.1 Shadow Mapping

Shadow mapping, also called as projective shadowing, was introduced by Lance Williams in his 1978 paper "Casting Curved Shadows on Curved Surfaces" [Williams, 1978]. Shadow mapping is a technique where shadows are created by testing whether a pixel is visible from the light source's view. The decision, whether to shadow or not is based on comparing the z-buffer value of the light source's view. A Shadow map is constructed by rendering the scene from the light's point of view. From this rendering, the depth map is saved usually to a texture the size of which affects (as in 3.1.3) the final fidelity of the shadow. After the shadow map is stored, a normal rendering of the view is done from the camera's point of view.

![Figure 3. Shadow mapped scene with two cubes casting a shadow.](image)
The biggest drawback of shadow mapping is that the texture size decides the quality and aliasing might be present in shadows created with shadow mapping. In addition, the shadow maps have always hard edges, and with small texture size they can be irritating in the 3D scene. Several solutions have been proposed through the years to provide real like soft shadows with shadow maps (see [Hasenfratz et al., 2003]).

There have been various shadowing methods in virtual environments before and after William's paper. One of these, also used in real-time applications, is a technique called shadow volumes or stencil shadowing. This method was first proposed by Franklin Crow in 1977 [Crow, 1977]. The main advantage of shadow volumes against shadow mapping is that they are precise to the pixel, though shadow mapping is usually faster.

3.1.5. Texture Mapping

Texture mapping was made famous in 3D graphics applications by Edwin Catmull in the 1970s [Catmull, 1974]. In texture mapping a surface texture (a bitmap or a raster image) is added to a 3D model or a surface to add detail and definition.

3.1.5.1 Normal Mapping

In 3D graphics, the more detail polygon meshes have the more realistic they look. Normal mapping is a technique where details are added to objects without increasing the polygon count. The idea for normal mapping was introduced by Krishnamurthy and Levoy in their 1996 work "Fitting Smooth Surfaces to Dense Polygon Meshes" [Krishnamurthy and Levoy, 1996]. They presented the idea to take geometric details of a high polygon model and convert them into tensor product B-spline surface patches with accompanying displacement maps. Normal mapping is a variant to bump mapping that was introduced by James Blinn in 1978 [Blinn, 1978].

In normal mapping the lighting of bumps and dents in an object is faked to make it look more real. Usually a normal map is an RGB image of a more detailed version of the object where every channel in the image is X, Y, or Z coordinate that together correspond to the surface normal in 3D space. The actual object model, in the 3D scene, will not have bumps and dents, it is the lighting calculation that is done in the pixel shader with the normal map that adds the detailed and bumpy look. Normal mapping can give a good appearance of a complex surface with a low polygon count model. On the left, on Figure 4, the cube's colors have been calculated with a texture and with a normal map. The cube on the right has only the colorful texture.
Figure 4. The same texture used on a cube with normal mapping (left) and without normal mapping.

Even though normal maps are usually created from detailed polygon meshes, they do not have to be more detailed, normal maps can be used only to add a 3D feel to 2D textures and objects as shown in Figure 4. With the availability of shaders, in graphics hardware, normal mapping has become widely used in real-time computer graphics. Today’s image processing programs like Adobe Photoshop [Adobe, 2011] and Gimp [Gimp, 2011] can create a normal map with a desired depth for any texture.

As the graphics computing power advances, more advanced methods are coming available in real-time graphics. These include techniques such as parallax mapping [Kaneko et al., 2001] and displacement mapping [Cook, 1984].

3.1.5.2 Advanced Bump Mapping
Parallax mapping (also referred to as offset mapping and virtual displacement mapping) is a more advanced method compared to bump mapping and normal mapping that adds even more depth to models in a 3D scene. Parallax mapping was introduced by Kaneko et al. in 2001 with their article “Detailed Shape Representation with Parallax Mapping” [Kaneko et al., 2001]. Parallax mapping adds the capability to represent the motion parallax effect that is missing in previously explained techniques. It uses a per-pixel image distortion process to represent detailed shape on a single
polygon. The actual texture is not distorted, but the texture coordinates are shifted for each drawn pixel as the texture is mapped to the polygon.

Displacement mapping is an advanced mapping technique that makes genuinely rough surfaces by changing the geometric position of vertices. It does not “fake” the rough bumpy surface effect like the previous techniques. The nature of the technique makes it still quite heavy for real-time rendering.

3.2. Computer Generated Haptics

Computer generated haptic feedback must feel real to users, i.e. solid objects must feel solid and the feedback must be continuous without unintended vibrations. Haptic rendering is the process where feedback is computationally generated to the user. When a user touches a haptics object, the proxy, which is moved by the haptic device, is pushed back from the inside of the object and the force feedback is generated to perceive the phenomenon of touching a solid object.

1 kHz update rate must be provided by the applications haptic loop to offer users a realistic haptic feedback. With an update rate below 1 kHz, there could be some vibrations and oscillations felt by the user. Therefore, a 1 kHz processing loop has become the standard in haptic application programming interfaces. The market offers many haptic devices with different capabilities, strengths, and prices.

3.2.1. Haptic Rendering

Haptic rendering is a method where computational forces are displayed to the user by making him or her feel a tactual perception. With haptic rendering the user gets the sensation of touching and interacting with physical objects. Haptic rendering algorithm is responsible for computing the forces and generating the sense of touch in real time from a haptic interface that is interacting with a mathematical model of an object.

When haptic interaction is done by users, the haptic interface is pushed “through” a modeled object. The force is calculated by different algorithms from this penetration. There are methods that do a one-to-one mapping of position in space to force and there are methods to do a constraint-based mapping. Haptic renderers vary with different haptic toolkits. Though, constraint-based algorithms for haptic displays are nowadays commonly used.

3.2.1.1 A Constraint-based God-object Rendering Algorithm

Zilles and Salisbury introduced a constraint-based god-object method for haptics rendering [Zilles and Salisbury, 2001] that would remove the drawbacks of one-to-one mapping algorithms. The drawbacks in these volume methods were, as stated by Zilles and Salisbury:

1. It is often unclear which piece of internal volume should be associated with which surface.
2. Force discontinuities can be encountered when traversing volume boundaries.

3. Small and thin objects do not have the internal volume required to generate convincing constraint forces.

The presented god-object rendering algorithm functions better with these drawbacks. Haptic interface point cannot be prevented from penetrating virtual objects when touching them. A god-object is an additional variable that presents the virtual location of the haptic interface. In free space, the haptic interface point and the god-object are in the same position, but when the haptic interface moves into an object the god-object remains on the surface. It will not penetrate the virtual objects and it presents the point where the haptic interface would be with infinitely stiff objects. The god-object location is computed to be the point that's distance is the minimum surface location to the haptic interface point. This method eases the calculation of force direction compared to volume based one-to-one mapping algorithms.

The constraint-based god-object method works well with static and immovable objects, but when the scene has dynamic and physically moving objects it has a serious drawback – the god-object point can end up inside a solid object. This happens due to the tradition of modelling objects by only their surfaces. Even though the haptic loop runs at 1 kHz, when the modelled object and the god-object move to the opposite direction and should collide the god-object goes through the surface of the object. Furthermore, because of the small numerical errors, polygons of modelled objects that share a common edge often contain gaps and the god-object point can “fall” into these gaps, into solid objects. As an enhancement to the god-object renderer, and to resolve these drawbacks, a new rendering method was introduced by Ruspini et al, [1997].

3.2.1.2 The “Ruspini” Rendering Algorithm

To prevent the haptic interface point from going through surfaces and objects Ruspini et al. presented a massless spherical shape virtual proxy based rendering method for haptic rendering. In this method, the radius of the proxy is made large enough in the virtual scene not to behave badly with triangular mesh gaps (see Figure 5, on the right) and dynamic moving objects. Because the Ruspini renderer is a constraint-based method like the god-object, it maintains two positions: physical position and the proxy position as seen in Figure 5. The rendering algorithm has been referred to as “Ruspini renderer” by the name of its inventor.
3.2.2. Force Feedback Haptic Devices

Today, there are many haptic devices available that are capable of producing various degrees-of-freedom (DOF) high-fidelity force feedback. These haptic devices act as a haptic interface with what users interact with the virtual scene.

Sensible Technologies [Sensible, 2011] has the PHANTOM product line with many different haptic devices such as the 6DOF Phantom Premium and the 3DOF PHANTOM OMNI (see Figure 6). With their product line they can offer different haptic devices that can meet the expectations of research and commercial customers. In addition, Force Dimension offers haptic devices for mainly research purposes. Their line up consists of well-known Omega devices (see Figure 6) with three, six, and seven degrees-of-freedom capabilities. Besides the Omega series, Force Dimensions has Delta series with larger workspace and a device called Sigma with unique 7 active degrees-of-freedom. French based haptic company Haption [Haption, 2011] designs, manufactures and sells haptic devices for industrial and academic use. In 2008 Novint Technologies [Novint Technologies, 2011] introduced a significant competitor for other manufactures with its low-price-range Novint Falcon (see Figure 7).
The Novint Falcon device by Novint Technologies [Novint Technologies, 2011] is the first three degrees-of-freedom (3DOF) capable force feedback device designed for consumer market with a fairly low price. It offers three-dimensional touch workspace of 10 centimeters to each direction and up to 10 newtons of force capabilities with a position resolution of 400 dpi. It has been studied that the users tend to use less than 5 newtons of force when exploring virtual environments, because of this the force capabilities of the Novint Falcon are enough to make objects feel real and meet user expectations.
As default, the Novint Falcon device has as a changeable grip with four buttons. Different grips, like a gun grip, are provided to modify the experience to meet for example first person shooter games. Novint Falcon has a default SDK (Software Development Kit) with what it is possible to implement own applications that utilize the Novint Falcon device. In addition, different haptic APIs and toolkits support Novint Falcon and with its low price it has become quite popular in the haptics research community.

3.3. Algorithms and Techniques
During the last decades, many good algorithms and techniques have been published to speed up the development of 3D graphics. Some of these techniques have become standard in for example computer games, and without the usage of some of these visualization techniques 3D virtual worlds would seem outdated.

3.3.1. Constructive Solid Geometry (CSG)
Constructive solid geometry (CSG) is a solid modelling technique where complex objects are designed, and built, from simple primitive objects [Requicha and Voelcker, 1982]. Objects are usually built up from primitives that are constructed with a binary tree structure, where leaves represent primitives and nodes represent operations. The operations that are used to combine primitives are: union, intersection, and difference. In addition, the primitives are usually simplistic shapes such as cube, cylinder, and sphere.

Constructive solid geometry is often used in solid modelling in 3D computer graphics and CAD. CSG is useful in situations where simplicity of objects and mathematical precision is desired.
In addition, CSG can be used in games for example for level editing and destructing of environments.

3.3.2. Marching Cubes

Marching cubes is a popular 3D surface construction algorithm presented by W. Lorensen and H. Cline [Lorensen and Cline, 1987] at SIGGRAPH in 1987. Their publication "Marching Cubes: A High Resolution 3D Surface Construction Algorithm" is widely cited and has been the base in many studies in recent years surrounding 3D surface construction (see e.g. [Nielson, 2004; Kazhdan et al., 2007]).

The algorithm generates a solid triangle mesh from 3D scalar data using a divide-and-conquer approach. Marching cubes initial purpose was to help presenting medical data from, for example, computed tomography and magnetic resonance images. In the usual cases the 3D scalar field data is static. From the data, the surface is located from user-specified value (sometimes called an isovalue) and the triangles are created corresponding to that surface. Then, to achieve smooth shading, vertex normals are calculated often with gradient data.

As the name of the algorithm states, marching cubes involves "marching" (going) through a cube, created with eight values in the corners. These kind of three dimensional cubes can be called voxels, 3D pixels in some cases.

Smooth shading (see [Blinn, 1977]) is desired basically every time in today's real-time graphics. By calculating gradients, smooth shading can be added to marching cubes generated objects. Smooth shading hides rough edges that might come from the surface construction algorithm if the used scalar data set is not large.

3.4. Libraries and Tools

There are many freely available libraries and APIs that will help in creating a multimodal virtual environment. Without the available libraries the amount of work needed for the implementation part would be more greater.

3.4.1. LibSDL

Simple DirectMedia Layer [libSDL, 2011] is a cross-platform multimedia library designed to provide low-level access to audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer. It is popular and, for example, used in various games. It supports many operating systems from desktops to game consoles and mobile handsets. The portability is the key here. SDL is written in C, but works with C++ natively. SDL is distributed under GNU LGPL version 2. This license allows you to use SDL freely in commercial programs as long as you link with the dynamic library.
3.4.2. Haptic Application Programming Interfaces

There are some open source haptic application programming interfaces available. One of these is an open source haptics library called “HAPI” [H3D API, 2011]. HAPI is a part of a cross-platform open source haptic development platform H3DAPI that uses open standards like OpenGL and X3D with scene graph design. H3DAPI is written in C++ and offers support for multiple haptic devices. The HAPI is a good choice to be used in some applications because it is fully separate from the H3DAPI and can be used alone.

HAPI offers comprehensive C++ base classes for haptic handling with a haptic loop that is running at the speed of 1 kHz. New classes, to further enhance the functionality, can be done easily by inheriting some of the HAPI's base classes. The usage of HAPI with different haptics devices is quite easy because it supports many different devices and no additional program code is needed when changing the haptic device.

In addition to H3DAPI, there are other popular haptic application programming interfaces available. CHAI 3D [Force Dimension, 2011] is an open source set of C++ libraries that can be used in real-time simulation. It is mainly designed for education and research purposes. Furthermore, OpenHaptics [Sensible, 2011] is a toolkit for haptic development from SensAble. Like the H3DAPI and CHAI 3D it supports several commercially available force feedback devices and can be programmed with C++.

3.4.3. Physics Simulation in Virtual Environment

Computational Physics simulation has been used in real-time graphics for several years. There are several open source physics simulation libraries that support 2D, 3D or both for rigid, soft or both bodies. Bullet physics [Bullet, 2011] is a well regarded and extensive library for collision detection, rigid body, and soft body dynamics. It is an open source library and free for commercial use under the Zlip license.

In addition, there are other open source physics libraries that offer somewhat the same functionalities as Bullet Physics. To name a few there are ODE (Open Dynamics Engine) [ODE, 2011] and Box2D [Box2D, 2011]. Many of the available libraries should be good for simulation purposes and only the users preference can be the factor on selecting the right one. Though, simulation environments are usually created in three dimensions and therefore need a 3D capable physics library.

3.4.4. XML-File Support

The eXtensible Markup Language (XML) is designed to transport and store data. XML is designed to be self-descriptive and therefore it is a good choice to be used in loadable or storable settings. XML tags are defined by the implementor. This makes it usable in different environments. Xerces
[Xerces, 2011] is a freely available library to be used in XML-file parsing. It is usable with C++, C, and Java programming languages.

3.4.5. Auditory Feedback in Virtual Environment

Speech synthesis is a way to guide the user in a virtual environment or give feedback. In addition, it is a way to get visually impaired users involved as well. There are some freely available speech synthesis libraries. One of these is University of Edinburgh's Festival Speech Synthesis Systems [Festvox, 2011] that is written in C++. It runs on multiple platforms offering black box text to speech output.
4. Implemented Multimodal Applications

This chapter presents the actual applications and the techniques used in their implementation. The techniques and tools were presented in more detail in chapter three.

All three applications are intended for elementary school use. The applications are named after their features: the Density application, the Leverage application, and the 3D Construction application. The applications were designed after receiving requests for the particular application.

The features that were requested for the Density application were comparison of chemical elements and their attributes in the air and in liquid. At least the most common chemical elements in the periodic table should be accessible. Furthermore, the density of the different elements would be felt with a haptic device. The weight for the particular chemical element would be calculated from its density and from the volume of the object that is hold up with the haptic device. The volume for the object would be chosen to be the same as in the few sample chemical objects in the class room.

The Leverage application's functionality should represent the leverage with modifiable weight and movable fulcrum. Generally all the physics classrooms have a physical fulcrum set-up with what it is possible to study balance equation. The computer generated set-up should have countable values present. In addition, the Leverage application could be used to test own calculations by changing the mass of the burden and the position of the fulcrum.

The context of the 3D Construction application was not that clear in the beginning, and different school utilization like mathematics and art were suggested for its target use. Though, it should support and improve ones spatial perception in three-dimensional space. Furthermore, the application should teach the combining of primitive objects to a more complex objects.

The base for the Density and the Leverage application was designed in the Spring of 2008, and the first fully functional versions (see Figure 8) were tested in school context in 2009. The 3D Construction application was designed in the spring of 2009, and the prototype was tested during the fall of the same year. The application was further designed and developed in the Spring of 2010, and the final school testings were done in the Fall of 2010. In addition, during 2010 and 2011, all of the applications have had some visual enhancements and general fixes. The applications, presented in this thesis are the newest versions, where the author has seen that a significant graphical improvement makes a difference.
Figure 8. First fully functional versions of the Density application and the Leverage application.

4.1. The Application Framework and the Tools
The applications are programmed with C++. In addition, all applications load an XML-file in the initialization process. The Xerces-C++ library was used for XML-file parsing for reading and writing of settings and attributes. For different applications the file offers different things. The purpose for this XML-file was that the users or/and the teachers could easily modify the applications' and attributes' behaviour.

The graphics are done with OpenGL version 2.1. OpenGL was an obvious choice because Linux was the main development platform and the applications should work on both Linux and Windows PCs. All the objects are shaded and use vertex and fragment shader programs that are implemented with OpenGL shading language (GLSL). Texture mapping was used here and there to improve the looks of the environments. In addition, normal mapping was used to generate bumpiness in objects. Furthermore, a water surface was created with normal map and a du/dv map.

Shadow mapping was tested during the development process, but it was not fully integrated into the final application framework. Realistic shadows, give more realm to the environment. Though you should not overdo shadowing in virtual environments. Virtual environments that are used mainly in teaching and research should be clear and bright, so that the users can see clearly the objects and interact with them.

To ease the porting between different operating systems, Simple Directmedia Layer (libSDL) library was used. The library was mainly used for creation of window, selecting of contexts and loading of RGB textures.

With the Density and the Leverage applications, physics are in important role. Therefore a rigid body physics library "Bullet" was integrated into the framework. The physics library is tightly integrated and when you are creating new objects with the API, physical attributes (like mass) are given as a parameter for the constructor.
HAPI haptic rendering library from the H3DAPI was used in the framework. It was chosen because it is open source and fully functional with many haptic devices and has different haptic rendering algorithms available. With its adequacy to work with physics based moving objects and with vast amount of vertices (in the 3D Construction application), the Ruspini renderer was chosen to be used in the virtual environment applications. Virtual sphere proxy does not penetrate moving objects or fall into minimal holes in the vertices (as explained in chapter 3.2.1 and figure 5).

University of Edinburgh’s Festival Speech Synthesis Systems was integrated into the framework. The reason for the speech synthesis to be present was that at some point in the near future the applications could be tested with visually impaired pupils. Although, the speech synthesis system was only briefly tested and used to output selected primitive in the 3D Construction application and to output values in the Density and the Leverage application, it could be used more efficiently. The speech synthesis was not enabled during the school testing of the applications.

In all applications, the view is roughly designed as a cube. We have at least a back wall that confines the area where we can move and interact with objects. In addition, in the applications all objects are visible all the time and the view does not change. Only in the 3D Construction application some objects could be positioned behind other objects, and are only visible by rotating the "table" or viewed from the up-view. With this design, the view is kept simple and the haptic device operates in the same space all the time.

The workspace of the haptic device is not a cube, at least with the Novint Falcon. It has its maximum axis width when the grip is in the middle in other axes. This workspace somewhat creates the space where all the objects should be positioned, it is basically an octahedron shaped space.

4.2. The Density Application

Comparison of chemical elements and their attributes offers challenges because of the nature of the chemicals. There are only few elements in the periodic table that can be handled in a classroom, because of their expensiveness, hazardousness, or stability. Fortunately, in a virtual simulation we can mimic the attributes and behaviour of elements to that exactness that is feasible.

The first prototype of the Density application did not have liquids implemented, it had only the periodic table of elements and comparison of different densities with the haptic device was possible. The request from the elementary school teachers was that the user interface would be easy to understand and easy and fast to use. When we designed the user interface together, we came to a conclusion that a graphical presentation of the periodic table would be the easiest to understand and use. The navigation of objects is done with the directional keybad, and therefore it is as easy as possible and as fast as possible.
The application presents the periodic table of elements, where users can choose the chemical element they wish. They can study its shown attributes and its density in the air by the Novint Falcon haptic device. The attributes of a chemical element are shown on the left bottom corner of the application (see Figure 9). With the haptic device, the users can control a cube (torquise coloured cube in Figure 9) that simulates the attributes of the selected chemical element.

The cube can be grabbed by pressing a Novint Falcon's button. When grabbed, the cube can be moved around the workspace with the haptic device, and the weight of the chemical element is felt through a downward force. By pressing the same button again, the cube is released. It falls or ascends depending of its attributes compared to air.

In addition, the users can compare the density of chemical elements by dropping the cube to a liquid. Different liquids can be added to the application by the users. The cube, affected by different forces, behaves according to its density and the density around it. It sinks to the bottom of the pool or rises to the surface affected by buoyancy, or it is partially submerged depending on the densities of the selected liquid and the chemical element. Chemical elements that have smaller density than the air above sea level will ascend from the view.

![Density application with normal mapped periodic table and water effects.](image)

Figure 9. Density application with normal mapped periodic table and water effects.
The force feedback is felt through the haptic device like the actual weight of the selected element would be on top of one's hand. The weight is generated by sending y-directional force, in newtons, to the Novint Falcon device. The force is calculated from the current element's density, cube's volume, and the gravitation. In addition, the volume of the cube can be changed in the XML-file. This is helpful when comparing neighboring elements that have almost identical densities or the examination is done on a certain region in the periodic table, and the weight difference is not so noticeable on the haptic device. Even though the force capabilities of the haptic device is good, the density range in the periodic table is wide, when noticeable force feedback is desired from alkali metals to actinides. Through lessons, the volume for the cube was selected so that the first alkali metal Lithium generated weight of half a newton with its 0.5 g/cm³ density. From there on, the chemical elements and the difference in the weight was noticeable up to Novint Falcon's limit of a 10 newtons which was met with a chemical element that has a density of 10 g/cm³ or above. After going above that value, the device gave its maximum force.

Visual feedback complements simulation and demonstrates buoyancy. When observing densities of different chemical elements by touch, users can study the density of every chemical element, the attributes of which have been implemented in the instrument. This offers an advantage over ordinary chemical element teaching.

The application is constructed that way, that users can freely add new liquids with their attributes and update chemical elements with XML-file. The attributes of a liquid is shown on the right-bottom-corner of the application view (see Figure 9). Furthermore, the number of liquids is not restricted. In addition, the color of the liquid element changes when the liquid changes. The color value is computed from the density and the darker the grey gets, the greater the density value gets.

Shaders, presented in Chapter 3, make possible various realistic and visually pleasing effects. In the Density application, realistic looking lively water was implemented with planar reflection, normal map and du/dv map. A du/dv map is a derivate of a normal map and quite similar in the way that it stores directional information in a texture.

The application is robust and self-explanatory and therefore it is a good enhancement to chemistry teaching in elementary school. It offers all the periodic table elements that cannot be studied in real environments, because of their expensive value or composition.
4.3. The Leverage Application

You might have tried to balance a metallic leverage in your natural science studies and have probably succeeded in doing so. With computer simulation we can add value to that process. It stabilizes the environment and can give numerical output from the balancing effort. The functionality in the Leverage application has been close to the same in its whole iteration cycle. Only the visual aspect has had changes.

The goal in this application is to reach equilibrium with a changeable length leverage which is affected by the weight of a cubic object and by the force generated by the user with the haptic device. The leverage is positioned to the centre of the view (see Figure 10) and it is accessible with haptic device from the left and right sides of the fulcrum. The burden with a changeable mass is, in the starting position, positioned to the left. Both, the fulcrum and the burden can be freely positioned in x-axis.

Figure 10. Leverage application with changing values on the right.
Leverage is balanced by generating as much force to the other side of the leverage as the weight generates to the other side of the fulcrum. The current mass of the cubic weight can be changed by keyboards plus and minus keys. The application does not force the user to put the weight (force) on top of the leverage, the force can be generated to the bottom of the leverage as well.

Users can also freely select the position where they put the weight in. They can interactively choose in which direction and to what position they produce a force by hand. The force input is gathered from the haptic device when a user pushes the virtual proxy, which is moved by the haptic device, against a haptic object, the leverage. The force input from the haptic device and the virtual forces are calculated, and as a result new positions are set for the objects. In addition, the position of the fulcrum can be freely selected causing the length of the lever arm to vary. When changing the positions of the weight or the fulcrum, physics simulation is put to a halt with one of the Novint Falcon's buttons to avoid disturbance. The simulation can be continued using the same button.

The values used in the simulation is shown on the bottom-right-corner of the view (see Figure 10). With the values, the balance equation can be studied later on.

4.4. The Geometric Construction Application

The third application is a 3D geometric object construction application. It is the most complex of these applications and its usefulness might be in geometry studies or art.

The application uses a 3 DOF haptic device as its primary input output mechanism. Objects can and should be felt with the Falcon device and the users should not settle for the obvious visual feedback. Even though the constructed objects and the example objects can be rotated and viewed from different angles, there can be situations where all the surface shapes are not visible and the haptic feedback is the only available method to explore the objects thoroughly.

Objects are constructed by freely combining three primitive objects, ellipsoids, cuboids, and cylinders with Boolean operations. The available Boolean operations are Boolean union and Boolean difference. At one point, of the application's lifespan, Boolean intersection was also present. However, the decision was made to remove it totally to make the application more simplistic and unerrorprone for school context. With pilot testing, the Boolean intersection was seen as too complicated and unnecessary option for the purposes of the application.

The active primitive object can be changed with keyboard buttons left and right arrow buttons and it is made visible and modifiable within the workspace with a haptic device's button. The selectable primitive objects can be seen on the bottom of the application view in Figure 11. In the workspace, within haptic device's touch space, primitives can be positioned and modeled to desired shapes before making them solid and touchable. Furthermore, the 3D Construction instrument supports copying modified primitives and deleting previous primitives from the constructed solid model.
Combining of simple geometric objects with Boolean operations procedurally is a technique used in solid modeling called constructive solid geometry (CSG) presented in chapter three. This technique was the bottom idea when the designing process was started for this application. In Figure 11, the constructed object can be seen on the centre. Furthermore, the current object (in Figure 11) is created by a union of three ellipsoids and one cuboid, and difference of two cuboids.

Complex objects are generated by combining scalar fields of the primitives and calculating polygonal mesh for the resulting surface. The triangle mesh is generated with an algorithm by Lorensen and Cline called marching cubes that was presented in more detail in chapter three. It generates triangles that present the base for the graphical and haptic visualization of the constructed objects.

Figure 11. The 3D Construction application with a constructed object on the centre.

The graphical user interface of the application consists of three different views: main view, right-up-corner view, and the example view. The main view is where the complex surface is generated from primitives and where the example objects can be examined. The view in the right-up-corner presents the same view as the main view, only seen from the y-axel. The third view, on right-bottom-corner, holds the example object or the constructed object when the example object is in examination mode. Objects can be rotated around the y-axel and constructed freely. In addition,
with the application users can load their previous creations as example objects. This is done by saving constructed objects' primitives and combining Boolean operations in XML-format.

In addition, the application offers also a mode where users can select before-hand made objects and try to recreate them. All the created objects are saved in XML-format, so users can add countless model objects to the application. Later on, all the operations that were used in the creation can be compared from the XML-files.

The 3D Construction application offers plenty of functions and that is one the reasons that it take some time to learn to use it. Prior to the study at school, some comments on the pilot tests were that it is a bit CAD-like and therefore a bit complex.
5. Evaluation of Applications in School Context

As stated before, a study in a school context was made with the applications in use by Erika Tanhua-Piirainen et al. and a conference article was written from that study [Tanhua-Piirainen et al., 2010]. The aim of the study was to find out pupils' and teachers' experiences about new haptic augmented computer tools. This section evaluates the applications based on the findings in the school context study.

The study at Ylöjärvi elementary school was made in two phases. In the first phase, the Density and the Leverage applications based study was conducted in a real teaching environment with students' teacher involved. The second phase was done later on and it only tested the 3D Construction application in a more structured test situation. In both of the studies, a researcher was present to analyze and note on the behavior of the applications and feedback of the students. Both studies were evaluated with qualitative methods.

5.1. Haptic Applications for Physics

The focus group for the first study consisted of 8th grade pupils in Ylöjärvi elementary school. Two special themes: density and leverage were selected as the base for the haptic tools for this study. Even though, the study was made with only 8th grade pupils, in the design phase the idea was that the applications could offer different features for different age groups was kept in mind. More features could be used by a higher level students. The main research question in the first study was the following: What was the learning experience like, when haptic interface technology was utilized in the physics classroom?

Observations, with the applications, were made during two physics lessons. At the first lesson the Density application was used. With it, the periodic table of chemical elements can be explored to study different matters and their properties. Elements in the periodic table are not easy to study with traditional methods in a classroom environment. With virtual environment all unstable, toxic or radioactive elements are comparable and available to study in groups or by oneself. In the second haptic enhanced lesson, the Density application was used. The application offers studying of the leverage and with the balance equation. Both of the applications were used in pairs in the study.
5.1.1. Physics Study With Virtual Environments

Observations, with the applications, were made during two physics lessons. After the first lesson, the pupils answered to some questions concerning the experience:

1. What was it like to use a haptic application?
2. Which features were easy or challenging to use in the applications?
3. How could the applications be developed further?
4. Which other school subjects the haptic device could be suitable for and why?
5. Other comments concerning the lesson?

After the second haptic lesson four pupils were discussed with in more detail, and the whole class was interviewed briefly. In addition, pupils' learning experience was reflected one week after the classes if the haptic applications helped them to understand the physical subjects that were studied better. The physics teacher wrote a diary before the study and after both of the haptic lessons. Furthermore, the researcher who was present in the teaching situation discussed with the teacher after the lessons.

Before using the haptic applications, most of the pupils were expecting the lesson to be “interesting”, “different” or “something little strange”. The situation, at the classroom, was a bit uneasy and noisy because there was only one haptic device in use. Therefore, turns had to be taken even though two pupils were using the applications at the same time.

The most common answer from pupils, when asked about the pros in the Leverage application, was that the felt experience of the forces added value to learning when compared to traditional study instruments. The Leverage application was found to be realistic and very concrete. Before using the Leverage application, the teacher believed that the pupils understood the leverage phenomenon, because they were able to solve related exercises. But as the teacher wrote in her diary: “The haptic trial convinced me surprisingly, that only using it the subject was understood. When the lever arm expanded the torque decreased. A new, appropriate instrument was found to learn the relationship between forces when lever balance was achieved.”

The Density application was easy to use and the possibility to explore different materials with the haptic application was appreciated by the pupils. In addition, the feature of dropping the cube into a liquid was liked by the pupils. The pupils were enthusiastic when experiencing heavy elements and they also compared their experiences with the others after their turn. By their own words: “With the help of the haptic device it was easier to understand the different densities, as the masses of the different elements were possible to be felt in one's hand.”, “The application was good because it was possible to compare densities and the differences of the densities were easy to notice.”.
Overall the two physics simulation applications performed smoothly, but certainly there were few challenges. Some difficulties were observed in the Leverage application with its visualization of the force generated by the burden comparing to the force generated by the user in the opposite sides of the fulcrum. With the simulation environment the values of the forces changed quite rapidly. In addition, a small drawback with the Density was that the haptic device generated some vibration with very heavy elements. Though, this is due to the capabilities of the Novint Falcon device.

5.2. 3D Construction Application for Geometry

The second study that used the 3D Construction application was carried out with 5th grade pupils at Ylöjärvi elementary school. This study was focused on pupils own experiences as well. The idea was to analyse two different set-ups: a) one device for a pair of pupils, b) both of the users having their own devices. Four pairs of pupils participated in tests during two school days. The pupils filled a short questionnaire after the test and the application saved log data from all the tests that would be analysed.

The 3D Construction application was first demonstrated by one of the researchers. All the features were explained and showed in practice. This was essential, because the application offers many features and different buttons to be used. Pilot tests, at the usability laboratory, prior going to school, suggested that the application might be a bit CAD-like, and therefore somewhat difficult to use at first. The pupils started by practising together freely the features that the application offered. At this section they were instructed by the researchers, if needed, and they could make questions.

In the actual test, the users needed to select, a before created, model from the menu and build a similar model themselves. The first two tasks to build a model were done with a one Novint Falcon haptics device connected. The first object was built by the other pupil and the other pupil was instructing and the second task the other way around. Only in the third task two Novint Falcons were used, one for each. The keyboard, what was used for some of the input, was available for both pupils during all the three tasks.

In the tasks, all of the pairs collaborated well, it did not matter if one or two devices were in use. When both pupils had the haptic device in use, they first negotiated and usually used the devices in turns. Even though, both of the devices could have been used simultaneously. Overall the collaboration went well and the pupils seemed focused and enthusiastic, only half of them preferred the two-device set-up to the single-device use situation when asked after the test. The explanation for that was that “there is a need to agree all the time” and “the construction will be difficult” or “proceed poorly”. In addition, the pupils who preferred the two-device collaboration explained that it “was fun like this” and “sometimes it makes us laugh”.

Overall the 3D Construction application proved to be an immersive application for three-dimensional geometry. Though, the usage of many buttons and support for several functionality
that the application offered was a bit demanding at first, the pupils usually remembered them well and they also had small cheat sheet to help to remember. Collaboration was effective and in many cases if one of the pupils forgot something the other remembered it. For the pupils, the most challenging feature was three-dimensional primitive modelling, where basic primitives sphere, cube, and elliptic cylinder could be freely shaped into ellipsoid, cuboid, and elliptic cylinder respectively. The shaping was bound to the centre position of the current primitive and to the position of the haptic proxy, and was a bit hard to comprehend.

5.3. Results From the Studies
The teachers agreed that the haptics enabled virtual environments give additional possibilities to understand the learning content and thus to take into account the different learners in the class. In addition, the pupils seemed enthusiastic and interested on using multimodal virtual environments. In the study, the experienced usefulness of the haptic applications and the attitudes of the teacher and the pupils to these kinds of learning instruments were investigated. Pupils' learning was not quantitatively measured by no any means.

The researchers felt that the implemented applications all behaved well and the tests were carried out without any noticeably bugs or problems.
6. Discussion

As stated in the previous Chapter, the applications were well-liked in the Ylöjärvi elementary school in Finland. The applications are usable in their current stage, but with small modifications they could be better. Better looks and more usability could improve the user experience.

6.1. Comparison to Haptics Enabled Application Programming Interfaces

There are currently only few versatile application programming interfaces for graphics and haptics enabled virtual environments. H3DAMI [H3DAMI, 2011] and CHAI3D [Force Dimension, 2011] are one of the most popular applications programming interfaces. What makes H3DAMI great is its feasibility for non-programmers or beginners. It is possible to create complex scenes with only XML-style language X3D. With small Python scripts more advanced scenes are possible. These are the reasons why H3DAMI can be recommended for home use besides research. CHAI 3D is an open source set of C++ libraries for computer visualization, haptics, and interactive real-time simulation. It is best suitable for education and research purposes because it is tied to C++ programming only. It offers extensions, for example, for ODE physics [ODE, 2011] that help utilizing physics on scenes.

The implemented application programming interface and applications are not meant for non-programmers. Even though the applications offer some modifications through XML-file, it is not possible to fully change the behavior of the scenes with it. The Bullet [Bullet, 2011] physics engine is tightly integrated to the application programming interface and new physics obedient objects can be easily added. To get objects visible vertex and fragment shaders need to be connected to the object. In addition, color or texture(s) can be attached to an object. All these can be done by inheriting the object base class and adding extra properties if needed. Speech synthesis was not that tightly integrated, it was only studied briefly – how it could be integrated and used if needed in the future. The creation process for the application programming interface was to learn by doing and to try to stitch the different pieces together for a multimodal virtual environment. The created application programming interface was meant to be the best suited for these three applications. It was not meant to be all-round for different purposes and different scenes. It would have taken quite a lot more time to implement more all-round application programming interface. The main libraries that were used are OpenGL for graphics, HAPI for haptics, and Bullet for physics.

6.2. Comparison of the Applications in Their Study Environment

The applications were studied in school context and analysed how they fit to support education. As stated in Chapter 5, the teachers agreed that the haptics enabled virtual environments give additional possibilities to understand the learning content and thus to take into account the different learners in the class. The study by Erika Tanhua-Piiroinen et al. [2010] was qualitative and
therefore the actual learning process cannot be properly evaluated. The study by Wiebe et al. [2009] was quantitative and their finding was that there was no improvement to learning by using haptics enabled virtual environment. Even though it perhaps does not improve learning, it makes it more fun and can enthuse pupils to study more. Multimodal virtual environments can support and bring new ways for teachers to teach and motivate students.

6.3. Further Development of Haptics

One suggestion to further enhance the Density application would be to implement viscosity to the liquid. Viscosity effect, or force, cannot be computed the traditional way, it just does not suit the haptic loop. The reason for this is that the viscosity force strengthens when moving faster and the haptic device starts trembling. There have been some suggestions on utilizing friction to mimic the behaviour of viscosity (see e.g. [Ruspini et al., 2007]). At the time of the study, H3DAPiS HAPI [H3DAPi, 2011] did not support usable viscosity effect. Some other haptics enabled application programming interfaces like Chai 3D [Force Dimension, 2011] lists viscosity as its effect, but it has not been tested by the author. The viscosity effect could be implemented to HAPI to add one more study property to the Density application.

The Leverage application suffers a bit from the three-dimensional set-up. The user is practically using it in two dimensions, or at least wants to use it in two dimension, but the application is in three dimensions. Currently there is an invisible, to eye, haptic wall that prevents the haptic proxy falling backwards from the leverage. Though, it does not help the falling situation nearest to the viewer. The applications visualization could stay in its current form in three dimensions but the haptics could be done with a two dimensional approach.

The 3D Construction application suffers from the vast amount of vertices that it should handle. This could be handled in the application more efficiently with a faster surface construction algorithm, like the one presented by Kazhdan et al. [Kazhdan et al., 2007], that creates less polygons with less computing time for somewhat the same object precision. In addition, more efficient and better suitable tree structure could be used in haptic rendering. Furthermore, haptic rendering could be done with a different approach for example not generating object's haptic representation from vertices.

6.4. Enhancing the Visualizations

There are a lot that can be done to enhance the visualization of the applications. Though one has to keep in mind that the applications are meant for study purposes and, therefore, a more clinical style scene visualization and clarity are essential.

In the Density application, even though the water effect looks good at its current state, it is not mimicking the natural visual behaviour of water. Refraction would make the water more real-life
like. In addition, a bit of fog should be implemented that would blur further objects under the water.

High dynamic range rendering offers the possibility to do lighting calculations on a larger dynamic range. This method makes scenes more realistic and smaller details are more visible in any range of light. High dynamic range (HDR) rendering has become more common year by year in games since it was made possible in OpenGL 1.4 and DirectX 9.0 Shader Model 2.0. As a next step the author would definitely implement HDR to the application framework.

Displacement mapping effects could be used to add more dimensional feel to objects. In addition, currently used normal mapping could be used in more objects. The advanced methods, presented in chapter three, like parallax mapping, could be used in many objects. Even though, the environment should be a bit clinical looking, some textures could be used for planes (floors, walls).

Shadow mapping was presented in chapter three and tested during the development process of the applications. It could give a nice addition to the visuals, but it can also lessen the clarity of the objects and surroundings. Furthermore, shadow maps could be used but the shadows should be almost invisible or pale.
7. Conclusions

This thesis presented an implementation process of creating a multimodal virtual environment (MVE). The process included searching available open source libraries, finding the right algorithms and techniques for visual and haptic presentation, and merging them together as a single application programming interface.

In addition, this thesis evaluated the applications based on the award winning conference article by Tanhua-Piiroinen et al. "Haptic Applications as Physics Teaching Tools" [Tanhua-Piiroinen et al., 2010]. The study gave positive results on utilizing the presented multimodal applications in the elementary school context. In addition, the teachers agreed that the haptics enabled virtual environments give additional possibilities to understand the learning content. Furthermore, the students seemed genuinely enthusiastic and interested in using multimodal virtual environments along with traditional teaching content.

The created application programming interface was mainly designed for the three implemented applications: the Density, the Leverage, and the 3D Construction application. The application programming interface supports haptic rendering, 3D OpenGL graphics, and speech synthesis. It is designed to be used by a C++ programmer, though it offers modifiable settings and attributes through an XML-file. The implementation process was meant to be a learning process for the author and therefore the easiest and the most simple tools and libraries were not always chosen.

There are few good application programming interfaces that support 3D graphics and haptics that were presented in Chapter three. Though, often one would need to integrate a physics library or a speech synthesis library or implement something new to match with the features that were implemented in the created application framework.

As a suggestion, the author would strongly recommend the readers to find and try out a haptic virtual environment if possible. Haptics cannot be described and visualized satisfyingly without the sense of touch.
References

[ACM SIGGRAPH, 2011] ACM Special Interest Group on GRAPHics and Interactive Techniques,  


SIGGRAPH Proceedings for 4th annual conference on Computer graphics and interactive  
technology, 1977.

Proceedings of the 5th annual conference on Computer graphics and interactive techniques,  

[Box2D, 2011] Box2D, Box2D physics library's domain. box2d.org/ (Accessed December 2011).


Surfaces. Phd Thesis, Department of Computer Graphics, University of Utah, December  
1974.


Proceeding of the 4th annual conference on Computer graphics and interactive techniques,  


[Festvox, 2011], University of Edinburgh's Festival Speech Synthesis Systems. Festvox – speech  

[Force Dimension CHAI 3D, 2011] Force Dimension CHAI 3D, CHAI 3D – API (Set of C++  
libraries for haptics, visualization and interactive real-time simulation). www.chai3d.org  

(Accessed December 2011).


