

Practical approach for automation of high fidelity haptic models generation

By

Mina Haratiannezhadi

B.S., K.N.Toosi University of Technology, 2013

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Master of Science in Industrial Engineering
in the Graduate College of the
University of Illinois at Chicago, 2015
Chicago, Illinois

Defense Committee:

Dr. Pat Banerjee

Dr. Cristian J. Luciano

Dr. Michael Scott

Table of Contents

B.S., K.N.Toosi University of Technology, 2013	i
THESIS.....	i
Submitted as partial fulfillment of the requirements	i
Chicago, Illinois	i
List of Tables	iv
List of Figures.....	v
Summary.....	vi
Chapter 1: Introduction.....	1
Data pre-processing.....	2
Movement tracking.....	2
Haptics rendering.....	3
Graphics rendering.....	4
Chapter 2: Medical Image segmentation	6
2.1 Intensity based segmentation.....	8
2.2 Discontinuity based segmentation.....	8
2.3 Region growing segmentation	9
2.4 Clustering based segmentation	9
2.5 Atlas-based automatic segmentation.....	10
Chapter 3: 3D Volume Generation.....	11
3.1 Open Inventor library.....	13

3.2 MHD field description.....	13
3.3 Haptics rendering.....	15
3.3.1 Polygonal mesh haptics rendering	16
3.3.2 Volume haptics rendering.....	17
3.3.3 CUDA volume haptics rendering.....	19
3.3.4 Haptic slice.....	20
3.3.5 Deformation shader	20
3.3.6 Cursor	21
3.3.7 Haptic effect.....	22
Chapter 4: Masking	24
Chapter 5: Converting to 8 bits	27
Chapter 6: Generating a scene in Open Inventor format	29
Chapter 7: Conclusion and Future work.....	36
Bibliography	38
APPENDIX	43
Write an iv file.....	43
Read an iv file	43
Vita	44

List of Tables

Table 1 Description of haptic datasets	15
--	----

List of Figures

Figure 1 Haptic simulation steps	5
Figure 2 Example of a scene graph	13
Figure 3 MHD format	14
Figure 4 Hierarchy of haptic nodes in a scene	16
Figure 5 Example of volume rendering	18
Figure 6 Example of opacity mapping	19
Figure 7 Example of using deformation shader	21
Figure 8 Example of cursor	22
Figure 9 Segmentation mask labels	24
Figure 10 Segmentation as a mask application	25
Figure 11 Segmentation as a mask algorithm	26
Figure 12 Apply a segmentation mask on an image sudo code	27
Figure 13 Concert to 8 bits application	28
Figure 14 IV Generator form	29
Figure 15 Scene form	30
Figure 16 Making volume form	31
Figure 17 Making mesh form	32
Figure 18 Making surface form	33
Figure 19 New effect form	34
Figure 20 Read data from MHD file	34
Figure 21 Questionnaire	37

Summary

This work describes the process for turning a raw volume into a 3-Dimensional volume that can provide haptic feedback and suggests a novel approach for automation of this process. The generated files are runnable in ImmersiveTouch simulators and present the properties of a haptic virtual object. ImmersiveTouch simulators perform various surgical procedures and they have been used in many hospitals and medical schools for educational purposes. Since these simulators are capable of performing unlimited customized cases, it is not efficient to employ a technician in each institution to generate the models manually. Existing approaches need the user to know the framework of an object to apply the required changes to the template file.

Making these objects requires a thorough knowledge of the simulator dataset format and structure of files needed to run the model. One of the fundamental problems in this area is to find a solution to simplify this process for nonprogrammer users to customize virtual models. In this research, three softwares were designed and implemented to automate three main steps of object generation. The purpose of this practice is to automate all of the process of making the haptic model.

The main input of this process is medical images, which are in different sizes, and formats. All of the images should be converted to the standard 8bit format, and the noises should be eliminated. Different tissues must be segmented separately; in this case a mask might be needed to apply on an image to take a specific tissue. After all the required input files are prepared, the user inserts haptic properties of the object into the application, and the application generates the required files by

extracting header information of the image. ImmersiveTouch simulators could run the output of the software and display the virtual objects.

The output files of the software are identical to the manually generated files. A survey was presented to a number of users with different level of knowledge of ImmersiveTouch dataset formats and asked them to evaluate the software. The results of the survey demonstrate that the users find it easier to customize and generate the 3D models with the software than manually.

Chapter 1: Introduction

Performing a surgery is a technically complicated procedure, which requires years of experience to minimize the risk. This experience is not possible to gain without practice. It has been shown that simulation-based training has a significant role in transfer of surgical skills. (Ali Alaraj 2013)

Obtaining surgical skill sets require many hours of supervised intraoperative training. Using virtual surgical simulators, medical students can understand the surgical techniques and feel the procedure before participating in a surgery. This method should be as realistic as possible to reproduce the experience to decrease the surgical faults.

ImmersiveTouch surgical simulation platform has been designed in University of Illinois at Chicago, and provides real-time haptic feedback along with high-resolution display for this purpose. An electromagnetic head-tracking protocol perceives the user head movement and changes the display perspective accordingly. The user can see the 3D virtual environment through a half-silvered mirror. This method integrates the surgeon hands, virtual tool and virtual patient to simulate the surgery. (Banerjee and Charbel 2006) (Banerjee, et al. 2007)

ImmersiveTouch simulators perform surgical procedures from different disciplines. Many properties of objects including shape, texture and elasticity are perceived by sense of touch. Haptic technology allows users to create haptic virtual objects, which can be touched and manipulated (G., Virtual Reality: Touch / Haptics 2009) Haptic interaction involves various mechanical variables such as torques, masses

of materials, vibrations, stiffness, etc. These properties arise from the environment and the user (perceiver). Different mechanical signals contribute to haptic perception (Young, Tan and Gray 2003). Haptic simulators are programmable systems, which can reproduce these signals that are experienced in the real environment (G., Principles of Haptic Perception in Virtual Environments 2008).

The simulation should be as realistic as possible to recreate the surgical procedure. The user sees his hand holding a virtual tool in the same location and environment as of the real surgery, and touches the objects of the scene using the stylus.

The ImmersiveTouch software utilizes different functions to process and display the graphic and haptic models. These are integrated on the hardware platform.

Data pre-processing

Magnetic resonance or CT scanner Dicom data sets are collected from clinical records and all confidential personal data gets anonymized. Dicom data should be converted to MHD/RAW format to ease the process. All of the images are segmented and combined to create a 3D volume of the patient body. To specify the surface's haptic features a polygonal iso-surface corresponding to each part is extracted from the 3D volume and exported to the rendering module.

Movement tracking

A sensor attached to the stereoscopic glasses tracks head movements to calculate viewer perspective while the user moves his head to see the virtual patient's body. Electromagnetic trackers use orthogonal electromagnetic fields to compute the

relative location of a receiver with respect to a known transmitter. The receiver is attached to the goggles, while the transmitter is at a fixed reference position. (Gobbetti, Scateni and Zanetti 2002) Another sensor located inside of the SpaceGrips tracks the surgeon's hand to locate the tool and focus on the specific part of the anatomy using the light.

Haptics rendering

Using the position and orientation of the haptic device, the system calculates the collision between haptic device and 3D model. Based on characteristics of the 3D model, the user will feel the force feedback. Therefore, the surgeon can feel the different surfaces and texture of different tissues. The haptic characteristics and graphic rendering for the simulator was modified based upon feedback from surgical faculty and senior residents. Properties of haptic feedback are specified in Open Inventor file format, which are readable by the simulator. Open Inventor is an object-oriented open-source 3D toolkit for graphics programming. It is built on top of OpenGL, and helps programmers build graphical scenes and manage 3D interaction between objects (Open Inventor 2015). Predefined nodes make a scene and they need to follow certain organizational structure to be understood by the simulator.

The 3D models are built based on real medical images, which are in DICOM¹ format. DICOM images are producing by different medical imaging devices such

¹ Digital Imaging and Communication in Medicine

as radiology, cardiology imaging, and radiotherapy device (DICOM 2015). All of the DICOM images should be converted to MHD² format, which is used in the medical segmentation and visualization toolkits. MHD file includes header information of an image, and the data is saved on a raw file. In the other words, the raw file includes the image, and the MHD file shows the image format (ITK/MetalO/Documentation 2015).

Graphics rendering

The virtual environment contains virtual patient body displayed using 3D iso-surfaces, virtual tool and perspective camera node. The graphics are displayed using a high-resolution monitor and trans-reflective mirror. “The ImmersiveTouch system offers high display resolution (1600 x 1200 pixels) and high visual acuity (20/24.74), which is important to clearly see the depth markings of the virtual catheter and small details of the head anatomy.” (Lemole, et al. 2007)

² MetaImage medical format

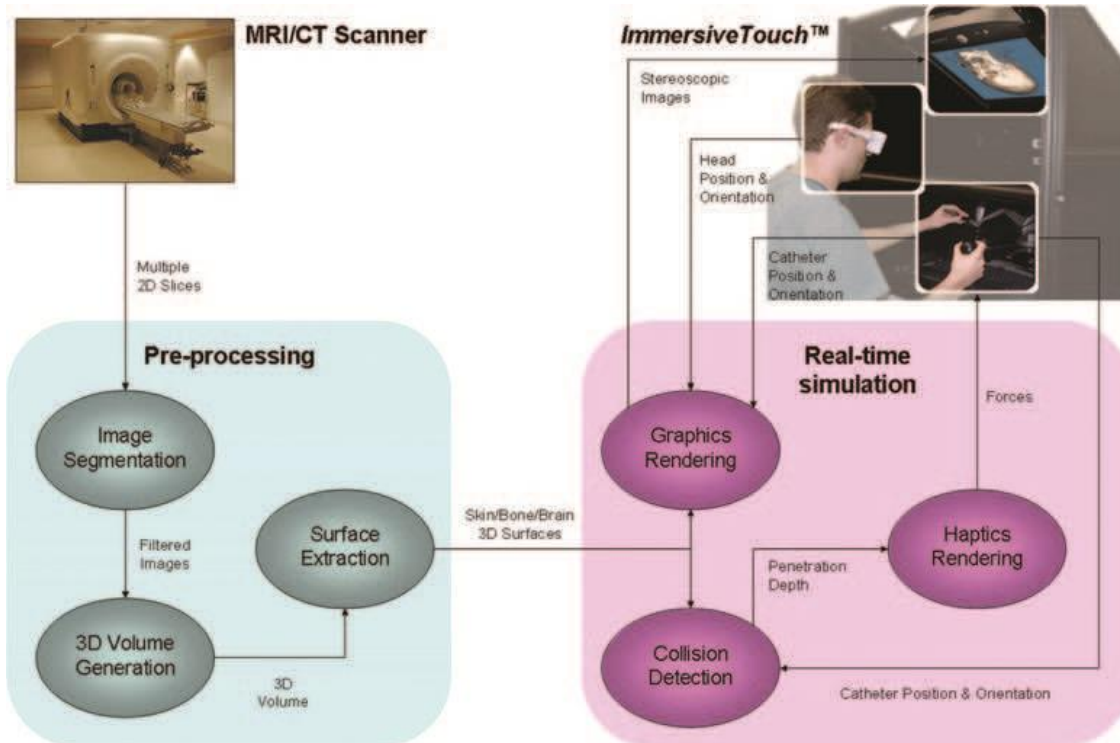


Figure 1 Haptic simulation steps

In this work our main focus is on data preprocessing section. An application to convert mhd/raw images to 8-bit format and an application to extract segmented image by applying segmentation as a mask were implemented. A platform to generate haptic modes was also designed and implemented. This application helps users to generate the haptic environment without any knowledge about open inventor syntax. In the first chapter we will discuss medical image segmentation methods, and a brief comparison between different medical image segmentation softwares.

Chapter 2: Medical Image segmentation

To facilitate visual interpretation of images and to ease the extraction of relevant information out of them, different image processing methods are employed. (Umamaheswari and Radhamani 2012) In medical image processing, the first step is to reduce the image noises captured by imaging system. In chapter 4, we present an application to crop 3D images and convert them to 8-bits. The next step is image segmentation, which means to identify different parts of the anatomy. Most of segmentation algorithms are based on discontinuities of the intensity values. The result of a segmentation process is saved in a binary image called *mask* according to the image intensities. In chapter 3, we present an application to apply a mask on the reference image to extract a specific part of image. Since segmentation of images is done manually, and it is difficult to reproduce the results because of the user involvement, it is a major bottleneck of image preprocessing in medical applications.

“Image segmentation is the process of dividing the individual elements of an image or volume into a set of groups, so that all elements in a group have a common property. In the medical domain, this common property is usually that elements belong to the same tissue type or organ. Segmentation allows visualization of the structures of interest, removing unnecessary information. Automated image segmentation, which aims at automated extraction of object boundary features, plays a fundamental role in understanding image content for searching and mining in medical image archives. “ (Smistad, et al. 2015) To make imaging systems and other information system in healthcare environments worldwide compatible with

each other, the Digital Imaging and Communications in Medicine (DICOM) standard is created as a cooperative international standard for communication of biomedical diagnostic and therapeutic information in disciplines that use digital images and associated data. “The DICOM standard, which includes a file format definition and a network communications protocol, is used in handling, storing, printing, and transmitting information in medical imaging. This standard also handles the integration of information produced by various specialty applications in the patient’s Electronic Health Record (EHR). It defines the network and media interchange services allowing storage and access to these DICOM objects for EHR systems. The National Electrical Manufacturers Association (NEMA) holds the copyright to the DICOM standard. Medical images in their raw form are represented by arrays of numbers in the computer, with the numbers indicating the values of relevant physical quantities that show contrast between different types of body tissue. “ (Startegic Document 2014)

One of the biggest problems in medical image analysis is image segmentation, which identifies the boundaries of objects such as organs or abnormal regions in images. Among some of the most common mathematical models are thresholding, region growing, edge detection and grouping, Markov Random Fields, deformable models, level sets and graph cut. (Qu, Huang and Jia 2008) For the best results different segments of the object should have definite boundaries, which is impossible in real world. In medical image segmentation automatic segmentation methods generate rough results. Achieving the appropriate results is impossible without experts’ intervention. Interactive segmentation methods have an important role in segmentation of regions accurately. Based on the amount and complexity

of information provided by user, the segmentation process may need different levels of user interaction. (Zhao and Xie 2013)

2.1 Intensity based segmentation

One of the simplest methods of image segmentation is based on the intensity levels or threshold based approach. This method makes different regions based on certain range of intensity values. There are two methods of thresholding; locally and globally. In global thresholding the object and background are getting divided, this method can be used to segment a single region out of an object. All of the pixels will be compared to the threshold value, the pixels that pass the threshold test are considered as object pixel and are assigned the binary value “1” and other pixels are assigned binary value “0” and considered as background pixels. The threshold based segmentation techniques are one of the most inexpensive methods, which can be performed on hardware level.

(Nagabhushana 2005)

2.2 Discontinuity based segmentation

This approach is based on the principle of intensity variations among the pixels of the image. The boundaries of the objects lead to creation of edges. The significant intensity differences among neighboring pixels in certain direction are defined as edges. (Ravi and Khan 2012)

ITK is an open-source, cross-platform system that provides developers with variety of software tools for image analysis. This system employs leading-edge algorithms for registering and segmenting multidimensional data. (Alves, et al. 2015) ITK-

SNAP is an open-source application developed with ITK library that provides semi-automatic segmentation using active contour methods. (ITK-SNAP 2014)

2.3 Region growing segmentation

This method works on the principle of homogeneity by considering the fact that the neighboring pixels inside a region possess similar characteristics and are different from the pixels in other regions.

The pixels are placed in a region based on their properties or the properties of their close neighbors. (Advanced Algorithmic Approaches to Medical Image Segmentation 2002) Each pixel is compared with its neighbor pixels, if the result is positive, it belongs to the region and the region is growing. The region stops growing when the similarity check fails.

2.4 Clustering based segmentation

Clustering a process of grouping the pixels based on their attributes. The objective of clustering techniques is to identify a portion of data with similar attributes like shape, texture etc.

There are various clustering methods; the most widely used are K-means algorithm and fuzzy C-means algorithm. The Clustering methods are usually divided as hierarchical algorithms and partitional algorithms. (Clustering methods 2010)

2.5 Atlas-based automatic segmentation

In atlas-based segmentation, a reference model is extracted, and it is used as a reference to identify anatomy parts and abnormalities. The reference atlas is built on the CT of a patient in standard position. In this method, automatic segmentation accuracy is highly dependent on similarity between the underlying atlas and the patient. (Daisne and Blumhofer 2013) Although this technique has been used for limited body parts, it is one of the most automated methods of segmentation.

Chapter 3: 3D Volume Generation

For a long time, human beings had been dreaming of a virtual world, which they can interact with objects, hear, touch and move them, as they were real. Humans explore objects by many steps. (Otaduy and Lin 2006) We locate the position of objects by watching them. Then we realize their shape and texture by touching their surface. Manual sensing of geometric characteristics, softness and texture of an entity happens through tactile and kinesthetic sensory systems, based on underlying neural inputs that respond to spatial and temporal distribution of forces on the hand.

Advanced computer graphics has gone a long way in this field. It enables the human tactual system to be stimulated in a control manner through haptic interfaces. These force feedback devices generate computer-controlled forces to help the user feel interacting with a natural environment. (Srinivasan and Basdogan 1997)

“The term haptic is the adjective used to describe something relating to or based on the sense of touch. Haptic is to touching as visual is to seeing and as auditory is to hearing”. (Otaduy and Lin 2006)

Sensory inputs along with controlled body motions activate the sense of touch.

Haptic rendering is defined as the process of computing and generating forces in response to user interactions with virtual objects. (Salisbury, et al. 1995)

Rendering algorithms that are based on a single contact point is called three degree of freedom haptic rendering algorithms. That is because any point in a 3-dimimensional environment has three degree of freedom, x, y and z. There are also

six degree of freedom haptic rendering algorithms, which deal with rendering the forces and torques arising from the interaction of two virtual objects. This is called 6-DoF, because the orientation of objects is involved in calculations.

There are two major ways of controlling a haptic device: impedance control and admittance control. In impedance control, the user moves the device, and the controller produces a force dependent on the interaction in the virtual world. In admittance control, the user applies a force to the device, and the controller moves the device according to the virtual interaction. Admittance control is more suitable for rendering a stiff virtual object, while impedance control is better for rendering a limp surface, because to render a stiff surface the device must react with large changes in force to small changes in the position, and impedance controlling might cause instabilities. Both of these controlling are deployed in haptic rendering systems for best result. (Ostaduy Tristán 2004)

To display and update a 3-dimensional scene, properties of existing objects are collected in a database called *scene*. The simulator program displays the sequence of graphics, and computes the haptic rendering based on the scene.

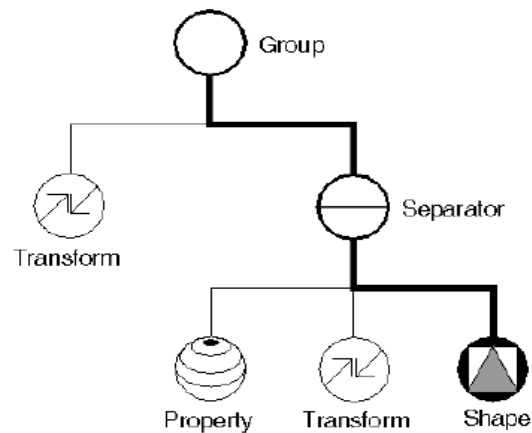
In this chapter, we describe an efficient way to draw a scene using Open Inventor library. Then we will present the principal of volume visualization.

3.1 Open Inventor library

Open inventor is a C++ library of objects and functions used to design interactive 3D graphics applications. It is set of blocks of code that enables user to write programs that use various graphics hardware features with high-level programming. All information about the objects including their, color, shape, texture, size and location in 3D space is placed in a scene database.

The scene database is node-based. Each piece of information, such as shape description geometric information, and surface material is defined as a node. The scene graph is hierarchy of nodes stored in Inventor database (Open Inventor 2015). The nodes of a scene graph include data and procedures that are needed to control the graphical and haptic scene.

Figure 2 Example of a scene graph



3.2 MHD field description

A raw file is a volume of image data stored in a single file. The volume can be of any dimensions. In order to visualize this data, properties of the image should be given.

To correctly load these images, the minimal information that you need to know is:

1. Number of dimensions
2. Size of each dimension
3. Data type
4. Name of the data file

Figure 3 MHD format

This assumes quite a bit about the image data. Specifically, it assumes that there is not any non-image data bytes (header data) at the beginning of the image data file, the voxels are cubes which means the distance spanned by and between a voxel in each coordinate direction is 1 unit, and the byte-order of the data in raw

```
ObjectType = Image
NDims = 3
DimSize = 256 256 32
ElementType = MET_USHORT
HeaderSize = -1
ElementSize = 1 1 2
ElementSpacing = 1 1 1
ElementByteOrderMSB = False
ElementDataFile = image.raw
```

file matches the byte ordering native to the machine we try to open the image on.

(ITK/MetaIO/Documentation 2015)

3.3 Haptics rendering

For virtual surgery, high visualization quality and real-time display are essential for a realistic simulation. (Lin, et al. 2014) There are three different ways to display 3-Dimensional datasets:

Table 1 Description of haptic datasets

Method	Technique	Pros	Cons	Example
SoHapticMesh	Off-line polygonal mesh rendering (triangle-based)	Faster graphics rendering Allows real-time deformation	Slower haptics rendering Not very detailed visualization (low polygonal count needed)	Skin
SoHapticVolume	Volume rendering (voxel-based)	More detailed visualization Faster haptics rendering Arbitrary volume size	Slower graphics rendering	Brain
SoCudaHapticVolume	Real-time polygonal mesh rendering (triangle-based)	Faster graphics rendering Faster haptic rendering Allows real-time material removal (e.g., bone drilling)	No real-time deformation allowed Volume size needs to be subsampled to multiple of 2 (e.g., 256 x 256 x 256)	Skull

A haptic surface should be defined along with a volume to demonstrate the haptic properties of the object. According to the volume rendering method, the surface can be defined as a polygonal mesh. To make a surface to produce an appropriate feedback while touching by the PHANToM³ (Massie 1996), multiple

³ PHANToM is a haptic device makes it possible for user to communicate with virtual scene by touching objects.

effects can be assigned to a surface. Also, to add deformability to a volume, a deformation node could be added to a haptic volume.

Various tools and devices might be used in a surgery; to assign these objects to the PHANToM, a cursor will be defined as a haptic object. The user will be able to switch between cursors and different states of a tool.

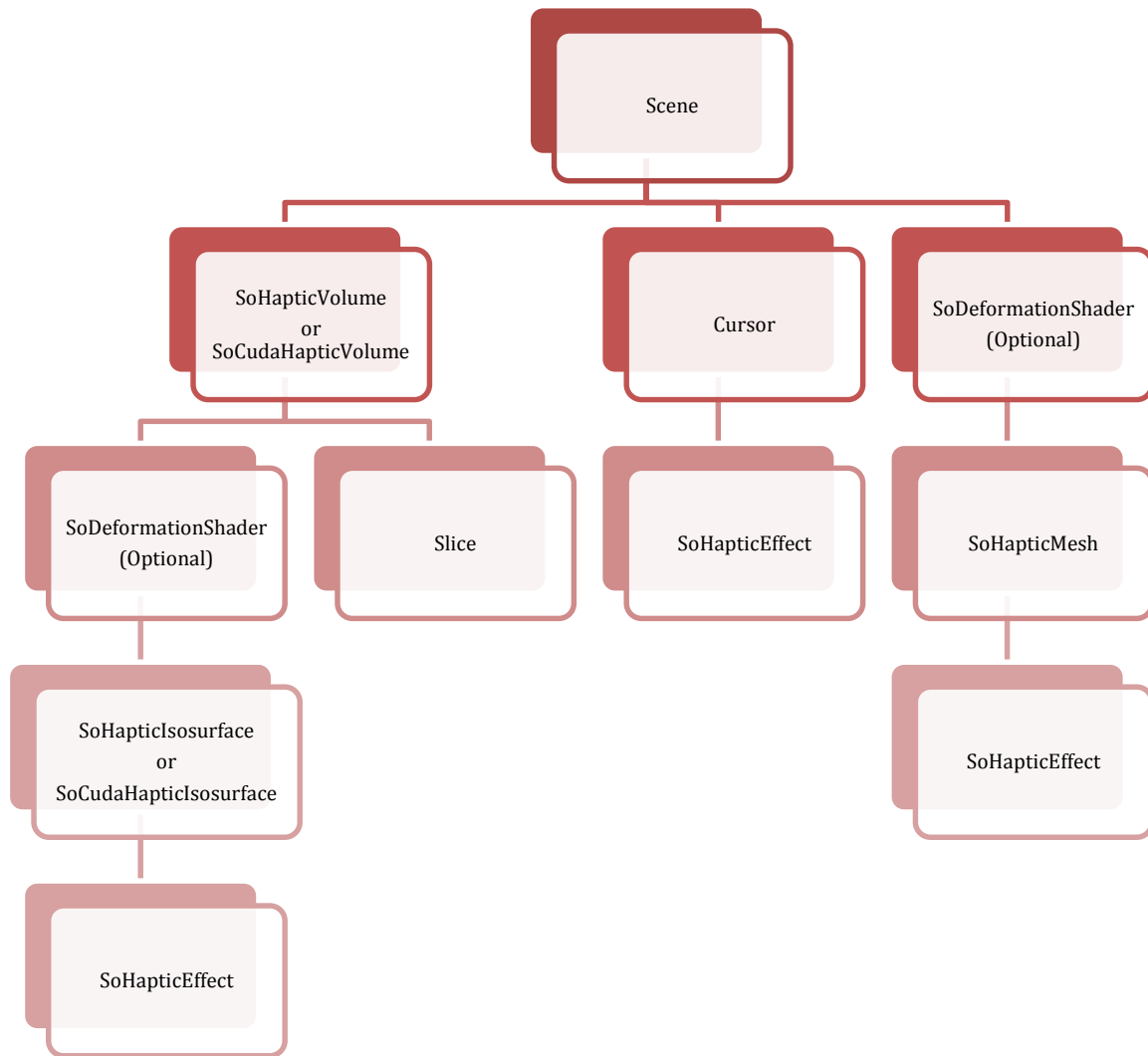


Figure 4 Hierarchy of haptic nodes in a scene

3.3.1 Polygonal mesh haptics rendering

Polygonal meshes are generated using a surface extraction algorithm, and they are represented as rigid polyhedra. As polygonal meshes are obtained from

isosurfaces the user cannot modify the model in runtime, which is necessary for volume removal. Though they render faster than haptic volumes and allow runtime deformation, they can be used in combination with haptic volumes to obtain the best efficiency (Rizzi, Luciano and Banerjee 2012). The fact that polygonal meshes were generated from isosurfaces prevented the user from dynamically modifying the model during the simulation, since it is computationally expensive to regenerate the whole mesh in real time. Having this ability is, though, an essential requirement for modeling surgical procedures where volume removal is frequently required, e.g., bone drilling.

Figure 3 shows the structure of a SoHapticMesh in Open Inventor format. A 3-dimensional vector graphic in VRML⁴ format is inlined in this file, which refers to the graphical representation of the object.

3.3.2 Volume haptics rendering

Volume haptization provides force feedback from volumetric datasets. For scalar data, they mapped either the voxel values to torque vectors or the gradient of voxel values to force vectors.

A haptic volume is constructed using an image in MHD/RAW format. These volumes are broadly used in bone surgery, when volume removal is necessary. However, graphics rendering of these volumes is slower than polygonal mesh, and they should be used along with polygonal meshes for visualization purposes (Rizzi, Luciano and Banerjee 2010).

⁴ Virtual Reality Modeling Language

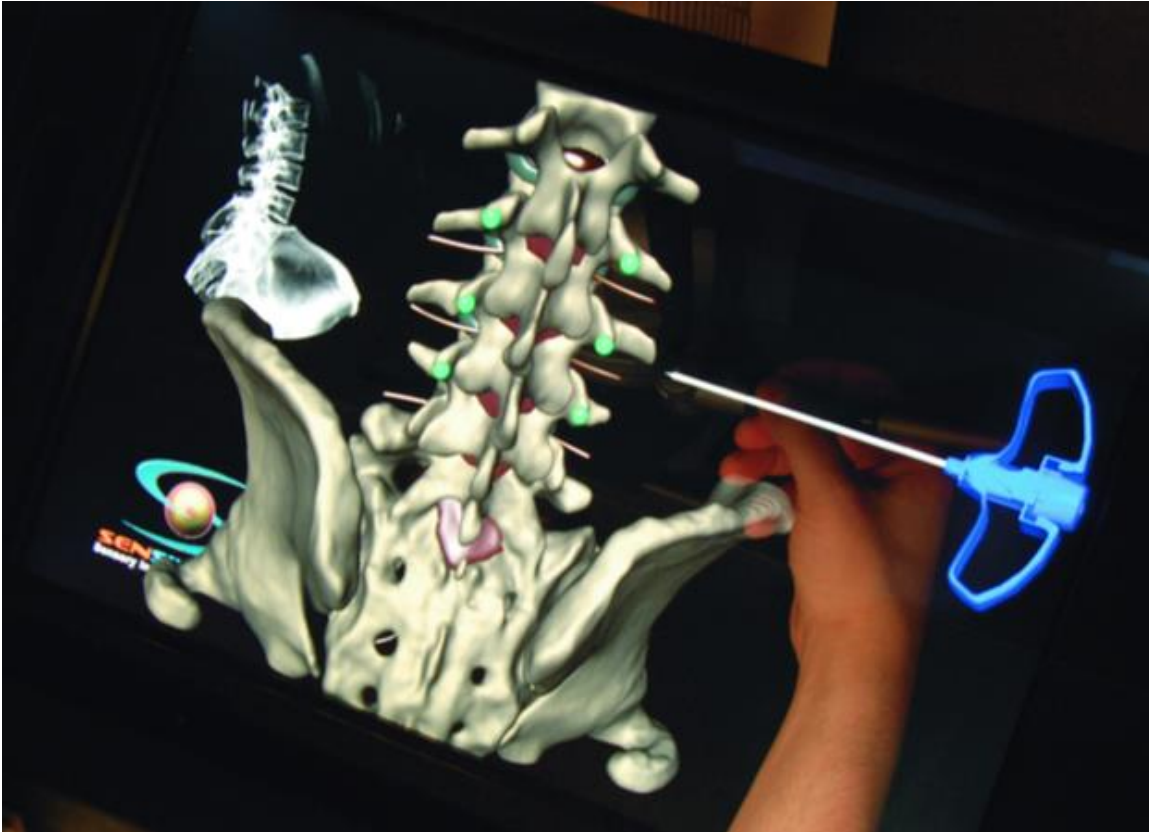


Figure 5 Example of volume rendering

For volume data, the standard proxy-based method cannot be used, since the volumetric data has no surface representations. A common way to introduce haptics to volume data is therefore to extract an intermediate local or global surface (Chen, Heng and Sun 2000) from which proxy-based surface haptics can be calculated.

The algorithms for surface haptics mostly perform on an explicit surface representation, for example polygons. The tip of the haptic instrument is prevented from penetrating into the object by providing the user with haptic feedback that pushes the instrument out of any virtual object. (Lundin, Ynnerman and Gudmundsson 2002) Isosurface node is an indexed triangle set, which should be

added as geometry to an object node. Graphical features and haptic properties of the model can be defined in this node (Corenthy, et al. 2012).

Most of these nodes are identical in haptic isosurface and polygonal mesh. A haptic isosurface has range and color nodes in addition. In the Scalar Opacity Mapping, the numbers along the X-axis are the density values between 0 and 256 for an 8-bit volume. Each point in this graph represents a number in the *range* node. Also, Each row contains 4 numbers: red, green, blue, and alpha (opacity). Each row of numbers coincides with a number of values in the *range* field.

Adding a material node can modify the appearance of the isosurface. The values for ambientColor, diffuseColor and specularColor are in RGB format.

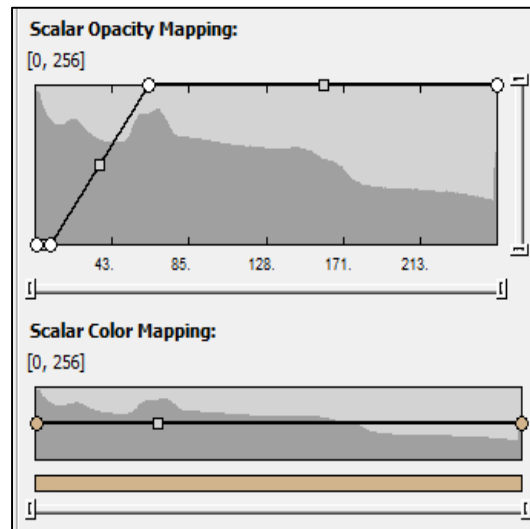


Figure 6 Example of opacity mapping

3.3.3 CUDA volume haptics rendering

CUDA is a general programming API, which provide direct access to the underlying parallel processors of the GPU, as well as full instruction sets, such as double precision computations and write operations at arbitrary locations. (Courtecuisse, et al. 2010) Haptic rendering implemented in CUDA is faster than reference CPU

implementation (Etheredge, Kunst and Sanders, Harnessing the GPU for Real-Time Haptic Tissue Simulation 2013) (Etheredge, Kunst and Sanders, Harnessing the GPU for Real-Time Haptic Tissue Simulation 2013). In this case, SoCudaHapticVolumes can be used to model hard tissues. The properties of this node are quite the same as SoHapticVolume and SoHapticIsosurface. In addition to the other properties of the volume, amount of memory reserved for the isosurface is specified in the node.

3.3.4 Haptic slice

If it is needed to visualize the orthogonal planes of the original volume (e.g., CT or MRI), collocated with the 3D model of the virtual patient, SoHapticSlice node can be added to the SoHapticVolume node.

3.3.5 Deformation shader

Physics engines offer multi-physics simulation capabilities, including rigid and deformable bodies, cloth, fluids, springs and characters, as well as collision detection and response algorithms. NVIDIA's PhysX, which can be improved using the physics processing unit or CUDA-enabled GeForce graphics processing unit, provides an optimized set of methods for physics-based simulation. This simulation engine applies position-based dynamics. (Maciel, et al. 2009)

To generate a deformable haptic model, a SoDeformationShader node should be added to the model. The user can customize the size of area of deformation while pushing the instrument into the haptic model, and the opacity of it.

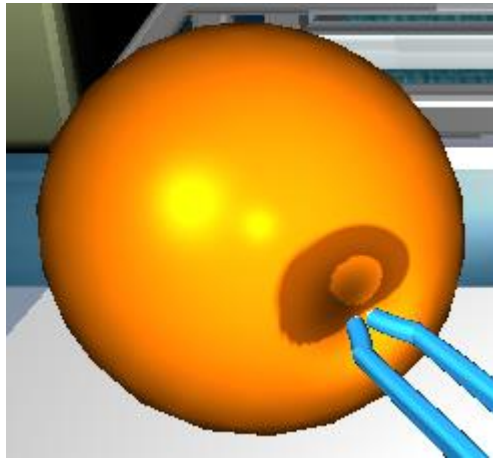


Figure 7 Example of using deformation shader

3.3.6 Cursor

Virtual tool modeling was the other important part of the virtual environment because the geometry or representation of the tool was essential to collision detection, force computation, haptic rendering, and data updating. (Lin, et al. 2014)

The cursor is the virtual instrument that is overlaid with the stylus (instrument the user is holding). In our applications, a portion of the instrument is left in its location when it is frozen. Therefore the instrument model is saved in two different formats, an internal and external view. The internal model is always shown when the instrument is active and the external model is left in the body when the instrument is frozen.

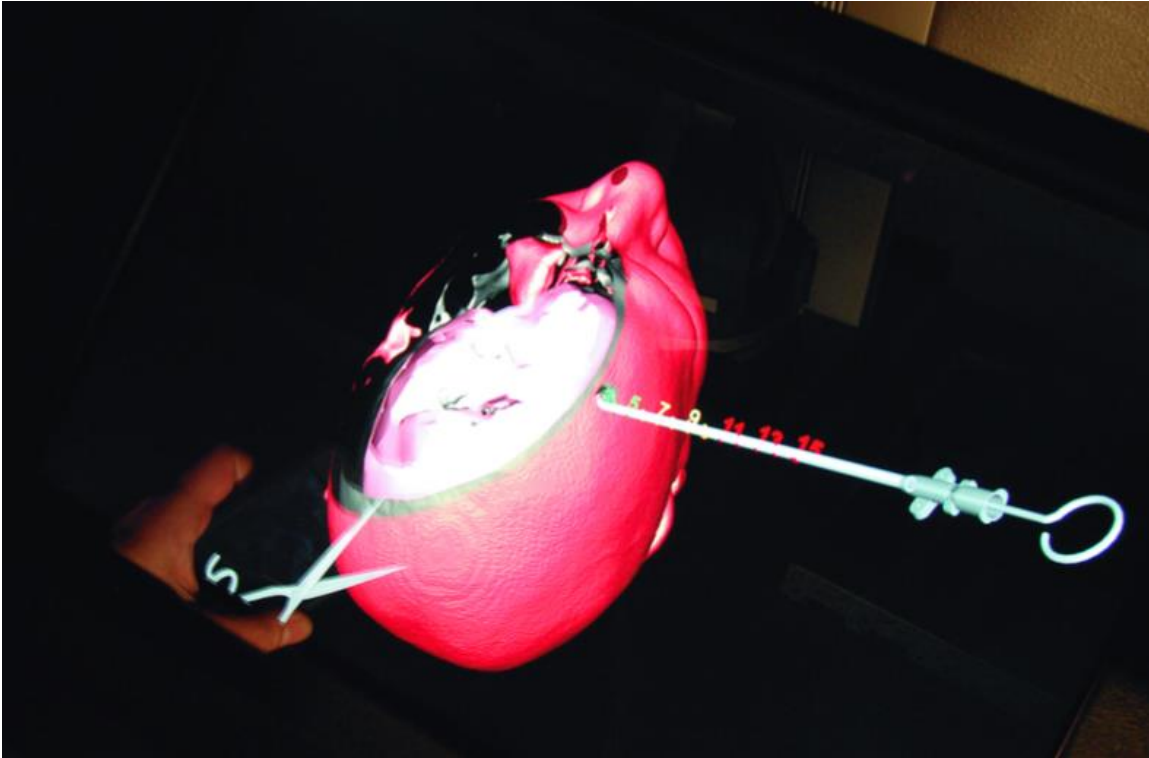


Figure 8 Example of cursor

3.3.7 Haptic effect

Each isosurface node may contain multiple haptic effects. As the PHANTOM's state changes, a specific haptic effect will trigger based on the haptic model, and will cause a haptic event. SoHapticEffect node defines the behavior of the vibration actuator. Properties of an effect can be customized by gain and magnitude node. Gain node value shows how fast the effect reaches its maximum value (magnitude). Following is the list of predefined haptic effects and their characteristics:

- Constant: Applies a constant force in a particular direction.
- Spring: Pulls the haptic device towards a particular point.
- Viscous: Based on current velocity, applies a force to resist the haptic device motion.

- Friction: Damped a user's velocity, and applies a stick-slip friction opposing the haptic device motion.
- Custom: Programmable force, useful to create new effects.
- Line: Forces the haptic device to follow a straight line motion in a particular direction (e.g. catheter insertion)
- Fulcrum: Restricts the haptic device motion to a pivot point (e.g. needle insertion)
- Vibration: Applies a constant sinusoidal vibration (e.g. drilling).

Chapter 4: Masking

In a haptic model, any tissue might demonstrate a specific behavior, and needs to be defined as a separate haptic node. In this case, original image is segmented. A segmented image is marked with a number of labels. Each label is assigned to a tissue. A segmentation mask is presented in MHD/RAW format. The size of this RAW file is equal to the original image. It means, in the segmentation mask each segment of data in the original file has been replaced by an integer number, which shows the assigned label. Labels are markings showing a range of data.

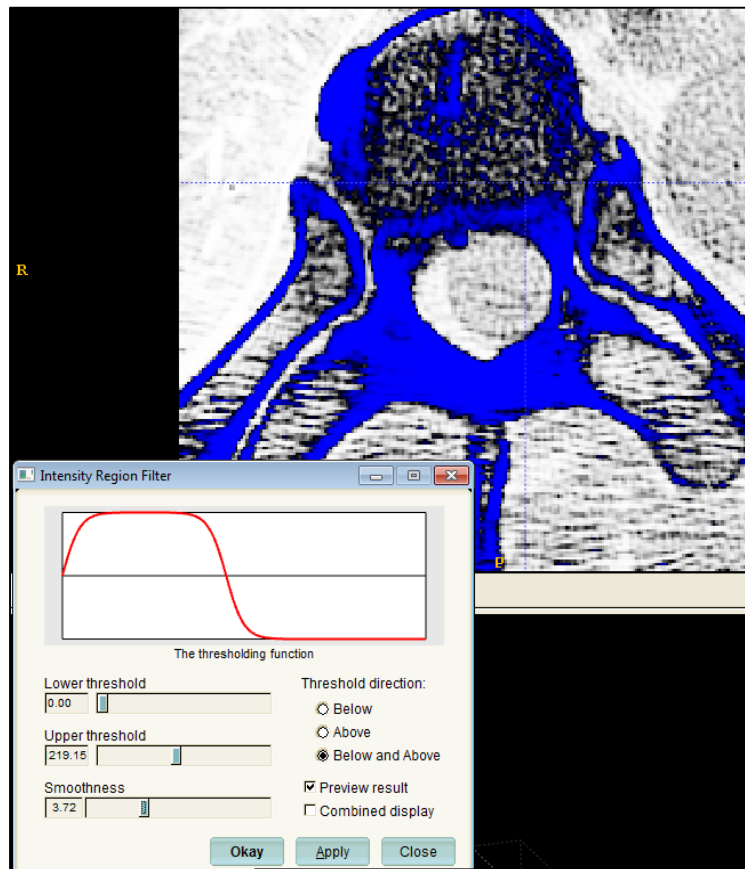


Figure 9 Segmentation mask labels

A program is presented in this chapter to apply this mask on the image, and return specific part of the 3D image. This software loads the original and segmented files,

and returns part of the original image, which has the stated range of labels. To perform this procedure, program reads both raw files, and compares the label of each segment of data with the label range; if this label is in the list of acquired labels writes the data segment in the output file; otherwise if the label is out of the range, that segment will be written as zero. Figure 10 shows graphical user interface of the application.

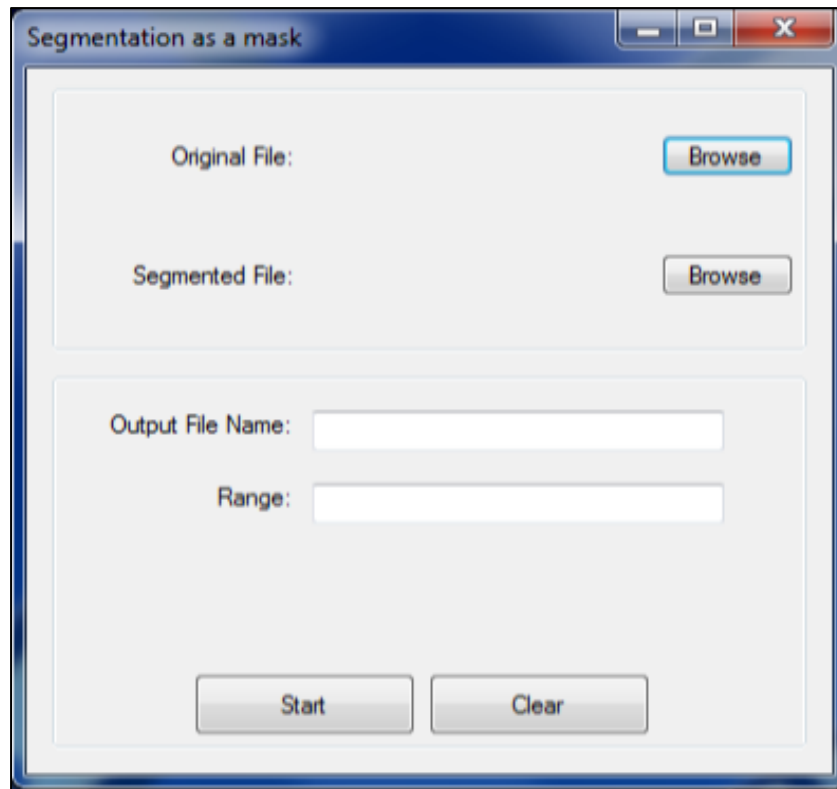


Figure 10 Segmentation as a mask application

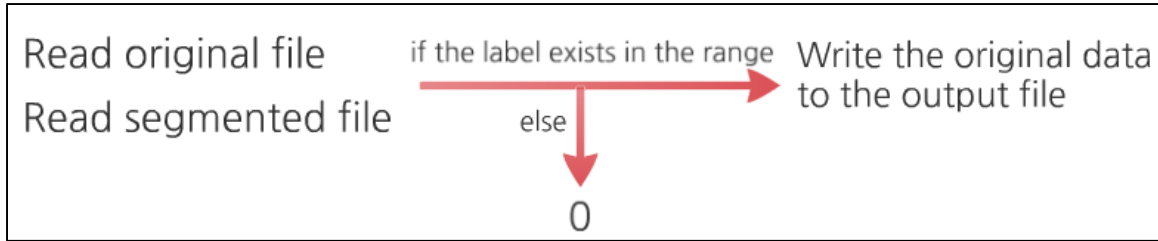


Figure 11 Segmentation as a mask algorithm

The program generates a new raw file including the specified parts of the original image. It is important for both images to be in exact same dimension size. Otherwise the segmentation does not match the original image.

Chapter 5: Converting to 8 bits

For best simulation performance, all of the images should be converted to 8-bit images. Also, images sometimes include some redundant data. By cropping the image, a portion of these noises can be eliminated. The resulted image is less in size and easier to segment, because we are dealing with smaller range of data.

An application presented in this chapter to crop a 3D image and convert it to an 8-bit image. This program loads an image in MHD/RAW format according to the header information, and prompts a range of data density. As the data range has been changed, each segment of data should modify accordingly. Data segments with negative value and less than minimum density will be assigned to zero (black), while which are positive and more that maximum density will save as 255 (white).

The other data segments will be recalculate according to the range.

Figure 12 Apply a segmentation mask on an image sudo code

```
Int vol = DimSize.x * DimSize.y * DimSize.z;
Int *input_buffer = new int [vol];
Unsigned char *output_buffer = new unsigned char [vol];
Fread (input_buffer, sizeof(int), vol, fp);
For ( int i =0 ; i < vol ; i++)
{
    if ( input_buffer [i] < 0 && input_buffer [i] < min)
        output_buffer [i] = 0;
    else if ( input_buffer [i] > 0 && input_buffer [i] > max)
        output_buffer [i] = 255;
    else
        output_buffer [i] = 255/ (max - min) * (input_buffer [i] - min);
}
```

Besides, a MHD format output file is generated. In this file element type is "MET_SHORT". Figure 13 shows the application's user interface.

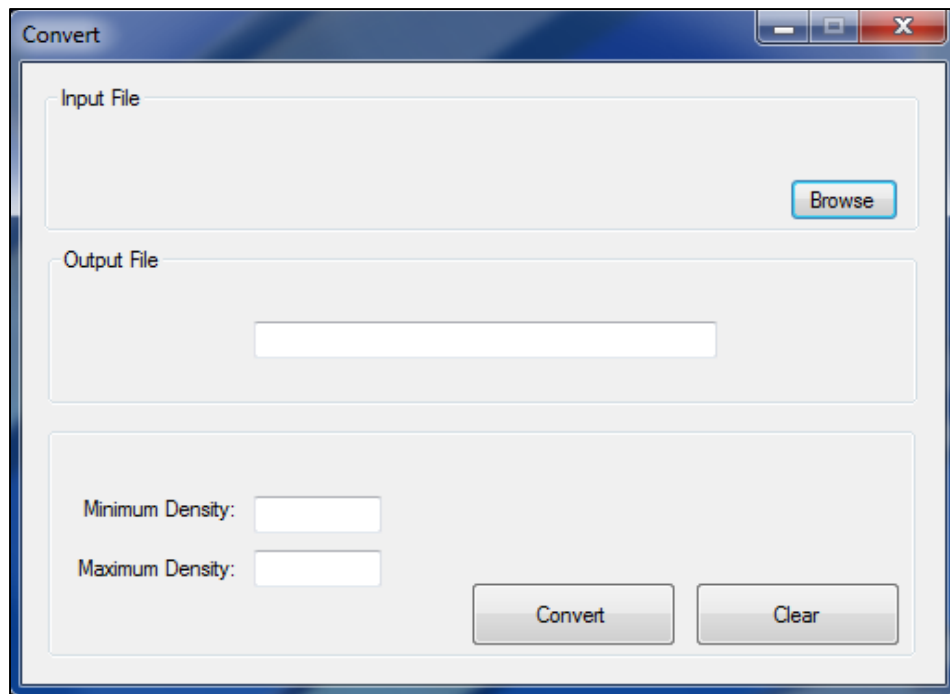


Figure 13 Concert to 8 bits application

Chapter 6: Generating a scene in Open Inventor format

After preprocessing all of the medical images, simulator readable files are ready to be generated. These files are in Open Inventor format, and they allow haptic rendering by ImmersiveTouch simulators. In this chapter, an application presented to generate a new scene or modifying an existing one. This program has been written in C++/CLI. All of the dynamic objects are defined in the “scene.iv” file. Static objects are programmed in the simulator application, and the user is unable to change them.

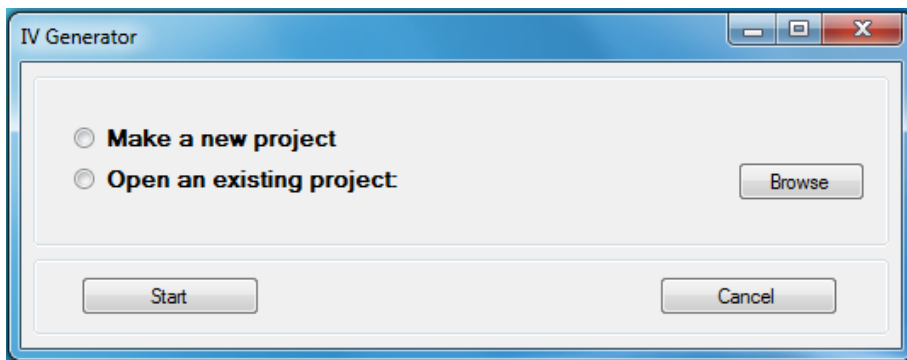


Figure 14 IV Generator form

To modify a model, user should insert directory of the “scene.iv”, and the application will retrieve all of the included files.

In the scene form, user can insert general properties of the model. User should specify, whether fluoroscopy is required in the procedure, and load cursor input files. Besides, the user can add desired effects to the cursor.

User will be able to edit or remove the generated files in each step. Also, an existing file can be imported to a model. For example, if a volume has been generated in another project, it can be added to the new project.

Figure 15 Scene form

Scene

Directory: C:\objects\Rhizotomy\Procedures

Volumes:

- skin_mesh
- bone_vol

☒ Fluoro: C:\objects\Rhizotomy\Models\256x256x512_dicom.mhd

☒ Cursor

Internal view: C:\objects\Rhizotomy\Models\Needle_int.WRL

External view: C:\objects\Rhizotomy\Models\Needle_ext.WRL

Active

- ☒ effect_line.LINE.20,5

To add a new volume, user should fill “Making Volume” form. User should specify the input MHD file and select type of the haptic volume. User will be able to add multiple surfaces to the volume. Type of the surface is selected based on the volume type.

Polygonal mesh does not include any surface. In this case, by selecting haptic mesh as type of the volume, application asks for haptic properties of the mesh and does not let user add any surface to the mesh. User should check whether the mesh is touchable. Also, if deformation is allowed in the mesh, the option should be selected.

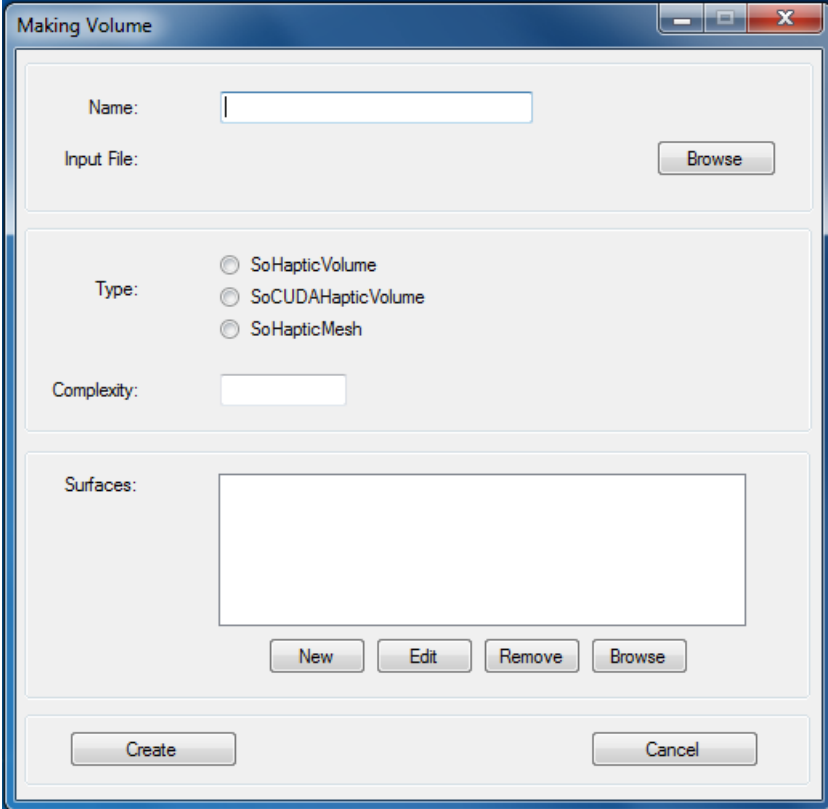
The image shows a software dialog box titled "Making Volume". It has a standard Windows-style title bar with minimize, maximize, and close buttons. The dialog is divided into several sections. The top section contains a "Name:" label followed by a text input field, and an "Input File:" label followed by a "Browse" button. The middle section contains a "Type:" label followed by three radio button options: "SoHapticVolume", "SoCUDA HapticVolume", and "SoHapticMesh". Below the radio buttons is a "Complexity:" label followed by a text input field. The bottom section contains a "Surfaces:" label followed by a large empty rectangular box. Below this box are four buttons: "New", "Edit", "Remove", and "Browse". At the very bottom of the dialog are two buttons: "Create" on the left and "Cancel" on the right.

Figure 16 Making volume form

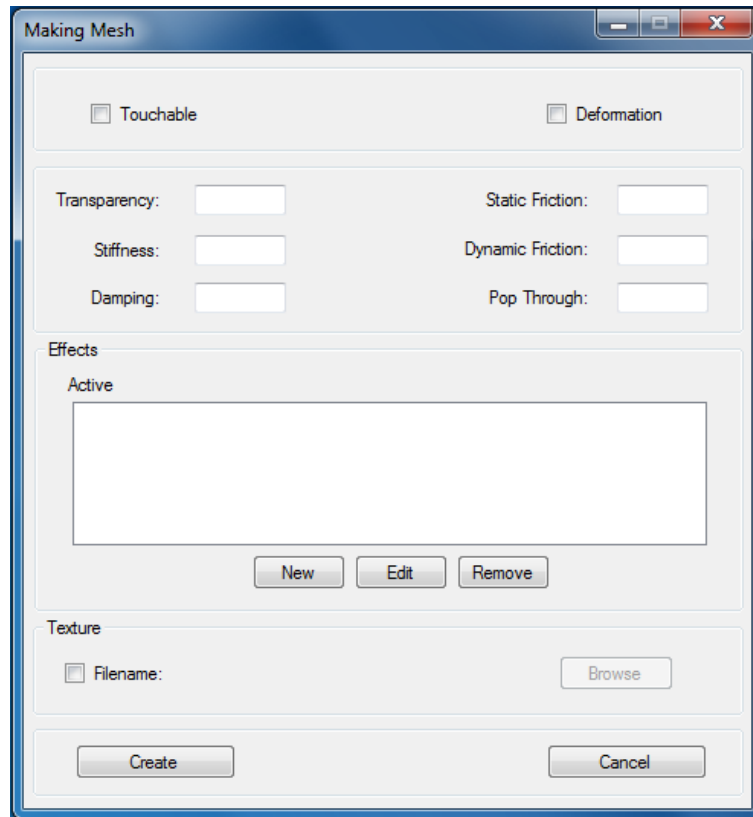


Figure 17 Making mesh form

According to the surface type, required fields are enabled. Haptic properties of a surface are similar to a mesh. User should select whether the surface is for a hard tissue or a soft one. Application will insert a default value for density range according to the selected type of tissue. User will be able to adjust the range in runtime.

Making Surface

Name:

Type: ☒ SoHapticIsosurface ☐ SoCudaHapticIsosurface ☐ Slice Max Triangles Fraction:

☐ Touchable ☐ Deformation

Transparency: Static Friction:

Stiffness: Dynamic Friction:

Damping: Pop Through:

Range: ☐ Hard tissue ☐ Soft tissue ☐ Use proxy

Effects

Active

New Edit Remove

Create Cancel

Figure 18 Making surface form

There is no need to fill the form to add a “slice.iv” file to the project. Application generates the file automatically.

User should fill “New Effect” form to add a new effect to a haptic node. Type of the effect should be selected from the list of all effects. Magnitude and gain of the effect are inserted in this step.

Figure 19 New effect form

The image shows a Windows-style dialog box titled "New Effect". It has a standard title bar with minimize, maximize, and close buttons. The main area contains four input fields: "Name:" followed by a text box, "Type:" followed by a dropdown menu, "Magnitude:" followed by a text box, and "Gain:" followed by a text box. At the bottom of the dialog, there are two buttons: "Create" and "Cancel".

To generate the haptic nodes, which require an input MHD file, the program extracts the required information out of the MHD file; for example, dimensions size, offset, etc. To extract information of a MHD file, the program reads each line of the text file and, loads data in an array. The first index of array includes the field name. According to the field name, data will be extracted from a specific index of the array. For example, in figure 19 the size of each dimension is extracted from DimSize in the MHD file.

Figure 20 Read data from MHD file

= line

DimSize	=	X	Y	Z
i= 0	1	2	3	4

X = line [2]
Y = line [3]
Z = line [4]

Before opening a new form, the information of the parent form is passed to the new form. According to this information, certain restrictions may apply.

All of the enabled fields are required, and the user will be notified if one of them is missing. By this approach, the chance of runtime errors has been reduced.

In each form, by pressing the “Create” button the corresponding IV file will be generated, and the relevant list in its parent’s form will update accordingly.

In the library of each form, two constructor functions have been defined. One is for making a new object, and the other one is for loading an existing file. If the user wants to edit any file, the application finds the file, and extracts the node’s information by looking up specific keywords in the file. The data will be passed to the relevant form constructor function, and the related form will be filled by the extracted data. After the user makes the modifications, the new file will replace the original file.

To remove a file from the project, the name of the model should be removed from the list. Also, the generated file should be deleted from the project directory.

Chapter 7: Conclusion and Future work

In the given models, haptic properties of each object have been specified using experimental evaluation. This approach may not have enough accuracy, and many devices have been designed to estimate mechanical characterization of a tissue. In this case, different kinds of stimuli are applied to the objects and the corresponding effects are measured. For example, it is possible to calculate the stiffness distribution with a direct method from a set of ultrasound images (Dupont, et al. 1999). Different methods have been proposed to measure the haptic properties of the scene. It is suggested to apply these features in the designed application in order to go forward to make the whole procedure automatically generated.

To evaluate the designed applications, thirteen people asked to participate in a survey. In this survey, required properties of a Rhizotomy procedure was given to the participants, along with a video tutorial of the application. They were supposed to make the model using the software. Participant with different level of programming knowledge were selected to participate in this survey. The ImmersiveTouch simulator was able to run all of the produced models. The participants were asked to rate easiness of each task according to their experience. Figure 21 shows the list of questions of the questionnaire.

Field of study:
Years of experience on Immersive Touch
Have you ever created a 3D model manually?
How clear is the software manual?
Could you perform all of the steps according to the manual?
Did you have any problem making a new model?
Did you have any problem editing an existing model?
How easy was making a new rendering model?
How easy was making a new haptic surface?
How easy was making a new polygonal mesh model?
How easy was adding a haptic effect?
How easy was to remove a file?
How easy was to edit a file?
Did you experience any unexpected runtime errors?
How user-friendly is the software?
Questions or comments:

Figure 21 Questionnaire

The results of the survey demonstrate %54 of the participants had no experience making a 3D model manually. Also, 92% of them find the software manual clearly documented. They were able to follow all of the steps according to the manual, and they found the software user interface easy to use. All of the users made the whole procedure scene successfully, and all of the participants agree that they could create, edit, and remove any file conveniently. None of the users experienced any runtime errors during software evaluation. Also, according to the participants' reports, making a procedure took them 20 minutes in average, which is a good improvement.

Bibliography

1. "Advanced Algorithmic Approaches to Medical Image Segmentation."
Edited by Kamaledin Setarehdan and Singh Sameer. 2002.
2. Ali Alaraj, Fady T. Charbel, Daniel Birk, Mathew Tobin, Cristian Luciano, Pat P. Banerjee, Silvio Rizzi, Jeff Sorenson, Kevin Foley, Konstantin Slavin, Ben Roitberg. "Role of Cranial and Spinal Virtual and Augmented Reality Simulation Using Immersive Touch Modules in Neurosurgical Training." *Neurosurgery*, 2013: 115-123.
3. Alves, Raquel S., Diogo Borges Faria, Dorval C. Costa, and Joao Manuel Tavares. "Analysis of gated myocardial perfusion spect images based on computational image registration." *2015 IEEE 4th Portuguese Meeting on Bioengineering (ENBENG)*, 2015: 1-2.
4. Banerjee, P., and Fady Charbel. "On-Demand High Fidelity Neurosurgical Procedure Simulator Prototype at University of Illinois using Virtual Reality and Haptics." *Accreditation Council for Graduate Medical Education (ACGME) Bulletin*, 2006: 20-21.
5. Banerjee, Pat, Cristian J. Luciano, Michael Lemole, Fady T. Charbel, and Michael Y. Oh. "Accuracy of ventriculostomy catheter placement using a head- and hand-tracked high-resolution virtual reality simulator with haptic feedback." *Journal of Neurosurgery* 107 (2007): 515-521.
6. Chen, K. W., P. A. Heng, and H. Sun. "Direct haptic rendering of isosurface by intermediate representation." *Virtual Reality Software and Technology*, 2000.

7. "Clustering methods." In *Data mining and knowledge discovery handbook*, edited by Lior Rokach and Oded Maimon. Springer, 2010.
8. Corenthy, L., J. San Martin, M.A. Otaduy, and M. Garcia. "Volume haptic rendering with dynamically extracted isosurface." *Haptics Symposium (HAPTICS), IEEE*. 2012.
9. Courtecuisse, Hadrien, Hoeryong Jung, J ér émie Allard, Christian Duriez, Doo Yong Lee, and St éphane Cotin. "GPU-based Real-Time Soft Tissue Deformation with Cutting and Haptic Feedback." *Biophysics and Molecular Biology* (Elsevier), 2010: 159-168.
10. Daisne, Jean-François, and Andreas Blumhofer. "Atlas-based automatic segmentation of head and neck organs at risk and nodal target volumes: a clinical validation." 2013.
11. *DICOM*. 08 15, 2015. <http://medical.nema.org/Dicom/about-DICOM.html> (accessed 08 15, 2015).
12. Dupont, Pierre E., Capt Timothy M. Schulteis, Paul A. Millman, and Robert D. Howe. "Automatic Identification of Environment Haptic Properties." *Presence: Teleoperators and Virtual Environments*, 1999.
13. Etheredge, C. E., E. E. Kunst, and A. J. B. Sanders. "Harnessing the GPU for Real-Time Haptic Tissue Simulation." *Journal of Computer Graphics Techniques 2* (2013).
14. Etheredge, C. E., E. E. Kunst, and A. J. B. Sanders. "Harnessing the GPU for Real-Time Haptic Tissue Simulation." *Computer Graphics Techniques 2* (2013).

15. G., Robles-De-La-Torre. "Principles of Haptic Perception in Virtual Environments." *Human Haptic Perception*, 2008: 363-379.
16. G., Robles-De-La-Torre. "Virtual Reality: Touch / Haptics." *Encyclopedia of Perception 2* (2009): 1036-1038.
17. Gobbetti, Enrico , Riccardo Scateni, and Gianluigi Zanetti. "Head and Hand Tracking Devices in Virtual Reality." In *3D Image Processing*, 287-292. Springer Berlin Heidelberg, 2002.
18. *ITK/MetaIO/Documentation*. National Library of Medicine Insight Segmentation and Registration Toolkit (ITK). 07 14, 2015. <http://www.itk.org/Wiki/ITK/MetaIO/Documentation> (accessed 07 14, 2015).
19. *ITK-SNAP*. 11 14, 2014. <http://www.itksnap.org>.
20. Lemole, Michael, Pat Banerjee, Cristian Luciano, Sergey Neckrysh, and Fady Charbel. "Virtual reality in neurosurgical education: part-task ventriculostomy simulation with dynamic visual and haptic feedback." *Neurosurgery*, 2007: 148-9.
21. Lin, Yanping, Xudong Wang, Fule Wu, Xiaojun Chen, and Chengtao Wang. "Development and validation of a surgical training simulator with haptic feedback for learning bone-sawing skill." *Journal of Biomedical Informatics* (Elsevier) 48 (2014): 122-129.
22. Lundin, Karljohan, Anders Ynnerman, and Björn Gudmundsson. "Proxy-based Haptic Feedback from Volumetric Density Data." 2002.
23. Maciel, Anderson, Tansel Halic, Zhonghua Lu, Luciana P. Nedel, and Suvranu De. "Using the PhysX engine for Physics-based Virtual Surgery with Force Feedback." *Int J Med Robot*, 2009: 341-353.

24. Massie, T. H. "Initial Haptic Explorations with the PHANToM: Virtual Touch through Pointer Interaction." Master Thesis, Massachusetts Institute of Technology, 1996.
25. Nagabhushana, S. "Computer Vision and Image Processing." (New Age International Publishers) 2005.
26. *Open Inventor*. 10 13, 2015. <http://oss.sgi.com/projects/inventor/> (accessed 10 13, 2015).
27. Otaduy Trista ñ, Miguel Angel. *6-DoF Haptic Rendering Using Contact Levels of Detail and Haptic Textures*. Dissertation, Chapel Hill: University of North Carolina at Chapel Hill, 2004.
28. Otaduy, Miguel A., and Ming C. Lin. *High Fidelity Haptic Rendering*. Morgan & Claypool Publishers, 2006.
29. Qu, Wei, Xiaolei Huang, and Yuanyuan Jia. "Segmentation in Noisy Medical Images Using PCA Model Based Particle Filtering." *Proceeding of SPIE - The International Society for Optical Engineering*, 2008.
30. Ravi, S, and A M Khan. "Operators Used in Edge Detection: A Case Study." *International Journal of Applied Engineering Research* 7 (2012).
31. Rizzi, S.H., C.J. Luciano, and P.P. Banerjee. "Haptic Interaction with Volumetric Datasets Using Surface-based Haptic Libraries." *Haptics Symposium, IEEE*. 2010.
32. Rizzi, Silvio H., Cristian J. Luciano, and and P. Pat Banerjee. "Comparison of Algorithms for Haptic Interaction With Isosurfaces Extracted From Volumetric Datasets." *Computer Information Science Engineering*, 2012.

33. Salisbury, J. K., D. Brock, T. Massie, N. Swarup, and C. Zilles. "Haptic Rendering: Programming Touch Interaction with Virtual Objects." *Proceedings of the ACM Symposium on Interactive 3D Graphics*. Monterey, 1995.
34. Smistad, Erik, Thomas L. Falch, Mohammadmehdi Bozorgi, Anne C. Elster, and Frank Lindseth. "Medical image segmentation on GPUs - A comprehensive review." *Medical Image Analysis* 20, 2015: 1-18.
35. Srinivasan, Mandayam A, and Cagatay Basdogan. "Haptics in virtual environments: Taxonomy, research status, and challenges." Edited by Pergamon. 21 (1997): 393-404.
36. "Strategic Document." *NEMA*. 08 25, 2014.
<http://dicom.nema.org/dicom/geninfo/strategy.pdf>.
37. Umamaheswari, J, and G Radhamani. "Hybrid Denoising Method for Removal of Mixed Noise in Medical Images." *International Journal of Advanced Computer Science and Applications* 3 (2012): 44-47.
38. Young, J. Jay, Hong Z. Tan, and Rob Gray. "Validity of Haptic Cues and Its Effect on Priming Visual Spatial Attention." *HAPTICS '03 Proceedings of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS'03)*, 2003: 166.
39. Zhao, Feng, and Xianghua Xie. "An Overview of Interactive Medical Image Segmentation." *Annals of the BMVA* 2013 (2013): 1-22.

APPENDIX

To generate the iv files several functions are used in the application development.

In this section, the logic behind these functions is described.

In order to make the user be able to edit the models after generating them, all of the files should have a specific format.

Write an iv file

To be able to read the iv files, each field should be in a single line. Also, the value of each field should be in front of it, separated by a single tab. (e.g. StaticFriction 0.5). When the user presses the “Create” button in each form, a function will be called by program, which has a template of the related model (e.g. soHapticVolume). The program replaces the value of each field in the template. If the model includes a number of different nodes (e.g. soHapticEffect), the program will add as many nodes as stated in the form to the parent node.

Read an iv file

By selecting any model in the application and pressing the “Edit” button, the program will read the file from “Procedures” directory. Name of each node is identical to the iv filename. Based on type of the model, the program reads the file line by line, and looks up the keywords. The value in front of each keyword is assigned to a variable. The program uses these values, and calls the form constructor to refill the form. When the user modifies the form and presses the “Create” button the file will be regenerated, and it replaces the original one.

Vita

Name: Mina Haratiannezhadi

Education: BS. Computer Software Engineering, K. N. Toosi University of
Technology, Tehran, Iran, 2013

MS. Industrial Engineering, University of Illinois at Chicago,
Chicago, Illinois, 2015