Hybrid Surgery Cutting using Snapping Algorithm, Volume Deformation and Haptic Interaction

Jie Peng, Ling Li, Andrew Squelch

1 First Author Curtin University, jie.peng@postgrad.curtin.edu.au
*2 Corresponding Author Curtin University, L.Li@curtin.edu.au
3 Curtin University, A.Squelch@curtin.edu.au

Abstract

In this paper, we propose algorithms to generate realistic cut simulations on hybrid deformable anatomy objects consisting of volumetric data and iso-surfaces. A 3-dimensional node snapping algorithm is presented to modify the surface topology of the objects, without adding new elements. Smooth cut is generated by duplicating and displacing mass points that have been snapped along the cutting path. A volumetric deformable model is employed underneath the surface, with the internal structure and material properties of the heterogeneous objects revealed along the opening. A 3D Chainmail deformation algorithm is used for the deformation of the volumetric model to enhance the realism. A haptic device is integrated into the simulation system as a cutting tool to trigger the progressive cutting procedure, and to feel the different volumetric components. The simulator incorporates the simulation of surgical prodding, pulling and cutting. Advanced features include the separation on the cut surfaces and post-cutting deformations like wrinkle effect. The proposed cutting techniques can be used in surgical simulation or other virtual simulations involving topological modification of heterogeneous soft materials to enhance the fidelity and realism.

Keywords: Mesh-cutting, Direct volume rendering, Direct volume deformation, Haptic interface

1. Introduction

Simulation of procedure tasks has the potential to bridge the gap between basic skills training outside the operating room (OR) and performance of complex surgical tasks in the OR [33]. Cutting simulation is the key component in surgical simulation as cutting is a common operation in both conventional and minimal invasive surgeries. Most surgical tasks begin with an incision to expose the surgical region. However, it is a challenging task to simulate cutting operations since the object topology is changing in real-time and a large amount of computation is usually required for realistic cutting simulation.

In surgical simulation, it is generally not necessary to provide a physically-based simulation of the internal forces involved in the deformation, as a huge computational cost would be required. Instead, it is more important to show realistic internal structures at the cutting site as the visual clue, especially for surgical training and planning purposes [1]. Therefore, priority should be given to the realistic surface deformation and smooth cutting before there is an opening. Afterward, the users often focus on the exploration of the interior features.

Surface-based cutting methods can provide smooth and realistic cutting effect at a low computation cost, but a great deal of information about the interior structures and the material properties of the heterogeneous tissues are discarded. Meanwhile volumetric models can incorporate appropriate internal structures and material properties for situations in which high-fidelity virtual environments are required.

Volumetric models can be implemented by polygonal representations, which are usually obtained after a time consuming process of explicit segmentation and reconstruction. Special algorithms are required for representing and modelling heterogeneous deformable volume objects [1]. As an alternative approach, direct volume rendering allows the efficient visualization of tomo-graphic 3D image data, using implicit segmentation based on transfer functions for colour and opacity values. Combining with direct volume deformation, it is capable of presenting a great deal of information about internal structures at the cutting site to enhance the fidelity and realism for the purpose of surgical training and planning.
A hybrid method was hence proposed to deal with the cutting of heterogeneous objects, which consists of surface cutting and interior volume deformation. The surface mesh cutting is implemented using the node snapping algorithm and the interior volume is represented and manipulated by direct volume rendering and deformation. A haptic device is integrated into the cutting simulation system as the cutting tool so that users can feel the different materials within a deformable object as well as interact with it through touching and cutting.

The remainder of the paper is organized as follows: In Section 2, related existing work is reviewed. Our cutting simulation system is described in Section 3, which includes systems overview, as well as details on surface mesh cutting, surface deformation, direct volume rendering, direct volume deformation and haptic interaction. Some simulation results are presented in Section 4 with the evaluation of the system performance. Section 5 concludes the paper.

2. Related work

In most of the existing cutting simulations, cutting operation is considered only on surface models or homogeneous volumetric models. In general, surface based models are followed by some special procedures, such as the groove creation [1][3], to generate an illusion of volumetric cuts. In [26], nonlinear elasticity of deformable objects has been modelled using nonlinear strain tensors or nonlinear spring coefficients. However, these approaches are only capable of dealing with deformable objects that consists of a single material type. More recently, adaptive hexahedral simulation meshes based on octree refinement have been widely employed [2]. These approaches require an explicit correspondence between the simulation elements and the embedded surface vertices. In addition, when cutting into volumetric objects, their internal volumetric structures have to be created at an extra cost. We aim at creating a hybrid model with realistic cutting effect on the outer surface and on the underneath heterogeneous volume model with detailed interior structure, which means users could explore and understanding the three dimensional spatial relations of the complex anatomy in the related region.

Cutting simulation starts with surface incision. Surface mesh cuttings are generally implemented by three types of topological modification approaches, e.g. element removal, mesh subdivision and mesh adaptation. The element removal technique basically removes the elements that are intersected by the cutting tool, which has been applied in [5]. Despite of its simplicity and computational efficiency, this method cannot present smooth cutting because of visual artefacts and the cut surfaces look unnaturally jagged. More appealing visual representations of incisions are made possible with mesh subdivision methods. They usually classify a cut according to the different rotational invariant intersection states. Predefined subdivisions of mesh elements are then performed. In the context of medical applications, this was first introduced in [6] for cutting of tetrahedral meshes with predefined planes. This approach has been refined in [7], where partial incision of mesh elements as well as progressive cutting is taken into account. Finally, [8] discusses the use of a state machine to keep track of different incisions in tetrahedral meshes. All these approaches often involve considerable increase of element count, ranging from 5 to 17 new elements per incised tetrahedron. This problem could be ameliorated with mesh adaptation approaches as suggested in [9] and [11]. The main idea is to approximate a cutting path with existing edges or surfaces. It enables mesh incisions without large increase of element count and occurrence of small elements. Unfortunately, their work is only limited to homogeneous models which means soft tissue layers around the cutting site could not be presented though they are common in real surgery. Finally, a heterogeneous deformable object modelling technique has been proposed in [10] to model heterogeneous soft tissues and to present different materials and structures. However, this method means that for every different soft biological tissue model, a very complicated and time consuming algorithm has to be applied before any deformation and manipulation.

In contrary, volumetric models can represent a great deal of information about interior structure and mechanical properties of heterogeneous tissues without any extra procedures. In fact, whenever the internal structure is important for the appearance or behaviour of a graphical object, a volumetric object representation is necessary [13]. Kaufman et al. introduced the field of volume graphics in [14]. There are three basic classes of volumetric object representations in computer graphics: 3D sampled data, particle system [4]; and geometric meshes for modelling deformable objects using mathematical techniques such as FEM [21] or mass-spring systems [19][20].
Sampled volumes lack a sense of connectivity that is necessary for modelling complex tissue deformation or the cutting of volumes into distinct pieces. Particle systems have been used to model objects that are highly deformable and to model separation and joining of such objects [15]. However, as particle systems lack a representation of internal structure; they are not suitable for modelling most materials.

The geometric meshes used in FEM and mass-spring systems are most common. Of these two, the latter is more widely used, because it is more accurate and can accommodate different material properties through a small number of parameters, but it suffers from the drawbacks including unrealistic behaviour for large deformations and the difficulty in finding appropriate values for the spring constants to produce realistic behaviour.

Another promising volumetric object representation is linked volume, in which each element in a sampled volume is explicitly linked to its six nearest neighbours. These links are stretched, contracted, and sheared during object deformation and deleted or created when objects are cut or joined. Based on this volumetric representation, a novel approach to soft tissue deformation called 3D Chainmail is presented by Gibson [24]. It is geometrically-based but is capable of simulating material properties to some extent. The algorithm is extended to model the differences between types of tissue and their interaction in [25].

Computer haptics refers to methods for creating haptic impression to a human user via a specialized piece of hardware (haptic interface or display). To feel the 3D object, the user needs to be in direct contact with the haptic interface device. The device is, in turn, controlled by haptic rendering algorithms (software). This connection creates a closed loop that continuously exchanges force signals and position signals between the user and the virtual 3D objects.

In the introductory text on haptics-interactions by Salisbury et al. [28], it is stated that haptic-interaction is a very important component in VR applications, in line with the importance of visual and auditory components. Whereas visual and auditory media are one-directional from the media to user, the haptic device is bi-directional in nature. The haptic device both receives interaction input and responds with forces. According to [28] this is often referred to as the single most important feature of the haptic interaction modality. A haptic force-feedback interface has been widely used for conceptual design, collaborative design, virtual prototyping and sculpting in VR environment [17][18][27].

Given the discussion above, existing surface models lack the interior structure information while existing volumetric model could not offer smooth realistic real-time cutting. In this paper, we propose a hybrid cutting simulation system which applies the mesh adaption as the surface cutting approach and 3D Chainmail as the volume deformation method. Haptic interaction is integrated into the system and the volumetric model is rendered by direct volume rendering.

3. Methodologies

3.1. System overview

Our hybrid cutting system takes a two-step approach. The first step generates a smooth surface cutting, while the second step performs direct volume deformation & rendering and mixes them with surface cutting. The complete system pipeline consists of the following five major operations:

**Surface deformation:** When a tool collides with a deformable object, the object’s surface deforms until the applied force becomes larger than the yield limit of the material being simulated. The deformation is determined by the intersection point and the direction of tool motion.

**Surface cutting:** When the breaking strength of the material is reached, further tool motion causes the object to be cut and followed by a rupture. This is a progressive process and the underlying geometric model is updated using a node-snapping technique.

**Surface wrinkle generation:** During the progressive cutting, collisions between the blade and the geometry are constantly detected. Once a cutting operation has finished, the next step is to pull open the cut to expose the surgical region. Due to the elasticity of the soft tissue, the wrinkle effect will be generated along the cut path on the surface during the pulling.

**Volume deformation:** A cutting gutter on the underneath volumetric model is revealed along the cut opening to enhance realism. The gutter is implemented by a direct volume deformation algorithm called the Divod Chainmail [22] based on the Chainmail algorithm and the Enhanced Chainmail algorithm.
Volume rendering and mixing: The last operation performs volume rendering and mixes it with surface illustration.

3.2. Surface deformation

Various techniques can be found in literature for the simulation of deformable objects. These techniques can be categorized into two main approaches: geometrically based approaches and physically based approaches. The geometrically based modelling approaches do not account for the physics of deformation, but are simpler to implement. In contrast, the physically based approaches, attempt to model the underlying physics, but are computationally intensive. In order to simply the system, we choose the very rudimentary surface deformation algorithm called Coordinate Deformer in H3D [29], which is very realistic and effective. H3D is a haptic extension of the X3D scene-graph API that renders a scene graphically and haptically - a scene’s objects have graphical properties (e.g., colour) and haptic properties (e.g., friction). It is entirely written in C++, and it uses OpenGL and HAPI for graphics and haptics rendering. The Coordinate Deformer displaces the points of a mesh within a radius of the stylus tip of the haptic device. The displacement is determined by a mathematic function called shape function.

3.2.1. Pushing effect by Gaussian Function

The default shape function in H3D is the Gaussian function as depicted in Figure 1, which will fit the displaced mesh points to a bell shaped surface that resembles a Gaussian surface with the apex at the contact point of the stylus. We implement the pushing effect before the rupture using the Gaussian function since it fits very well with such effects observed in real life. Figure 2 illustrates the initial deformation of the object surface before the yield limit is reached. During the deformation, a force vector is calculated from the number of displaced points, modified by the mesh’s material parameters and sent to the haptic device. This force model increases the resistance felt by the user as more and more of the mesh is displaced.

3.2.2. Wrinkle effect by Decay Function

The default shape function in H3D is the Decay function as depicted in Figure 3, which will fit the displaced mesh points to a decaying wave that resembles a Decay surface with the apex at the contact point of the stylus. We implement the wrinkle effect before the rupture using the Decay function since it fits very well with such effects observed in real life. Figure 4 illustrates the initial deformation of the object surface before the yield limit is reached. During the deformation, a force vector is calculated from the number of displaced points, modified by the mesh’s material parameters and sent to the haptic device. This force model increases the resistance felt by the user as more and more of the mesh is displaced.

Figure 1. Gaussian function

Figure 2. Pushing effect

Figure 3. Decay function

Figure 4. Wrinkle effect
In a real surgery, when a surgeon uses a retractor to hold back soft tissues such as skin to open up the cut, there will be fold along the cutting edge, because of the surface elasticity [16]. Wang et al. [30] used two hand-held haptic devices to simulate the contact with retractors. Retractors or spatulas are used in surgery to separate the cut surfaces while the surgeon attempts to make further cuts or to focus on the region of interest. However, they did not implement the wrinkle effect when the cutting edges are pulled away by the retractor. In addition, complex calculation is required to obtain the intersections between the retractor polygon and the surface object. Correa and Silver [31] have implemented a very realistic wrinkle effect along the cut, but the deformation is a part of the rendering pipeline, which differs from traditional deformation methods, where deformation is considered as a modelling problem and a new mesh is required. It does not suit our purpose since our deformation is based on the mesh. We design a decay function to displace the mesh points along the cutting path to simulate the wrinkle effect. In Figure 3, the x axis represents the distance of a mesh point to the cutting path, and the y axis represents the depth of the deformation along the direction of penetration depth of the surface (i.e., the distance the haptic device has penetrated the surface). That is, the further the mesh point is from the cutting path, the less displacement it is applied. Our decay function is able to produce very realistic and smooth wrinkle effects with very little computational cost. Figure 4 is an example of the wrinkle effect applied on a surface mesh of a female chest.

3.3. Surface mesh cutting

In this section, we will present an efficient cutting algorithm based on node-snapping that is capable of visually simulating progressive cutting with minimum increase in the number of new elements.

Node-snapping methods have been used in geographic information systems for nodal coordinate adjustments of digitized data and CAD tools for identification of features and the cleanup of data with node merging and weeding. Nienhuys and van der Stappen [11] proposed a similar algorithm for the modification of object geometry. Later Lin and Lim [1] applied the approach to progressive cutting.

![Image of cutting path intersects with mesh points](image)

**Figure 5.** Cutting path intersects with mesh points

The node-snapping method starts with collision detection. After the initial collision of the haptic tool with the object surface, intersection points between the cutting path and the underlying polygon edges are calculated as shown in Figure 5. It is worth mentioning that the marked intersection points are only potential as the cutting is a progressive process. Whenever an intersection point is calculated, the local area of the mesh is updated before move to next intersection point. Figure 6 illustrates the details of the process. As the cut progresses, the mesh point nearest to the intersection point is snapped to the cutting path and the cut will be generated by dividing the mesh model along the cutting path and temporarily ends here. This is especially important for visual realism when the cutting tool moves very slowly and the cut should still be updated in real time. The reoriented polygon edges continuously follow the cutting path.
For each intersection point, after the node-snapping, the next task is to open up the cut by duplicating and displacing vertexes that have been snapped along the cutting path. First, snapped points, except the starting and ending points of the cut, are duplicated twice and directly displaced at two sides of the cutting path such as vertex points $S_i$ and $S_j$ for $S_i$ in Figure 7. The displacement direction is perpendicular to the cutting path. The original snapped points such as point $S_i$ are then deleted. All polygon edges connected to the deleted points are reconnected to their duplicated points. As shown in Figure 7, edges originally connected to point $S_i$ are now connected to point $S_{i1}$ or $S_{i2}$, depending on which side of the cut they are located at. In particular, both points $S_{i1}$ and $S_{i2}$ are connected to the starting cut point $S$. After some points on the surface model are snapped, duplicated and deleted, surface triangles in the vicinity of the cut need to be updated. Triangles are updated according to the reconnected edges near the cut.

The displace vector $X_{i1}$ and $X_{i2}$ in Figure 7 is defined on the tangent plane of the original vertex. The tangent plane normal of a vertex is calculated as the average normal of its neighbouring triangles. Let $W$ be the cut opening width at point $i$, which currently is a user-defined controlling parameter, the displacement vectors of the duplicated points can be calculated as the following equations:

$$X_{i1} = \frac{W}{2} V_{co}$$
The new positions of the duplicated mesh points can be calculated by

$$X_{12} = -\frac{W}{2} V_{co}$$

where $V_{co}$ is calculated according to Figure 8 as:

$$V_{co} = \frac{1}{|V_{tool} \times V_{sn}|} V_{tool} \times V_{sn}$$

where $V_{tool}$ and $V_{sn}$ are the unit vectors along the direction of the tool travelling and the tangent plane normal at node $S_t$, respectively.

Although this cut opening method does not follow the physical law, the generated cut is unconditionally smooth with high level of realism. This method is computational more efficient than the physically based numerical integration methods where the cut is generated by the spring forces of disconnected springs. By duplicating and displacing mass points that have been snapped along the cutting path, node-snapping algorithm could generate very smooth cut without adding new elements. Figure 9 shows the cut opening results on the surface on different resolution. New edges are generated and connected to their neighbour mesh points. Figure 9(b) also shows an example of multiple cuts on the same mesh.

![Figure 9](image)

**Figure 9.** Surface cut on the mesh plane

### 3.4. Direct volume deformation

We apply the direct volume deformation algorithm (Divod Chainmail) to simulate the gutter on the cutting site. The Divod Chainmail is based on the Chainmail algorithm and the Enhanced Chainmail algorithm.

The main advantage of the 3D Chainmail algorithm is its performance as each element has to be processed at most once. This allows the algorithm to work on a large data set and still produce interactive response times. A topology change can be easily done by linking or unlinking elements. However, the 3D Chainmail algorithm has some major drawbacks. First of all, it is restricted to the use of rectilinear grids. It assumed at most six neighbours (left, right, top, bottom, front and back) and made assumptions about their respective positions. Secondly it only works on homogeneous data. Schill et al. introduced an algorithm to enable the modelling of inhomogeneous data [25]. The basic idea is to change the chain boundaries of the elements. The movement is governed by the shape of the boundary assigned to a chain mail element. Different types of tissue are modelled with different shapes of chain regions. The problem with the introduction of inhomogeneous chain regions is that it cannot be proven that each element only has to be processed once. Hence, the speed advantage of the chain mail algorithm is lost. Schill et al. solved this problem by using sorted lists during the neighbour movement calculation. This increases the computational cost but the algorithm is still able to produce...
interactive frame rates. Later, Christopher proposed the Divod Chainmail algorithm [22] for local
direct volume deformation. It consists of three parts: deformation, mapping and memory management.
The first part calculates the deformation of the Chainmail object. The second part is responsible for
mapping the original volume data to the Chainmail object and mapping the deformed Chainmail object
back to the volume data. The third part handles the loading of the necessary portions of the Chainmail
object into memory. A typical volume contains 512X512X64=2^24 elements hence the memory used to
model the entire volume is huge. The Chainmail algorithm calculates local deformations hence it is
sufficient to only load the affected parts. A memory management system allows holding only those
parts in memory which are required for the calculation. In addition, each voxel carries its own set of
parameters, and this allows both homogeneous and heterogeneous volume to be handled in the same
way. Figure 10 is a closed-up illustration of the process of pulling apart the tissues along the cut by
Divod algorithm.

![Figure 10](image)

**Figure 10.** Direct volume cutting (a) a thin layer is cut through the sphere dataset, (b) right side is
pulled away, (c) left part is pulled away and (d) left and right sides are being pulled away
simultaneously and formed the cut groove

### 3.5. Direct volume rendering and mixing

Direct volume rendering (DVR) has become a popular technique in visualization, which allows
scientists to gain insights into their data set through the display of materials of varying opacities and
colours. Although the volumetric data set is interpreted as a continuous function in space, for practical
purposes it is represented by a uniform 3D array of samples. In graphic memory, volume data is stored
as a stack of 2D texture slices or as a single 3D texture object. Each voxel corresponds to a location in
data space and has one or more data values associated with it, so classification of these voxels to assign
colour and opacity is critical in obtaining useful visualizations that help to provide understanding into a
data set. This classification function is known as transfer function and plays an important role in
volume rendering to correctly interpret the volumetric content.

A variety of DVR methods exist, but all are based on the idea that voxels are assigned a colour and
a transparency mask. In this project, we choose 3D texture mapping volume rendering. Texture-based
volume rendering techniques perform the sampling and compositing steps by rendering a set of 2D
gameometric primitives inside the volume. Each primitive is assigned texture coordinates for sampling the
volume texture. The proxy geometry is rasterized and blended into the frame buffer in back-to-front or
front-to-back order. In the fragment shading stage, the interpolated texture coordinates are used for a
data texture lookup. Next, the interpolated data values act as texture coordinates for a dependent
lookup into the transfer function textures. Illumination techniques may modify the resulting colour
before it is sent to the compositing stage of the pipeline.

The surface and volume models are integrated together by putting the surface and the volume into a
same scene and ensuring their coordinates coincident. By doing this, we have implemented the hybrid
model in a simple, effective and flexible way. The surface and volume could be rotated and
manipulated as a whole; meanwhile they can be treated separately when necessary. For example, when
cutting on the surface, we can turn the haptic of the volume model off so that the underneath volume
model would not affect the surface cutting.

### 4. Implementation and results
The proposed hybrid cutting techniques are implemented on a 2.4 GHz dual-CPU workstation with 2 GB memory and a high-end graphics card. The whole system is based on the open source H3D API and Volume Haptics Toolkit (VHTK). Both are created by SenseGraphics AB. VHTK extends H3D API by introducing the scene-graph nodes necessary for loading volumetric data, handling and processing the data and for using the data to produce both visual and haptic feedback.

SensAble Technologies’ PHANToM is used to provide force reflectance and feedback for haptic interaction. The PAHNToM provides 6 degrees of force feedback device allows the user to explore object models using the sense of touch. Eventually, the force feedback device will provide valuable sensory feedback to the surgeon during simulation of tissue deformation and cutting.

In order to maintain the consistency between the surface and volume, we use the marching-cube algorithm to extract the iso-surface from the volumetric datasets as our surface mesh. Two examples of surgical cutting simulation on the surface are shown in Figure 11. Figure 11(a) shows the screen shot of a progressive surgical cutting on the foot surface, while Figure 11(b) shows the surgical cutting on the female breast surface. Both surfaces are obtained by iso-surface extraction from the corresponding volumetric data.

![Figure 11. Surface cut on mesh models](image)

Our cutting simulation system has been applied on dataset from CT scan of a female upper torso to simulate the spine surgery. The size of the torso data is 384x384x240. The volumetric data is first shown through direct volume rendering as in Figure 12: (a) shows the whole volume, while (b) shows only the bones within the volume. The first step in a spine surgery is to cut the skin, which is then opened to reveal the second layer, consisting of fat and muscles. An incision is next made on the second layer to display the spine bones. Figure 13 illustrates the whole process.

![Figure 12. Direct volume rendering](image)

![Figure 13.](image)

(a) A closer view of skin cutting, (b) Skin deformation: widen up the cut and the wrinkles are formed (c) A thin cut on the volume (d-f) Volume deformation: the muscles are pulled away to reveal the underneath bones.
Figure 14(d) shows a picture of spine surgery from a commercial company CREAPLAST [32] and Figures 14(a) to (c) show some snapshots of our simulations. It can be seen that our system is able to provide a realistic environment for the junior surgeons to practice the spine surgery at a low cost and, as a result, improve patient safety.

![Figure 14](image)

**Figure 14.** (a-b) Different volume opening width to reveal the underlying bone, (c) The simulation result, including skin incision and volumetric deformation, and (d) An illustration spine surgery from CREAPLAST.

Our system is also applied to simulate the bone drilling procedure in a knee surgery. We use a knee dataset from CT scan. The volumetric data size is 186X229X305. The data is segmented into bone, muscle, fat and skin with the designed transfer function as illustrated in Figure 15. Bone drilling involves material removal rather than material deformation. The drill effect is implemented by removing the voxels which are located inside the haptic stylus. The simulation of a drill procedure is illustrated in Figure 16. From the final result of drill effect in Figure 16(c), and with the integration of the haptic device providing force feedback and possible simulation of blood on the bone, we can say that our system is capable of offering the potential of a realistic, safe, controllable environment for novice doctors to practice surgical operations, allowing them to make mistakes without serious consequences.

![Figure 15](image)

**Figure 15.** (a) Different materials of the knee are rendered by specific transfer functions, (b) only bone is displayed and (c) anatomy illustration of knee (d) CT knee image.

![Figure 16](image)

**Figure 16.** (a) the cutting process on the surface, (b) the cut is widened up to reveal the underneath bones and starts to use the haptic device as the tool to drill on the bone and (d) the final result after the drill procedure.

5. Conclusion and future work
In this paper, cutting techniques are proposed for surgical simulation on the hybrid deformable models of surface and volume. 3D node snapping and topology modification approaches are presented to generate the smooth surface on the outer surface. The cut can be manipulated and widened up to present heterogeneous interior structure and material properties of the underneath volumetric deformable models. The proposed cutting techniques can enhance the fidelity and realism of surgical simulation. Our closest goal of future work is to use two haptic devices in our system, so that the second haptic device can assists the manipulation of the object, either by “holding” the mesh or by affecting the manipulation directly. The blood produced during surgeries will also be simulated in the future.

6. References


