Creating Continuous Force Feedback for Haptic Interaction of Volume Data Sets

Y. Liu S. D. Laycock
School of Computing Sciences, University of East Anglia
Norwich, NR4 7TJ, UK
{yu.liu|s.laycock}@uea.ac.uk

ABSTRACT

Interacting with volumetric models via a haptic device presents an effective way of perceiving details concerning the models internal structures. Approaches to facilitate this range from interacting directly with the volume data to interacting with a polygonal surface derived from the data. Previous approaches have utilised a force field to provide continuous forces such as the Force-Map method which assigns a force vector at any position in the virtual environment. Nevertheless, the Force-Map method is still limited in simulating fast moving drilling due to the fact that there are no forces inside the volume. It suffers from a pop through problem when the virtual drill quickly moves against the volume object. To circumvent this problem, the work presented in this paper introduces a Level-Box method to improve the Force-Map method by encoding the object's internal area into a number of levels which not only enables the user to touch the volume object by using a Force-Map, but also accelerates the Force-Map update procedure when drilling. Users can select from a variety of virtual tools to gain continuous and smooth force feedback during the drilling of volumetric data which increases the applicability of the approach.

Keywords

Volume haptics, Marching cubes, Force-Map haptic rendering, Level-Box,

1 INTRODUCTION

The potential for the use of volumetric data in medical applications has been well established. Recent developments in graphics accelerator cards have enabled systems to render large and complex volumetric data sets in a variety of different rendering styles, aiding the observer's perception of the data. Previous work in interactive simulation of volumetric data has focused primarily on visualization. By integrating haptic technology, an important emerging area related to volumetric visualization has developed to build up a visual haptic system which enables the user to interact with the volume data via a haptic feedback device. The visualizations that were linked with haptic feedback devices to enable the user to touch the volumetric data were introduced in 1993 by Iwata and Noma. They used their approach for the haptic interaction of data produced in Computational Fluid Dynamics. In this case a force could be mapped to the velocity and torque mapped to the vorticity [5]. Virtual Sculpting systems linked to haptic feedback devices have been available for many years; however, these often do not ensure the modified data remains faithful to the characteristics of the original volumetric data. In this paper, a Level-Box approach to improve the Force-Map haptic rendering method for drilling into surfaces based on the volumetric data is presented.

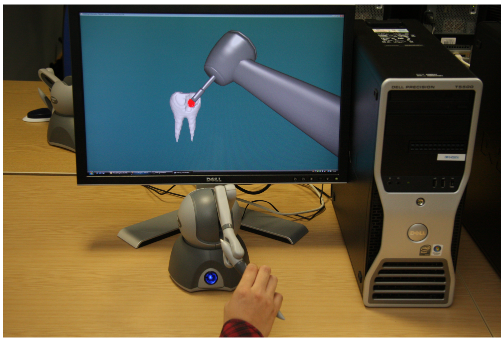


Figure 1: The visual-haptic system illustrating drilling into a volumetric object constructed from CT data.

The major objective for the design of the visual haptic system is to gain a fast haptic and graphic refresh rate at which the calculations must be efficiently performed. Based on the results of analyzing human factors, an update rate of 1KHz is required in order for a user to perceive stable and smooth haptic feedback from the visual haptic system. This is in contrast to the visualization which must update at approximately 30Hz to ensure the graphic scene is perceived as a smooth and continuous animation. If the haptic update frequency is lower than 1KHz, an obvious vibration can be felt from the haptic device. One objective of this work is to create a system which can accurately render volume data at sufficient rates for both the visualization and the haptics. For the field to move beyond today's state of the art, researchers must surmount a number of technological barriers. Firstly, the volume data updating algorithm must be fast, especially considering the fact that the surface representation of the volume data may be constructed from millions of triangles. Secondly, the haptic feedback should be rendered such that when the probe point is moving across the voxel boundaries a continuous force is returned to the user. Lastly, since the haptics and visualization calculations will be performed in separate threads, mechanisms are required to ensure that each thread can be updated in a safe manner.

2 PREVIOUS WORK

A large proportion of the previous volume haptic rendering approaches have concentrated on the use of a surface-based haptic rendering technique. An intermediate surface can easily be extracted using Marching Cubes to enable forces to be calculated utilizing a standard constraint-based method [15, 4]. However, this suffers from stability problems which occur when the surface is updated. This motivates researchers to develop algorithms which directly haptically render the isosurface extracted from the volumetric data. The direct volume haptic rendering approach is capable of providing a way to generate force feedback directly from the volume data without extracting an intermediate representation. Even though it is able to represent the force at any position in the volume data, the haptic feedback generated by this method suffers from force instabilities since it is difficult to properly decide the rendering parameters in the force function. This is especially the case when the function is changing during the process, such as when drilling or milling, in real applications. Moreover, forces may vary significantly in strength and direction which sometimes can not be represented by a simple mapping method.

Morris et al. [12] simplifies the computations for drilling through the use of another point-shell method to compute haptic interactions and bone erosion for spherical drill bits. In contrast to the work of Pflesser et al. [13], Morris et al. use the data within the spherical tool to perform bone removal as opposed to sampling points on the tool's surface. Both of these approaches limit the user to drilling with a spherical drill. Eriksson et al. [2] proposed a haptic milling surgery simulator using a localized Marching Cubes algorithm for the visualization. To improve the stability they employed a direct haptic rendering method with mechanisms to remove fall-through issues. The data inside the virtual drill is set to a vector pointing to the centre of the voxel. The output force is the sum of all those vectors. This approach works well when the drilling tool moves in a small area, but a "kicking" would result when the haptic test points move across the cubes' boundaries.

A Force-Map method is proposed by Liu and Laycock [6] to solve these problems which encode the whole virtual 3D space in an invisible map for haptic rendering and is able to generate smooth force feedback. It allows arbitrary shapes of drilling tools. But simulators are still limited to haptic rendering methods which use the surface based haptic rendering approach for touching the object. What is more, the force calculation suffers from the pop through problem due to the Force-Map only being calculated near to the surface. This is particularly likely to occur when the operation is performed by a fast moving drilling tool. In order to alleviate these issues, the work presented in this paper introduces a Level-Box method to improve the Force-Map haptic rendering algorithm which enables the visual haptic system to use a single approach to rendering for the standard interaction and also when drilling. Additionally, it can more efficiently update the Force-Map to gain smooth and stable force feedback during drilling into the volume data.

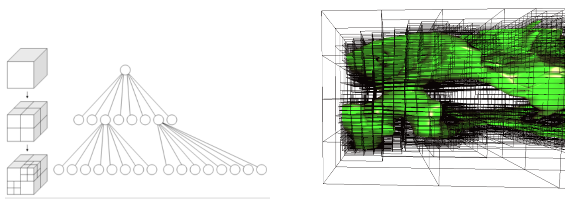
McNeely et al [10] proposed a distance field method to give an advance warning of any potential contacts between the tool and the objects. They extend the voxelization of an object beyond its surface into free space surrounding the polygonal object, marking free-space voxels with different integer values that represent a conservative estimate of distance-to-surface expressed in units of voxel size. The work presented in this paper uses a similar distance field idea to encode the non-surface free-space voxels into a number of layers according to the Euclidean distance to the surface. In contrast to McNeely's work, we encode the internal voxels of the volume object in this work with our Level-Box approach. The method is described in detail in Section 5.

Yau et al [14] also proposed a visual haptic system for training dental students by using surfel models. They use an octree based box to define the internal area of the teeth, when the drilling changes the shape of the teeth models, the internal boxes are dynamically updated which increases the octree level to create a modified surface. In spite of the advantages of using variable shapes of drilling tools, the haptic rendering update occurs under 1 kHz which does not meet the requirement of a stable haptic rendering system.

3 VOLUME DATA MODIFICATION

The volume-based representation is a natural choice for rendering a collection of digital images produced by medical scanning technologies such as Magnetic Resonance Imaging (MRI) or Computed Tomography (CT). There are a variety of graphical rendering techniques for visualizing the three dimensional data, often with options to display the material properties such as density and viscosity within the voxels. This has the potential to greatly enhance a user's performance in medical and scientific three dimensional data exploration.

When using the Marching Cubes algorithm [8], a volume can be interpreted by generating polygons representing the surface, typically constrained to a specified value of the data. But extracting the global iso-surfaces from the volume data based on Marching Cubes can be time consuming especially when the volume data is derived from many high resolution digital images. However, in this work a local Marching Cubes algorithm is employed to enable the surface to be updated efficiently. The values of the volume data surrounding the haptic stylus can be adjusted to less than a surface threshold value depending on the application. By considering the material properties of the data contained within a voxel the rate at which the data is removed can be adjusted. Once the data has been updated, the local Marching Cubes approach recomputes the surface surrounding the stylus. The volume that is updated depends on the resolution of the volume data and the shape of the tool used for the interaction.



(a) (b)

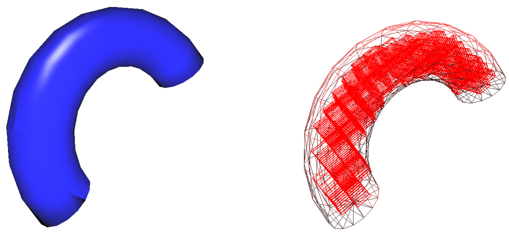
Figure 2: (a) Octree data structure, (b) Pelvis data construction using Octree data structure.

To handle large data sets, an Octree based structure [3] is employed which enables the data to be changed dynamically in an efficient manner. The Octree based structure uses a hierarchical representation of the data to efficiently detect and update localized changes to the data [11]. Each node in the octree represents a cell which contains triangles. Initially paths in the octree from the root to a leaf (voxel) will only be created if triangles forming the surface reside in the voxel, Figure 2. If the haptic stylus reaches a region and edits the data where no surface triangles are present then a new surface is likely to result. At this point the octree is updated by traversing from the root to the leaf containing the modified data, creating any new cells for the octree that do not previously exist. If the data changes such that an octree cell no longer contains triangles on the surface, then the triangles and octree cells are removed from the structure.

The efficiency of the approach is affected by the chosen depth of the octree. There is a trade-off between the quality of the visualization and the efficiency of the approach. If a small octree depth is used fewer voxels containing large triangles will result, which can often exhibit undesirable edge aliasing. Conversely, too many voxels caused by higher octree depths will increase the computational load of updating the surface during tool-object intersection. The Octree depth selection also depends on the size of the volume data. If the grid is too small then the visualization is more complex when dealing with a huge number or a large area of volume data.

4 SURFACE EXTRACTION AND MODIFICATION

The visual haptic system presented in this paper is able to function with an arbitrarily shaped drilling tool composed of polygons. This extension strives further than other work which only employs simplistic objects, such as single spheres or cylinders as the drilling tools represented by implicit functions.



(a) (b)

Figure 3: (a) Original Polygonal tool, (b) Identification of internal boxes via flood fill.

A grid of cells is constructed to encompass the whole object. Then a flood fill algorithm can be used to determine the cells that are inside the virtual tool. This method starts by choosing a cell known to be inside the tool object. Subsequently, it iteratively checks the 26 surrounding boxes until the boundary ones are reached. The approach results in all the interior boxes being labelled as interior. Figure 3 shows the steps for voxelising the internal volume of an arbitrary polygonal tool. The scale of the tool may also be easily adjusted to satisfy the specific requirements of a given application.

During the running of the program the polygonal tool interacts with the object derived from the volume data. To be able to effectively modify the data whilst drilling the volume, data points within the tool's bounding box are tested to determine if they are inside the tool's volume. Firstly, each data point must be checked with the three dimensional grid of cells to detect if the point is

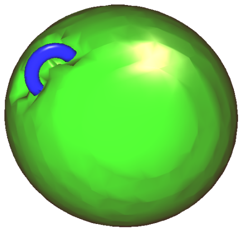
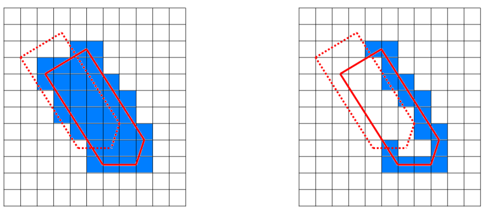


Figure 4: Polygonal tool and object interaction.

either in a boundary or interior cell. If a data point is located in a boundary cell, then it will be further checked against the tool's surface triangles located in the cell. After these steps, the values of all the data points inside the tool will be modified. After the data has been changed, the bounding box volume around the modified data points can be utilised to perform a local Marching Cubes algorithm to generate a new surface from the modified volume data.

The efficiency of the method discussed above largely depends on the size of the tool. The larger the tool used to interact with the data, the more voxels that need to be updated and recalculated by the Marching Cubes approach. This limits the use of the complicated tool implementation. Typically the haptic stylus moves slowly during drilling, especially when the tool interacts with rigid objects such as bones. The volume of data that must be changed between the adjacent graphic frames may differ by only a small amount, or indeed maybe exactly the same when the drilling tool does not move across a small voxel.



(a) (b)

Figure 5: The red outlines represent the tools between two adjacent graphic frames when drilling. The dotted outline represents the drill tool at the previous frame, whilst the solid outline represents the drill at the current frame. The blue boxes represent areas that need to be calculated by the Marching Cubes algorithm. (a) represents the full updata, whilst (b) illustrates our approach.

In this situation, it is not necessary to update the whole bounding box in each graphic frame because of the largely overlapping area. Alternatively, the update step can only consider the new area compared to the data area in the previous frame, as shown in Figure 5(b), which avoids calculating the overlapping voxels twice in two frames. By using this method, the computation of the tool-object interaction is dramatically improved even when dealing with large polygonal tools. First of all, the modified data is detected for later use. Then the voxels containing the modified data are chosen to regenerate the new surface, as shown in Figure 5(b).

5 HAPTIC RENDERING

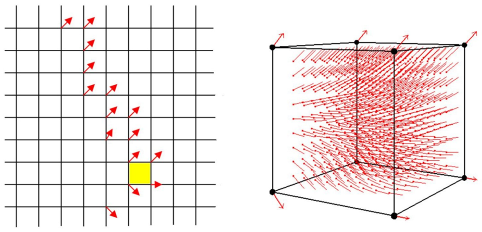
5.1 Force-Map Algorithm

The haptic rendering method described by Eriksson et al. [Eri05] suffers from force discontinuities when the tool moves between the encoded cubes. Sample points in this work are tested for contact with the volume data. Given a sample point position, a vector calculated from the occupancy force-map can be output. By using this method, the force feedback is stable and smooth even though it has a similar force cube encoding system. The force vectors stored in the data are calculated based on the local surface, which also benefits from the advantages of the surface based haptic rendering approach. The synchronisation of updating the graphic and haptic loops enhances the fidelity of the virtual visual-haptic system when applied to real applications. The following steps outline the Force-Map haptic rendering method adopted for a surface representation of dynamically changing voxel data.

Initially all the normals of the triangles contained in each octree leaf node (voxel) are averaged to result in a single force vector representing the data in the voxel. The larger the voxel is, the more volume data points lie within it. Additionally, only the data inside the voxel is assigned to a force vector while others are set to none. After this initialisation step, all the data near to the surface is set to a force vector which approximately equals the closest surface normal.

When the surface is updated in the haptics thread the data points that are found to lie inside the new voxel are set to a force value based on the triangle's face normal. If there is more than one triangle in the voxel, the averaged face normal will be used. Some force values in the old surface might also need to be updated since the triangles forming the surface in the voxel may have changed.

The force vectors stored in the data must be combined appropriately before being returned to the haptic device. When the virtual drilling tool moves into the volume data, a haptic test point checks the surrounding eight data values in the three dimensional space. These eight data values are referred to as the force cube in this work. The corners of the force cube contain the force vectors stored in the data. Tri-linear interpolation is employed here to enable an interpolated force vector to be calculated for any position inside the force cube. Another advantage of using the tri-linear interpolation method is that the haptic test point can be



(a) (b)

Figure 6: Force-Map haptic rendering, the red arrows represent the force vector. (a) The yellow square indicates one force cube displayed in two dimensions. (b) The same single force cube in three dimensions.

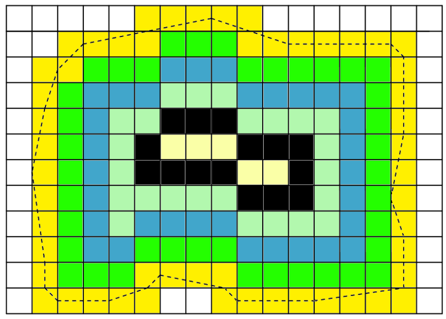


Figure 7: Level-Box construction.

smoothly moved from one force cube to another without any force discontinuities occurring between them.

5.2 Level-Box Method

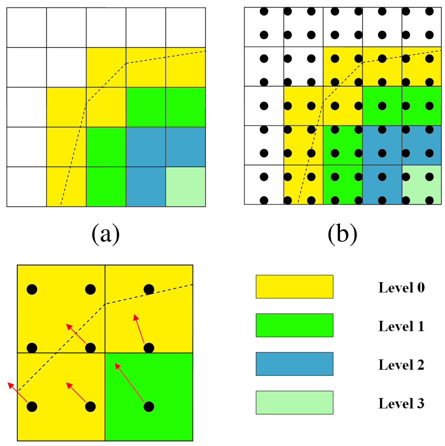
In our previous work [7], two different haptic rendering methods are employed depending on the user interaction with the volume data. When touching, a surface based method is employed. The Force-Map method is only used for the drilling. The system needs to switch between two totally different haptic rendering methods which can cause problems with regard to the consistency of the forces. Previously the Force-Map method only set force vectors close to the surface, preventing it from being employed when the user is touching the surface. If the user quickly pushes the tool toward the volume object, it will pop through the Force-Map.

In order to overcome this problem and enable the system to use one haptic rendering method, this paper introduces a new Level-Box method as an enhancement to the Force-Map approach. The area inside the volume will be partitioned into different layers. The data points in each layer will be assigned a force vector. The deeper the layer is, the larger the force vector will be set to the data in that layer. Firstly, the whole volume is partitioned into small boxes which are called Level-Boxes in this paper. The size of each Level-Box matches the size of the Octree leaf used to construct it.

5.2.1 Level-Box Construction

The Level-Boxes outside the volume object are labelled as level -1, as shown by the empty boxes in Figure 7. The boxes with the surface triangles are then labelled as level 0, as shown by the yellow boxes. After that, the neighbouring boxes of level 0 are set to level 1(represented by green boxes). This step is repeated a number of times, until the level box reaches the centre of the volume and every box has been assigned to a level.

The next step is to assign a force vector to each data point. Basically, the data in the high level boxes will be set to a larger force vector. Figure 8 (a) shows one corner of the whole volume. Figure 8(b) shows the data position which is also the Force-Map corner position in 2D. The data in the higher level is set to a force vector with the direction of the average of the neighbouring lower level boxes. As shown in Figure 8(c), the yellow box (level 1) has one data point inside. The force vector direction will be decided by the neighbouring yellow boxes but with larger scale. Then the data in level 2 is decided by level 1. Following this logic, the centre data has the greatest force vector.



(a) (b)

(c)

Figure 8: . Level-Box construction. (a) A small area of the level boxes, (b) The data position in the level boxes, the black points represent the data, (c) The force in the high level green box is decided by the force in low level yellow boxes.

When the Level-Boxes are constructed, the tool is able to gain the correct force feedback. The deeper it goes into the volume object, the larger the force will be which is sent to the haptic device. The Force-Map makes sure that the force is continuous and smooth.

5.2.2 Level-Box Updating

In the Level-Box construction step, the system also sets up a link between adjacent lower and higher level boxes. The force vectors in the higher level boxes are decided by the lower level ones, thus any changes in the lower box will affect the Force-Map in the neighbouring higher level box. In this circumstance, if the drilling tools modify the surface level boxes, the interior high level boxes get updated correspondingly. This link helps the low level boxes to quickly find the high level box related to it.

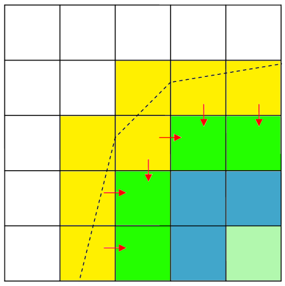
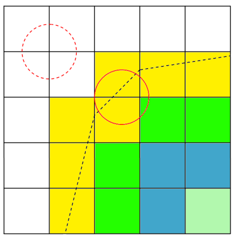
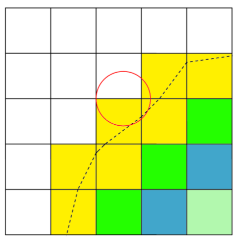


Figure 9: Level-Box link between high level and low level boxes. The arrows represent the links between a level 0 box and its neighbouring level 1 boxes.

When the tool moves towards the volume object during drilling, the surface will be recalculated based on the position of the sphere. If there is no surface in the level boxes anymore, they are changed to level -1, while the surface boxes are set to level 0. By using the link, the neighbouring ones will reduce the level because it is closer to the surface. Since the level numbers are updated, the scalar of the force vectors inside is also changed based on which level they are located in. The user is able to detect the difference of the surface after the drilling.



(a)



(b)

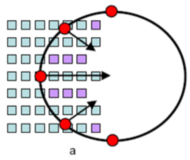
Figure 10: Level-Box updating. (a) The tool drills into the volume object. The dotted sphere represents the previous position of the drilling tool, while the solid one represents the current position, (b) The surface is updated and the level 0 boxes are changed. Consequently, the high level boxes are affected.

5.3 Multi-point Haptic Rendering

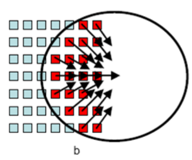
For any real application, drilling with a single point does not lead to a realistic result. An approach involving multiple test points approximating the drilling tool is usually preferred. In this work, a number of hapic points are distributed approximately around the surface of the drilling tool. At each time step, each haptic point is tested in the constructed Force-Map to calculate the contribution to the overall haptic force.

5.4 Arbitrary Tool Haptic Rendering

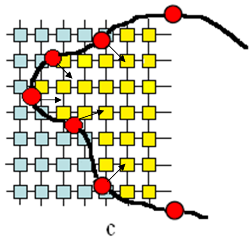
Pflesser et al. [13] proposed a haptic system for virtual temporal bone surgery which uses a modified version



(a)



(b)



(c)

Figure 11: Arbitrary tool haptic rendering. (a) Petersik's et al. [13], (b) Morris et al. [1], (c) Arbitrary tool for changing data. Red points represent the haptic test points. Yellow points represent the data removed by the of drilling tool.

of the Voxmap-Pointshell algorithm [9]. Their approaches sample the surface of the drilling instrument and then generate appropriate forces at each sampled point. A number of samples are distributed around the drill and a ray-tracing approach is then employed to calculate the force vectors towards the tool centre, which can subsequently be combined to generate the overall force returned to the haptic feedback device. The ray tracing algorithm has the potential to miss voxel data located between two rays due to an insufficient sampling as Figure 11(a) shows. Morris et al. [1] also present a method which calculates the force by counting the data points inside the tool. The force direction points to the centre of the drilling tool. Unfortunately, the haptic rendering method only allows sphere drilling (Figure 11(b)). In this work, the multiple points are located on the surface of the tool to calculate the force in the Force-Map respectively. All the force vectors inside the drilling tool are set to none and when the tool touches and drills the volume the next time, the user can detect the previously modified area.

5.5 Multi-Layer Rendering

In many applications, the properties of the simulated materials differ depending on the location being drilled. This is particularly the case in medical and dental applications where the material properties of each voxel must be considered. For example drilling through soft tissue should be very different to drilling through rigid bone. We demonstrate that the Force-Map haptic rendering method can be extended to use Multi-Layer volume data so that the trainee can feel underlying structures and material properties, such as teeth and bones. In detail, the Force-Map method can easily incorporate this issue by simply setting a scaled force vector where the scaling factor is related to the neighbouring voxel data.

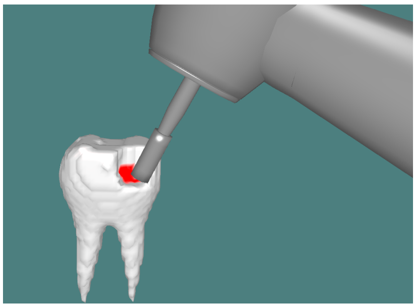


Figure 12: Multi-Layer haptic rendering.

5.6 Tangent Force Rendering

In order to enhance the force fidelity, this work also implements the tangential force on tools which is an important property of the drilling application by using the Force-Map haptic rendering algorithm. The direction of the tangent force is opposite to the tools rotation direction on the surface of the volume object.

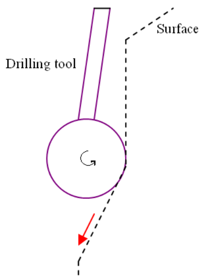


Figure 13: Tangent force implementation.

The tangent force also depends on the drilling speed of the tool and the properties of the drilling material. This haptic system allows users to choose a range of the haptic drilling speed from 200000 R/min to 400000 R/min. A faster speed will result in a greater tangent drilling force in the tangent direction of the tool-surface-contact points. Different material properties also affect the tangent force. This work allows multi-layer applications; the tangent force differs when the drilling tool moves through different materials.

6 RESULTS

Figure 14 illustrates a procedurally generated sphere along side a surface representation of a human pelvis. The surface was extracted from 87 CT slices obtained at the Norfolk and Norwich University Hospital, UK.

The work has been tested on a Two Quad Core 2.26 GHz processor PC with a NVIDIA Quadro FX580 graphics card. To provide haptic feedback a PHANToM Omni device, produced by SensAble Technologies has been employed. By using the system, a user can drill into rigid objects using arbitrary types of tools constructed from polygons.

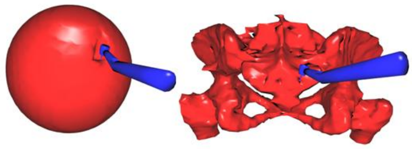


Figure 14: The left sphere-like object is created procedurally whilst the right hand image was extracted from 87 CT image slices. Each slice contains 256 X 256 pixels.

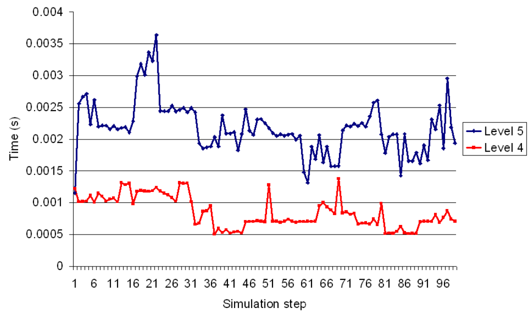


Figure 15: A graph presenting the time taken to update the surface during drilling with a polygonal tool. The blue line shows the result which uses octree level 5. The red line shows the result which uses octree level 4.

The volume of the tooth has been calculated from a data set. This data has been sampled to create a triangular surface mesh. Figure 15 shows the time required to perform the surface modification and Force-Map updates during rendering, which allows users to efficiently obtain visual cues. The Force-Map can be sampled at a higher rate in the haptic feedback loop to obtain stable force feedback. In Figure 15 the blue line shows that if the octree depth is five, the display has higher resolution but this increases the update time.

7 CONCLUSION

In this paper a Level-Box method is introduced for assisting a Force-Map haptic rendering algorithm to achieve real-time drilling of volumetric objects. In order to gain more realistic force feedback for drilling applications, arbitrary tool model selection has been implemented in this work, for tools based on implicit equations.

This paper addresses some of those challenges, specifically in the context of simulating stable and smooth force feedback. To further ensure that the fidelity of the simulator is at an acceptable level, the future work will involve the integration of drilling sound and drilling dust simulation.

A video demonstrating the program can be downloaded from the following link. http://www.urbanmodellinggroup.co.uk/drilling.wmv.

REFERENCES

[1] E. Ruffaldi E, D. Morris D, T. Edmunds T, F. Barbagli, D. K. Pai, S. Superiore, and S. Anna. Standardized evaluation of haptic rendering systems. *Proceedings of the 14th IEEE Haptics Symposium*, 2006.

[2] M. Eriksson, H. Flemmer, and J. Wikander. A haptic and virtual reality skull bone surgery simulator. *Proceedings of World Haptics*, March 2005.

[3] J. D. Foley, A. V. Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principle and practice*. Addison-Wesley, 1996.

[4] C. Ho, C. Basdogan, and M.A. Srinivasan. Haptic rendering: Point and ray based interactions. In *Proc. of the Second PHANToM Users Group Workshop*, 1997.

[5] H. Iwata and H. Noma. Volume haptization. In *Proc. of IEEE Symp. on Research Frontiers in Virtual Reality*, pages 16–23, 1993.

[6] Y. Liu and S.D. Laycock. The force-map haptic rendering algorithm for drilling into volume data. In *WSCG 09*, pages P17–20, 2009.

[7] Y. Liu and S.D. Laycock. A haptic system for drilling into volume data with polygonal tools. In *TPCG 09*, pages 10–16, 2009.

[8] W. E. Lorenson and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM SIGGRAPH*, pages 163–169, 1987.

[9] William A. Mcneely, Kevin D. Puterbaugh, and James J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *ACM SIGGRAPH*, pages 401–408, 1999.

[10] William A. Mcneely, Kevin D. Puterbaugh, and James J. Troy. Six degree-of-freedom haptic rendering improvements. In *Haptics-e,3*, 2006.

[11] M.Eriksson, M.Dixon, and J.Wikander. A haptic vr milling surgery simulator using high resolution ct data. *Proceedings of Medicine Meets Virtual Reality*, 14:138–144, 2006.

[12] Dan Morris, Christopher Sewell, Nikolas Blevins, Federico Barbagli, and Kenneth Salisbury. A collaborative virtual environment for the simulation of temporal bone surgery. In *In MICCAI*, pages 319–327, 2004.

[13] A. Petersik, B. Pflesser, U.Tiede, and K. H. Höhne. Haptic rendering of volumetric anatomic models at sub-voxel resolution. In *10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, page 66, 2002.

[14] H. T. Yau, L. S. Tsou, and M. J. Tsai. Octreebased virtual dental training system with a haptic device. *Computer-Aided Design & Applications*, 3:415–424, 2006.

[15] C. Zilles and J. Salisbury. A constraint based godobject method for haptic display. *IEEE Conference on Intelligent Robots and Systems*, pages 146–151, 1995.