Approximate reconstruction of meshes after material removing

Félix Paulano
Universidad de Jaén
Dep. Informática
Campus Las Lagunillas
23071, Jaén, Spain
fpaulano@ujaen.es

Juan J. Jiménez
Universidad de Jaén
Dep. Informática
Campus Las Lagunillas
23071, Jaén, Spain
juanjo@ujaen.es

Antonio Martínez
Universidad de Jaén
Dep. Informática
Campus Las Lagunillas
23071, Jaén, Spain
amalbala@ujaen.es

Rubén Pulido
Universidad de Jaén
Dep. Informática
Campus Las Lagunillas
23071, Jaén, Spain
rpulido@ujaen.es

ABSTRACT
Boolean operations are complex, so it is difficult to perform them in real time. Sometimes, the goal is only to reconstruct the model. In that case, accuracy is not too important and other approaches can be performed. However, the reconstruction of the model must satisfy some requirements like smoothness or velocity. In this paper, a method to reconstruct a model after a cut is presented. This method can be applied to simulate medical procedures, such as the rejection of damaged tissues, or applied to virtual sculpting. A haptic device has been used to test the effectiveness of the method. Tests have shown that the elimination and the reconstruction are performed in real time.

Keywords
Mesh reconstruction, Boolean operation, Material removing, Concavity regeneration, Surgery simulation, Virtual Sculpting, Simulation tools

1. INTRODUCTION
Boolean operations allow performing unions, intersections, differences and other operations between two solid models. This kind of operations is traditionally based on the marching cubes algorithm [NY06]. These operations are very costly because they are extremely complex and its result aims to be exact. For that reason, it is very expensive to use them in a real time application. In simulations, other approaches faster than Boolean operations must be used and accuracy is not usually a key factor.

Our aim is to perform difference operations between two solid models (A-B) in an approximated way. Specifically, the first solid (A) has at least a part slimmer than the second one, and the second solid (B) act as a tool to remove portions of the first solid (figure 1). This is because the solid must be inside the tool so that it can be cut.

This kind of operations is commonly used in virtual simulations, such as virtual surgery or virtual sculpture. In this case, Boolean operations cannot normally be used because they are so costly. New approaches that perform this operation in real time are needed. These new approaches do not have to be exact, but they have to allow real time interaction.

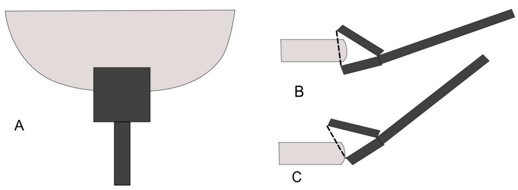

Figure 1. Example of application of the method. A – An aerial view. B, C – Side view of two separate cases.

This paper presents an approach to perform an approximate reconstruction of a mesh. This approach allows realizing a reconstruction that can be applied to real time simulations. The obtained reconstruction is not exact, but in some cases, such as virtual surgery or sculpture, it is an advantage over other approaches. This is because our method obtains a smooth surface after the cut, simulating the real cut of a certain tool with specific types of tissues. In the next section, some works recently published, related to this research area, will be described. Then, steps of the simulation will be enumerated in a general way and the method will be described in detail. In addition, some special cases will be described. The fourth section will show the simulations that have been performed in order to apply the method. Finally, the simulation results will be presented as well as a brief conclusion.

2. BACKGROUND

In the bibliography, there are some recently works that propose new Boolean operations approaches.

Wang [Wang10] presented a method to perform approximate Boolean operations of two freeform polygonal meshes using Layered Depth Images (LDI). A trimmed adaptive contouring method is developed to reconstruct the mesh surface from the LDI samples near the intersected regions and then suture it to the boundary of the retained surfaces. This method can perform Boolean operations of freeform solids in a few seconds. Jing et al. [JWBC09] proposed an approach to perform Boolean operations on polygonal meshes that can be applied to both closed meshes and open meshes. They use a collision detection algorithm based on OBB trees to speed up the intersection between each two triangles. Then, the intersection region is obtained from the intersected triangles and the intersection segments. Zhou et al. [ZWSW*10] proposed a Boolean operation method based on L-Rep model of 3D entities. The speed of the method is improved by changing the three dimensional spatial analysis into a one-dimensional calculation.

These approaches can perform a Boolean operation in a several tenths of a second. However, it is not enough to apply them in real time simulations. For that reason, other methods that allow performing the approximated operation in real time must be developed.

There are many works that perform a mesh reconstruction using a method based on the Delaunay triangulation [Shewchuk02]. These works perform a homogeneous triangulation. However, our aims are to develop a quick and robust method for simulating a cut so the size and the form of the new generated triangles could be non-homogeneous. Moreover, we know the approximate shape of the resulting mesh before the reconstruction; hence the performance of the method can be improved using that information. Other approaches [Frisken99] propose a linked volume representation that enables physical modeling of object interactions, such us deformations or interactive objects deformations. [NS00] presents an interactive algorithm for an interactive linear FE deformation simulation. Moreover, runtime changes of the mesh can be realized because the process requires no global precomputation.

We propose a method that allows performing an approximate mesh reconstruction after removing triangles. The difference from other approaches is that our method first performs a material removing and then reconstruct the hole of the mesh, obtaining an approximated solution near to the real cut of the tool. The result is a smooth mesh on the border of the operation (on the border of the hole) (figure 2) which is suitable for some type of operations with this type of tools (e.g. for virtual sculpting or surgical simulations). Moreover, enables real time interaction.

3. DESCRIPTION OF THE PROCESS

In order to exemplify, it has been taken into account that the mesh used to remove material is shaped like a cuboid. However, this mesh can be any shape possible, such as a sphere, a cylinder, a star, or otherwise. Hence, the method can be easily applied to other solids. Our approach can be divided in the next steps:

1. Removing triangles. The triangles of the solid A that are inside the cuboid are removed.
2. Transforming the hole. The hole created after removing triangles is transformed in a convex concavity.
3. Projecting the cuboid points. The four cuboid vertices that are nearest to the solid A are projected on it. These four projected points will be used to reconstruct the mesh.
4. Classifying the sutured points. Projected points and sutured points are classified into four quadrants.
5. Generating new triangles. The last step is to reconstruct the mesh using all previous calculations.

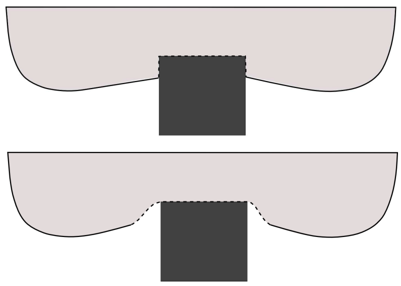

Figure 2. Top, example of a mesh reconstruction using a Boolean operation. Bottom, example of a mesh reconstruction using our approach.

After applying the method, the cut is simulated and the resulting mesh is closed and approximated to the Boolean operation between the trajectory of the tool and the original mesh. In addition, the appearance of the cut is smooth.

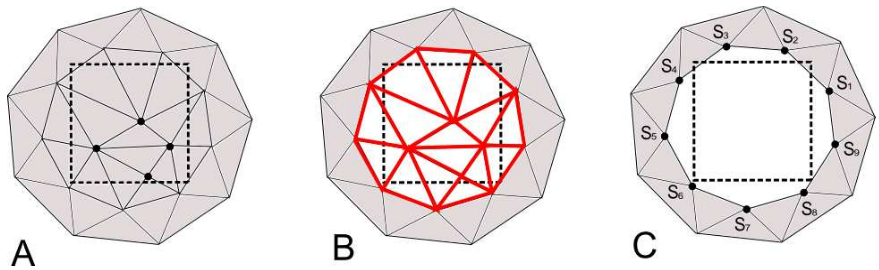

Figure 3. 2D schema of the method used to remove triangles. A – Points to remove. B – Triangles to remove (red). C – Suture Points (s_1, …, s_9)

Removing triangles

Before reconstructing the mesh, triangles of the solid A that are inside the cuboid must be removed. In order to achieve this, triangles that have at least one of its vertices inside the cuboid are deleted.

To simplify this procedure, a spatial decomposition has been performed using a tetra-tree [JFSO06]. This data structure is built in an initial step, so it does not reduce the method efficiency. A tetra-tree is an hierarchical space decomposition defined in the whole space. At its first level, the tetra-tree divides the entire space into eight tetra-cones. In the following levels of the hierarchy, each tetra-cone is homogeneously divided into four new tetra-cones, as shown in figure 4. The tetra-tree is subdivided until reaching one of the following conditions:

- The maximum level of subdivisions is achieved. This level is previously defined.
- The tetra-cone to subdivide has less triangles than a threshold.

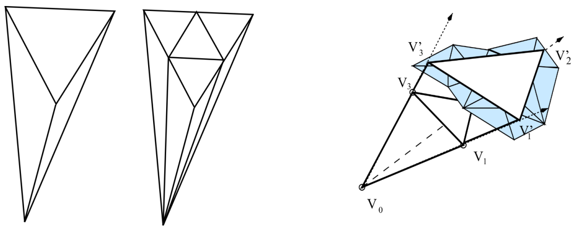

Figure 4. Left, a representation of the division of a tetra-cone. Right, a scheme that represents the bounding tetrahedra associated with a mesh.

Hence, the cudoid only have to interact with the triangles belonging to the tetra-cones where it is situated. The complexity of the calculation associated with the removing is reduced to the tetra-cone space.

This type of spatial decomposition allows us to quickly locate the nearest object part (tetra-cones) where the points are situated. The tetra-tree also allows us to discard triangles for removing in an optimal way, due to the adjustment obtained by bounding tetrahedra [JS08] associated with each tetra-cone. This is represented in figure 4.

The tetra-tree has been chosen because fits the mesh better than other approaches [JS08], such us an octree. In addition, the tetra-tree allows classifying triangles quickly and robustly because is based on barycentric coordinates. While the removed triangles are being determined, the topology of the hole is calculated. In order to achieve this, the triangles that have only a vertex inside the cuboid are used. The two vertices that are outside the cuboid are marked as suture points. In order to simplify the reconstruction, the topology of the suture points is stored, sorting them in opposite counter clock wise. This procedure is shown in figure 3.

Transforming the hole

In this step, the hole created after the elimination of triangles is transformed in a convex concavity using the topological information about its edges. The algorithm can be divided into three parts.

First, the concavities are located (Figure 5). Points that form the hole, which are marked previously as suture points, are studied in groups of three consecutive points. If the sign of the matrix determinant formed by three consecutive points is negative, a concavity is found. During this process, consecutive concavities are grouped into one (e.g. p_1-p_4 in Figure 5).

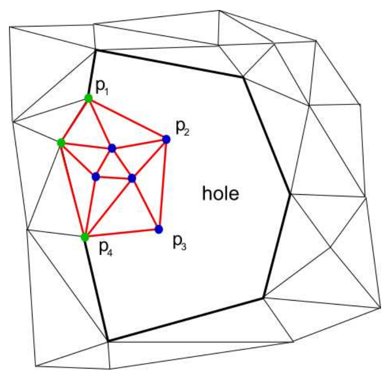

Figure 5. Schema of the method used to eliminate the concavity. Red – Triangles to remove. Blue – Points used as input in the iterations of the algorithm. They are also removed. Green – Points to reconstruct.

Second, once all the concavities are located, they are converted into convexities. First at all, the first and the last point that form the concavity are stored (p_1

and p_4). Then, the remaining points that form the concavity are processed in a loop. This loop is repeated until there are no more input points.

The loop has the following steps:

- Triangles that contain the input points are determined.
- Points belonging to those triangles are studied to check if they are inside or outside the concavity:
  - Points that are inside will be input points in the next iteration of the algorithm.
  - If two points of a triangle are outside the concavity, they will be stored as points to reconstruct and their topology will be saved.
  - Points which have been input points in this iteration will be discarded.

When there are no input points, the loop is finished, obtaining then a set of points to reconstruct. This procedure is shown in figure 5. The first and the last point of the concavity are within the set of points to reconstruct.

Finally, the points to reconstruct are used in groups of two consecutive points to generate new triangles (figure 6). The midpoint of the segment that goes from the first point of the concavity to the last point of the concavity is calculated. This point is used as a third point of each triangle.

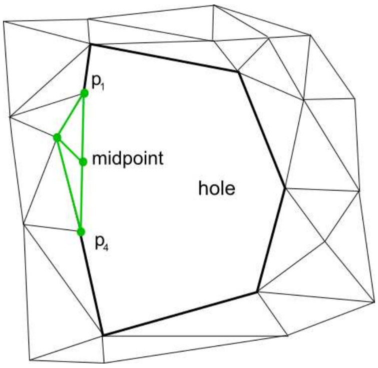

Figure 6. Schema of the hole reconstruction. Green – New triangles added.

The elimination of the concavities in the hole allows simplifying the subsequent generation of new triangles, preventing cross segments. Moreover, the cut obtained is smooth, so some simulations are more realistic than an exact approach.

Projecting the cuboid points

The four points representing the cuboid that are nearest to the solid are used to reconstruct it, so these points are projected on the solid using the algorithm proposed by [JSF10]. This is done to simulate the trajectory of the tool to perform cutting. In order to simplify the procedure, two planes representing the solid are determined. Hence, the points are projected on one of these two planes, instead of the solid.

In the general case (figure 1, B), two planes are calculated. These planes (figure 7) represent the top and the bottom of the solid so they are called upper average plane and lower average plane respectively. A medium plane is used for dividing the triangles in the tetra-cones scope in which the tool is included. Then the triangles in the top and in the bottom are used to obtain two average normals. The upper average plane is defined by the triangles whose normal is similar to the top average normal. On the other hand, the lower average plane is defined by the triangles whose normal is similar to the bottom average normal. In both cases, an error must be considered. Then, the upper points are projected on the upper average plane and the lower points are projected on the lower average plane, as shown in figure 7.

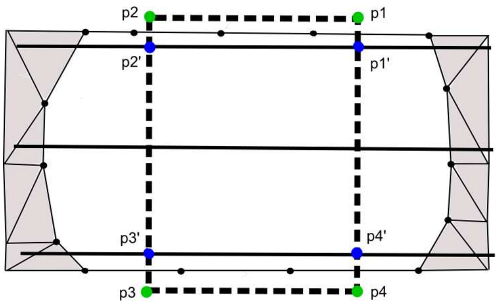

Figure 7. A 2D scheme that shows the projection of the cuboid points (p1, p2, p3, p4) on the upper and lower average planes (p1', p2', p3', p4').

If any of the points cannot be projected on its respective plane, an oblique cut is detected (figure 1, C). In a oblique cut, the part of the object to be cut is not completely within the tool. This is usually because the tool is not aligned with the object. In that case, one auxiliary plane is used to project those points. To calculate the auxiliary plane, some calculations are made in real time. The removed triangles that are situated between the upper average plane and the lower average plane are marked as central triangles. The auxiliary plane is defined by the average point and the average normal of the central triangles.

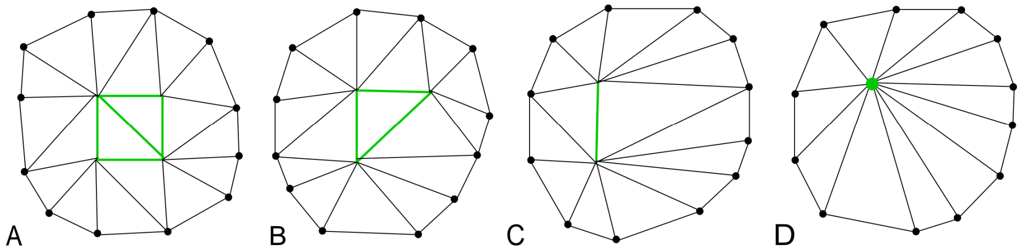

Figure 8. Different cases of generation of new triangles. The procedure depends on the number of projected points that are inside the solid: 4 (A), 3(B), 2 (C), 1(D).

After projecting them, the four projected points are used to reconstruct the solid. However, only the projected points that are inside the solid mesh are used. To check this, the inclusion algorithm by Feito [FT97] is used, because it allows determining if a point is inside a mesh without the need to perform complex calculations, such as solving a system of equations. In addition, to further reduce this problem, the previous spatial decomposition using a tetra-tree [JFSO06] is also utilized.

Classifying the suture points

In order to simplify the reconstruction of the mesh, the suture points and the projected points are classified into four quadrants. The central triangles, which were calculated in the previous step, are also utilized in this procedure. Two perpendicular planes (figure 9) that pass through the average point of the central triangles are calculated. These planes are also perpendicular to the planes forming the cuboid.

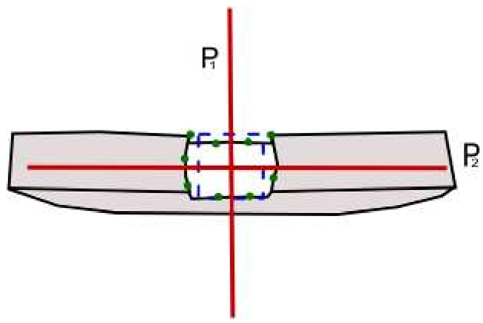

Figure 9. Representation of the two planes (red) forming the quadrants used to divide the suture points (green) and the cuboid vertices (blue).

Both planes divide the previously projected points into four quadrants, assigning each point to a different quadrant. The suture points are also classified using these four quadrants.

Generating new triangles

The last step is to generate new triangles to close the mesh. In order to achieve this, all previous calculations are used.

First, the projected points are used to build a structure. This structure will be used as a patch and it depends on the number of projected points that are inside the solid (figure 8):

- In the case that the four projected points are inside the solid mesh, this structure will be a square.
- The structure will be a triangle if there are three points inside the solid.
- If there are only one or two points inside the solid, it will not build any structure. In this case, the origin of the reconstruction will be the projected point or the segment joining the two projected points respectively.

Second, in each quadrant a sub-mesh is built. In order to achieve this, a triangle is built using each two consecutive points. The projected point assigned to each quadrant is used as the third point in each triangle. If one projected point is outside the mesh, the triangles of its associated quadrant are generated using the projected point of the upper or lower quadrant.

Finally, in order to join the sub-meshes built in each quadrant, new triangles are generated using the projected points and the boundary suture points in each quadrant.

Special cases

In our method, there are some cases that must be treated specially. These cases have been described and can be solved by adapting the general procedure.

In the general case, only a hole is created after removing the triangles. However, several holes may be generated after that (figure 10). In that case, each hole must be treated separately.

This special case is detected in the stage when the hole becomes convex. In this stage, the suture points are read in a loop. The loop finish when all the suture points are read or the first suture point is reached again. If the first suture point is reached again before all suture points are read, the special case is detected. In that case, all unread points are included for processing in a new loop. Each time the first point is founded again, a new hole is detected.

Once the holes are detected, each hole is converted into a convex shape. In order to achieve that, the algorithm explained in the previous section is applied to each hole. Once all the holes become convex, the four cuboid front points are projected on the planes as in the general procedure (figure 10). Then, all the holes are reconstructed. In order to achieve this, triangles are generated using every two consecutive points belonging to the hole.

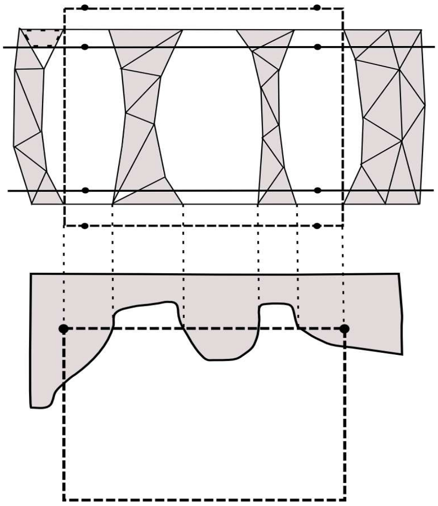

Figure 10. A special case. Three holes have been generated instead of only one. Bottom, an aerial view before cutting.

The average points of the holes are used as the third point of each triangle in the hole reconstruction, as shown in figure 11. Finally, if the two left projected points are inside the solid, the hole nearest to the left is reconstructed using the two left projected points. For that, the approach used in the general case when only two projected points are inside the solid is utilized (figure 11). Similarly, this procedure is repeated with the right projected points and the hole nearest to the right.

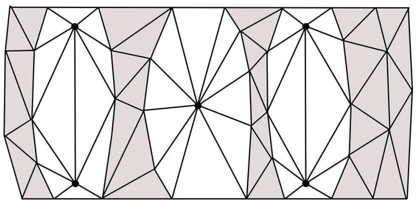

Figure 11. Generation of new triangles in a special case. The central hole is reconstructed using only a point. The others two hole are reconstructed using two projected points.

4. EXPERIMENTAL RESULTS

In this work, a method to realize an approximate reconstruction of a mesh has been implemented. The data structures and algorithms implemented allow simulating the removal and reconstruction of triangles in real time.

In order to prove the method, a virtual meniscus arthroscopy has been simulated. Specifically, a radial injury is treated. This kind of injuries is usually treated removing the damaged area [RBM09] to attempt to keep the stability. Figure 12 shows a representation of the area of the knee treated in the simulation, and a real image of the knee during an arthroscopy. Hence, in the simulation the first solid represents the meniscus and the second solid represents the surgical tool used to remove the damaged tissue.

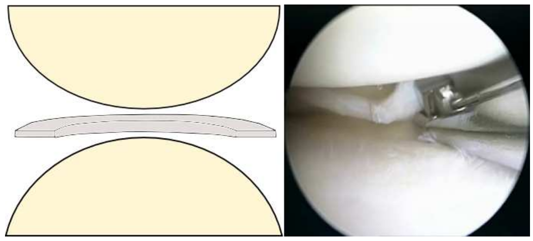

Figure 12. Left, a scheme that represents the area of the knee treated in the simulation. Right, an image of a real meniscus arthroscopy

A haptic device has been used to improve the interaction. Specifically, a Sensable Phantom Omni© and its associated software has been used. The haptic device simulates the surgical tool that is used during the intervention. The surgical tool movement is calculated through the transformation matrix of the haptic device. Moreover, the coordinates of the points that represent the device are calculated to determine the collisions with other elements. The device feedbacks a simple force based on Hooke's law when it detects a collision.

Collision detection

The collision detection [LG98] between the surgical tool and all the triangles that form the meniscus is complex. Moreover, obtaining a real time feedback is complex too, because the refresh rate of the haptic device is 1 kHz [MCL08]. The collision detection involves an intersection algorithm between the surgical tool and the meniscus.

To reduce the problem, the previous spatial decomposition of the meniscus is used, reducing the number of triangles involved in the intersection test. Even reducing the size of the problem through a spatial decomposition, it is still complex to calculate the intersection with the surgical tool. To simplify the collision detection, it only works with a number of representative points of the cuboid and the handle. Then, only these points are classified by the tetra-tree, reducing the collision detection to a point in polyhedra test [JFSO06].

The representative points of the cuboid are their vertices and central points of their faces. To represent the surgical tool handle, taking into account that it is shaped like a small cylinder, a set of points belonging to the cylinder axis are used. These points are shown in figure 13. Once the representative points are calculated, tetra-cones that contain at least one of them are obtained. These tetra-cones define the space of the meniscus where the intersection will be calculated. To calculate the intersection, the point in polyhedron algorithm by Feito [FT97] has been used.

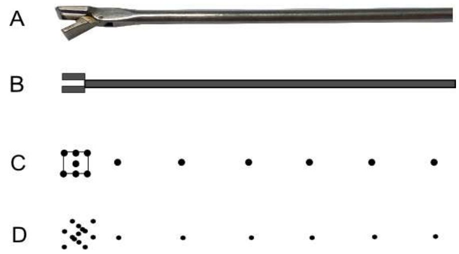

Figure 13. A – The real surgical tool. B – 2D Representation of the surgical tool. C – 2D representation of the tool points used to simplify the collision detection. D – 3D representation of those points.

The use of representative points improves the performance of the simulation, because it allows determining if the surgical tool collides with the elements involved in the simulation in real time, therefore the tool cannot pass through the meniscus.

Mesh reconstruction

One of the device buttons has been used to actuate the tool. If this button is pressed when the surgical tool is close to the meniscus, the tool is actuated, removing a portion of the meniscus. To provide feedback to the user in this action, it has applied a small force. In addition, the tool movement can be restricted as in a real intervention. Therefore, the instrument can be force to only swing through the input. Nowadays, in many cases the repair of radial meniscus injuries is treated by removing the damaged part. Our aims are to apply the proposed method to simulate the rejection of the damaged tissue. Our approach avoids to perform costly Boolean operations between elements [FHK01], giving us a real time interaction. The main advantage of using this approach, instead of Boolean operations, is that our procedure is faster [JWBC09] and response time is a key factor in haptic devices. Boolean operations can obtain more accurately results. However, the accuracy in the meniscus cut is not a critical feature. In figures 14 and 15, results of the simulation are shown. In these images, front and oblique cuts are displayed.

Results

To perform the experiment has been made a subdivision of the mesh that represents the meniscus. This procedure allows us to obtain more complex meshes to measure the performance of the method.

First, it has been measured the time it takes to reconstruct the mesh using different tetra-tree subdivisions. As shown in the table 1, until the fourth subdivision, the more the tetra-tree is subdivided, the better the times obtained are. Moreover, the time it takes to reconstruct without a tetra-tree has also been measured. The results show that the use of our method to remove and reconstruct the mesh, as well as the use of a tetra-tree, enables a real time interaction. Although the refresh rate of the haptic device should be 1 kHz [MCL08], it is impossible to push the button at this frequency, so real time interaction is achieved.

Number of triangles	Time using a level-2 tetra-tree (ms)	Time using a level-3 tetra-tree (ms)	Time using a level-4 tetra-tree (ms)	Time not using a tetra-tree
13618	12,67	8,53	5,87	17,33
25258	29,93	23,33	14,67	73,65
55864	107,07	75,33	42,2	200,67

Table 1. Reconstruction using different tetra-tree subdivisions as well as not using a tetra-tree.

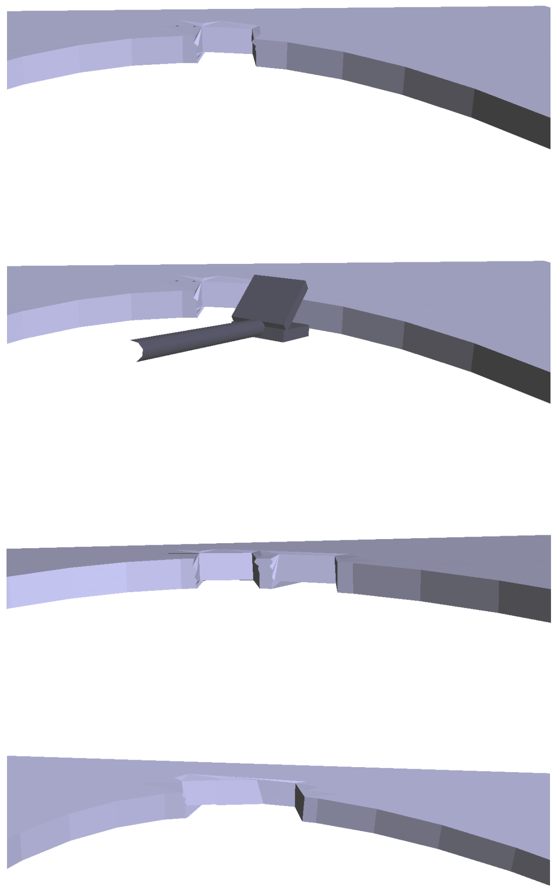

Figure 14. Some images of the simulation. They show front cuts in the meniscus.

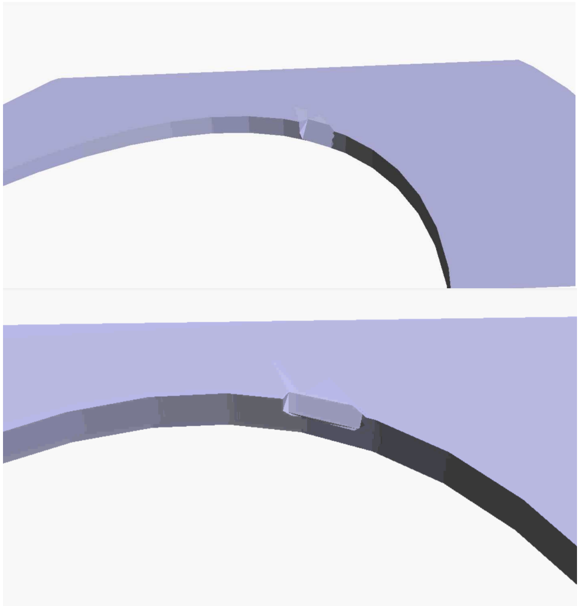

Figure 15. Some images of the simulations. They show oblique cuts in the meniscus.

5. CONCLUSION

In this paper, a method that allows performing an approximate mesh reconstruction after mesh removing has been presented. In contrast to Boolean operations, our method enables real time simulation.

To exemplify the method, it has been considered that the mesh used to remove material is shaped like a cuboid. However, this mesh can easily be any shape. For that, the projected points must be chosen according to the shape of the mesh, instead of the cuboid. The same happens with the points used in the collision detection.

To prove the method, a virtual meniscus arthroscopy has been performed. Specifically, it has been focused on radial injuries. In the future, our method can be applied to perform other simulations, such us other surgery operations or virtual sculpting.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Education and Science and the European Union (via ERDF funds) through the research project TIN2007-67474-C03-03, by the Consejería de Innovación, Ciencia y Empresa of the Junta de Andalucía through the research projects P06-TIC-01403 and P07-TIC-02773, and by the University of Jaén through the research project UJA-08-16-02, sponsored by Caja Rural de Jaén.

6. REFERENCES

[FHK01] Farin, G., Hoschek, J., Kim, M. Solid Modeling. Handbook of Computer Aided Geometric Design. Elsevier Science, 2001.

[Frisken99] Frisken-Gibson, S.F. Using linked volumes to model object collisions, deformation, cutting, carving and joining. Visualization and Computer Graphics, vol. 5, pp. 333-348.

[FT97] Feito, J.R., Torres, J.C. Inclusion test for general polyhedral. Computers & Graphics 21, pp.23-30, 1997.

[JFSO06] Jiménez, J.J., Feito, F.R., Segura, R.J., Ogáyar, C.J. Particle oriented collision detection using simplicial coverings and tetra-trees. Computer Graphics Forum 25, pp.53-68, 2006.

[JS08] Jiménez J.J., Segura, R.J. Collision detection between complex polyhedra. Computers & Graphics 32, vol 4, pp.402-411, 2008.

[JSF10] Jiménez, J.J., Segura, R.J., Feito, F.R. A robust segment/triangle intersection algorithm for interference tests. Efficiency study. Computational Geometry 5, pp.474-492, 2010.

[JWBC09] Jing, Y., Wang, L., Bi, L., Chen, J. Boolean Operations on Polygonal Meshes Using OBB Trees. Environmental Science and Information Application Technology, vol.1, pp.619-622, 2009.

[LG98] Lin, M.C., Gottschalk, S. Collision detection between geometric models: A survey. In Proc. Of IMA Conference on Mathematics of Surfaces, pp37-56, 1998.

[MCL08] Ming, C., Lin, M.O. Haptic Rendering: foundations, algorithms, and applications. A K Peters, Ltd, 2008.

[NS00] Nienhuys, H., Stappen, F. Combine finite element deformation with cutting for surgery simulations. In EUROGRAPHICS 2000.

[NY06] Newman, T.S., Yi, H. A survey of the marching cubes algorithm. Computers & Graphics 5, vol.32, pp.854-879, 2006.

[RBM09] Richmond, J.C., Bono, J.V., McKeon, B.P. Knee Arthroscopy, 2009.

[Shewchuk02] Shewchuk, J. R. Delaunay refinement algorithms for triangular mesh generation. Computational Geometry 1-3, vol.22, pp.21-74, 2002.

[Wang10] Wang, C. Approximate Boolean Operations on Large Polyhedral Solids with Partial Mesh Reconstruction. Visualization and Computer Graphics, no.99, pp.1-10, 2010.

[ZWSW*10] Zhou, L., Wang, D., Sheng, Y., Wang, Y., Li, P. Three dimensional Boolean operation based on L-Rep model. Geoinformatics, 2010 18th International Conference on, pp.1-6, 2010.