Improving Mesh Quality of Extracted Surfaces using SurfaceNets

P.W. de Bruin[1], F.M. Vos[2], S.F. Frisken-Gibson[3], F.H. Post[1], and A.M. Vossepoel[2]

Abstract

Finite-element modelling is a standard way for simulation of soft tissue deformation. For proper modelling, triangular surface models must fulfil requirements of accuracy, smoothness, and conciseness. Several techniques proposed in the literature do not meet these requirements. In this paper we extend a new technique called SurfaceNets, which can create a globally smooth triangle mesh that retains fine detail.

Keywords: *Finite elements, mesh generation, surface rendering, SurfaceNets.*

1 Introduction

1.1 Hysteroscopic removal of myomata

Over the past few years, endoscopic surgery has become a well established practice to perform minimally-invasive surgical procedures. A typical application is the hysteroscopic removal of uterine fibroids (or myomata). These benign tumors are a common gynaecological pathology causing disturbed pregnancy and excessive menstrual bleeding.

Conventionally, transvaginal ultrasonic imaging and MRI is applied to diagnose the presence and extent of myomata. The removal is typically done through a minimally invasive operation with a hystero-resectoscope. This instrument consists of a loop electrode mounted on an endoscopic camera tube. In order to create a working space for the hysteroscope, the uterine cavity is inflated with a distension medium. Next, the fibroid is removed by piecewise cutting it from the uterus.

While doing so the surgeon has to be very careful not to perforate the uterine wall. On the other hand, incomplete removal of the fibroid requires additional surgery to remedy the patient's complaints. Also, intraoperative navigation is extremely difficult because the uterus shows a homogeneous area in which there are very few orientation clues. To plan the operation the surgeon has to build a mental 3D model from 2D image slices (MRI or ultrasound). This process is hampered by the low resolution of the images, a poor signal to noise ratio and the complex geometry of the organs. Apart from the inadequate visualization, the usefulness of preoperative information is limited, as there will be considerable deformation of the internal organs during the operation.

To model organ deformation for intraoperative planning a strategy is adopted consisting of the following stages:

1. data acquisition through MRI;
2. image segmentation;
3. deformable model generation;
4. model deformation to simulate inflation of the uterus (using intraoperative calibration via 3D ultrasound);
5. enhanced intraoperative visualization.

In order to model the tissue deformation, many authors have proposed finite element representations of the relevant structures [3, 2, 9]. Initialization of such models is commonly achieved through supervised segmentation of preoperative image data. Often, the latter classification is accurate at pixel level. Using the Marching Cubes algorithm [10], the result is converted to a collection of triangles that represent the surfaces of relevant organs. This rep-

[1]Computer Graphics & CAD/CAM, Delft University of Technology
[2]Pattern Recognition Group, Delft University of Technology
[3]MERL – a Mitsubishi Electric Research Laboratory, Cambridge, MA USA

resentation is well suited to be imported in environments for finite element analysis.

Requirements which the triangular models should meet to enable proper modelling for mechanical analysis are:

- *accuracy* the representation of the organ surface geometry should be sufficiently accurate;
- *smoothness* sharp corners should be avoided as these can cause artifacts such as internal stress concentrations. Therefore, the model should conform to the smooth organ boundaries;
- *conciseness* to achieve fast response times, the number of elements (triangles) in the model should be minimal;
- *triangle quality* the triangles in the mesh should predominantly have equilateral sides to avoid visualization artifacts as well as finite element errors.

As noted previously, segmentation commonly results in a binary image (i.e., classification at pixel level). Application of Marching Cubes to the binary data results in a triangulated model that is neither smooth nor accurate (assuming that subpixel accuracy is required).

Some solutions to this problem proposed in the literature were inadequate. For example, Gaussian prefiltering of the binary image (before applying Marching Cubes) reduces the accuracy, and significant anatomical detail may be lost, while insufficient smoothness is achieved [7]. In addition, the Marching Cubes surface will typically have a very large number of triangles. To reduce the number of triangles, various mesh decimation techniques have been proposed [6, 11, 1]. However, since the decimation is applied without the context of the original data, it yields a reduction in the number of triangles and possibly and increased smoothness without guaranteeing the accuracy of the surface.

An obvious approach to fulfil the requirements is by adjusting the Marching Cubes triangles on the basis of the original greyscale data. It might be expected that the result can be decimated without sacrificing too much accuracy due to the smoothness of the mesh.

1.2 Outline

Recently, a technique called *SurfaceNets* has been described to optimize a triangle mesh derived from binary data [8]. In this article, this concept is extended to incorporate greyscale data. A number of techniques is examined to smooth triangle meshes without losing accuracy.

The paper is structured as follows. Section 2 gives a global description of the SurfaceNet technique. Then, in Section 3 the effectiveness with regard to the requirements mentioned earlier is evaluated. Finally, in Section 4 we will summarize our findings and draw conclusions.

2 Techniques

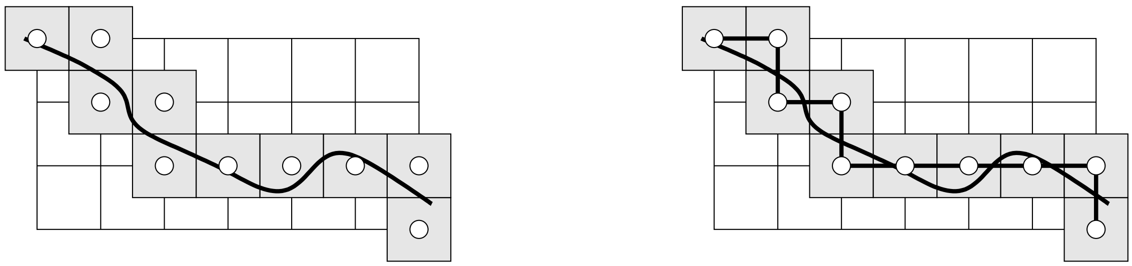
2.1 Generating a SurfaceNet

The goal of the SurfaceNet approach is to create a globally smooth surface that retains the fine detail present in the original greyscale data. The generation of a net for binary objects consists of the following four stages [7]:

1. Identify nodes of the SurfaceNet;
2. Create links between the nodes;
3. Relax node positions to achieve a globally smooth surface while satisfying constraints on node movement;
4. Triangulate the SurfaceNet.

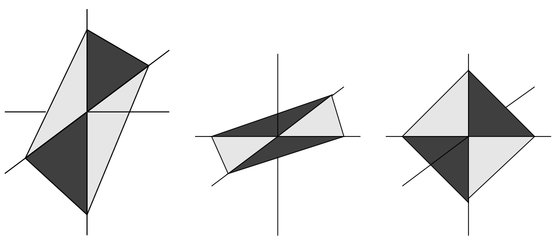
In this Section we will present a basic explanation of this strategy (largely following [7]), i.e., it is assumed that a binary segmentation of the original data exists. Subsequently, three improvements will be introduced that utilize the greyscale image data during relaxation of the SurfaceNet.

The first step in creating a SurfaceNet is to locate the cells that contain a surface node. A cell is defined by 8 neighboring voxels in the binary segmented data (see Figure 1 presenting the 2D case as illustration). If all voxels have the same binary value, then the cell is either entirely inside or entirely outside of the object. If, however, at least one of the voxels has a binary value that is different from its neighbours, then the cell is a "surface cell". The net is initialized by placing a node at the center of each surface cell (step 1). Subsequently, links are created with nodes that lie in adjacent surface cells (step 2).



(a) Before linking the nodes. (b) After linking the nodes.

Figure 1: Building a SurfaceNet. The white squares represent voxels, the thick black line represents the edge of an object and the gray squares are cells with nodes represented by white circles in the center.



(a) (b) (c)

Figure 2: Each node can be connected to 6 neighbours, creating 12 possible triangles.

Assuming only face connected neighbours, each node can have up to 6 links (corresponding to right, left, top, bottom, front and back neighbours). Once the SurfaceNet has been defined, each node is moved to achieve better smoothness and accuracy ("relaxation", step 3) subject to some constraints. In the following sections we will elaborate on this.

After relaxation the SurfaceNet can be triangulated in order to form a 3D surface model (step 4). As illustrated in Figure 2, there are 12 possible triangles joining each node to a pair of neighbours. By determining which pairs of neighbours are present, possible surface triangles are identified (see [7] for further details). The resulting triangle mesh can be rendered using standard 3D graphics techniques.

2.2 Introducing smoothness

Once a SurfaceNet has been defined (i.e., after step 2 in the above algorithm), the node positions are adjusted to "improve" the surface. This smoothing is desirable to remove furrows and "terraces" due to quantization. Let us first only consider the smoothness of the net.

One way to smooth the surface is to move every node to the average position of its linked neighbours [5]. The vector $\vec{a}$ pointing from the current position of the node $\vec{p}_{\text{old}}$ to the average position is calculated as:

$$\vec{a} = \frac{1}{N}\sum_{i=1}^{N}\vec{p}_i - \vec{p}_{\text{old}} \tag{1}$$

where $\vec{p}_i$ corresponds to the position of a linked neighbour and N is the total number of neighbours of this node.

It may well be that the average position is outside the original cell, thereby diverging from the initial segmentation. To impose conformance, the "relocation" vector $\vec{a}$ is constrained to stay within the boundaries of the original cell by the function c (see also Figure 3):

$$\vec{p}_{\text{new}} = \vec{p}_{\text{old}} + c\,(\vec{a}) \tag{2}$$

Here, c is defined to satisfy the proper constraint of the node position.

The relaxation is implemented in an iterative manner by considering each node in sequence and calculating a relocation vector for that node. The SurfaceNet is updated only after each node in the net has been visited. This

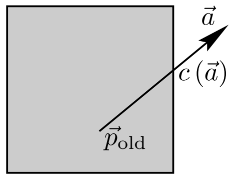


Figure 3: Position constraint of a node. If $\vec{p}_{\text{old}} + \vec{a}$ is outside the cell boundary, the function c is used such that $\vec{p}_{\text{old}} + c(\vec{a})$ is on the cell boundary.

procedure is repeated until the number of iterations exceeds a preset threshold, or when the largest relocation distance is less than a given minimum value.

2.3 Increasing accuracy

The smoothing technique described in the previous Section ignores all greyscale information in the dataset after building the SurfaceNet. The nodes shrink-wrap around the object without considering accuracy. However, this is the best approximation when the binary segmentation is the best estimate of the object.

If the object surface can be estimated to lie at an isosurface of the image data, this isosurface can be used to increase the accuracy of the SurfaceNet. Let us assume that the true object surface can be obtained by drawing an isocontour (at level I_{iso}) in the original greyscale data. For instance, in many CT based applications the Marching Cubes algorithm is used to approximate the object shape in this way. By definition, at a given point the greyscale gradient vector is perpendicular to the isosurface through that point. Thus, to enhance accuracy a node can be displaced along the gradient vector to the isosurface. This is expressed as:

$$\vec{g} = \text{SIGN}\left(I_{\text{iso}} - I\left(\vec{p}_{\text{old}}\right)\right) \nabla\vec{p}_{\text{old}} \qquad (3)$$

Here, SIGN is a function that returns the sign of its argument, d is the distance to the isosurface, $I(\vec{p}_{\text{old}})$ is the interpolated intensity and $\nabla\vec{p}_{\text{old}}$ is the normalized gradient vector in $\vec{p}_{\text{old}}$. The latter vector is obtained either by a central difference method or by convolution with Gaussian derivatives.

The node position is updated by:

$$\vec{p}_{\text{new}} = \vec{p}_{\text{old}} + c\left(d\vec{g}\right) \qquad (4)$$

As in Equation 2, c imposes a position constraint on the node to stay within the original cell. The distance to the isosurface d can be estimated by interpolation.

2.4 A combined approach

By combining the methods presented in Section 2.2 and Section 2.3, we obtain a surface that fits the isosurface of the data and is globally smooth. To combine these features, a node should be displaced to get the best smoothness within the isosurface. The combination is made by first calculating the projection $\vec{a}_p$ of the averaging vector $\vec{a}$ on the plane perpendicular to the gradient $\vec{g}$ (cf. Equation 1 and 3):

$$\vec{a}_p = \vec{a} - \vec{g}\left(\vec{a} \cdot \vec{g}\right) \qquad (5)$$

Subsequently, the combined displacement function is defined as:

$$\vec{p}_{\text{new}} = \vec{p}_{\text{old}} + c\left(\vec{a}_p + d\vec{g}\right) \qquad (6)$$

This formula combines relocation towards the isosurface with smoothing in the orthogonal plane. Again, c ensures that the new position of the node always lies within the confines of the original surface cell.

In Section 3 we will evaluate the strategies now defined.

3 Results

To evaluate the relative effectiveness of the presented techniques, the SurfaceNet is compared with Marching Cubes, which is a standard isosurface extraction tool [10]. The effectiveness of each technique will be tested against the requirements introduced in Section 1. The conciseness is not important for the comparison because Marching Cubes and SurfaceNets generate almost equal amounts of triangles. Each of the remaining requirements is measured as follows.

- A measure expressing the local smoothness of a polygon mesh is given in [12]. As a first step, the angles α_i of all triangles around a vertex are summed. If all triangles connected to a vertex are coplanar this sum is equal to 2π. A measure of the local smoothness at a vertex is defined by $|2\pi - \sum \alpha_i|$, which is then averaged over all vertices;

- A simple and direct measure for triangle quality is found upon division of the smallest side of each triangle by its largest side.

If the triangle is equilateral this expression is equal to 1;

- The accuracy is expressed by the modified Hausdorff distance that represents the mean distance of the generated mesh to a reference shape [4]:

$$H_{\text{ave}}(S_1, S_2) = 1/N \sum_{p \in S_1} e(p, S_2) \quad (7)$$

where e is the minimum distance between a point and a surface, and S_1 and S_2 are two surfaces.

Using these measures, the following experiment is conducted. Two volumes, one containing a greyscale image of a sphere and the other containing a greyscale image of a cube were created. An isosurface is extracted using Marching Cubes and using a SurfaceNet with each of the three different relaxation algorithms presented in Section 2. These surfaces are compared to the exact reference shape. Next, the number of triangles in each of the meshes is halved and quartered (using the Qslim decimator [6]) and the surfaces are again compared to the reference shape. The results of all the measurements are listed in Table 1.

In Table 1 the columns containing H_{ave} clearly demonstrate better accuracy of the SurfaceNet than Marching Cubes (e.g. compare the rows MC and SNAG). This effect is particularly evident for the cube. The averaging method alone gives rather poor results. This can be explained by two factors. First, the method disregards all information in the data. Secondly, the SurfaceNet is shrink-wrapped around the cube and the sharp corners eventually protrude during shrinking. Decimation results in slightly less accuracy (best illustrated by the sphere, see for instance SNAG).

Regarding mesh quality, no significant differences are observed for the cube. However, the sphere yields slightly better performance (compare MC with SNAG).

The most prominent differences are found in the columns “smoothness”. The SurfaceNet approach evidently gives better results.

Finally, it can be seen that the number of triangles generated with the SurfaceNet technique is slightly higher than with Marching Cubes.

In addition to the presented results, several other experiments were performed. Figure 4 shows the meshes generated by Marching Cubes and SurfaceNets from a greyscale dataset with two overlapping spheres. Figure 5 and Figure 6 respectively show a close-up and an overview of the meshes generated from a greyscale 256x256x61 MRI dataset of the adbomen of a female patient. The terracing effects are clearly visible in the Marching Cubes mesh, and much less visible in the SurfaceNets mesh. Finally, Figure 7 shows Marching Cubes and SurfaceNets meshes generated from a CT-dataset of an ankle.

4 Conclusions

Finite element modelling is a standard way to simulate soft tissue deformation. For proper modelling, triangular mesh models must satisfy requirements of accuracy, smoothness and conciseness. Several techniques proposed in the literature do not meet these requirements (e.g., Marching Cubes in combination with low pass filtering or mesh decimation).

In this paper we extended a new technique called SurfaceNets, and evaluated three variants. Optimization of a triangle mesh was performed by averaging vertices, stepping in the direction of the gradient to the isosurface, and a combined approach. The latter combination yields more accurate surface descriptions than Marching Cubes.

From visual inspection of test objects, the meshes generated by a SurfaceNet appeared to be of higher quality than those created by Marching Cubes. However, expressing this by a quality measure, no significant differences were found. Also, no significant results were found regarding the number of triangles. But the SurfaceNets meshes are more suitable for finite element modelling as they are significantly smoother.

We conclude that SurfaceNet creates a globally smooth surface description that retains fine detail.

Acknowledgements

This research is part of the MISIT (Minimally Invasive Surgery and Intervention Techniques) programme of the Delft Interfaculty Research Center on Medical Engineering (DIOC-9). The work described here was carried out at MERL – a Mitsubishi Electric Research Laboratory in Cambridge, MA (USA).

Table 1: The results using a grayscale cube and a grayscale sphere. The Marching Cubes cube has 864 vertices and 1724 triangles. The SurfaceNet cubes have 866 vertices and 1728 triangles. The decimated meshes have half and quarter the number of faces of the original mesh. The Marching Cubes sphere has 4536 vertices and 8204 triangles. The SurfaceNets have 4442 vertices and 8880 triangles. The decimated meshes have 0.5 or 0.25 the number of faces of the original mesh. The number of SurfaceNet relaxations is fixed to 5 in all cases. (SNA = averaging, SNG = gradient based, SNAG = averaging and gradient based)

	Cube			**Sphere**		
Type	H_{ave}	Quality	Smoothness	H_{ave}	Quality	Smoothness
MC	0.1344	0.66	0.0145	0.018582	0.56	0.6012
MC/2	0.1298	0.51	0.4583	0.014100	0.71	1.0974
MC/4	0.1291	0.39	1.0478	0.019490	0.65	1.0475
SNA	0.3198	0.68	0.0157	0.200700	0.67	0.0044
SNA/2	0.3194	0.55	0.0313	0.203800	0.64	0.0073
SNA/4	0.3160	0.45	0.0621	0.204700	0.65	0.0123
SNG	0.0934	0.67	0.0145	0.010998	0.64	0.0032
SNG/2	0.0932	0.53	0.2786	0.01361	0.65	0.0057
SNG/4	0.0931	0.46	0.3576	0.0215	0.73	0.0122
SNAG	0.0869	0.68	0.0147	0.010596	0.67	0.0030
SNAG/2	0.0847	0.59	0.0607	0.012787	0.68	0.0057
SNAG/4	0.0849	0.54	0.0620	0.021316	0.71	0.0122

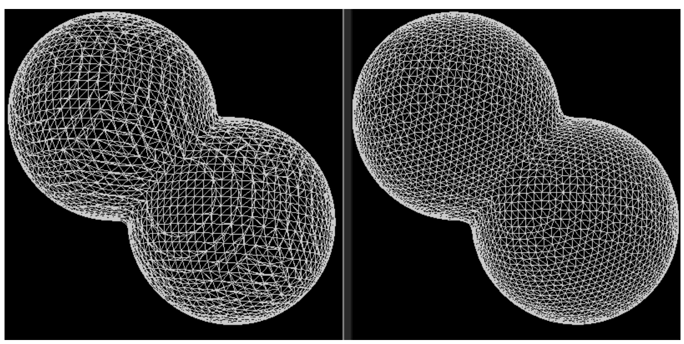


Figure 4: Two spheres partly overlapping. Meshes generated by Marching Cubes (left) and SurfaceNet (right). Both meshes have the same number of triangles.

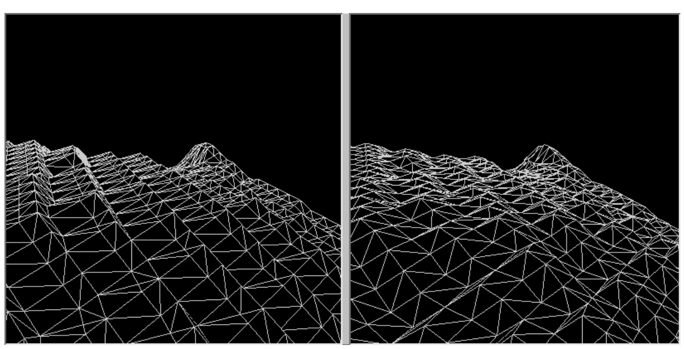


Figure 5: Generated mesh using Marching Cubes (left) and SurfaceNets (smoothing+gradient) (right). A close-up of a part of the uterus is shown.

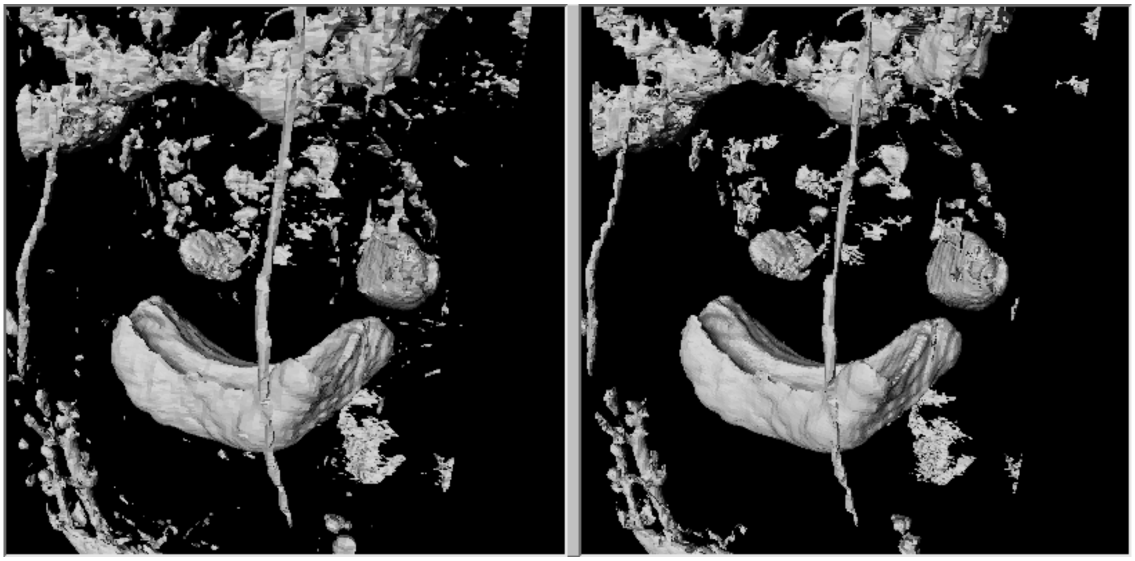


Figure 6: View of a bladder extracted using Marching Cubes (left) and using SurfaceNet (right).

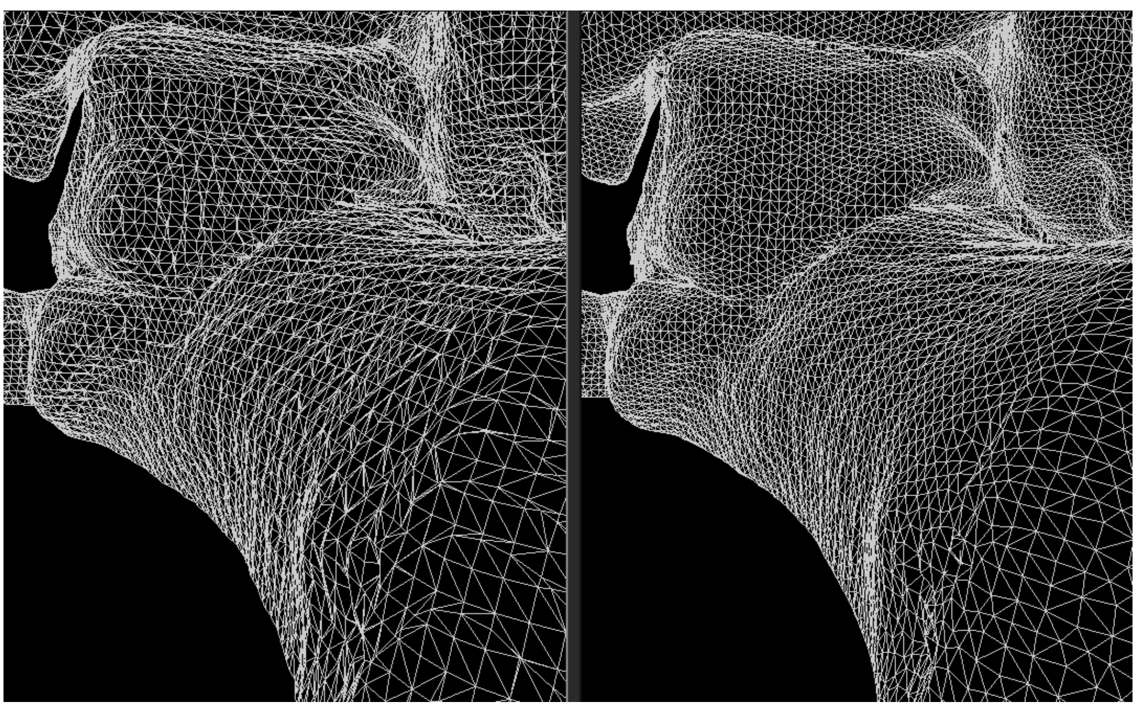


Figure 7: Mesh generated by Marching Cubes (left) and SurfaceNet (right) on a grayscale image of an ankle. The dataset is a CT-scan with dimensions $132 \times 141 \times 69$.

References

[1] Allamandri, F., Cignoni, P., Montani, C., and Scopigno, R. Adaptively adjusting marching cubes output to fit a trilinear reconstruction filter. Tech. Rep. xx, Istituto per l'Elaborazione dell'Informazione - Consiglio Nazionale delle Ricerche, Pisa, Italy, 1997. http://vcg.iei.pi.cnr.it.

[2] Bro-Nielsen, M. Finite element modelling in surgery simulation. *Proceedings of the IEEE Special Issue on Virtual & Augmented Reality in Medicine 86*, 3 (Mar. 1998), 490–503.

[3] Cotin, S., Delingette, H., and Ayache, N. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics 5*, 1 (1998), 62–73.

[4] Dubuisson, M., and Jain, A. A modified Hausdorff distance for object matching. In *Proceedings, 12th IAPR International Conference on Pattern Recognition, Conference A (Jerusalem, Israel, October 9–13, 1994)* (Oct. 1994), IEEE Computer Society Press, Los Alamitos, CA, 1994, 566-568, pp. 566–568.

[5] Field, D. Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods 4*, 6 (1988), 709–712.

[6] Garland, M. *Quadric-Based Polygonal Surface Simplification*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1999. Software and thesis available at: http://www.cs.cmu.edu/~garland/quadrics/.

[7] Gibson, S. Constrained elastic surfacenets: generating smooth surfaces from binary sampled data. In *Proceedings Medical Image Computation and Computer Assisted Interventions, MICCAI '98* (1998), pp. 888–898. http://www.merl.com/reports/TR99-24/.

[8] Gibson, S. Using distance maps for accurate surface representation in sampled volumes. In *Proceedings 1998 IEEE Symposium on Volume Visualization* (Oct. 1998), pp. 23–30. http://www.merl.com/reports/TR99-25/.

[9] Koch, R., Gross, M., Carls, F., von Büren, D., and Fankhauser, G. Simulating facial surgery using finite element models. In *Proc. ACM SIGGRAPH 96* (1996), pp. 421–428. New Orleans, USA, August 4-9 1996, Computer Graphics Proceedings.

[10] Lorensen, W., and Cline, H. Marching cubes: a high resolution 3D surface construction algorithm. In *Proc. ACM SIGGRAPH'87* (July 1987), pp. 163–169.

[11] Montani, C., Scateni, R., and Scopigno, R. Decreasing isosurface complexity via discrete fitting. Tech. Rep. xx, Istituto per l'Elaborazione dell'Informazione - Consiglio Nazionale delle Ricerche, Pisa, Italy, Dec. 1997. http://vcg.iei.pi.cnr.it.

[12] Veron, P., and Leon, J. Shape preserving polyhedral simplification with bounded error. *Computers & Graphics 22*, 5 (1998), 565–585.