INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2003; **58**:189–207 (DOI: 10.1002/nme.775)

Multiple material marching cubes algorithm

Ziji Wu[1] and John M. Sullivan, Jr[2,*,†]

[1] *Thayer School of Engineering, Dartmouth College, Hanover, NH 03755, U.S.A.*
[2] *Department of Mechanical Engineering, Worcester Polytechnic Institute, Worcester, MA 01609, U.S.A.*

SUMMARY

The accurate reconstruction of three-dimensional (3D) boundary surfaces from two-dimensional (2D) medical images is a crucial procedure in most applications of computational biomedical engineering. This paper addresses an innovative system that efficiently reconstructs accurate, multiple-material, 3D surface meshes from 2D medical images. It is based on an enhanced marching cubes algorithm, the multi-material marching cubes algorithm (M3C), which extracts boundary surfaces between different materials within one sweep of the image stack in an integrated manner. The continuity and integrity of the surfaces are ensured with this robust algorithm. Surface adjustment algorithms were also revised to adapt to the multiple-material nature of the system. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: surface reconstruction; marching cubes; multi-material; surface adjustment

INTRODUCTION

Three-dimensional (3D) surface reconstruction from two-dimensional (2D) medical images, such as stained histology, MR, or CT, is an active research field. Realistic data and object visualization requires accurate surface delineations. Surface reconstruction is also a critical component for successful solid-volume mesh creation. The surface reconstruction strategy comprises four steps: (1) image segmentation—to delineate boundary outlines on each 2D medical image; (2) surface reconstruction—to triangulate the individual boundary line segments of adjacent medical images; (3) smoothing—to mitigate the stair-stepped behaviour in the surface creation process; and (4) simplification—to reduce the surface mesh density. This paper focuses on the latter three steps. A discussion of error control is also presented. Numerous examples throughout the paper demonstrate the advancement of this work.

*Correspondence to: John M. Sullivan, Jr. Department of Mechanical Engineering, Worcester Polytechnic Institute, Worcester, MA 01609, U.S.A.
†E-mail: sullivan@wpi.edu

Contract/grant sponsor: NIH; contract/grant number: P01 CA80139
Contract/grant sponsor: National Cancer Institute
Contract/grant sponsor: NSF; contract/grant number: DBI-9604937

Received 30 January 2002
Accepted 4 November 2002

MULTI-MATERIAL MARCHING CUBES ALGORITHM (M3C)

The marching cubes algorithm dominates the medical field as an ideal surface reconstruction strategy from pixel-based images, such as stained histology, CT and MR images. The marching cubes algorithm is simple, fast, robust, and easy to implement [1].

The traditional or single-material marching cubes algorithm (SMMC) is a binary decision routine. If multiple tissues need to be reconstructed, SMMC is used iteratively. Consider the situation where a physician or researcher is interested in the blood flow into a kidney. The SMMC marches through each image layer and extracts the kidney(s) as a single material surface. The triangulated surface has nodal locations based on the pixel resolution of the image. A second application of the SMMC marches through each image layer and extracts the blood vessels as a single material surface. A significant effort is required to merge the nodes and elements on the interface between the two materials. Inevitably, there will be voids or mismatched nodes and elements. Hence, these two distinct materials are not an integral system, even though they can be displayed together. This iterative procedure can provide wonderful visualizations of surfaces, but one cannot perform virtual endoscopy (VE) simulations or use the geometries to create valid contiguous finite element meshes. If these individual geometries are smoothed and/or simplified, the continuity problem increases dramatically.

All of the existing implementations [1–5] and variations [6–11] of marching cubes algorithm are based on binary decision schemes. None of them address reconstructing integral models of multi-material domains without resorting to manual intervention. We have developed an enhanced marching cubes algorithm, the multi-material marching cubes algorithm (M3C), which alleviates this fundamental constraint on the SMMC algorithm [12]. The multi-material marching cubes system creates valid, contiguous 3D surfaces for as many materials requested within a single sweep through the images. This algorithm also overcomes the ambiguity problem in SMMC [4, 5]. The basic strategy of the system is explained initially with the 2D version, the multiple-material marching squares algorithm (M3S). Subsequently, the extension to 3D is presented.

Multi-material marching squares

The marching squares algorithm (MS) is the 2D version of the marching cubes routine. It extracts boundary contours from 2D images. A square grid is overlaid on the region of interest. Functional values are sampled at the grid vertices and assigned a binary value—a ‘1’ if the grid vertex is within the desired material, ‘0’ otherwise. First, an individual square is positioned at one corner, say, the lower left. It samples the four binary values at the vertices then indexes one unit to the right and repeats the sampling process. When the square reaches the end of the row, it indexes to the next row. The iteration terminates when it has reached the upper right corner. There are four vertices in a square, hence 2^4 or 16 possible nodal value combinations. A contour line segment is created whenever two adjacent vertices differ in binary value. Figure 1 shows the sixteen possible combinations of a square cell [2]. Every adjacent square cell must have the same contour intersection location on shared edges. Consequently, after a full sweep of the image is performed the contour segments will form contiguous contour(s) of the specified material.

The binary nature of the algorithm supports placing the contour intersection at the mid-point along an edge. Some investigators have linearly interpolated along an edge seeking a

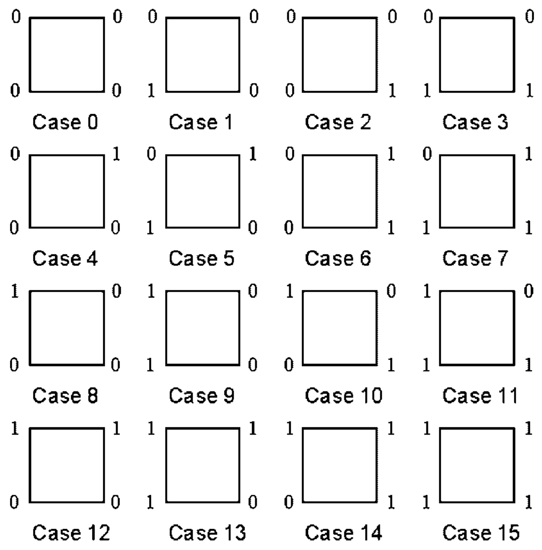

Figure 1. Vertex value combinations (16) for a single material marching square.

specific threshold value in a gray scale image. We have found potential flawed situations with that strategy in 3D. Numerical artefacts such as unrealistic surface platelets and mismatched surfaces can result. These situations do not arise with the edge mid-point strategy. Further, for segmented multi-material images threshold interpolation is inappropriate. Consequently, both our single and multiple material strategies use edge mid-points.

An ambiguous situation can exist when opposing diagonal vertices of a square have the same material ID as shown in cases 5 and 10 of Figure 1. A variety of solution techniques were discussed in Reference [2]. We introduce the concept of connectivity priority, that is, a user can define a material connectivity priority list. An underlying objective of this work is to create valid 3D finite element models suitable for numerical analyses. Frequently, the physics of the situation dictate the desired connectivity, i.e. to segregate or join material regions. Material 1 could be the fluid in a channel while material 2 being a globule of some form. Prioritizing the connectivity allows the physics (such as fluid pressure) to remain continuous. This paper assumes that continuity will be maintained when a square has the same material ID on an opposing diagonal. If both diagonals have this property the smaller the material ID, the higher the connectivity priority. This treatment eliminates the ambiguity completely in 2D and 3D situations with or without multiple materials.

2D algorithm

In a multi-material square, the four vertices can have up to four different status values or material IDs. There are $4^4 = 256$ possible combinations for contours to pass through the square. If a square has no more than two different material IDs, it is solved using a traditional MS algorithm.

Alternately, a multi-material strategy is used. Each edge of the square has 2 end vertices. If the material IDs differ between these vertices a mid-edge node is created and a contour

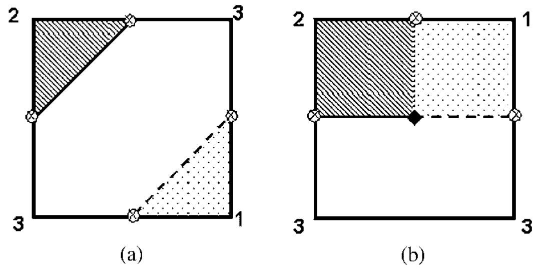

Figure 2. Possible contour segments for squares with three distinct material IDs.

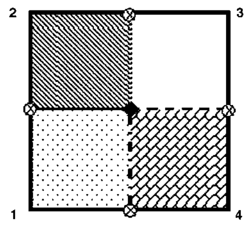

Figure 3. Contour segments for squares with four distinct material IDs.

will pass through this location. Consider the two possible situations for a square with three distinct material IDs, as shown in Figure 2. In Figure 2(a), the repeated material ID vertices are diagonal with one another. In this situation the continuity of flow is maintained for that material. The other situation has the repeated material ID vertices along an edge. In this situation a face-centred node is created and the contour segments emanate from this centre node to the mid-edge nodes. Continuity of contour segments in adjacent elements is guaranteed with this strategy. If four distinct material IDs exist in a square a face-centred node is created with contour segments emanating to the mid-edge nodes, Figure 3. This strategy is robust and provides discretization at the resolution level of the grid.

The following rules summarize all possible situations (256 cases):

Rule 1. If the square has no more than two distinct materials, the standard marching squares is used to draw the boundary line segments. When ambiguous cases occur, the material with a higher priority remains contiguous.

Rule 2. If the square has three distinct materials and the two vertices with the same material ID are on a common diagonal, they remain contiguous.

Rule 3. Otherwise, a face-centred node is introduced and contour segments emanate from it to each mid-edge node.

2D example

A 2D example was created using the multi-material marching squares algorithm, as displayed in Figure 4. The original image was cryosection slice 1972 of Visible Male dataset from Visible Human Project [13]. The dimensions were 2048 by 1216 pixels with pixel resolution of $0.33\,\text{mm} \times 0.33\,\text{mm}$. The outlines or contours generated automatically with the multi-material marching squares with grid size $0.33 \times 0.33\,\text{mm}$, is shown in Figure 4(b). The algorithm extracted boundary line segments of all 12 different materials from the segmented image with fidelity.

Multi-material marching cubes algorithm

The 3D analogy of the marching squares routine is the marching cubes algorithm. It uses a cube created from eight pixels; four each from two adjacent slices, as shown in Figure 5. The binary decision process is the same as the 2D situation. If an edge (12 for the cube)

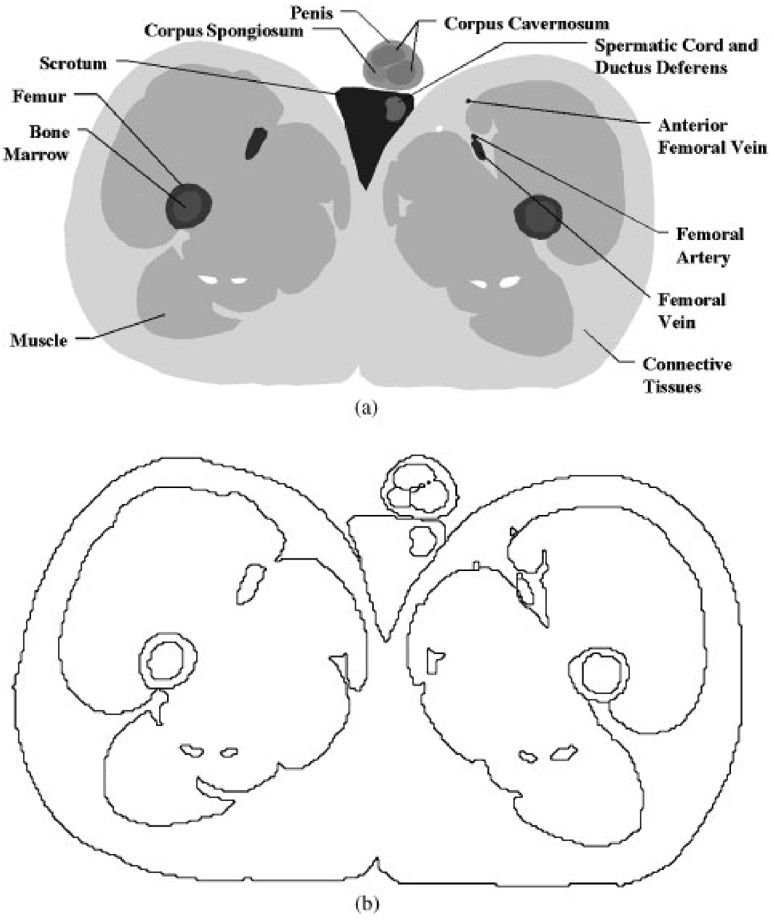

Figure 4. A multiple material marching squares (M3S) example: (a) segmented image; and (b) M3S resultant boundary segments.

has two different material vertices, a mid-edge node is created. However in 3D, the boundary separations form triangular facets within the cube. Since there are eight vertices and two states (binary) in each cube, there are 2^8 or 256 possible surface combinations. An effective lookup table created by Bourke expedited the surface reconstruction for the single-material marching cubes strategy [3]. The lookup table has 256 entries for rapid deployment. The number of unique triangular faces can be reduced to 15 by using symmetry [2]. However, the 256 lookup table strategy eliminates numerous conditional checks within the code and is preferred.

The single-material marching cubes algorithm creates individual surfaces wonderfully. However, most biological applications require multiple material interactions. These interactions cannot be simulated or modeled with the SMMC strategy. The multi-material marching cubes algorithm developed in this research effort eliminated this roadblock.

Since there are 8 cube vertices that can take up to eight different materials, there are 8^8 or 16 777 216 possible cases. The huge number of cases makes a lookup table strategy infeasible.

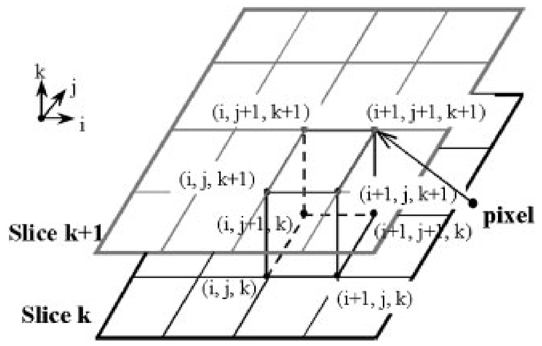

Figure 5. Marching cubes algorithm.

Our strategy is to analyse the cases and categorize them according to certain features so that robust procedural rules could be established.

3D algorithm

Two fundamental constraints of the algorithm exist. One constraint is that each face shared by adjacent cubes has the same break down pattern. The second constraint is to ensure that each unique material within the cube has separating surface(s) from all other materials.

These constraints can be addressed in a two-step process. First, conforming patterns are generated on the six cube faces. Then, partitioning of the materials within each cube is performed without introducing additional lines or nodes on the cube faces. The multi-material marching squares algorithm provides unique solutions for all possible 2D situations. This routine is used to determine the breakdown pattern on all six faces of a cube. Again, surface nodes are created only at mid-points along edges or at the centre of the cube faces. This system ensures conforming faces when viewed from either adjacent cube. Several examples of this first stage are illustrated in Figures 6 and 7. The procedure works flawlessly on each face as it does in the 2D case. Further, each face maintains total continuity along all edges.

The second stage involves triangulating the line segments created on each face in the first stage without introducing additional nodes or line segments on any cube face. We handle situations differently based on the number of face centres.

If a cube has no face centres, all surface contour segments form independent closed loops as shown in Figure 6(a) and 6(b). Therefore, a simple divide-and-conquer triangulation scheme works fine [2]. The triangles retain the same pair of material IDs as the original segmented loops, Figure 6(c) and 6(d).

If a cube has face-centred nodes, it must have at least two [12]. When face-centred nodes exist, some line segments form open-ended polylines with the open ends at face centres, Figure 7. Triangular patches need to be created from these line segments that isolate the material regions from one another without introducing new lines on the cube faces. This latter requirement guarantees the partition pattern on each face conforms to that created by the neighbouring cube sharing the face.

Consider a cube with only two face centres. All open loops become closed if the two face-centred nodes are connected. Each closed loop segment has two material IDs that are

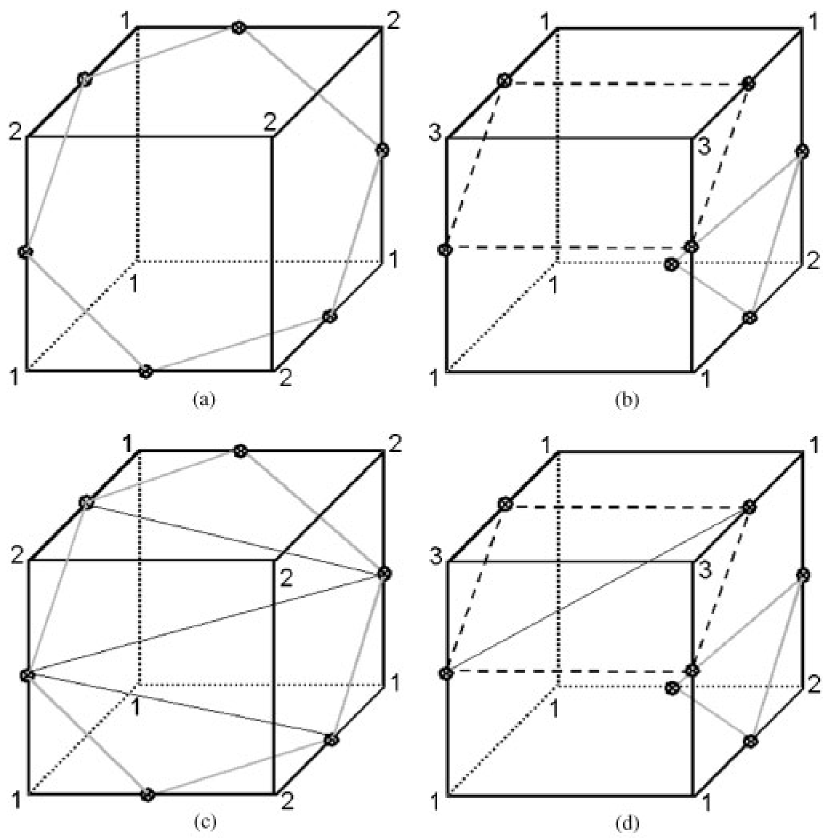

Figure 6. Surface contour segments form independent closed loops: (a) two materials, no face centre; (b) three materials, no face centre; (c) triangulated two materials, no face centre case; and (d) triangulated three materials, no face centre case.

common with all other segments in that closed loop. The triangles formed from this loop are classified with those two common material IDs. An example is provided, Figure 8, which corresponds to Figure 7(a).

When three or more face-centred nodes exist on a cube, a cube-centred node is created and assigned all the material IDs of that cube. Connecting cube-centred node to each of the face-centred nodes forms the necessary closed boundary polygons between different materials. Each of the closed loops is assigned the only two material IDs common to each segment of the loop. Triangle patches are constructed by connecting the cube-centred node to each non-adjacent node in the loop. Any unique material loops not involving a face-centred node are triangulated as discussed previously. An example is illustrated in Figure 9, which triangulated the loops displayed in Figure 7(c).

Cubes containing cube-centred nodes imply the adjacency of numerous materials. Consequently to create the material-separating surfaces from the centre of the cube or voxel is reasonable.

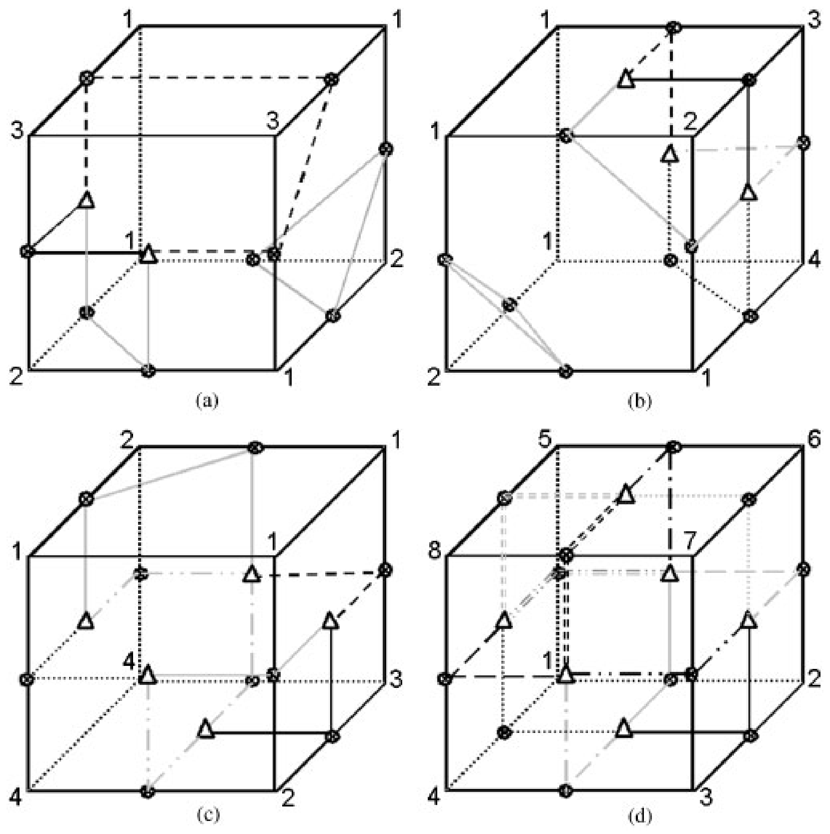

Figure 7. Examples of line segment formations on the cube faces: (a) three materials, two face centres; (b) four materials, three face centres; (c) four materials, five face centres; and (d) eight materials, six face centres.

3D preliminary example

A 3D surface example was created from segmented images of the Visible Male Dataset in VHP [13, 14]. Eight distinct materials were selected within the kidney region. The example used 131 images from slice 1560 to slice 1690 in the dataset. The pixel resolutions of the images were $0.33\,\mathrm{mm} \times 0.33\,\mathrm{mm} \times 1\,\mathrm{mm}$ in x, y, and z, respectively. The eight materials selected were left and right kidneys, duodenum, liver, ascending colon, descending colon, transverse colon, small intestine, and spleen, in descending priority of connectivity. The grid size used was $2 \times 2 \times 2\,\mathrm{mm}$. The surface mesh had 106 454 nodes and 214 103 triangular elements. Figure 10 depicts the posterior view of this kidney region.

A surface mesh of this region had been created previously using the single material marching cubes routine [14]. However, material regions were disjoint and not suitable for numerical computation as an integrated system. The multi-material marching cubes surface mesh was created with a single sweep of the 131 segmented images and maintained conforming interface elements between adjacent materials. A cut-away view into the small intestine and duodenum

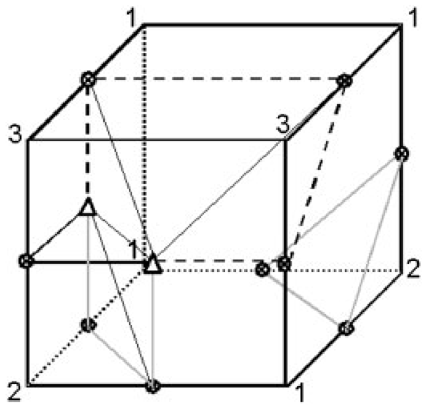

Figure 8. Triangulation of segments in a cube with two face centres.

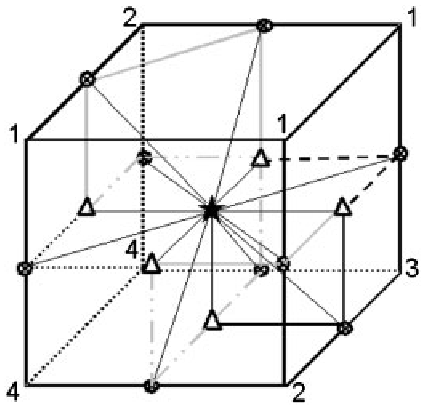

Figure 9. Triangulation of segments in a cube with more than two face centres.

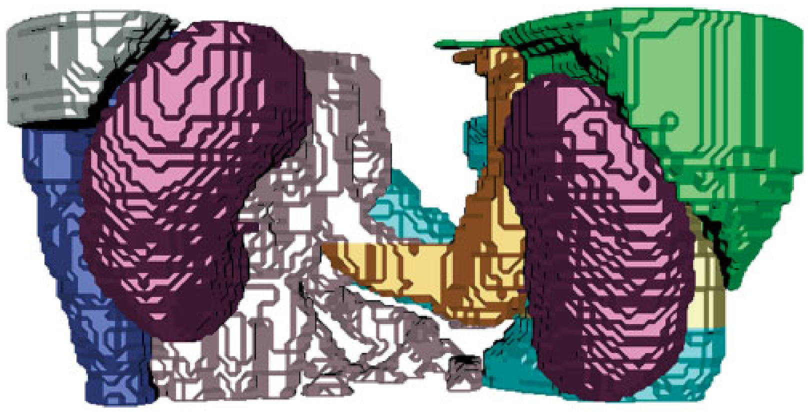
Figure 10. Surface mesh of kidney region from M3C.

region shows the separating surface between the two material classifications, Figure 11. Some of the triangles were removed for the sake of clarity. As shown in this figure, surface facets with different material IDs share the same nodes and edges at the junction.

M3C is robust, fast, and all adjacent cube faces match flawlessly. The M3C surface model exhibits the same stair-step shaped surface characteristics as those of SMMC. The model can also be massive depending on the pixel resolution of the medical images and the marching cubes grid size. Generally speaking, the resulting surfaces are too dense and severely hinder the computational efficiency for both visualization and volume mesh generation purposes, as does the SMMC algorithm. However, unique to the M3C algorithm, every faceted surface separates two materials. No overlapping physical spaces or voids exist. The surface model, albeit large, is valid numerically.

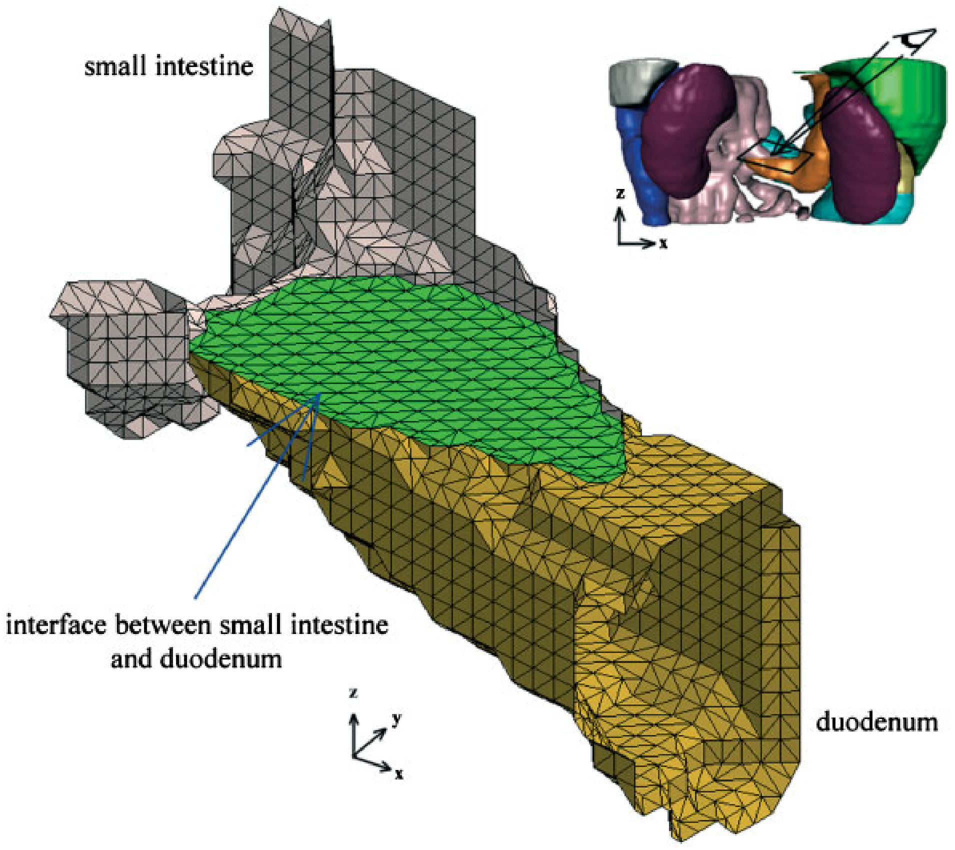

Figure 11. Interface created between two adjacent materials.

SMOOTHING

The purpose of smoothing in this work is to mitigate the stair-stepped characteristics of the marching cubes routine to obtain more realistic appearances of the surface models. During smoothing the topology of the dataset is not modified, only the geometry. A simple, yet effective technique is Laplacian smoothing [2].

The Laplacian smoothing of a node p_i at position $\mathbf{x}_i$ can be expressed as

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda \mathbf{V}_{ij} = \mathbf{x}_i + \lambda \sum (\mathbf{x}_j - \mathbf{x}_i) \quad \forall j : 0 \leqslant j < n$$

where $\mathbf{x}_{i+1}$ is the new co-ordinate position, and $\mathbf{x}_j$ are the positions of nodes p_j connected to p_i, and λ is a user specified relaxation factor. Typically, λ ranges between 0 and 1; and the process is executed repeatedly.

Existing versions of the Laplacian smoothing algorithm only focus on single material domains [2, 15–17]. More sophisticated smoothing routines exist but they too retain the single-material focus [18, 19]. We present a successful, robust classification strategy that can be applied to the various smoothing routines, which preserves the integrity of the multi-material

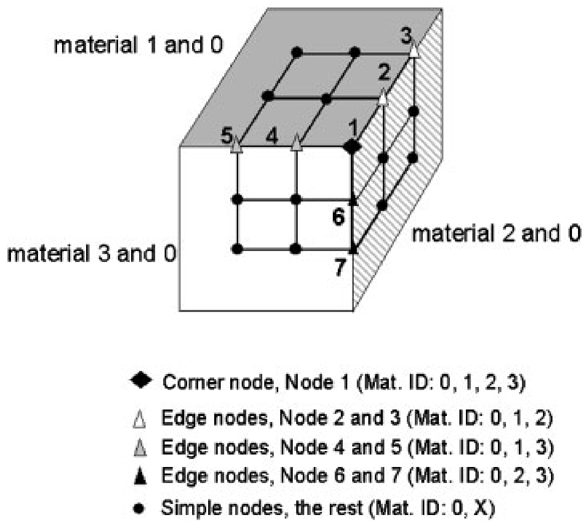
material 1 and 0
material 3 and 0
material 2 and 0
1
2
3
4
5
6
7
◆ Corner node, Node 1 (Mat. ID: 0, 1, 2, 3)
△ Edge nodes, Node 2 and 3 (Mat. ID: 0, 1, 2)
△ Edge nodes, Node 4 and 5 (Mat. ID: 0, 1, 3)
▲ Edge nodes, Node 6 and 7 (Mat. ID: 0, 2, 3)
● Simple nodes, the rest (Mat. ID: 0, X)

Figure 12. Illustration of node definitions.

domains. The viability of the strategy is demonstrated using the Laplacian smoothing algorithm. Multi-material surface models created by M3C can share up to eight different materials within a single cube. These nodes need special consideration in the adjustment processes. The nodes are classified as simple, edge, or corner-based as follows:

Simple node. A simple node can only have two material IDs, exactly the same as in the single-material marching cubes system. These two materials are the minimum common set of all adjacent node material IDs.

Edge node. An edge node has at least three material IDs associated with it. Further, there are two and only two adjacent nodes that contain, at minimum, all the same material IDs in their sets, as does the node in question. For example, in Figure 12, Node 2 has a material ID set $\{0, 1, 2\}$. It also has two adjacent neighbours with at least the same three material IDs, i.e. Node 1 has $\{0, 1, 2, 3\}$ and Node 3 has $\{0, 1, 2\}$. Hence, according to the definition, Node 2 is an edge node.

Corner node. A corner node also has at least three material IDs associated with it. However, it does not satisfy the edge-node requirements. Again, as displayed in Figure 15, Node 1 has a material ID set, $\{0, 1, 2, 3\}$. None of its neighbouring nodes has this set of material IDs, at minimal. Consequently, Node 1 is classified as a corner node.

In order to preserve the topologic and geometric features, all corner nodes are tagged as fixed. They are not moved or deleted during the surface mesh adjustments. Edge nodes are restricted to motions along the edge path; and simple nodes are adjusted using the single-material smoothing algorithms.

The surface mesh originally shown in Figure 10 was smoothed with relaxation factor λ of 0.5 and 6 iterations, Figure 13. Substantial reductions in the stair-stepped nature of the geometry are evident.

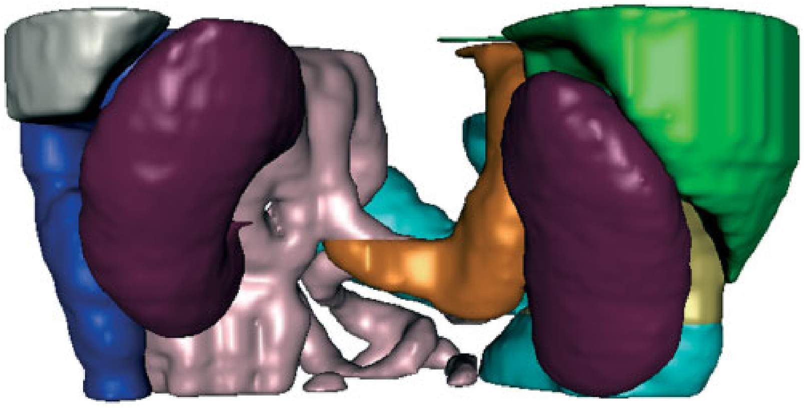

Figure 13. Smoothed eight material surface mesh within kidney region.

SIMPLIFICATION

A critical and fundamental research problem encountered in the visualization of complex models is the development of methods for storing, approximating and rendering the very large data sets that define them. Visualizing surface models created from biomedical images has the attractiveness of fine, highly resolved surface renderings. However, the number of primitives, i.e. triangles, generated by marching cubes algorithm can prohibit reasonable rendering speeds [20]. The M3C exhibits these same characteristics as does the SMMC, that is, the number of triangles created can be huge. Consequently, surface mesh simplification is often desired.

Once again, existing surface mesh simplification algorithms can only handle single material surfaces [20–23]. Erikson [22], Cignoni *et al.* [23] provided a thorough comparison of the current simplification algorithms. One of the algorithms, the decimation algorithm, proposed by Schroeder, Zarge, and Lorensen is a publicly available algorithm [2, 21]. Its goal is to reduce the total number of triangles in a surface mesh, while preserving the original topology and forming a good approximation to the original geometry. It works by iterative elimination of vertices chosen upon local geometric optimality criteria.

Each simple node is surrounded by triangles separating the same two materials. Each segment connected to a simple node shares exactly two triangles. An average plane is fit through all the nodes connected to the simple node. The plane passing through the local region is computed as an area-averaged plane using the attached triangles. The distance from the simple node to the plane is compared to the length criterion for node deletion [21].

Each edge node has two adjacent nodes N1 and N2, each having at least the same material IDs as the edge node being analysed. The distance between the node and the line segment formed by N1 and N2, as in Figure 14, is compared to the length criterion for node deletion.

Simple and edge nodes that satisfy the decimation criterion ($d < d_{\mathrm{cri}}$) are deleted individually. All triangles attached to that node are deleted as well. The hole in the mesh is patched using a local triangulation process. If the deleted node was a simple node, the hole is topologically 2D and is patched with a recursive 3D divide-and-conquer technique introduced in Reference [2]. If the deleted node was an edge node, the intersection between different

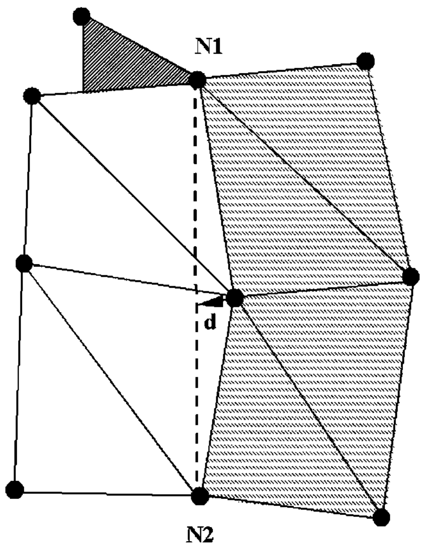

Figure 14. Error evaluation of an edge node.

materials is retained during the triangulation. Hence, the triangles connected to this node are grouped into different sub-sets each having the same material ID combination. Each sub-set is triangulated using the divide-and-conquer algorithm. This decimation process is also conducted iteratively until either the overall reduction rate reaches a user desired level or there are no nodes left that satisfy the deletion criterion.

The smoothed surface mesh in Figure 13 was decimated with a 50% reduction target. The result is displayed in Figure 15. The final surface mesh contained 47 833 nodes and 96 861 triangles. A visual inspection shows the close match to the original input. This decimated dataset contained approximately 45% of the original primitives. Hidden line views of both meshes are displayed in Figure 16(a) and 16(b), respectively. The original mesh has uniform high-density elements. The decimated mesh is non-uniform and quite sparse in planar regions. Note that, after surface adjustments, such as smoothing and decimation, the M3C surface model is still an integrated system.

EXAMPLE

As a final example demonstrating the capability of this system, a more complete kidney region was reconstructed. Though we used same set of images as in the previous examples, many more materials were reconstructed simultaneously, they are:

Digestive system: duodenum, liver, ascending colon, descending colon, transverse colon, ileum, small intestine, and spleen.

Skeletal system: T12-L3 bodies, intervertebral discs, spinous processes, and transverse processes.

Nervous system: the spinal cord.

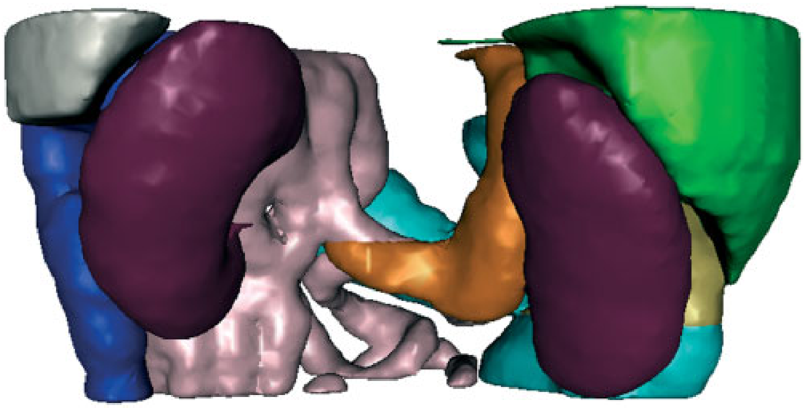

Figure 15. Decimated eight material surface mesh within kidney region.

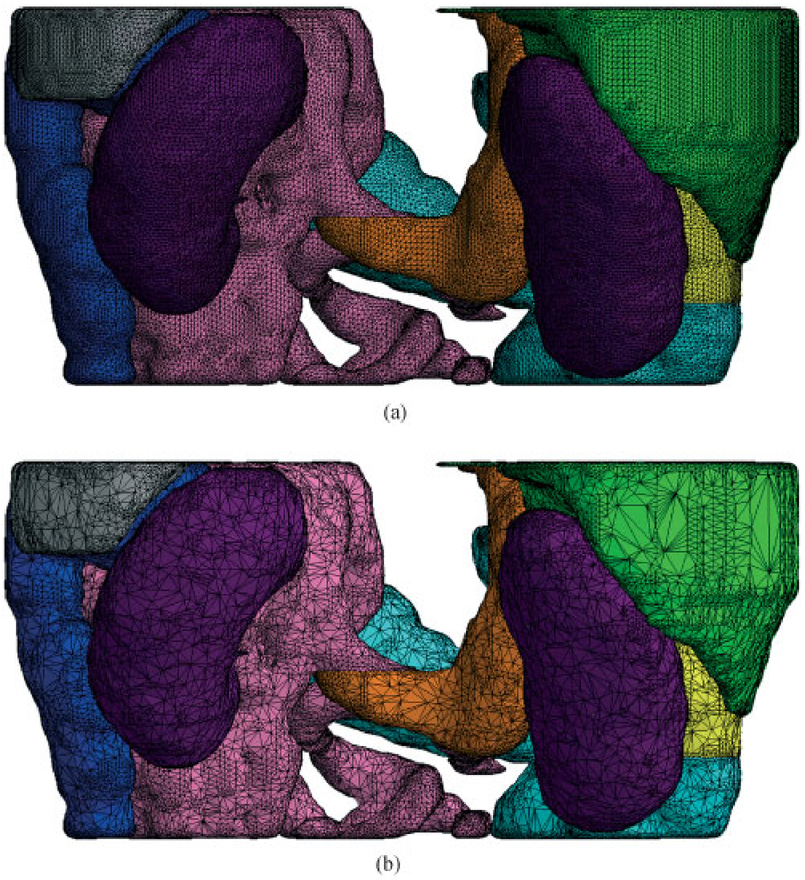

Figure 16. Hidden line views of the surface meshes before and after decimation: (a) surface mesh before decimation; and (b) surface mesh after decimation.

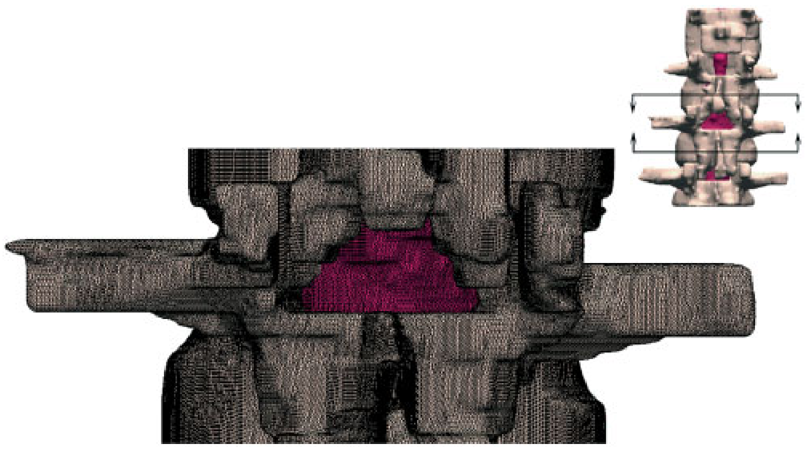

Figure 17. Posterior view of a portion of smoothed surface mesh of spinal bones and spinal cord in the kidney region at full resolution.

Circulatory system: left and right kidneys, inferior vena cava, jejunum vein, colic veins, splenic and superior mesenteric veins, superior mesenteric vein, portal vein, renal vein, renal artery, and abdominal aorta.

In order to extract small features in the region during the surface reconstruction, the size of the cube was $0.33\,\text{mm} \times 0.33\,\text{mm} \times 1\,\text{mm}$, which is the pixel resolution of the images, or, in other words, the maximum resolution for marching cubes algorithm. The surface mesh had 2 827 422 nodes and 5 665 456 triangles. The entire 5.7 million element surface mesh was smoothed as an intact, multi-material surface. Limitations of the visualization software and hardware prevented a meaningful display. However, portions of the spine and cord surfaces are displayed at full resolution after smoothing in Figure 17. Subsequently, the surface mesh was decimated with a reduction ratio of 95% specified. This simplified mesh contained 83 685 nodes and 170 035 triangles. A rendered posterior view of this region is displayed in Figure 18(a). In order to show the arbitrary geometry of the blood vessels in the region, semi-transparent posterior and transverse views of this surface model are shown in Figure 18(b) and 18(c), respectively. Figure 19 displays the sub-surface meshes of the skeletal system along with the nervous system after 95% decimation, which can be compared to the pre-decimation mesh displayed in Figure 17.

VOLUME SHRINKAGE DURING THE PROCESS

Inevitably, some volume changes occur during the reconstruction process. These changes can occur at each process step. The descending colon, reconstructed in the previous section, was extracted from the parent surface mesh of the kidney region to serve as an example.

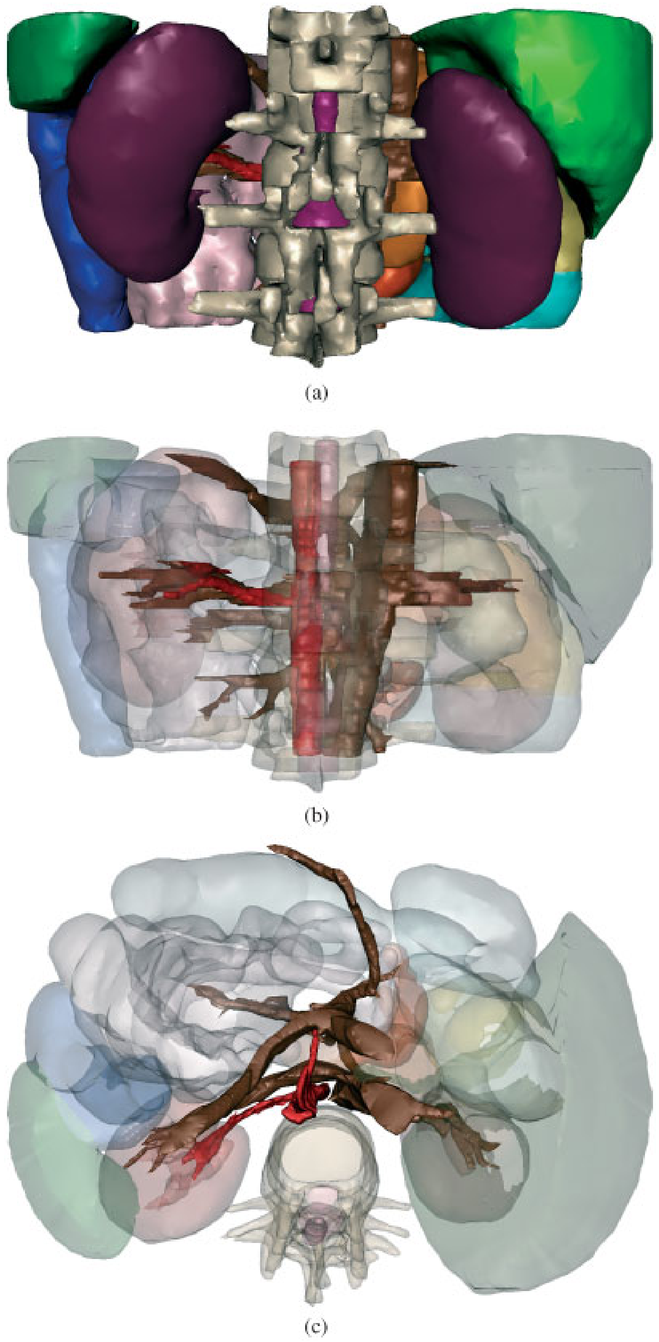

Figure 18. Views of smoothed and decimated surface model of the kidney region: (a) posterior view; (b) semi-transparent posterior view; and (c) semi-transparent transverse view.

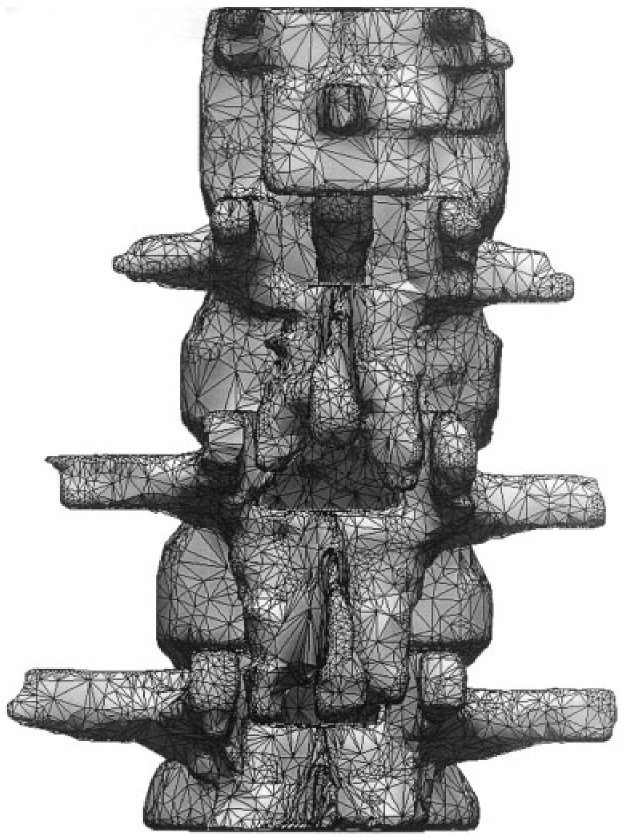

Figure 19. Posterior view of smoothed and decimated surface model of spinal bones and spinal cord in the kidney region.

The volume deviation of M3C depends on the grid size used in the process. Reconstruction nodes are placed at the centre, be it on an edge, face, or within the body of the cube. The maximum distance that a surface node can be biased from the physical boundary is a half-grid size. Note that our node placement along an edge is the same as that of the Discretized Marching Cubes algorithm, a variation of SMMC [6, 8]. This deployment strategy is more accurate than other marching cubes based algorithms, which could have a full grid size deviation. In the example presented in the previous section, this maximum deviation was 0.165 mm in x, y directions, and 0.5 mm in the z direction, respectively.

The volumes of the M3C, smoothed, and decimated surface models of the descending colon were 143.8, 140.0, and 139.1 cm^3, respectively. Smoothing resulted in 2.64% shrinkage, while decimation introduced only 0.67% shrinkage. The overall volume loss was 3.27% in this example. Given that the final smoothed and decimated surface model contained only about 3% of the nodes and elements compared to the original M3C model, we believe the representation maintains high fidelity to the original anatomy.

CONCLUSIONS

A fully automated surface mesh reconstruction system was presented. It generates multi-material surface models within one sweep of stacked, segmented images in an integrated fashion. It is robust and fast. All adjacent cube faces match; and all distinct materials are bounded by triangular patches. The interfaces between adjacent materials are unique and conform to both material boundaries. No voids or overlaps exist.

Similar to a SMMC system, surface mesh adjustment algorithms are applied to the reconstructed multi-material surface models. Both smoothing and decimation methods were revised to adapt to the multiple material nature of this integrated strategy. During the adjustments the topology of the surface models was maintained well even when it was represented with less than 5% of the original discretization.

By defining a material priority list, assigning grid size, and controlling the smoothing and decimation parameters, users can achieve a very efficient surface representation with high accuracy and complete integrity.

ACKNOWLEDGEMENTS

This work was supported, in part, by NIH grant P01 CA80139 awarded by the National Cancer Institute and NSF Grant DBI-9604937.

REFERENCES

1. Lorensen WE, Cline HE. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics* 1987; **21**:163–169.
2. Schroeder W, Martin K, Lorensen B. *The Visualization Toolkit*. Prentice-Hall: Englewood Cliffs, NJ, 1997; ISBN 0-13-954694-4.
3. Bourke P. Polygonising a scalar field. http://www.swin.edu.au/astronomy/pbouke/modelling/polygonise/, valid January 2002.
4. Nielson GM, Hamann B. The asymptotic decide: resolving the ambiguity in marching cubes. *IEEE Proceedings of the Visualization Congress* 1991; San Diego, CA, 83–91.
5. Matveyev SV. Approximation of isosurface in the marching cube: ambiguity problem. *IEEE Proceedings of the Visualization Congress* 1994; 288–292.
6. Montani C, Scateni R, Scopigno R. Discritized marching cubes. *IEEE Proceedings of the Visualization Congress* 1994; Washington, DC, 281–287.
7. Gueziec A, Hummel R. Exploiting triangulated surface extraction using tetrahedral decomposition. *IEEE Transactions on Visualization and Computer Graphics* 1995; **1**(4):328–342.
8. Li J, Agathoklis P. An efficiency enhanced isosurface generation algorithm for volume visualization. *The Visual Computer* 1997; **13**:391–400.
9. Lucas L, Gillard D, Remion Y. A new unsupervised cube-based algorithm for iso-surface generation. *Computer Networks and ISDN Systems* 1997; **29**:1737–1744.
10. Chan SL, Purisima EO. A new tetrahedral tessellation scheme for isosurface generation. *Computers and Graphics* 1998; **22**(1):83–90.
11. Treece GM, Prager RW, Gee AH. Regularised marching tetrahedral: improved iso-surface extraction. *Computers and Graphics* 1999; **23**:583–598.
12. Wu Z. Accurate and efficient three-dimensional mesh generation for biomedical engineering applications. *Ph.D. dissertation*, Worcester Polytechnic Institute, 2001.
13. The Visible Human Project. http://www.nlm.nih.gov/research/visible/visible_human.html, valid January 2002.
14. Kulkarni AA. Surface model creation for biomedical applications using a marching cubes technique. *M.S. Thesis*, Worcester Polytechnic Institute, 2000.
15. Vollmer J, Mencl R, Müller H. Improved Laplacian smoothing of noisy surface meshes. *Computer Graphics Forum* 1999; **18**:131–138.
16. Amenta N, Bern M, Eppstein D. Optimal point placement for mesh smoothing. *8th ACM-SIAM Symposium on Discrete Algorithms*, 1997.

17. Canann S, Stephenson M, Blacker T. Optismoothing: an optimization-driven approach to mesh smoothing. *Finite Elements in Analysis and Design* 1993; **13**:185–190.
18. Taubin G. A signal processing approach to fair surface design. *SIGGRAPH* 1995; Los Angeles, CA, 351–358.
19. Schneider R, Kobbelt L. Geometric fairing of irregular meshes for free-form surface design. *Computer Aided Geometric Design* 2001; **18**:359–379.
20. Shekhar R, Fayyad E *et al*. Octree-based decimation of marching cubes surfaces. *IEEE Visualization Conference* 1996; San Francisco, CA, 335–342.
21. Schroeder W, Zarge J, Lorensen W. Decimation of triangle meshes. *Computer Graphics* 1992; **26**(2):65–70.
22. Erikson C. Polygonal simplification: an overview. *Technical Report TR96-016*, Department of Computer Science, University of North Carolina, Chapel Hill, 16 February 1996.
23. Cignoni P, Montani C, Scopigno R. A comparison of mesh simplification algorithms. *Computers and Graphics* 1998; **22**(1):37–54.