Two algorithms for the three-dimensional reconstruction of tomograms

H. E. Cline, W. E. Lorensen, and S. Ludke
General Electric Company, Corporate Research and Development, Schenectady, New York 12301
C. R. Crawford and B. C. Teeter
General Electric Medical Systems, Milwaukee, Wisconsin 53201

(Received 3 October 1986; accepted for publication 23 February 1988)

Three-dimensional (3-D) surface reconstructions provide a method to view complex anatomy contained in a set of computed tomography (CT), magnetic resonance imaging (MRI), or single photon emission computed tomography tomograms. Existing methods of 3-D display generate images based on the distance from an imaginary observation point to a patch on the surface and on the surface normal of the patch. We believe that the normalized gradient of the original values in the CT or MRI tomograms provides a better estimate for the surface normal and hence results in higher quality 3-D images. Then two algorithms that generate 3-D surface models are presented. The new methods use polygon and point primitives to interface with computer-aided design equipment. Finally, several 3-D images of both bony and soft tissue show the skull, spine, internal air cavities of the head and abdomen, and the abdominal aorta in detail.

Key words: three-dimensional imaging, medical diagnostic images, computer graphics, algorithms

I. INTRODUCTION

A three-dimensional (3-D) surface reconstruction of a contiguous series of two-dimensional computed tomography (CT) or magnetic resonance (MR) slices resembles a photograph of the anatomy. Three dimensions have been applied in craniofacial surgery,[1,2] complex fractures,[3] heart volume measurements,[4] and in understanding the geometry in cases that are difficult to interpret from the 2-D slices.

An image of the anatomy consists of the visible surfaces of tissues computed by scanning the data and projecting surface patches onto the view plane. In previous 3-D algorithms,[5-11] voxel size limits the resolution of 3-D reconstructions, resulting in 3-D images that appear blocklike or stepped compared to the smooth surfaces of real tissues. Attempts to produce smoother surfaces,[8] by averaging over the neighboring voxels, reduce the detail of the image. Existing methods for 3-D display generate images based on the distance[9,10] from an imaginary observation point to a patch on the surface and on the estimated surface normal of the patch.[8]

We have discovered two algorithms that create 3-D surface representations and use gradient shading to produce images of both bony and soft tissues. The first method creates a polygonal surface format that interfaces with computer-aided design (CAD) graphics accelerators. The second algorithm generates point surface coordinates and normal vectors that are projected onto a raster display. Examples of images generated from CT studies of the head and abdomen are presented to show the utility of the algorithms for the visualization of the anatomy of both the bony and soft tissue. We discuss the implementation of the algorithms.

II. ALGORITHMS

Production of 3-D medical images involves two steps; surface reconstruction and surface display. Both algorithms have a common shading method and differ only in the type of surface element produced.

To shade the surface of the 3-D image projected onto the view plane, an intensity is calculated from the component of the unit normal vector parallel to the view direction. The gradient vector of the 3-D density function estimates the surface normal direction since the gradient is perpendicular to surfaces of constant density. Gradient shading contributes to the image quality by giving contrast that depends on the surface orientation. The idea of using the gradient vector to shade a surface has been used before[12,13]; however, we calculate the normalized gradient vector at the surface point which is new. If the 3-D data form a rectangular lattice in space with a unit cell of dimensions a, b, and c, the gradient vector $g = (g_x, g_y, g_z)$ is estimated from the density function by taking central differences between the densities, $f(x_0, y_0, z_0)$, evaluated at the lattice points (x_0, y_0, z_0)

$$g_x = \frac{f(x_0 + a, y_0, z_0) - f(x_0 - a, y_0, z_0)}{2a}, \tag{1}$$

$$g_y = \frac{f(x_0, y_0 + b, z_0) - f(x_0, y_0 - b, z_0)}{2b}, \tag{2}$$

$$g_z = \frac{f(x_0, y_0, z_0 + c) - f(x_0, y_0, z_0 - c)}{2c}. \tag{3}$$

To locate the surface, the density function is linearly interpolated in the space (x, y, z) between the lattice points (x_0, y_0, z_0) and compared with the value C of the desired surface.

0094-2405/88/030320-08$01.20

Thus the surface of interest is

$$f(x, y, z) = C. \tag{4}$$

The gradient vector defined at each lattice point is trilinearly interpolated over the voxel to give a local value of the gradient vector, **g**, at the desired surface. Smoother surfaces are obtained with the interpolated gradient vector than by using the gradient at each data point. The unit surface normal **n** is the gradient vector divided by its magnitude, $|\mathbf{g}|$. The surface appears smooth because the interpolated gradient vector is continuously varying with distance the across voxel boundaries.

A. Polygonal 3-D surface reconstruction

The polygonal algorithm, called *marching cubes*, converts an array of density data into a polygonal format by sequential tessellation of a logical cube (voxel) constructed from eight pixels; four each from two adjacent slices. There are many ways for a surface to intersect a voxel and they can be classified by noting which of the eight voxel vertices are inside the tissue of interest. Each voxel vertex has two possible states: inside or outside the surface. Each possible configuration has an index, calculated by labeling each vertex with one of two colors depending on whether the vertex is inside or outside the surface. Although there are 256 ways to color eight vertices with two colors, there are only 15 topologically distinct patterns. Actually, according to Polya's theorem,[14] there are 23 distinct ways of coloring the eight vertices of a cube with two colors; however, taking into account complementary patterns, where the colors are reversed, only 15 distinct patterns occur, Fig. 1. The patterns 11 and 14 are mirror images that are distinct with respect to the rotation operation because one is left handed and the other right handed.

The number of vertices, or weight, inside a surface classifies each voxel. There is but one pattern with weight 0, one with weight 1, three with weight 2, three with weight 3, and seven with weight 4, resulting in a total of 15 patterns (Fig. 1). The trivial case 0 occurs if each vertex density is less than the surface constant; the surface does not intersect the voxel and no triangles result. If only one vertex density is less than the surface value, a single triangle, cutting the three adjacent edges, results, case 1. In case 2, the surface cuts across two vertices at the ends of a common edge. Two triangles describe the quadrilateral surface that is, in general, nonplanar. Although there are two possible ways to represent this quadrilateral, our experience indicates that either representation is valid. It is important to choose patterns that match at common faces to prevent holes in the surface. Case 14 is a mirror image of case 11. They are treated as separate cases because one is left handed and the other right handed and they cannot be rotated to coincide with each other.

Once the 15 cases are tessellated, the triangle vertices are permuted according to the 24 elements of the cube rotation group, and complemented to yield the tessellations of the entire set of 256 configurations. The algorithm stores each tesselated configuration in a table that contains the edges for each triangle. The locations of the triangle vertices are calculated by linearly interpolating the voxel edge to find the intersection point. The CT or MR instrument generates a 3-D array of integers stored in a series of slice files. The files are scanned two at a time by marching the voxel between two successive slices, hence the name marching cubes. Comparing each voxel vertex with the surface density constant c, the algorithm forms an 8-bit case index. For each index, a series of triangles are generated from a lookup table. Using the surface value constant, linear interpolation between the voxel vertices produces the triangle vertex value. This interpolation is repeated for a set of three points defining a triangle. To produce a shaded image, the normal vectors for each polygon are calculated from the gradient vector at each voxel vertex by interpolating along the appropriate voxel edges. A set of three vertices defining a triangle and the corresponding three unit normal vectors are sent to a graphics display processor. Three normal vectors at the triangle vertices are required to generate smooth shading, rather than using the normal to the triangle that would be used for flat shading. The triangles are projected onto the viewing plane and an intensity assigned to each vertex. The display processor performs scan conversion of each triangle, and grades the intensity linearly across each surface facet, yielding a smooth shaded image.[15] A depth buffer was used to remove the hidden surfaces.

To summarize, the marching cubes algorithm includes the following steps:

(1) Input the 3-D data, the surface constant, view orientation, scale factors, cut planes, and nature of the cut surfaces.

(2) Scan two consecutive slices and calculate an index formed by comparing the eight density values with the surface constant.

(3) Tessellate each voxel configuration by retrieving edges from a precalculated table.

(4) Interpolate the densities at the triangle vertices with respect to the desired surface density value along the voxel edges to locate the three triangle vertices.

(5) Interpolate the voxel vertex gradient vectors at the triangle vertices and normalize to find the vertex normal vectors.

(6) Transform the vertices and the normal vectors to find the intensity at the triangle vertices.

(7) Triangles projected on the view plane are shaded by smoothly grading the intensity values and hidden surfaces are removed via a depth buffer.

B. Point 3-D surface reconstruction

The algorithm of *dividing cubes* was developed to eliminate the scan conversion step of the polygonal display method. In medical images there are many surface elements and the triangles created by the polygonal algorithm may occupy only a small area on the raster display. As the number of facets increases, the size of each triangle decreases and approaches the pixel size. For complex medical images, it is more efficient in memory and time to display point primitives on the raster display directly.

The dividing cubes algorithm subdivides the voxels into small cubes that lie on the surface of the object and projects the intensity calculated for each cube onto the viewing plane, forming a gradient shaded representation of the three-dimensional object (Fig. 2). The voxel scale is chosen to make the small cubes equal to the pixel size on the raster display.

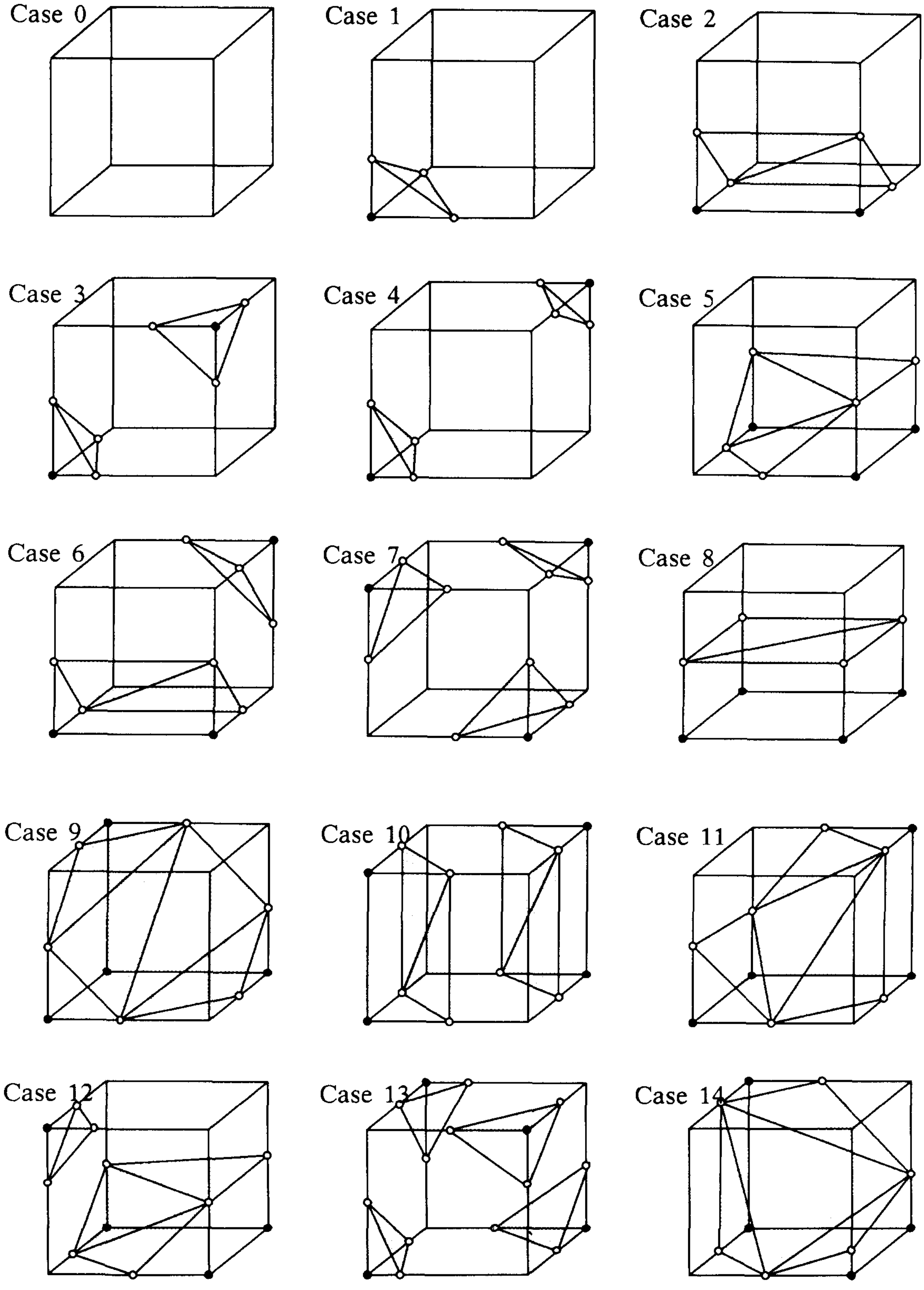

FIG. 1. There are 15 topologically distinct patterns required to tessellate to a surface intersecting a cubic or rectangular voxel. The black filled vertices are those that lie inside the surface. Each pattern separates the vertices inside the solid from those exterior to the solid by a polygonal surface intersecting the edges at points shown as open circles. Taking into account the symmetry of a cube reduces the number of configurations from 256 to the 15 cases shown above.

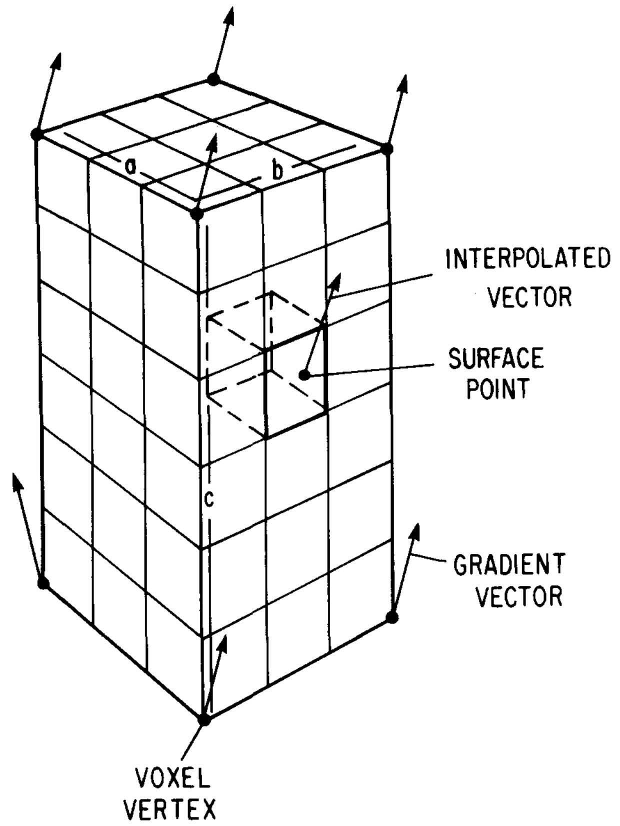

FIG. 2. Subdivision of a voxel into small cubes increases the resolution using the dividing cubes algorithm. The gradient vector is calculated at the voxel vertices and linearly interpolated to estimate the surface normal at the center of each cube intersecting the surface. In the case illustrated, the voxel was divided into 3×3×6 cubes.

The display resolution, rather than the voxel size, limits the image resolution. The idea of choosing a surface element equal to the pixel size is the basis of "ray-tracing" algorithms used for 3-D display of medical objects.[15] In ray tracing the surface is calculated in the image space rather than the object space. We use an object space algorithm to form a display list allowing rapid manipulation of the surface without the need to recalculate a surface for each view orientation. In a previous object space method,[5] a solid model was constructed from voxels. Here, instead of building a solid model, we construct the surface of constant density since it takes much fewer points to describe the geometry of a surface than that of a solid. A surface of constant density may be mathematically described by setting the density function equal to a constant, Eq. (4). By subdividing the voxel, we interpolate the data in three dimensions to approximate the continuous density function with a fine cubic lattice of integers. This subdivision increases the accuracy of the interpolation. However, the finer the subdivision, the more surface points need be processed. The densities at the corners of the small cubic element are calculated from the eight voxel vertices by linear interpolation. The algorithm calculates an index for each voxel by comparing the cube vertices with the valuè of the desired surface as in the polygonal algorithm. Each cube is classified as being inside the surface, outside the surface or intersecting the surface. A cube inside a surface has eight cubically connected densities greater than the constant density of the surface, a cube outside the surface has eight densities less than the surface, otherwise the cube intersects the surface. For those cubes that lie on the surface, we interpolate the gradient vector at each voxel vertex at the center of each surface cube to provide an estimate the surface normal vector. Both the surface points and corresponding normal vectors are transformed and projected onto the viewing plane. A depth buffer removes the hidden surfaces.

The algorithm of dividing cubes includes the following steps:

(1) Input the 3-D data, the surface constant, view orientation, scale factors, cut planes, and nature of the cut surfaces.

(2) Read consecutive slices into memory and advance these slices as data are being processed.

(3) At each of the eight voxel vertices, calculate gradient vector components by taking differences between forward and backward neighbors along each axis.

(4) Divide each voxel into $a \times b \times c$ cubes that are the size of the display pixel and are associated with eight density values obtained from interpolation of the voxel.

(5) Scan the voxels and test for surface intersections where some but not all the eight density values exceed the surface constant.

(6) Interpolate the gradient vector at each cube intersecting the surface.

(7) Calculate the light intensity at each surface point by the scalar product by the projection of the normal vector along the view direction.

(8) Rotate and project each cube onto the view plane and paint the pixel with the calculated intensity if it has not been previously painted with a closer cube.

III. CT DATA

A male patient, age 12, with an orbital encephalocele was examined with a GE CT 8800 scanner (GE Medical Systems, Milwaukee, WI) using a 1.5-mm collimator with 0.8-mm pixels. The study consists of 93 slices, 1.5 mm apart, resulting in a total dosage of 8–10 rads. The 2-D slices start from a position above the orbits and cover most of the head and neck.

A 30 axial slice CT study of the abdomen was made at the University of California Radiology Department at San Francisco. A GE CT 8800 scanner produced 5-mm slices with 1.3-mm pixels. The voxel size was 1.3×1.3×5 mm and the resolution of each slice was 256×256.

IV. RESULTS

Computed tomography 3-D surfaces of the head and body, processed with the above algorithms, show the quality of the resulting 3-D images and illustrate techniques developed to reveal the anatomy in 3-D. A head study of a 12 yr old male with an encephalocele, a birth defect in the nasal suture of the skull where the brain protrudes, represents a comprehensive 3-D CT exam. The 3-D reconstruction shows a hole in the skull at the nasal suture near the left eye socket.

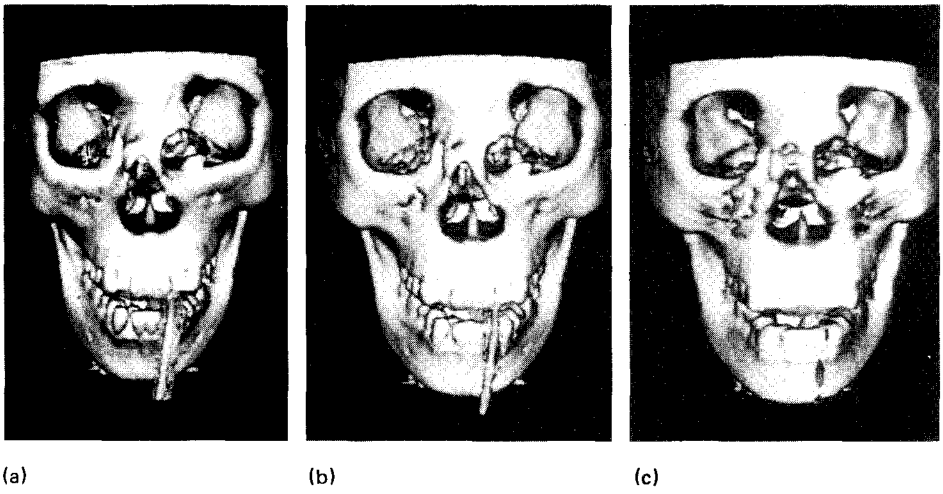

Fig. 3. A 3-D reconstruction of the surface of the skull from CT data of a 12 yr old male with a birth defect in a suture of the skull near the left eye socket. The polygonal surface is shown at different resolutions (a) 256×256×93, (b) 128×128 ×93, and (c) 64×64×93.

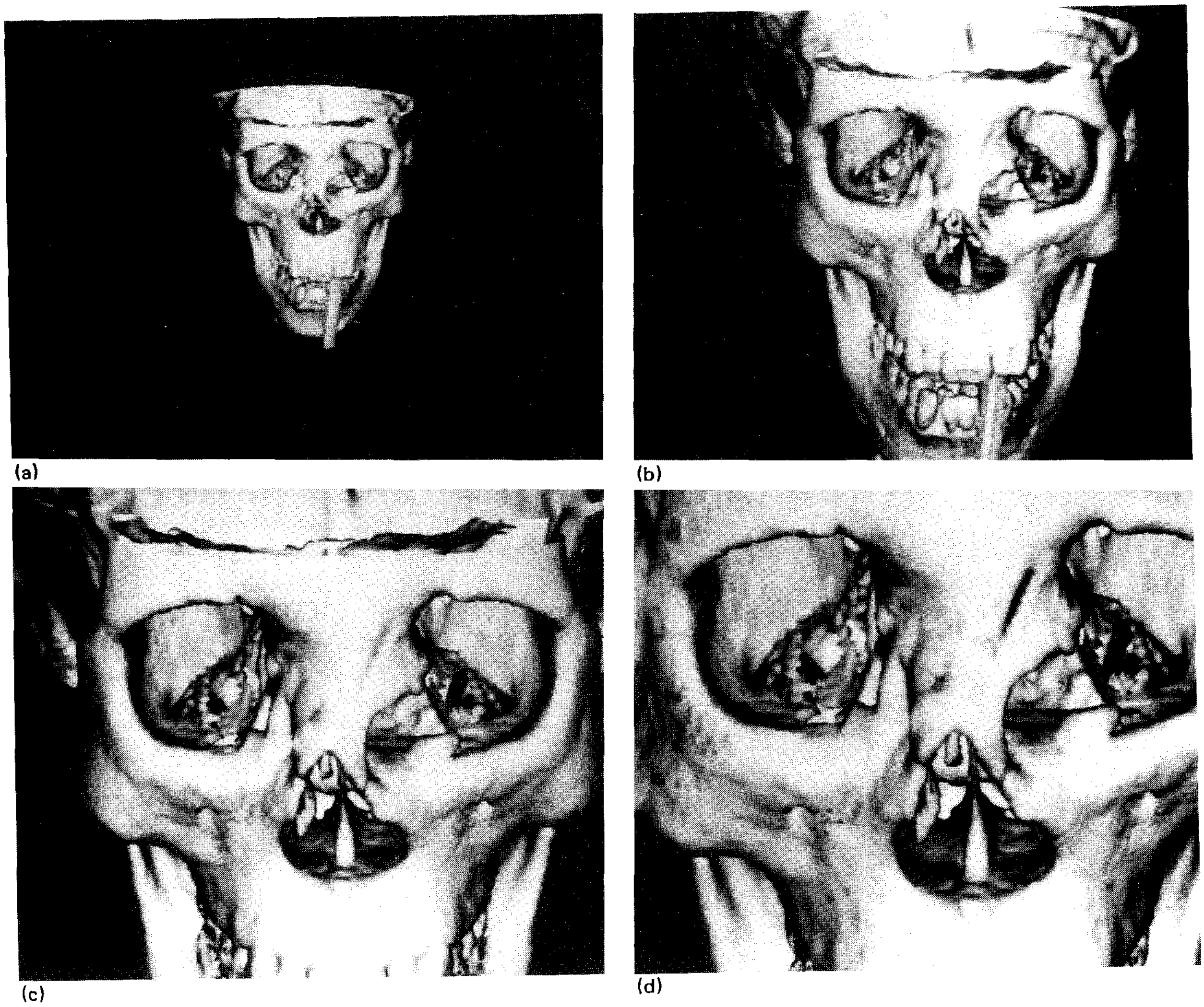

Fig. 4. The results of the point algorithm show 3-D reconstruction at different voxel subdivisions and magnifications (a) 1×1×2, (b) 2×2×4, (c) 3×3×6, and (d) 4×4×8.

First, consider images of the skull produced in the polygonal format at different resolutions, Fig. 3. The resolution of the data was reduced by averaging down the slices, reducing the number of triangles in the wire frame surface, (a) $256\times256\times93$ resolution, 550 000 triangles (b) $128\times128\times93$ resolution, 200 000 triangles, and (c) $64\times64\times93$ resolution, 76 000 triangles. Reducing the number of triangles reduces the time required to manipulate the 3-D image. Interactive image cuts, rotation, and scaling help locate the region of interest. However, as the triangle size increases, surfaces with fewer polygons lack the fine detail visible in the full resolution models.

Second, consider the point algorithm used to display 3-D images directly on a raster display device. The size and quality of the 3-D image resulting from the point algorithm depends on the number of cubes used to subdivide the voxel. Each cube matches the pixel size of the raster display. For the skull, the resolution of the 3-D image and the size of the image on the raster display was varied by changing the number of cubes along each voxel dimension. Figure 4 shows the results of voxel subdivision into (a) $1\times1\times2$ cubes, 350 000 points, (b) $2\times2\times4$ cubes 1 100 000 points, and (c) $3\times3\times6$ cubes 2 500 000 points. The perceived resolution increases as the number of points in the surface increases; however, the resolution of detail is limited by the information content of the data and little is gained by magnification to (d) $4\times4\times8$ cubes, 4 400 000 points.

A variety of surface cuts uncover more information about the internal anatomy in Fig. 5. Texture obtained from the x-ray density may be mapped onto the cut plane, Fig. 5(a). The CT data reformatted on the cut plane are used to obtain the density of the cut tissue. By rotating the image, a cut may be viewed from a different perspective, Fig. 5(b). Displaying skin and bone together shows the relationships of the two surfaces, Fig. 5(c). By closing the cut of the bone, it appears solid, while the skin surface remains open to view the skull. Finally, an open cut of the skull exposes internal surfaces, Fig. 5(d).

Simultaneous display of the cut surfaces of soft tissue and bone helps to determine the geometry of different tissues in one view. To generate multiple surfaces, we specify two surface constants and change the voxel classification step to include both constants. This multiple surface technique was used in the body study, where the abdominal wall was cut away to expose the abdominal aorta while simultaneously displaying the surfaces of skin, muscle, and bone, Fig. 6. By

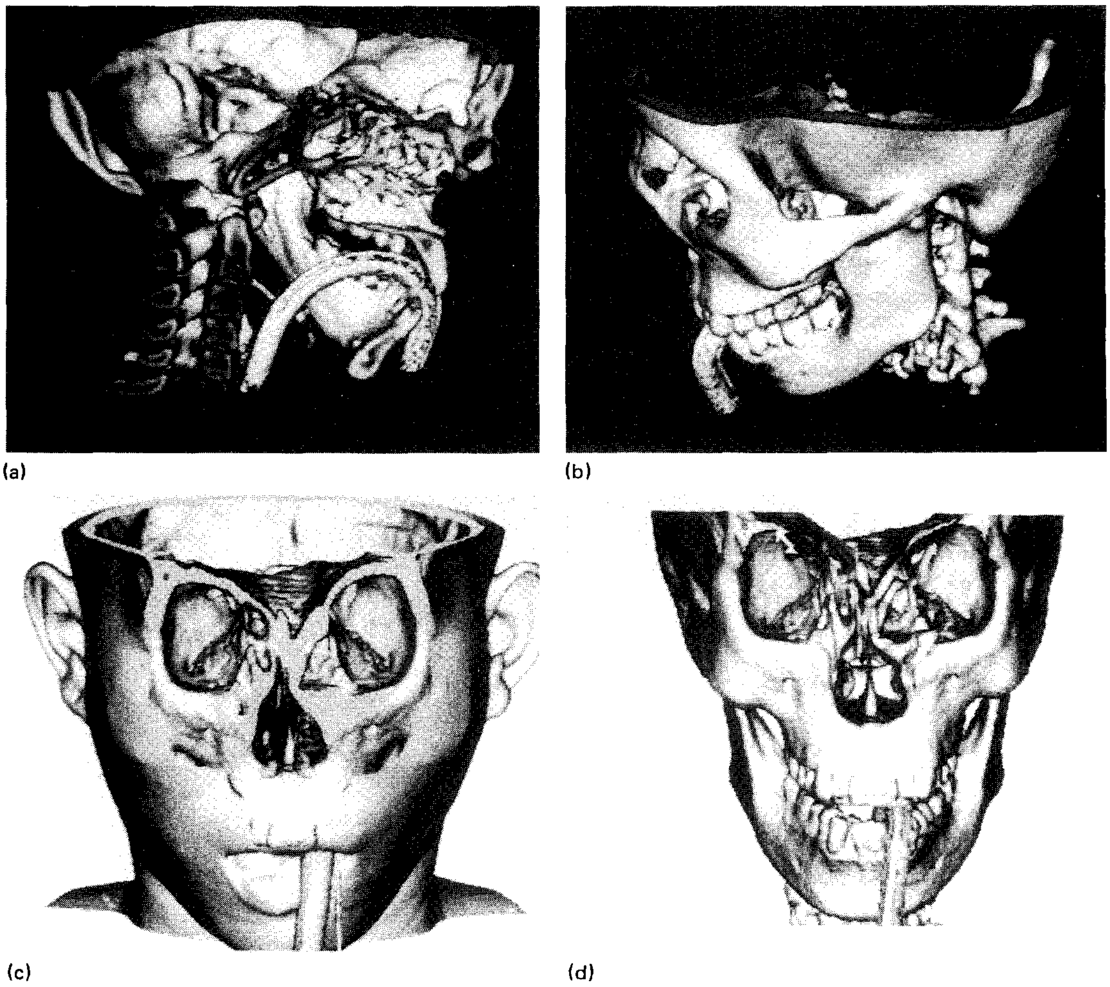

FIG. 5. (a) A sagittal cut surface exposes the internal surfaces of the skull. Mapping the original 2-D data onto the cut surface shows the bone tissue density. (b) The skull is rotated to view the left side. (c) Cutting both the surfaces of skin and bone exposes the defect in the skull, the cut surface is solid for bone and open for soft tissue. (d) The skin surface is removed and the bone surface cut is open.

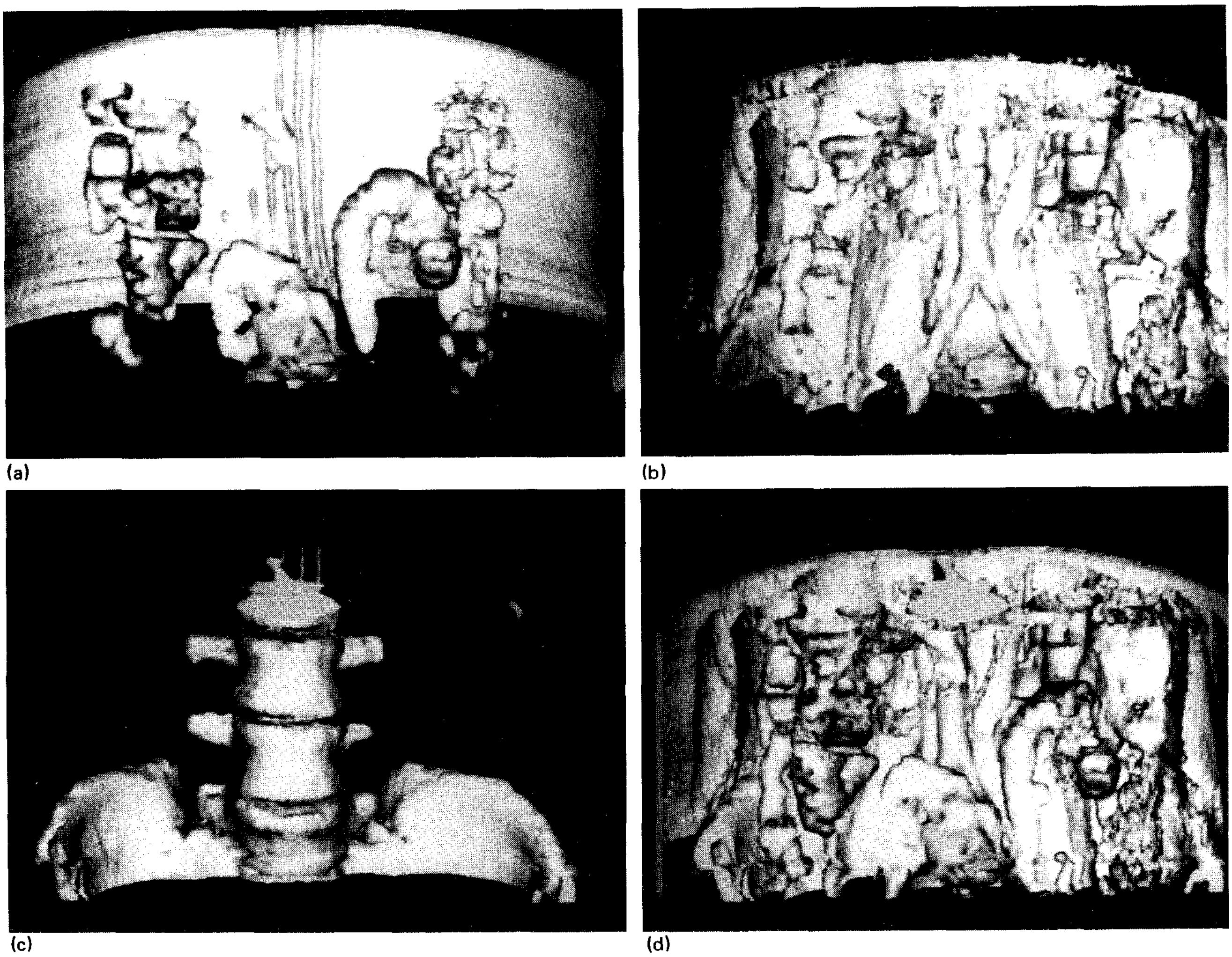

FIG. 6. An anterior view of the abdomen cut posterior to the abdominal wall to expose internal tissue (a) the surface of skin and the internal air filled cavities of the intestines, (b) the surface of muscle revealing the abdominal aorta, (c) the surface of bone exposing the spine and pelvis, and (d) a composite view of skin, muscle, and bone exposing the internal tissue of the abdominal cavity.

viewing the surfaces of skin, muscle, and bone separately we may differentiate the surfaces displayed in the composite view.

V. DISCUSSION

Two algorithms were presented using gradient shading to improve the image quality and linear interpolation to increase the apparent image resolution. In the first algorithm, a polygonal format produces a surface representation that can interface with computer-aided design (CAD) graphics accelerators, allowing interactive manipulation of images. The second algorithm uses a point format to more efficiently match the pixels on a raster display. Both algorithms produce high quality images using gradient shading and interpolation of the voxel. Recently, ray-tracing methods[13,16,17] have also yielded comparable high quality images. The algorithms presented here are object space methods in contrast to the image space ray-tracing technique. An advantage of the object space method is that the surface of interest is stored in a display list for subsequent transformations while an image space method is reapplied for each new view. The methods may be thought of as a single gradient shading algorithm differing in only the format of the surface elements. An important advantage of the polygonal surface representation is its use in computer graphics and computer-aided design; consequently, hardware designed for other applications may be used to display 3-D polygonal medical images. The point method gives the same high quality and efficiently matches a raster display, eliminating the need to scan convert polygons. Without special purpose hardware, the point format is preferred. Furthermore, the point method gains economy of storage and speed by using integers rather than the floating point numbers required by the polygonal method. For example, the point algorithm implemented on a GE Quick 9800 scanner rotates images in 2–5 s by using the array processor that is included in the system.

The gradient method presented here provides a more accurate surface normal; consequently, the images appear smooth. The improved resolution of the new algorithms results from interpolation of the voxel and the gradient vector.

One could envision a 3-D display work station for examining the anatomy and planning clinical procedures with the referring physician. Useful features showing the anatomy of the head and body from CT data are as follows:

(1) view 2-D CT slices while varying the window and level interactively;

(2) select the surfaces of interest and generate a 3-D image;

(3) rotate, translate, and scale 3-D images to locate the region of interest;

(4) cut images—open surface to view internal tissue, solid surface to model the tissue, and textured surface to show x-ray density;

(5) display two or more surfaces;

(6) measure distances, areas, volumes, orientations;

(7) rapid playback of preprocessed 3-D images or 2-D CT slices.

Using software that implements the high-resolution methods, we have illustrated the above features in a CT study of the head and body. To rapidly manipulate the images, the software first created a display list of polygons on a host computer and then displayed with a graphics accelerator. In the future, the rendering time of complex 3-D images will decrease and the anatomy will be manipulated in real time.

These new algorithms use gradient shading to produce improved image quality in CT studies of both the head and body. Soft-tissue and bone surfaces, shown simultaneously, display the anatomy with unprecedented detail. Comprehensive 3-D studies of over 250 clinical cases have been carried out to determine the medical significance of examining the anatomy with the 3-D methods described.

ACKNOWLEDGMENTS

We thank Dr. J. F. Schenck and W. Leue of our laboratory for stimulating discussions and providing the CT data of the body. Dr. D. C. Hemmy, of the Medical College of Wisconsin provided the data generated by Dr. D. J. David, of Children's Hospital.

[1] D. C. Hemmy, D. J. David, and G. T. Herman, Neurosurgery **13**, 534 (1983).

[2] V. M. Vannier, J. L. Marsh, and J. O. Warren, Radiology **150**, 179 (1984).

[3] D. L. Burk, D. C. Mears, D. H. Kennedy, L. A. Cooperstein, and D. L. Herbert, Radiology **155**, 183 (1985).

[4] E. A. Hoffman and E. L. Ritman, Radiology **155**, 739 (1985).

[5] G. T. Herman, R. A. Reynolds, and J. K. Udupta, Proc. Soc. Photo-Opt. Instrum. Eng. **367**, 3 (1982).

[6] E. Artzy, G. Frieder, and G. T. Herman, Comput. Graphics Image Proc. **15**, 1 (1981).

[7] G. T. Herman and H. K. Liu, Comput. Graphics Image Proc. **9**, 1 (1979).

[8] L. Axel, G. T. Herman, J. K. Udupa, P. A. Bottomley, and W. A. Edelstein, J. Comput. Assist. Tomogr. **7**, 172 (1984).

[9] E. H. Farrell, IBM J. Research and Development **27**, 420 (1983).

[10] M. W. Vannier, J. L. Marsh, and J. O. Warren, Comput. Graphics **17**, 263 (1983).

[11] D. J. Meagher, Proc. Comput. Graphics 343, **2** (1984).

[12] D. Gordon and R. A. Reynolds, Comput. Graphics Image Proc. **29**, 361 (1985).

[13] K. H. Hohne and R. Bernstein, IEEE Trans. Med. Imaging **5**, 54 (1986).

[14] F. Harary, *Graph Theory* (Addison–Wesley, Reading, MA, 1969), p. 180.

[15] H. Gouraud, IEEE Trans. Comput. C-20, 623 (1971).

[16] H. K. Tuy and L. Tuy, IEEE Comput. Graphics Appl. **4**, 29 (1984).

[17] D. A. Talton, S. M. Goldwasser, R. A. Reynolds, and E. S. Walsh, *Proceedings of the National Computer Graphics Association*, Philadelphia, PA, March 1987.