DISPLAY OF 3D MEDICAL IMAGES

Yiu-Wing Tam
Advanced Technologies Department
Alberta Research Council, Calgary
Alberta, Canada. T2E 7H7

Wayne A. Davis
Department of Computing Science
University of Alberta, Edmonton
Alberta, Canada. T6G 2H1

ABSTRACT

Three-dimensional display of medical data usually involves three steps: (1) image segmentation, (2) merging of 2D data into a 3D representation, and (3) display of 3D object. This paper gives a survey of various object representation and display techniques. A simple and efficient method to produce high quality shaded images is proposed. The method is termed the *normal-averaging method* and is a subset of the gradient shading method. The method uses both the distance of the object from the light source and the object surface orientation as the basis for shading. The theory and its implementation are discussed and images produced by several display technqiues are presented and compared.

KEYWORDS: 3D, Three-Dimensional, Reconstruction, Display, Medical Image, Voxel, Image-Space, Voxel-Based, Normal-Averaging.

1. INTRODUCTION

The availability of medical imaging modalities such as computed tomography scanners, nuclear medicine systems, ultra-sound scanners, digital subtraction angiography systems and the magnetic resonance imaging systems, enable clinicians to view the internal structure of human organs without performing surgery. Such systems produce three-dimensional (3D) information in the form of a series of parallel two-dimensional (2D) images, each of which displays a cross-section of the internal structure. Inspecting the 2D images one after the other does not give clinicians full understanding of the shape of the objects they wish to view. With the aid of computer imaging devices, 3D images of objects can be produced. The display of 3D data usually involves the following three steps:
(1) image segmentation,
(2) merging of 2D data into a 3D representation,
(3) display of 3D object .

Image segmentation can be regarded as a preprocessing step being applied to the data. It involves grouping parts of a generalized image into units that are homogeneous with respect to one or more characteristics. Very often the object of interest is buried inside objects of non-interest. For instance, a physician interested in bone study would like the soft tissue data removed from the CT (Computed Tomography) images. One aspect of image segmentation frequently used is the detection of boundaries; *edge* detection in 2D images and *surface* detection in 3D images. Since a lot of work has been done in the area of image segmentation in the past decade, edge or surface detection will not be discussed in this paper, interested readers are urged to read the paper by Fu and Mui [4] for an overview.

Once the images are segmented, the information can be used to produce a data structure (according to some specified data model) so that the 3D objects can be represented. Having the 3D representation of the object defined, one can then visualize the object by projecting the 3D objects onto a 2D screen. Among these projections are wire-framing, particle-framing and shading.

This paper is divided into two main sections. Section 2 gives a survey of 3D representation techniques as well as a comparison of several 3D display techniques based on image quality. Section 3 presents two gradient shading methods; one recently developed by the authors and one proposed by Gordon and Reynolds [7]. A new shading technique (based on the framework proposed by Gordon and Reynolds) is also introduced. This new technique produces very good images.

2. 3D OBJECT REPRESENTATION TECHNIQUES

In order that detected objects can be visualized, methods have to be developed to represent the extracted surface (or, in other words, to reconstruct the

This research was supported in part by the National Sciences and Engineering Research Council of Canada under grant A7634.

3D surface), such that they can be displayed on a graphics srceen. A number of surface representation techniques have been proposed. The following discusses several 3D surface representation methods.

2.1 The Cuberille Method

A method proposed by Herman and Liu [9] represents 3D surfaces by means of a *cuberille*. A cuberille is a partitioning of space into equal cubes by three orthogonal sets of equally spaced parallel planes. The volume elements (*voxels*) become components of a cuberille ***S*** and all surface elements are faces of ***S***. In this method, each face α is associated with two numbers:

(1) b(α) the brightness of the displayed surface α, which is calculated by a shading rule.

(2) d(α), the distance of the center of α from the observer.

The shading rule is a variant of Warnock's [8] rule in which the shading value depends on the cosine of the angle the face makes with an assumed direction of light and on the distance of the point from an assumed light source.

A number of display techniques, based on the cuberille representation, have been proposed by Chen and Herman et al. recently [1], such techniques will be discussed later.

2.2 The Triangular Tile Techniques

Another approach to represent 3D surfaces is by means of polygonal patches. A variety of polygons can represent surfaces of the reconstructed structure; the most popular and simplest of these is the triangle. A number of approaches have been proposed [2, 5, 11].

2.3 Other Techniques

A different approach, adopted by Wu, Abel and Greenberg [15], gives rise to a technique which reconstructs surfaces with a spline approximation. In this method, B-splines represent the sectional contours and cardinal splines interpolate between the contour planes.

Yet another approach that should be mentioned is *Octree-Encoding* [6, 12]. It is a geometric modeling technique in which 3D objects can be represented to any specified resolution (up to the resolution of the original image) in a hierarchical 8-ary tree structure or *octree*.

The simplest form, perhaps, to represent a 3D object is by means of the original object space -- a volume. Each element of the volume is a *voxel*. The object is then depicted by all *non-transparent* voxels (i.e. voxels with non-zero intensity values). This representation is called the *voxel-based method*. The following describes some 3D display techniques that can be applied to the above mentioned representation methods.

2.4 TECHNIQUES OF DISPLAYING 3D OBJECTS ON 2D PLANES

Displaying 3D objects on a 2D plane requires depth-cues to create the illusion of the third dimension. Wire-framed objects usually use hidden surface removal or motion parallax to create a feeling of 3D. Other methods use shading to simulate both the object surface characteristics and the position and orientation of the surfaces with respect to the light sources. In general terms, the shading value or the intensity of a point on a surface depends on three factors: (1) the amount of ambient light, (2) the amount of diffuse reflection and (3) the amount of specular reflection. Thus the equation for the shading intensity of a point on the surface can be written as:

$$I = I_a + I_d + I_s$$

where

I_a = the amount of ambient light,

I_d = the amount of diffuse reflection, and

I_s = the amount of specular reflection.

In medical applications, the term I_s is not often used due mainly to the fact that it is computationally expensive and that I_d gives sufficiently good quality images. Thus in a medical application

$$I = I_a + I_d .$$

The amount of diffuse reflection depends on the cosine of the angle between the incident ray and the surface normal and the distance from the point light source to that particular point on the surface. Thus

$$I_d = f(\cos \theta, d)$$

where

θ = angle between the incident light ray and the surface normal,

d = distance from the light source to the surface.

In theory, illumination is inversely proportional to the distance from a point light source to the surface. In practice, the factor $1/d^2$ does not produce natural looking images, so $1/d$ is usually used. Chen and Herman et al. [1] suggest a general form which can be used to describe many shading methods (some of which will be discussed below). The formula is as follows:

$$S(P) = \left[\frac{M - L}{2R} \; (R - d)\, N(\theta) + L \right]_{L}^{M} \qquad \textbf{(2.1)}$$

and

$$\left[V \right]_{L}^{M} = \begin{cases} L, \text{ if } V < L, \\ V, \text{ if } L \le V \le M, \\ M, \text{ if } M < V. \end{cases} \quad (2.2)$$

In this formula, P is a point on the surface of an object and $S(P)$ is the shading value of point P. M is the maximum intensity that can be prouced by the display hardware and L is the amount of ambient light. d is the distance of P from the source of light and θ is the angle between the direction of light and the estimated surface normal at P. R is the radius of the smallest sphere that encloses the object. $N(\theta)$ is a function of θ which simulates the properties of a diffuse reflecting surface.

2.4.1 Distance Shading

The simplest shading method, perhaps, is distance shading. This form of shading does not require the complex process of estimating the object normals of surfaces therefore it is very computationally efficient. The formula for distance only shading can be obtained by setting $N(\theta)$ in Equation **2.1** to 1. In the voxel-based environment, d is the distance from the centre of the voxel to the light source. Distance shading produces smooth looking images due to the fact that the distance d for neighboring faces does not differ too much. In a practical sense, however, this is a handicap, because minute textural information will be lost. Another difficulty is that distance shading does not take into account surface orientation; faces on either side of an edge have similar distance from the light source. This is an important drawback because real-edge information cannot be detected.

2.4.2 Constant Shading

A simple shading method frequently used during the emerging stage of computer graphics is constant shading. This form of shading can be applied to objects with surfaces represented by a polygonal mesh. The shading value on each polygon is contant and is estimated in terms of the surface normal of the polygon and the direction of the light source. If Equation **2.1** is used, $N(\theta)$ will be in the form

$$N(\theta) = \cos\theta$$

where θ is the angle between the surface normal of the polygon and the direction of the light source. Chen and Herman [1] suggest a more general form

$$N(\theta) = [\cos(\frac{\theta}{n})]^{p}$$

and recommend, after extensive experiments, that p=0.6 and n=2. A major problem with constant shading is the strong appearance of artificial edges at certain rotation angles. When compared with distance shading, however, this method is still preferable since more details of the objects can be visualized.

2.4.3 Phong Shading

In the 1970's two smooth shading methods revolutionized the world of computer graphics: intensity interpolation, or Gouraud shading and normal vector interpolation, or Phong shading. Phong shading is computationally more expensive than Gouraud shading, but is visually preferable since it is less susceptible to the Mach-band effect. Phong shading is regarded as the *de facto* standard for image quality. Since the principles of Gouraud and Phong shadings are well known, they will not be discussed here.

2.4.4 Image-based contextual shading

In an attempt to overcome the problems caused by artificial edges in the cuberille environment [9], Herman and Udupa [10] proposed the image-based contextual shading method. This method tries to eliminiate artificial edges caused by constant shading. The idea is to assign a shading value to points on a face that depends not only on the orientation of the face itself, but also on the orientation of adjacent faces (the four faces which share an edge with the original face). Denote the angle between the normal to a face f and the direction of light as θ_0 and let θ_1, θ_2, θ_3 and θ_4 be similarly defined angles for the four adjacent faces. Then the shading value of all points on the face f is determined by Equation **2.1** evaluated at the centre point with

$$N(\theta) = \sum_{i=\theta}^{4} W_i \left[\cos(\frac{\theta_i}{n}) \right]^{p}, \qquad \sum_{i=\theta}^{4} W_i = 1.$$

Number of visible critical faces in the neighborhood	0	1	2	3	4
Critical	1/16	1/8	1/4	1/2	1
Noncritical	1	15/16	7/8	3/4	1/2

Table 2.1 The weight W_0 used in the image-based contexual shading method.

The weights W_i depend on whether the face f is *critical* or *non-critical*, and on the number of visible adjacent faces which are critical. A critical face is a face whose normal is less than 45° with the direction of light, and the rest are called non-critical faces. Herman and Udupa proposed a unique scheme of assigning weights. Briefly, if f is critical, then a zero weight is assigned to adjacent critical faces; if f is non-critical then zero weight is given to adjacent non-critical faces. Based on a predefined set of values for W_0, see Table 2.1, the values of W_i, for $1 \le i \le 4$ are calculated by the formula $(1\text{-}W_0) / n$ for $n > 0$ (where n is the number of visible critical faces). If $n = 0$, W_i is assigned a value of zero.

The image-based shading method has been used mainly in the cuberille environment. In such an implementation, a one byte neighbor-code is associated with each face of the cuberille (there are only 81 possible arrangements of the faces). Since there are only 81 possible values of $N(\theta)$, these values can be stored in a look-up table. Thus contextual shading is no more expensive than constant shading, and yet produces better images. However, because of the arrangement of critical and non-critical faces, the surface intensity over the whole image varies unnaturally with the direction of light.

2.4.5 Normal-based contextual shading

In order to solve the rotational invariant problem of the image-based shading method, Chen and Herman [1] propose a similar technique called the normal-based contextual method. The implementation of this method uses the same adjacent faces arrangement and the same neighbor-code data structure. However, instead of using weighting factors to calculate $N(\theta)$, object normals are calculated based on the orientation of a face f as well as the orientation of the adjacent faces. The object normal, ω, of a centre point P of a face f is defined as the cross product of vectors μ (in the u-w plane) and ν (in the v-w plane), see Figs 2.2a and 2.2b. The vectors μ and ν can each have (independently) one of five possible directions, depending on the adjacent faces in the positive and negative u-directions (for μ) and v-directions (for ν). The exact method by which μ is determined is illustrated in Fig. 2.3 for each of the nine arrangements of adjacent faces in the positive and negative u-directions. A similar method can be used to obtain ν. With the above algorithmic changes, the normal-based method solved the rotational invariant problem of the image-based shading method; that is, the texture pattern does not change appreciably as the object is rotated. This method of shading does produces good images, and the quality is found to be comparable with the Phong shading method, but with a considerable speed up in computation.

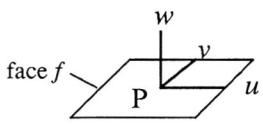

Fig. 2.2a Local coordinate system.

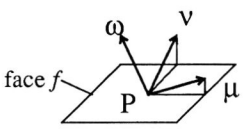

Fig. 2.2b Object normal, ω, affected by adjacent faces.

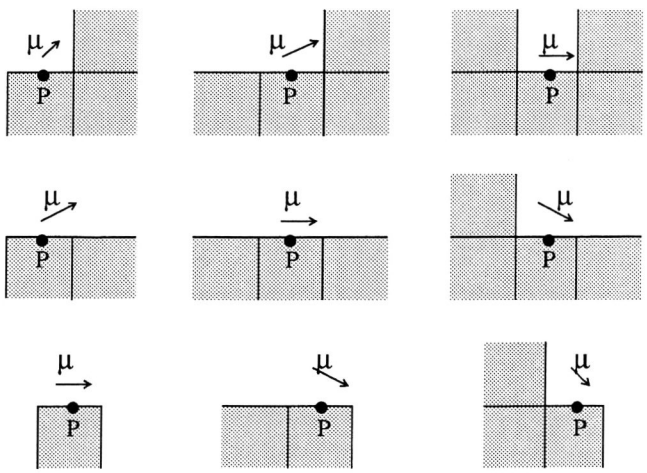

Fig. 2.3 Determination of vector μ. There are 5 possible directions of μ (making angles $\pm 45°$, $\pm 26.6°$, and $0°$ with the positive u-axis (Fig 2.2a)). These angles are determined by the orientation of the adjacent faces.

2.4.6 Gradient Shading

With the exception of distance shading, the aforementioned shading methods may be described as *object-space shading* methods due to the fact that extra data (face-orientation, vertex normals, or a neighbor-code) need to be stored with the object representation. Object-space methods have the disadvantage that the normals must be recalculated if the object is modified. For systems allowing rapid interaction with the object (e.g. in surgical planning), this problem is of great importance. To avoid this problem two methods have been proposed: (1) the voxel-based method [3] recently developed and (2) the image-space shading method proposed by Gordon and Reynolds [7]. Both methods can be termed as a *gradient shading* method since both methods make use of the gradient vectors of the object surfaces to find the shading values. These methods will be discussed in detail in the next chapter and a new display technique is proposed that is based on the image-based gradient shading method. This technique produces images of higher quality than that produced by Gordon and Reynolds.

3. NEW TECHNIQUES IN GRADIENT SHADING

In this section two gradient shading methods are discussed: one recently developed [3] and the one proposed by Gordon and Reynolds [7]. A new technqiue is also proposed that produces image of better quality than the two methods just mentioned. Before going into details, the foundations on which these methods are based will be laid out.

Unlike the object-space shading methods which require extra data to be stored with the object representation, the gradient shading method uses no more data than a complete pre-image obtained by coordinate transformation, hidden-surface removal and scan-conversion. One suitable pre-image is the *Z-buffer* used for hidden-surface removal. The Z-buffer contains, for each pixel, the distance from the light source to the closest point on the object which projects onto that pixel. The Z-buffer, or in other words, the *distance matrix*, then contains a complete representation of the visible surface of the object of the form $z = z\,(x,y)$, see Fig. 3.1.

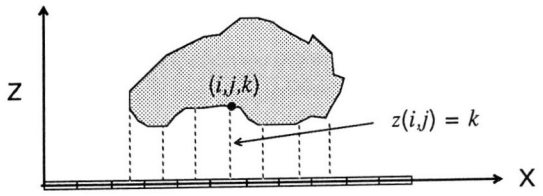

Fig. 3.1 Distances of a cross-section of an object from a light source with parallel rays. (The light source here is assumed to be located on the XY-plane).

The object normal **N** is then computed for every point in terms of the gradient vector ∇z at that point. From the normal **N**, the light direction **L**, the distance, and the radius of the minimal spanning sphere, the shading intensity of a point on the object can be easily obtained using Equation **2.2**. Since the distance information is the only data needed, any other technique besides the Z-buffer algorithm, can be used to obtain a distance matrix.

Both the voxel-based method proposed in [3] and the image-space gradient method proposed by Gordon and Reynolds [7] require the distance matrix as the pre-image. Given an M x M distance matrix, each entry of the matrix can be represented by

$$z_{i,j} = z\ (x = i, y = j), \qquad (1 \le i \le M, 1 \le j \le M).$$

From the distance matrix the normal at any point of the object surface $z = z\ (x,y)$ can be obtained from the gradient vector

$$\nabla z = \mathbf{N} = \left(\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}, -1 \right).$$

The two methods mentioned differ in the way $\partial z/\partial x$ and $\partial z/\partial y$ are obtained and the way the shading intensities are estimated.

3.1 The Voxel-Based Shading Method

The voxel-based method is designed for efficient display of 3D objects without sacrificing image quality. Like other gradient shading methods, it makes use of the distance matrix to estimate surface normals. To obtain the shading value using Equation **2.2** as described in chapter two, the voxel-based method assumes $N(\theta) = \cos\theta = \mathbf{N} \bullet \mathbf{L}$. The efficiency comes from the fact that the $N(\theta)$s are pre-generated, and that during calculation of the intensity values, the $N(\theta)$ can be obtained by a simple table look-up. The following describes how the look-up table is derived. The object normal of a point on the surface of the object can be represented by

$$\mathbf{N} = \left(\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}, -1 \right) \approx \left(\frac{\Delta z_x}{\Delta x}, \frac{\Delta z_y}{\Delta y}, -1 \right).$$

To get Δz_x (the difference in depth in the x-direction), the following schema is proposed, for an M x M distance matrix:

backward difference

$$\Delta z_{xb} = 2\ (z_{i,j} - z_{i-1,j}), \qquad (2 \le i \le M, 1 \le j \le M),$$

forward difference

$$\Delta z_{xf} = 2\ (z_{i+1,j} - z_{i,j}), \qquad (1 \le i \le M - 1, 1 \le j \le M),$$

central difference

$$\Delta z_{xc} = z_{i+1,j} - z_{i-1,j}, \qquad (2 \le i \le M - 1, 1 \le j \le M),$$

In this case $\Delta x = \Delta y = 2$. And without loss of generality, assume **L** = (0, 0, -1) which is in the direction of sight, one can deduce that

$$N(\theta) = \frac{\mathbf{N}}{|\mathbf{N}|} \bullet \frac{\mathbf{L}}{|\mathbf{L}|}$$

$$= \frac{\left(\frac{\Delta z_x}{\Delta x}, \frac{\Delta z_y}{\Delta y}, -1 \right) \bullet (0, 0, -1)}{\left| \left(\frac{\Delta z_x}{\Delta x}, \frac{\Delta z_y}{\Delta y}, -1 \right) \right| \bullet \left| (0, 0, -1) \right|}$$

$$= \left[\left(\frac{\Delta z_x}{\Delta x} \right)^2 + \left(\frac{\Delta z_y}{\Delta y} \right)^2 + 1 \right]^{-1/2}$$

Since $\Delta x = \Delta y = 2$, therefore

$$N(\theta) = \frac{2}{\left[\Delta z_x{}^2 + \Delta z_y{}^2 + 4 \right]^{1/2}}$$

$$= f(\Delta z_x, \Delta z_y).$$

Since the value of N(θ) depends only on two parameters which are finite and within a small bound, a look-up table can be defined (for values of N(θ)) with rows indexed by Δz_x and columns indexed by Δz_y as shown in Fig. 3.2.

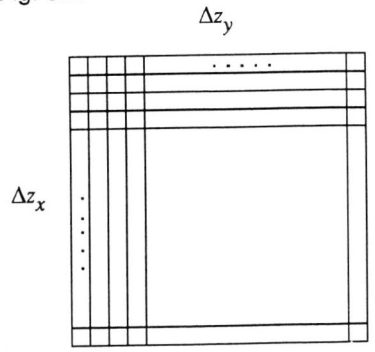

Fig. 3.2 A shading intensity table indexed by Δz_x and Δz_y.

For an n x n x n cubic array,

$$-n \le \Delta z_x \le n,$$

and

$$-n \le \Delta z_y \le n.$$

The size of the look-up table will be $(2n+1) \times (2n+1)$.

Once the table is generated, it can be stored in a file for later use. The look-up table can be applied to other images having the same or lower resolutions. Thus, a table generated for the 64 x 64 x 64 array can be used on a 16 x 16 x 16 array image provided the same values of Δz_x and Δz_y are used. By choosing larger values for Δz_x and Δz_y, say, 4, 6, 8, ... etc., smoother images can be produced. It is suggested, however, that images of lower resolution should use smaller values and images of higher resolution should use larger values. The analogy here is like looking at paintings of different sizes. For smaller paintings, one needs to get a closer look, and for larger paintings, one have to stay back a short distance in order to have a better perception of what is painted there.

3.2 The Image-Space Gradient Shading Method

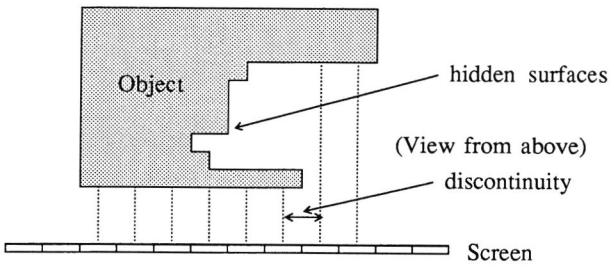

Fig. 3.3 Discontinuities in distance due to hidden surfaces.

The voxel-based method produces high quality images in an efficient way provided that the object to be displayed has continuous surfaces. In cases where there are discontinuities due to hidden surfaces (see Fig. 3.3), the quality of the image degraded due to the fact that the object normal is incorrect at discontinuities. Gordon and Reynolds [7] proposed a weighted average function for better estimation of surface normals at discontinuities.

To obtain the $\partial z/\partial x$ (a similar way can be used to obtain $\partial z/\partial y$), Gordon and Reynolds [7] proposed the following schema for the backward, forward and central differences:

backward difference

$$\delta_b = z_{i,j} - z_{i-1,j}, \qquad (2 \le i \le M, 1 \le j \le M),$$

forward difference

$$\delta_f = z_{i+1,j} - z_{i,j}, \qquad (1 \le i \le M-1, 1 \le j \le M),$$

central difference

$$\delta_c = \frac{1}{2}(z_{i+1,j} - z_{i-1,j}), \qquad (2 \le i \le M-1, 1 \le j \le M).$$

To solve the problem of discontinuities between objects, the following equation is used,

$$\frac{\partial z}{\partial x} = \frac{W(|\delta_b|)\ \delta_b + W(|\delta_f|)\ \delta_f}{W(|\delta_b|) + W(|\delta_f|)}, \qquad (3.1)$$

where, for $\delta \ge 0$,

$$W(\delta) = \begin{cases} 1, & \text{if } \delta \le a, \\ \varepsilon, & \text{if } \delta \le b, \\ \frac{1+\varepsilon}{2} + \frac{1-\varepsilon}{2} \cos\left(\frac{\delta - a}{b - a}\pi\right), & \text{otherwise.} \end{cases} \qquad (3.2)$$

A similar expression is used for $\partial z/\partial y$. ε, a and b are empirical constants: ε is a small number large enough so that division by 2ε maintains sufficient precision. In the implementation described in [7], $\varepsilon=10^{-5}$ is used. a and b are parameters which control the points at which δ's change their weights: a is the maximal δ at which the large weight (W=1) is given, and b is the minimal δ at which the small weight ($W=\varepsilon$) is given. For δ's between a and b the weight varies continuously with the cosine function. This weight function minimizes discontinuities in the image when the orientation of the object is changed. This formula ensures that small δ's are given large weights and large δ's are given small weights. If W_b equals W_f then the central difference is obtained; this will be the case independent on the values of W_b (or W_f). The parameters a and b ensure that the appropriate normal is obtained for rapidly changing slopes. Gordon and Reynolds [7] found that a=2, b=5 produces good images.

3.3 The Normal-averaging Shading Method

In the display of objects (in CT tomograms) represented by voxels, a difficult dilemma is: on one hand it is required that the object should be displayed with as much detail as possible, preferably by some form of shading that will accentuate the curvature of the surface; on the other hand artifacts, which is a result of the digitizing process, should be eliminated or deemphasized. However, when surface curvatures are emphasized strongly, artifacts stand out sharply. By smoothing the image, artifact patterns could be reduced, but at the risk of eliminating important details. Both methods mentioned above produce high quality images. When displaying objects which have steep slopes or rough surfaces, however, a strong sense of "squareness" resulted. A new gradient shading technique is now proposed that minimizes the artifacts and produces very high quality images. This method is termed the *normal-averaging method.*

The voxel-base method and the image-space gradient method share a common feature; both methods make use of the four horizontal and vertical neighbors (Fig. 3.4a) for obtaining surface normals. These neighbors are called the *adjacent* neighbors, and the neighbors on the four corners the *diagonal* neighbors (Fig. 3.4b). The normal obtained by either the adjacent

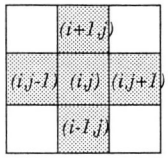

Fig. 3.4a
A 3 x 3 portion of the distance matrix. Shaded parts are the centre and its adjacent

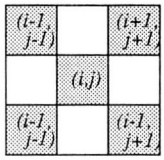

Fig. 3.4b
A 3 x 3 portion of the distance matrix. Shaded parts are the centre and its diagonal

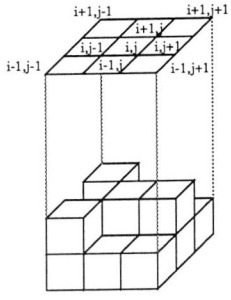

Fig. 3.5a A 3 x 3 portion of the distance matrix and the corresponding object surface.

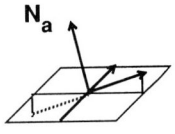

Fig. 3.5b The normal based on adjacent neighbors.

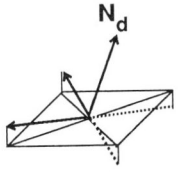

Fig. 3.5c The normal based on diagonal neighbors.

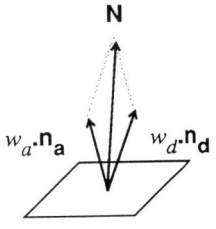

Fig. 3.5d The averaged normal.

neighbors or the diagonal neighbors, however, does not always represent the actual surface orientation. For instance, Figs. 3.5a, 3.5b, and 3.5c show one example out of the many situations where the actual surface normal is poorly estimated by either method alone. A new technique is therefore proposed for getting a better estimation of the the surface normal.The method we have adopted is to estimate the actual surface normal by means of :(1) $\mathbf{n_a}$, the unit normal (i.e. normalized $\mathbf{N_a}$) obtained by means of the adjacent neighbors only, and(2) $\mathbf{n_d}$, the unit normal (i.e. normalized $\mathbf{N_d}$) obtained by means of the diagonal neighbors only, and(3) two weighting factors w_a and w_d.As shown in Fig. 3.5d, the actual normal **N** can be obtained by the equation:

$$\mathbf{N} = w_a.\mathbf{n_a} + w_d.\mathbf{n_d}.$$

The two weighting factors are empirical parameters for adjusting the direction of **N**. If $w_a = w_b = 1$, then **N** will be equally "tilted" between $\mathbf{n_a}$ and $\mathbf{n_b}$. If the *closeness* of the neighboring elements is taken into account with respect to the central element, then using the fact that adjacent neighbors are 1 unit away and the diagonal neighbors are 2 units away, we can assign w_a=1 and w_b=1/ 2. To obtain the surface normal of adjacent neighbors, $\mathbf{N_a}$, use equations **3.1** and **3.2** and the following backward, forward, and central differences for $\partial z/\partial x$:

backward difference

$$\delta_b = z_{i,j} - z_{i-1,j}, \qquad (2 \le i \le M,\ 1 \le j \le M),$$

forward difference

$$\delta_f = z_{i+1,j} - z_{i,j}, \qquad (1 \le i \le M - 1,1 \le j \le M),$$

central difference

$$\delta_c = \frac{1}{2}(z_{i+1,j} - z_{i-1,j}), \qquad (2 \le i \le M - 1,1 \le j \le M).$$

Similar techniques can be used to obtain $\partial z/\partial y$. For the surface normal of diagonal neighbors, $\mathbf{N_d}$, use equation **3.1** and **3.2** and the following backward, forward and central differences for $\partial z/\partial x'$ (define X' to be the the diagonal axis in the SW-NE direction and Y' to be the diagonal axis in the SE-NW direction) :

backward difference

$$\delta_b = z_{i,j} - z_{i-1,j-1}, \qquad (2 \le i \le M,\ 2 \le j \le M),$$

forward difference

$$\delta_f = z_{i+1,j+1} - z_{i,j}, \qquad (1 \le i \le M - 1,1 \le j \le M - 1),$$

central difference

$$\delta_c = \frac{1}{2}(z_{i+1,j+1} - z_{i-1,j-1}), \qquad (2 \le i \le M - 1,2 \le j \le M - 1).$$

And these differences for $\partial z/\partial y'$:

backward difference

$$\delta_b = z_{i,j} - z_{i-1,j+1}, \qquad (2 \le i \le M,\ 2 \le j \le M),$$

forward difference

$$\delta_f = z_{i+1,j-1} - z_{i,j}, \qquad (1 \le i \le M - 1,1 \le j \le M - 1),$$

central difference

$$\delta_c = \frac{1}{2}(z_{i-1,j+1} - z_{i+1,j-1}), \qquad (2 \le i \le M - 1,2 \le j \le M - 1).$$

By letting $N(\theta) = (cos\ \theta)^p = (\mathbf{N}\cdot\mathbf{L})^p$ and using Equation **2.2** and **2.3**, the shading intensity for each point of the object can be obtained. The normal-averaging method does give better quality images than the previous two methods. In order to produce even smoother images, reduce the angular dependency on the cosine function by keeping p<1. Experiments show that p=0.6 produces high quality images.

3.4 Experimental Results

The object used to illustrate the images produced by the gradient shading methods is obtained from SPECT (Single Photon Emission Computed

Tomography) imaging of a liver. The SPECT images is obtained from the General Electric 8800 CT/T transmission tomography scanner. The maximum size of the box that excloses the liver is 64 voxels. The SPECT data of the liver is preprocessed by applying a global thresholding segmentation technique. In Figs. 3.6a-f the images shown are produced by different gradient shading techniques, in the order:

(a) voxel-based shading method,
(b) image-space gradient shading method with diagonal neighbors,
(c) image-space gradient shading method with adjacent neighbors and p=1,
(d) normal-averaging method with p=1,
(e) image-space gradient shading method with adjacent neighbors and p=0.6, and
(f) normal-averaging method with p=0.6.

The orientation of the liver is a coronal image at a 30° tilt upwards in the vertical plane. To demonstrate the flexibility of gradient shading, exposed views (Figs 3.7a-d) of the liver are also provided. The exposed views are generated from a coronal image with "cuts" at distances of 20, 25, 30, and 35 pixels from the first image plane at which the light source is assumed to be located.

The voxel-based shading method tends to emphasize artifacts. This is apparent in Fig. 3.6a. As discussed earlier, the voxel-based shading method is particularly poor in approximating surface normals at discontinuities, this can be seen in Fig. 3.6a where there is a steep slope between the upper part and the lower part of the right liver half (patient's right liver half, i.e. left half shown in the photo).

The image-space gradient shading method reduces the artifacts due to surface discontinuities by using a weighting function (**3.1** and **3.2**), the result of which is smoother images at discontinuities. This can be seen in Fig. 3.6c. Two kinds of images based on the image-space gradient shading method are produced: one based on diagonal neighbors (Fig. 3.6b) and the other based on adjacent neighbors (Figs. 3.6c). The diagonal-neighbor method tends to produce smoother images than the adjacent-neighbor method, however, many details of the image are lost. The image-space gradient shading method, in general, produces better images than the voxel-base shading method, yet artifacts are still quite noticeable.

The newly developed normal-averaging method seems to produce the best quality images among all the gradient shading methods discussed in this paper. The results can be seen in Fig. 3.6d.

To illustrate the effect of angular dependency on the cosine function in $N(\theta) = (cos\ \theta)^p$, in Figs. 3.6e and 3.6f the effect of the parameter p on images is shown. For the two figures the images are generated with p=0.6. Fig. 3.6e is produced by the image-space gradient shaing method using adjacent neighbors, Fig. 3.6f is produced by the normal-averaging shading method. By comparing Fig. 3.6e with Fig. 3.6c and Fig. 3.6f with Fig. 3.6d, it can be seen that images with p=0.6 give smoother shades than images with p=1. In some cases, more detail can be seen at surfaces with steep slope. In other cases, object surface details are not so apparent on the smoother images with p=0.6. On the basis of this dilemma, it is suggested that parameter p could be used as a software "knob" for fine tuning the images to the desired quality.

To demonstrate further the capability of the gradient shading method for producing high quality exposed views of objects, four exposed views of the liver with "cuts" at 20, 25, 30, and 35 voxels from the first image plane are also included. The hollow space inside the liver is due to a preprocess of global threshold segementation being applied on the SPECT images. In the case of surgical planning, seeing the exposed views of internal organs are invaluable to clinicians.

The shaded images are generated on an Apollo DN560 workstation using C as the programming language. It took an average of 80 seconds to produce a distance matrix from a 64 x 64 x 64 voxels image. It took 0.5 seconds to produce a shaded image using the voxel-based shading method, 1 second for the image-space gradient shading method, and 1.5 second for the normal-base shading method. In general, it takes far less time to produce the shaded image than to find the distance matrix. An interesting topic for further research is to implement the distance matrix and gradient shading techniques on special hardware or parallel processors. This provides the oppurtunity for displaying the image in near-real time or even real-time.

5. CONCLUSIONS

The following steps for displaying 3D image data have been described and discussed:

(1) merging of 2D data into a 3D representation, and
(2) 3D object display.

Also investigated are three gradient shading methods; one proposed by Gordon and Reynolds [7] and two other new methods proposed by the authors. The newly proposed normal-averaging shading method gives the best image quality among the three methods. Gradient shading methods, in general, produce high quality images more efficiently than many other shading methods and it is a tool that is indispensable in the area of interactive surgical planning. One prominent feature of the gradient shading is its efficiency in providing exposed views of objects where other shading techniques find difficult to match.

ACKNOWLEDGEMENTS

We would like to thank Dr. R. Hooper and the Department of Nuclear Medicine, Cross Cancer Institute, Edmonton, Alberta for providing image data for

experimentation throughout the research. We also thank the Advanced Technologies Department of the Alberta Research Council, Calgary, Alberta for providing the computing equipment for completing this research. Special thanks go to Dr. Brian Barge and Mr. Maurice Engler (both of the Advanced Technologies Department) for their constant encouragement.

REFERENCES

[1] L. Chen, G.T. Herman, R.A. Reynolds, and J.K. Udupa, "Surface Shading in the Cuberille Environment", *IEEE Computer Graphics and Applications*, Vol. 5, Dec. 1985, pp. 33-43.

[2] H.N. Christiansen and T.W. Sederberg, "Conversion of Complex Contour Line Definitions into Polygonal Element Mosaics", *Computer Graphics*, Vol. 12, Aug. 1978, pp. 187-192.

[3] W.A. Davis and Y.W. Tam, "An Efficient Method for Displaying 3D Images", *Conf. Proc. - Mini and Microcomputers and Their Application*, June 1985, pp. 201-204.

[4] K.S. Fu and J.K. Mui, "A Survey on Image Segmentation", *Pattern Recognition*, Vol. 13, 1981, pp. 3-16.

[5] H. Fuchs, Z.M. Kedem, and S.P. Uselton, "Optimal Surface Reconstruction from Planar Contours", *Comm. ACM*, Vol. 20, Oct. 1977, pp. 693-702.

[6] R. Gillespie and W.A. Davis, "Tree Data Structures for Graphics and Image Processing", *Proc. of the 7th Conf. of CMCCS*, June 1981, pp. 155-161.

[7] D. Gordon and R.A. Reynolds, "Image Space Shading of 3-Dimensional Objects", *Computer Vision, Graphics, and Image Processing*, Vol. 29, 1985, pp. 361-376.

[8] H. Gouraud, "Computer Display of Curved Surfaces", *IEEE Trans. Computers*, Vol. C-20, June 1971, pp. 623-629.

[9] G.T. Herman and H.K. Liu, "Three-Dimensional Display of Human Organs from Computer Tomograms", *Computer Graphics and Image Processing*, Vol. 9, 1979, pp. 1-21.

[10] G.T. Herman and J.K. Udupa, "Display of Three-Dimensional Discrete Surfaces", *Proc. SPIE*, Vol. 283, 1981, pp. 90-97.

[11] E. Keppel, "Approximating Complex Surfaces by Triangulation of Contour Lines", *IBM J. Res. Development*, Vol. 19, Jan. 1975, pp. 1-96.

[12] D.J. Meagher, "Geometric Modeling Using Octree-Encoding", *Computer Graphics and Image Processing*, Vol. 19, 1982, pp. 129-147.

[13] W. Tiller, "Rational B-Splines for Curve and Surface Representation", *IEEE Computer Graphics and Applications*, Sept. 1983, pp. 61-69.

[14] M.W. Vannier, J.L. Marsh, and J.D. Warren, "Three-Dimensional Computer Graphics for Craniofacial Surgical Planning and Evaluation", *Computer Graphics*, Vol. 17, July 1983, pp. 263-266.

[15] S.H. Wu, J.F. Abel, and D.P. Greenberg, "An Interactive Computer Graphics Approach to Surface Representation", *Comm. ACM*, Vol. 20, Oct. 1977, pp. 703-712.

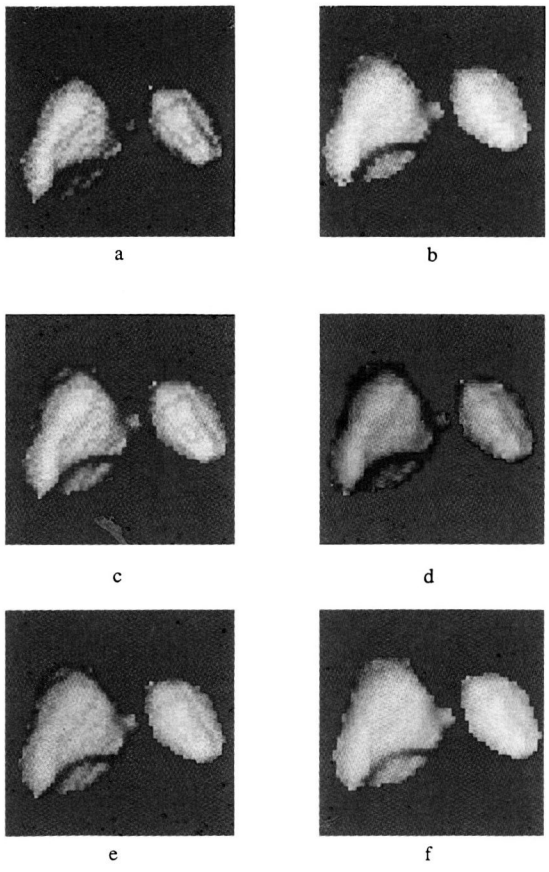

Fig. 3.6 3D display of a liver (coronal view tilted 30° upwards in the vertical plane).

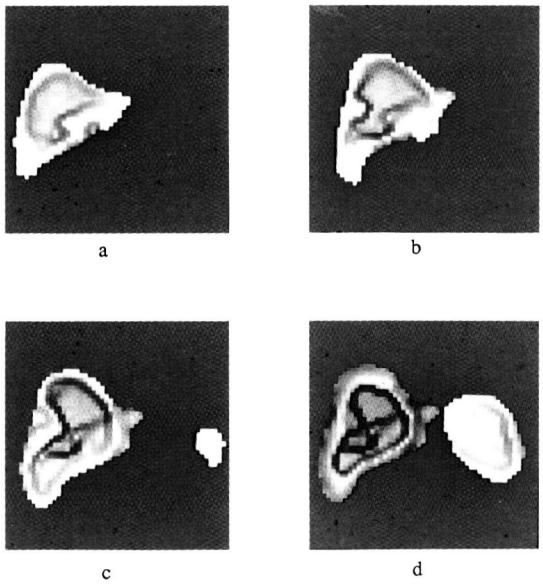

Fig. 3.10 3D display of four exposed view of a liver (with cuts at 20, 25, 30, and 35 voxels respectively from the first image plane).