

## Short Paper

---

# Surface Rendering for Multi-axial Cross Sections

MING-DAR TSAI AND MING-SHIUM HSIEH\*  
*Institute of Information and Computer Engineering  
Chung Yuan Christian University  
Chungli, Taiwan 320, R.O.C.  
E-mail: tsai@ice.cycu.edu.tw*  
\**Orthopaedics and Traumatology Department  
Taipei Medical College Hospital  
Taipei Medical College  
Taipei, Taiwan 110, R.O.C.  
E-mail: shiemin@mail.tmc.edu.tw*

This study presents, for the first time, an automatic algorithm for surface rendering of a volume consisting of multi-axial cross sections. In computer graphics, although a medical volume is usually considered to consist of sequential parallel cross sections, multi-axial parallel sections that constitute a multi-axial volume have also been frequently used in recent clinical cases. Herein, a multi-axial volume is assumed to consist of multiple one-axial subvolumes occupying the same space. The discrete ray tracing algorithm that searches for only a hit voxel in a conventional one-axial volume is extended to search for multiple hit voxels in a multi-axial volume. Quadratic isosurfaces reconstructed from the hit voxels are used to approximate the object surface. The rendering results, applied to clinical cases to improve diagnostic rates, indicate that 3D images obtained by applying our methods to multi-axial volumes are more clinically useful than those obtained from one-axial volumes.

**Keywords:** computer graphics, volume visualization, discrete ray tracing, isosurface reconstruction, medical application

## 1 INTRODUCTION

Visualization of medical volumes provides a non-invasive means of obtaining 3D geometry concepts about anatomic structures that can facilitate diagnosis. However, current volume visualization techniques still can not solve many clinical problems, such as visualizing multi-axial cross sections that constitute a multi-axial volume. Conventional volume visualization techniques can only deal with a volume consisting of a sequence of parallel cross sections. Clinicians do not take sections with small intervals owing to the expense involved. Therefore, a slender anatomic structure may easily go undetected if it

---

Received October 12, 1998; revised May 10 & August 17, 1999; accepted October 27, 1999.  
Communicated by Zen Chen.

lying in a direction nearly perpendicular to the axis of parallel sections. Recently, clinicians have used cross sections of suitable (transverse, coronal, sagittal, or even oblique) axes to resolve slender structures. They have also employed multi-axial sections to resolve slender structures in complicated clinical cases. Fig. 1 shows a bent and slender structure. This structure has two axes, and can be well resolved using two sets of cross sections parallel to the axes.

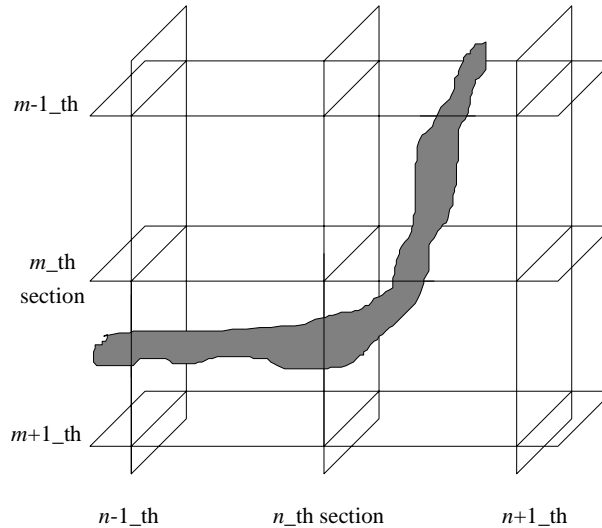


Fig. 1. Multi-axial cross sections for resolving a slender anatomical structure.

Payne and Toga proposed a surface reconstruction algorithm for a multi-axial volume [1]. Their system requires that the user draw the contours of an object and give directions for the contours on multi-axial sections. Next, the system verifies the consistency among the contours within and between the sections and then constructs a triangulated isosurface model based on the contours. It is clearly that their method is not an automatic algorithm, and that it depends on the space concept of the user. A situation in which the contours and the directions are not closed or consistent will cause the isosurface model to be ambiguous [1].

Two approaches to automatic isosurface reconstruction are widely used. One approach pre-generates triangulated isosurfaces before rendering is carried out. The triangles can be generated by the marching cube algorithm ([2]) or the tetrahedral decomposition algorithm ([3]). This approach is advantageous in that re-generating the triangles for different perspectives is unnecessary. Another merit of this approach is that the Gouraud-shading algorithm supported by most 3D rendering acceleration hardware can be applied to shade the triangles.

The other approach to automatic isosurface reconstruction uses the discrete ray traversing algorithm to search for a hit voxel of a ray and then reconstructs an isosurface patch using voxels neighboring the hit voxel [4-7]. The hit voxel is the first voxel traversed by a ray whose value exceeds the threshold of the isosurfaces (overthreshold), and whose previously traversed ones are under the threshold (underthreshold). In contrast to

conventional ray tracing algorithms, the discrete ray tracing algorithm tracks a ray through equally divided voxels and achieves higher efficiency through simple additions and comparisons. This approach does not need a huge amount of memory to store isosurfaces. Another merit of this approach is that higher order isosurfaces can be used to approximate surfaces of anatomic structures in order to obtain higher quality images [8]. This feature is important when visualizing small or slender anatomic structures in a multi-axial medical volume.

This study applies the discrete ray tracing algorithm and applies the local quadratic isosurface reconstruction to multi-axial volumes. Herein, a multi-axial volume is assumed to consist of multiple subvolumes occupying the same volume space. Each subvolume is equivalent to a conventional one-axial volume. The discrete ray tracing algorithm is extended to allow the ray to traverse all the subvolumes simultaneously. In our algorithm, if sections of any axis resolve an anatomic structure, the ray can search, as the discrete ray tracing algorithm does, for a hit voxel from the subvolume constituted by the sections. Then, our extended function allows the ray to search for hit voxels from other subvolumes if they also resolve the anatomical structure. As the result, our discrete ray tracing algorithm can accumulate all the resolution information in sections of different axes so as to render the anatomic structure. For every hit voxel, a quadratic isosurface is reconstructed by  $3 \times 3 \times 3$  neighboring voxels. Then, from all the reconstructed isosurfaces of the ray, we can choose the most near-viewpoint one for shading computation.

Clinical cases are also presented to demonstrate the feasibility of applying our algorithms and system. According to our results, the image quality obtained using quadratic isosurface approximation is better than that by obtained using polygon approximation. Moreover, the anatomic information from 3D images of multi-axial volumes is more useful than that from 3D images of one-axial volumes. The anatomic information and the image quality are helpful for improving the diagnostic rate in clinical cases.

The rest of this paper is organized as follows. Section 2 briefly introduces discrete ray tracing for one-axial volumes. Section 3 describes the proposed discrete ray tracing algorithm for multi-axial volumes. This section also introduces our method for reconstructing quadratic isosurfaces from medical volumes. Next, rendering results are presented in Section 4, which demonstrate the effectiveness of our system in diagnosing a spinal disease where anatomic structures for observation are resolved in different axes of sections. Concluding remarks and discussion are given in Section 5.

## 2. PRELIMINARY REMARKS ON DISCRETE RAY TRACING

Space division, frequently used to improve the efficiency of ray tracing, divides a volume into a set of uniform cubic elements called voxels [9]. The coordinates of the center of a voxel are integers. In contrast to ray tracing, which uses geometric representations for a 3D scene, discrete ray tracing is defined as using 3D discrete voxels to represent a 3D scene [5, 6]. Each voxel has 26 adjacent voxels: 8 vertex-sharing, 12 edge-sharing and 6 face-sharing neighbors. In a 6-ray traversal, the next traversed voxel can only be a face-sharing neighbor of the current voxel. It may be either a vertex- or an edge- or face-sharing neighbor in a 26-ray traversal. Corresponding to a simplified 2D example (Fig. 2), the next pixel in a 4-line (corresponding to a 6-ray) traversal can only

share an edge with the current pixel. These pixels share either an edge or a vertex in an 8-line (corresponding to a 26-ray) traversal.

$d_6$  denotes the distance traversed by a 6-ray traversal. For example,  $d_6$  between the voxels  $(x, y, z)$  and  $(x+a, y+b, z+c)$  is  $(|a|+|b|+|c|)$ .  $d_{26}$  represents the distance traversed by a 26-ray traversal. For example,  $d_{26}$  between the voxels  $(x, y, z)$  and  $(x+a, y+b, z+c)$  is the maximum of  $(|a|, |b|, |c|)$ . The average ratio of  $d_6$  to  $d_{26}$  has been reported to be about 1.84 as determined experimentally [7] or 1.866 as determined analytically [5]. Although  $d_{26}$  is shorter than  $d_6$ , 26-ray traversals have miss hit cases. According to Fig. 2, an object  $O$  undetected by an 8-line traversal is exactly hit by a 4-line traversal. The most efficient approach alternately uses 26- and 6-rays traversals [1, 4]. In this case, 26-ray traversals are used to rapidly traverse an empty space and, 6-ray traversals are used to traverse a space when approaching objects. For a scan conversed volume, a proximity flag can be used to specify voxels near an object [10, 11].

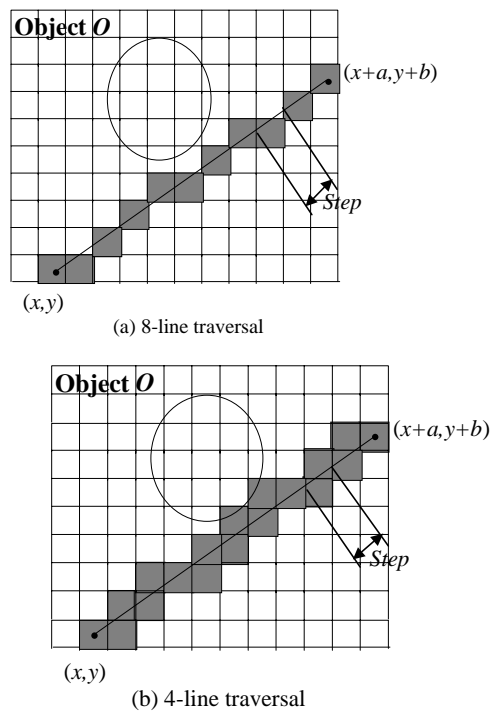


Fig. 2. Ray traversing in 2D view: (a) corresponding to a 26-ray traversal, (b) corresponding to a 6-ray traversal.

The discrete ray tracing algorithm based on incremental computations generates a sequence of voxels, where the next voxel is selected among adjacent neighbors of the current voxel. We will explain the discrete ray tracing algorithms (in this section and the following section) based on a ray that traverses from  $(x, y, z)$  to  $(x+a, y+b, z+c)$ , where  $a \geq b \geq c \geq 0$ . The computation is symmetric for  $b \geq a \geq c$ , for other orders and for negative values of  $a, b, c$ .

For a 26-ray traversal, the 3D-DDA algorithm is the simplest method. However, it requires the use of floating-point operations [6]. According to the corresponding 2D example (Fig. 2(a)), when the  $x$ -value is incremented pixel by pixel, the  $y$ -value is incremented by  $m$ , the slope of the line. Then, the  $y$ -value is rounded into an integer to determine whether the pointer accessing the next pixel should be incremented along the  $x$ -axis or diagonally. The 26-ray traversal algorithm (Fig. 3) replaces the rounding operations with two decision variables,  $e_y$  and  $e_z$  [12, 13], to achieve the fewest operations (three additions and two comparisons in a step [5]).  $e_y$  and  $e_z$  are initialized by the slope of the ray ( $a$ ,  $b$  and  $c$ ) and are incremented by different values  $dy_x$  or  $dy_{xy}$  (for  $e_y$ ) and  $dz_x$  or  $dz_{xz}$  (for  $e_z$ ), depending on whether  $e_y$  and  $e_z$  are positive or negative.  $dy_x$ ,  $dy_{xy}$ ,  $dz_x$  and  $dz_{xz}$  are forward differences computed from  $a$ ,  $b$  and  $c$ . If the decision variables are both negative, the pointer to the next voxel is incremented diagonally, indicating that the next voxel is a vertex-sharing neighbor. If the decision variables are both positive, the next voxel is a face-sharing neighbor. If one is positive and the other is negative the next voxel is an edge-sharing neighbor.

Procedure ray26()

```

{
  if ( $e_y < 0$ ) {
     $e_y += dy_x$ ;
    if ( $e_z < 0$ ) {
       $e_z += dz_x$ ;
       $ptr += offset_x$ ;
    }
    else {
       $e_z += dz_{xz}$ ;
       $ptr += offset_{xz}$ ;
    }
  }
  else {
     $e_y += dy_{xy}$ ;
    if ( $e_z < 0$ ) {
       $e_z += dz_x$ ;
       $ptr += offset_x$ ;
    }
    else {
       $e_z += dz_{xz}$ ;
       $ptr += offset_{xz}$ ;
    }
  }
}

```

Fig. 3. A 26-ray traversing algorithm [5].

In a 6-ray traversal, every voxel along the ray is face-adjacent to its predecessor. From the 3D-DDA algorithm for a 26-ray traversal, the  $x$ -coordinate can be shifted by a half step to obtain the 3D-DDA algorithm for a 6-ray traversal. The tripod algorithm (Fig. 4) uses three decision variables  $e_{xy}$ ,  $e_{xz}$  and  $e_{zy}$ , to avoid the need for floating point operations [5]. The next voxel can be determined by checking whether  $e_{xy}$ ,  $e_{xz}$  and  $e_{zy}$  are positive or negative.  $e_{xy}$ ,  $e_{xz}$  and  $e_{zy}$  are initialized by the slope of the ray ( $a$ ,  $b$  and  $c$ ) and in-

cremented or decreased by  $a_2$ ,  $b_2$  and  $c_2$ .  $a_2$ ,  $b_2$  and  $c_2$  are also computed from  $a$ ,  $b$  and  $c$ .

6-ray and 26-ray traversals can be alternated by transforming the values of the decision variables. For example,  $e_{xy}$  can be obtained from  $e_y$  by subtracting  $b_2$  [5].

```

Procedure ray6()
{
  if ( $e_{xy} < 0$ ) {
    if ( $e_{xz} < 0$ ) {
      ptr =+ offsetx;
       $e_{xy} += b_2$ ;     $e_{xz} += c_2$ ;
    }
    else{
      ptr =+ offsetz;
       $e_{xz} += a_2$ ;     $e_{xy} += b_2$ ;
    }
  }
  else{
    if ( $e_{zy} < 0$ ) {
      ptr =+ offsetz;
       $e_{xz} -= a_2$ ;     $e_{zy} += b_2$ ;
    }
    else{
      ptr =+ offsety;
       $e_{xy} -= a_2$ ;     $e_{zy} -= c_2$ ;
    }
  }
}

```

Fig. 4. A 6-ray traversing algorithm [5].

### 3. SURFACE RENDERING MULTI-AXIAL VOLUME OBTAINED BY MEANS OF DISCRETE RAY CASTING

Although no two voxels in a subvolume of a multi-axial volume overlap, they may partially overlap if they belong to different subvolumes. In the simplified 2D example shown in Fig. 5, two subvolumes  $A$  and  $B$  are mutually orthogonal to each other. The voxel  $\alpha$  belongs to  $A$ . The voxels  $\beta_1$ ,  $\beta_2$ ,  $\beta_3$ ,  $\beta_4$  and  $\beta_5$  belong to  $B$ .  $\alpha$  overlaps with  $\beta_1$ ,  $\beta_2$ ,  $\beta_3$ ,  $\beta_4$  and  $\beta_5$ . The discrete ray tracing and isosurface reconstruction algorithms can be applied to every subvolume as they can be to a one-axial volume. However, applying these algorithms to the whole multi-axial volume requires that some modifications be made for the following two reasons.

First, if a ray hits a voxel in a one-axial volume, the ray no longer will traverse in the volume. However, if the ray hits a voxel in a subvolume of a multi-axial volume, it must continue traversing in other subvolumes of the multi-axial volume. In the example shown in Fig. 5, when the ray is traversing  $\alpha$  in  $A$ , it is simultaneously traversing  $\beta_2$  followed by  $\beta_3$ , followed by  $\beta_4$  in  $B$ . Therefore, if  $\alpha$  is a hit voxel, whether or not  $\beta_2$ , then  $\beta_3$  and then by  $\beta_4$  is also a hit voxel must be checked. If one or more hit voxels are found in

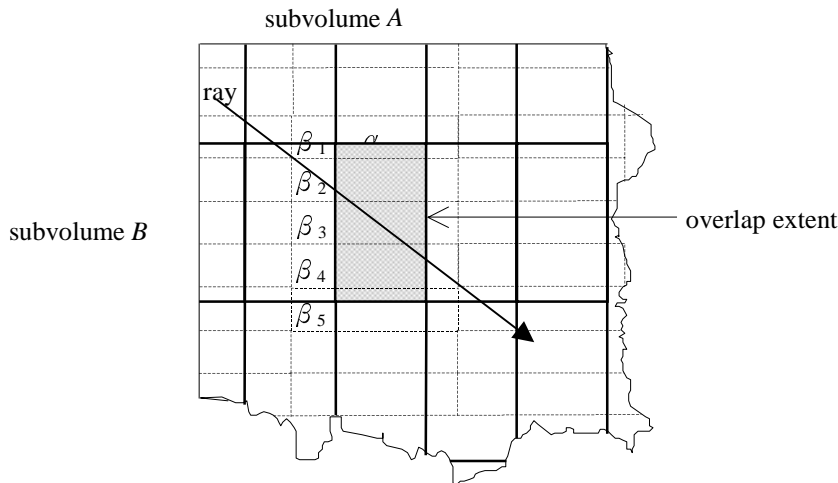
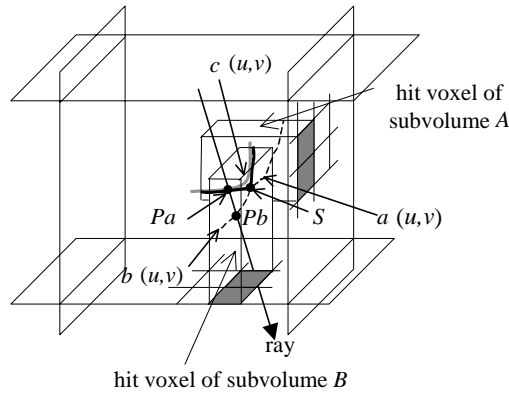


Fig. 5. Ray traversing in a multi-axial volume: a ray is simultaneously traversing the voxel  $\alpha$  in the subvolume A, and the voxel  $\beta_2$ , followed by  $\beta_4$  in subvolume B.

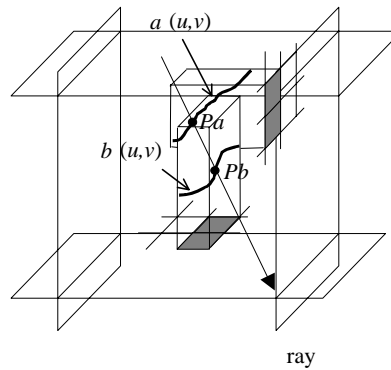
the area of overlap, the ray should stop traversing. Otherwise, the traversing should continue. Therefore, the discrete ray tracing algorithm should be modified as follows: 1) The ray traverses in all subvolumes. 2a) The ray completes traversing in the region of overlap in other subvolumes even it has already hit in some subvolume and then stops traversing to the next extent. 2b) The ray continues traversing to the next extent if therefore no hit voxels. In this study, we extend the 26-ray and the 6-ray algorithms as described by Cohen and Kaufman [5] to implement ray traversing in the multi-axial volume. The proposed algorithm is described in Section 3.1.

Second, no more than one isosurface patch is reconstructed for a ray in a one-axial volume. However, in a multi-axial volume, multiple isosurface patches may be reconstructed because the ray may hit multiple voxels. In the example illustrated in Fig. 6, two isosurface patches have been reconstructed because the ray hits one voxel in subvolume A and one in subvolume B. The isosurface  $a(u,v)$  is reconstructed from the hit voxel of A, and the isosurface  $b(u,v)$  is reconstructed from that of B.  $a(u,v)$  and  $b(u,v)$  may represent a continuous surface of an anatomical structure (Fig. 6(a)), different surfaces of a structure or surfaces of separate structures (Fig. 6(b)).

In the case of a one continuous surface of a structure (Fig. 6(a)),  $a(u,v)$  and  $b(u,v)$  intersect at  $S$ . (Although  $S$  is a point in Fig. 6(a), it is actually a 3D curve.) Either  $a(u,v)$  or  $b(u,v)$  is divided into one inner isosurface and one outer isosurface along the intersection curve. Only the outer isosurfaces (solid line) can be used to approximate  $c(u,v)$ , the real surface of the structure. The inner isosurfaces (broken line) are actually inside the structure and, therefore, should be given up. We use the most near-viewpoint intersection for shading computation as conventional ray tracing algorithms do. Because the intersections on the inner isosurfaces (like  $Pb$ , the intersection of  $b(u,v)$  with the ray) are further to the viewpoint than those on the outer isosurfaces (like  $Pa$ , the intersection of  $a(u,v)$  with the ray), we do not need to worry about whether any of the intersections on the inner isosurfaces become the most near-viewpoint.



(a) Isosurfaces representing one continuous surface



(b) Isosurfaces representing different surfaces

Fig. 6. Isosurfaces for approximating the surfaces of anatomic structures: ray and isosurface intersection are chosen, and then the most near-viewpoint intersection is chosen for shading computation.

In the case of separate surfaces of a structure or separate structures (Figure 6(b)), all the reconstructed isosurfaces are outer ones and can be used together to approximate the surfaces of structures. No inner isosurfaces make intersections invalid. Therefore, regardless of which case (Fig. 6(a) or Fig. 6(b)) is real, we can use the most near-viewpoint intersection for shading computation.

In this study, we use quadrics rather than polygons to approximate surfaces of anatomic structures, thus improving the image quality. Instead of using the  $3 \times 3 \times 3$  neighboring voxels of a hit voxel to determine a quadric as Webber [8] did, we use  $3 \times 3$  sample points calculated from the  $3 \times 3 \times 3$  neighboring voxels to determine the quadric. Although the  $3 \times 3$  sample points are calculated from the  $3 \times 3 \times 3$  neighboring voxels, some points can exceed the extent of the  $3 \times 3 \times 3$  voxels in sharply convex or concave surface cases. Doing so gives a more precise quadratic approximation, particularly for slender or small anatomic structures. The proposed method for isosurface reconstruction is described in Section 3.2.

### 3.1 Discrete Ray Casting Algorithm for a Multiaxial Volume

Fig. 7 illustrates our extended discrete ray tracing algorithm. Although voxels are cubic in the respective subvolume coordinate, they are cuboid in the world coordinate system. In addition, the actual sizes of different subvolumes may be not the same. Therefore, when a value is re-used in another coordinate system, a transformation occurs.

```

While (Tflag is 1 and RE < LV)
{
for (i from 1 to n)
{
    while (Ei < Vi and Ei < RE)
    {
        if (pflag is 1) {
            call ray6(ptri, Fflagi, exyi, exzi, ezyi, a2i, b2i, c2i);
            if (voxel(ptri) is overthreshold) {
                flagi=1; Tflag = 0; break;
            }
            else{ if (Fflagi is 1) Ei +=Stepi;}
        }
        else {
            call ray26(ptri, eyi, ezi, dyxi, dyxyi, dzxi, dzxyi);
            if (voxel(ptri) is overthreshold) {
                flagi=1; Tflag = 0; Fflagi=1; break;
            }
            else{
                if (voxel(ptri) is nearthreshold)
                {pflag = 1; }
                Ei +=Stepi;
            }
        }
    }
}
RE += RStep;
}

```

```

Procedure ray26(ptr, ey, ez, dyx, dyxy, dzx, dzxy)
{
    if (ey < 0) {
        ey += dyx;
        if (ez < 0) {
            ez += dzx;
            ptr += offsetx;
        }
        else{
            ez += dzxz;
        }
    }
}

```

```

        ptr += offsetxz;
    }
}
else{
    ey += dyxy;
    if (ez < 0) {
        ez += dzxz;
        ptr += offsetxz;
    }
    else{
        ez += dzxz;
        ptr += offsetxz;
    }
}
}

Procedure ray6(ptr, Fflag, exy, exz, ezy, a2, b2, c2)
{
    if (exy < 0) {
        if (exz < 0) {
            ptr += offsetxz;    Fflag = 1;
            exy += b2;    exz += c2;
        }
        else{
            ptr += offsetz;    Fflag = 3;
            exz += a2;    ezy += b2;
        }
    }
    else{
        if (ezy < 0) {
            ptr += offsetz;    Fflag = 3;
            exz -= a2;    ezy += b2;
        }
        else{
            ptr += offsety;    Fflag = 2;
            exy -= a2;    ezy -= c2;
        }
    }
}

```

Fig. 7. Discrete ray tracing algorithm for a multi-axial volumes.

Before a ray traverses, we compute the values used in the 26-ray and 6-ray traversing subroutines, including  $dy_x$  and  $dy_{xy}$  (for incrementing  $e_y$ ),  $dz_x$  and  $dz_{xy}$  (for incrementing  $e_z$ ) and  $a_2, b_2, c_2$  (for incrementing  $e_{xy}, e_{xz}$ , and  $e_{zy}$ ). Each subvolume has its own set of  $dy_x, dy_{xy}, e_y, dz_x, dz_{xy}, e_z, a_2, b_2, c_2, e_{xy}, e_{xz}$  and  $e_{zy}$  for the ray. When the ray is traversing in

a ( $i$ -th) subvolume, we compute the step ( $Step_i$ ), entry ( $E_i$ ) and exit ( $V_i$ ) of the subvolume.  $Step_i$  denotes a unit length the ray traverse across one voxel (as in the case of  $Step$  in Fig. 2) and is not changed during traversal of the same ray. The initial value of  $E_i$  is the distance from the viewpoint to the entry of the  $i$ -th subvolume. If the ray is in the 26-ray traversal or traverses across a  $x$ -face adjacent voxel in the 6-ray traversal,  $E_i$  is incremented by  $Step_i$  until  $V_i$ , where  $V_i$  denotes the distance from the viewpoint to the exit of the  $i$ -th subvolume.  $E_i$  is not incremented in the 6-ray traversal if the next voxel is  $y$ - or  $z$ -face adjacent.

$Step_i$ ,  $E_i$  and  $V_i$  are in the world coordinate system. From among the step, initial entry and exit values of all ( $n$ ) subvolumes, some values are chosen to control ray traversal among the subvolumes. Among the step values of all the subvolumes, the largest one is used as  $RStep$  to increment the ray coordinate  $RE$  until  $LV$ . The initial value of  $RE$  is the smallest initial entry of all the subvolumes.  $LV$  represents the exit of the multi-axial volume, which is the largest exit value. Under one  $RStep$  length, the ray traverses voxel by voxel in every subvolume until  $E_i$  is over  $RE$ . Because  $RStep$  is the largest step, the ray can traverse at least one voxel in every subvolume, and all hit voxels will be inside the extent of a  $RStep$ .

Every traversed voxel must be checked to see whether or not it is overthreshold must be checked. If a voxel does, the traversing iteration for this subvolume exits, the flag ( $flag_i$ ) indicating the hitting information of the  $i$ -th subvolume becomes 1 and the flag ( $Tflag$ ) becomes 0 to indicate that the ray has already hit. Then, ray traversing stops after the traversing of the  $Rstep$  extent.  $flag_i$  indicates whether the ( $i$ -th) subvolume contains a hit voxel.

In a 26-ray traversal, the flag  $Pflag$  is used to check whether or not the current voxel is near the threshold value. If it is, the ray traversal is switched to a 6-ray traversal. The near-threshold is used as the proximity flag specifying voxels near the surfaces of structures. Here, some threshold of another tissue is used as the near-threshold. For example, if the tissue for rendering is bone, the threshold of soft tissues becomes the near-threshold. Usually, the ray will not hit in the 26-ray traversal. However, whether or not the voxel is overthreshold in the 26-ray traversal is also checked.

The flag  $Fflag_i$  in a 6-ray traversal is used to indicate which face of the ray hit. This information is used when reconstructing the quadratic isosurface of the hit voxel.

### 3.2 Quadratic Isosurface Reconstruction Using 3×3 Sample Points

We use the following equation to represent the quadratic isosurface of a hit voxel:

$$p = aq^2r^2 + bq^2r + cr^2 + dqr^2 + eqr + fq + gr^2 + hr + k \quad (1)$$

The parametric ( $p$ -,  $q$ - and  $r$ -) axes of the isosurface are parallel to the primary axes of the hit voxel depending on which face of the voxel the ray hit. For example, if the ray hits the  $-x$ -face, the  $q$ -axis is set parallel to the  $y$ -axis, the  $r$ -axis is set parallel to the  $z$ -axis, and the  $p$ -axis is set parallel to the  $-x$ -axis.

The 9 coefficients of the isosurface are determined by  $P_{i,j}(i,j=-1,0,1)$ , an array of 3×3 sample points on the isosurface. In the example illustrated in Fig. 8, the voxel  $\alpha_{0,0,0}$  is

a hit voxel, indicating that it is overthreshold; its predecessor  $\alpha_{1,0,0}$  is underthreshold. Then,  $P_{i,j}$  is determined by  $s_{1,i,j}$  and  $s_{0,i,j}$ , the values of  $3 \times 3$  outside voxels  $\alpha_{1,j,k}(i,j=-1,0,1)$  and  $3 \times 3$  inside voxels  $\alpha_{0,j,k}(i,j=-1,0,1)$  using the following equation, where,  $T$  is the threshold value:

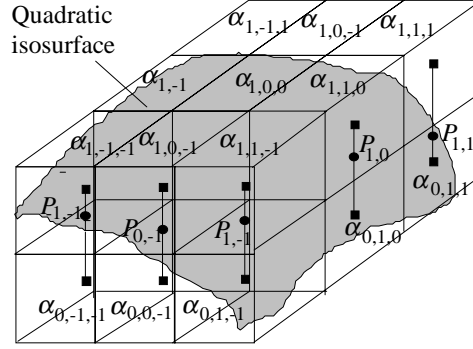


Fig. 8. Isosurface determination by  $3 \times 3$  sample points.

$$P_{i,j} = \frac{s_{0,i,j} - T}{s_{0,i,j} - s_{1,i,j}} \quad (2)$$

However, Equation (2) can only be applied in normal cases, in which the outside voxels are considered to be underthreshold and inside voxels overthreshold. If any outside voxel is overthreshold, a temporary point  $P$  is determined by the central voxel of the outside voxel array and the outside voxel using the following equation:

$$P = \frac{s_{1,i,j} - T}{s_{1,i,j} - s_{1,0,0}} \quad (3)$$

Then, the corresponding sample point is determined by  $P$ . For example, the value of  $\alpha_{1,1,0}$  is overthreshold in Fig. 9(a), where  $P$  is determined by  $\alpha_{1,0,0}$  and  $\alpha_{1,1,0}$  using Equation (3), and  $P_{1,0}$  is then calculated as  $P/P_{0,0}$ . If the outside voxel is overthreshold, whether or not its corresponding inside voxel is underthreshold is confirmed. If it is, a temporary point  $P$  is determined by the central voxel of the inside voxel array and the inside voxel using the following equation:

$$P = \frac{s_{0,0,0} - T}{s_{0,0,0} - s_{0,i,j}} \quad (4)$$

The sample point is then determined by  $P$ . For example,  $\alpha_{1,1,0}$  and  $\alpha_{0,1,0}$  are both underthreshold as shown in Fig. 9(b), where  $P$  is determined by  $\alpha_{0,0,0}$  and  $\alpha_{0,1,0}$  with Equation (4).  $P_{1,0}$  is then calculated as  $-P_{0,0}/P$ .

The parametric distances of  $q$  and  $r$  of all the sample points are  $-1$  or  $0$  or  $1$ , which simplifies the computation required to determine the coefficients of the quadric. The ( $p$ ) values of the sample points represent their positions in the  $p$ -axis direction. To avoid the

need for floating-point computation, we adopt Wallen's method, which divides 1 (the distance between two voxels) as 1,024-fraction [14]. The sample points can be by only at the fraction positions. Here, Equations (2), (3), (4) and the following equations only use integer computation.

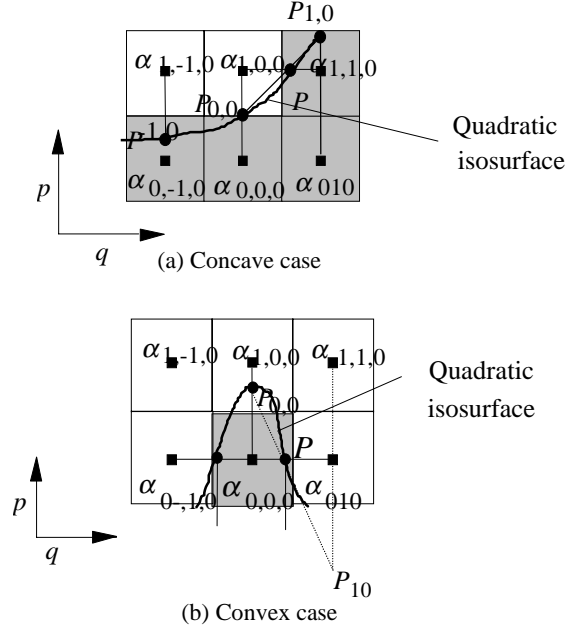


Fig. 9. Sample point determination in special cases.

$k$  is equal to  $P_{0,0}$ .  $c$  and  $f$  are determined by  $P_{1,0}$  and  $P_{-1,0}$  using the following equations:

$$c = \frac{1}{2}(P'_{1,0} + P'_{-1,0}),$$

$$f = \frac{1}{2}(P'_{1,0} - P'_{-1,0}),$$

$$P'_{1,0} = P_{1,0} - P_{0,0}, P'_{-1,0} = P_{-1,0} - P_{0,0}$$

Similarly,  $g$  and  $h$  are determined by  $P_{0,1}$  and  $P_{0,-1}$  using the following equations:

$$g = \frac{1}{2}(P'_{0,1} + P'_{0,-1}),$$

$$h = \frac{1}{2}(P'_{0,1} - P'_{0,-1}),$$

$$P'_{0,1} = P_{0,1} - P_{0,0}, P'_{0,-1} = P_{0,-1} - P_{0,0}$$

In addition,  $a$ ,  $b$ ,  $d$  and  $e$  are determined by  $P_{1,1}$ ,  $P_{-1,1}$ ,  $P_{1,-1}$  and  $P_{-1,-1}$  using the following equations:

$$a = \frac{1}{4}(P'_{1,1} + P'_{-1,1} + P'_{-1,-1} + P'_{1,-1}),$$

$$b = \frac{1}{4}(P'_{1,1} + P'_{-1,1} - P'_{-1,-1} - P'_{1,-1}),$$

$$d = \frac{1}{4}(P'_{1,1} - P'_{-1,1} - P'_{-1,-1} + P'_{1,-1}),$$

$$e = \frac{1}{4}(P'_{1,1} - P'_{-1,1} + P'_{-1,-1} - P'_{1,-1}).$$

$$P'_{1,1} = P_{1,1} - P_{0,0}, P'_{-1,1} = P_{-1,1} - P_{0,0}, P'_{-1,-1} = P_{-1,-1} - P_{0,0}, P'_{1,-1} = P_{1,-1} - P_{0,0}$$

After reconstructing the quadratic isosurface, we parameterize the ray into sub-volume coordinates, where the parameter  $t$  denotes the distance to the viewpoint if multiplying the determinant of the transformation matrix from the subvolume coordinate to the world coordinate. Using the ray equation and the quadric equation allows us to obtain two intersections with their parametric  $(p, q, r)$  values and  $t$ . If any of the  $p, q, r$  values of an intersection is not inside  $-0.5$  and  $0.5$ , the intersection is considered invalid since it is outside the extent of the hit voxel. Comparing the distances among all the valid intersections of all the hit voxels of the ray, we choose the one with the smallest distance and compute the surface normal on it for shading computation.

Because neighboring rays may hit the same voxel, our system reserves the quadrics of voxels hit by the last rays. Rays are cast in a sequential manner from passing through the pixel  $(0, 0)$  until  $(m, n)$ . Therefore, the quadrics of the voxels hit by the rays from passing the pixels  $(m-1, n)$  to  $(m, n-1)$  are reserved. If the ray hits any of these voxels, its quadric is not reconstructed again.

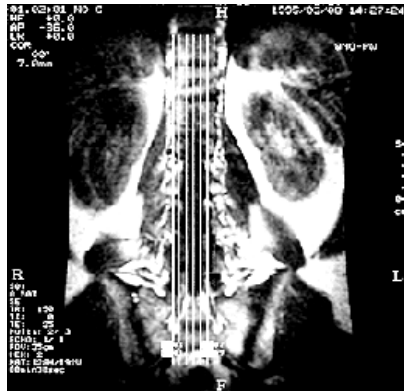
#### 4. EXAMPLE

Currently, the prototype system has been used in the Orthopedics and Traumatology Department of Taipei Medical College Hospital to assist diagnoses of spinal and joint diseases. A related study has demonstrated the effectiveness of our system in more accurately diagnosing a spinal disease and a herniated inter-vertebral disc (HIVD) [15]. In HIVD, substances of disc spaces may compress the spinal cord and roots and may be used to display neurological syndromes, including pain or numbness. Clinicians must know how and where disc spaces compress the spinal cord and roots in order to precisely diagnose related diseases. In diagnosing HIVD, particularly a far lateral disc (FLD), clinicians usually use both sagittal and transverse sections to observe the nervous system (the spinal cord and roots) and disc spaces. This is because sagittal sections are roughly parallel to the spinal cord and roots mainly used to observe the disc spaces. In addition, transverse sections are roughly parallel to the disc spaces mainly used to observe the spinal cord and roots. The resolution information in sagittal, and transverse sections are needed to diagnose HIVD.

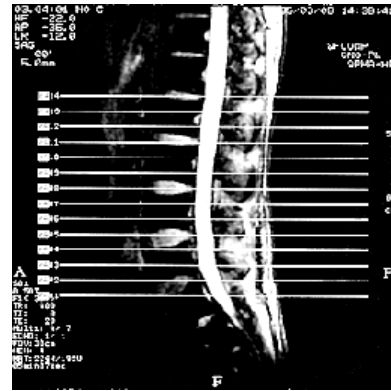
Although 3D images from conventional one-axial volumes can provide 3D geometry concept, they are still difficult to diagnose HIVD [16, 17]. This is due to the fact that the resolution information of sections of different axes can not be simultaneously used to

visualize anatomic structures. Therefore, 3D images obtained using our surface rendering algorithms can improve the diagnostic rate because the proposed algorithms can use all the resolution information in sections of different axes [15].

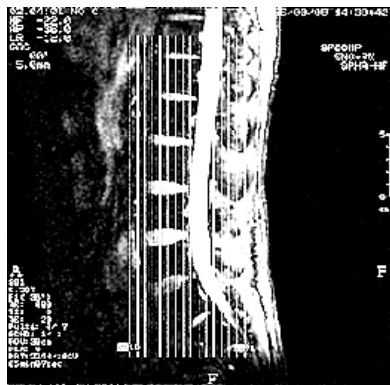
From the related study, we gave some surface rendering results for a typical case of FLD of lumbar HIVD (L-HIVD): a 40-year-old female suffering from right sciatica without low back pain for 1.5 years. The preliminary diagnosis based on some clinical findings was lumbar FLD herniation to the right at the disc spaces L4-5 (fourth lumbar spine - fifth lumbar spine) or L5-S1 (first sacrum spine). MRI was performed with 6 sagittal sections, 12 transverse sections, and 11 coronal sections of the spine under the clinical impression of L-HIVD L4-5 and L5-S1. Here, the coronal sections are used for comparison although they are seldom used for clinical observation of the spine. Fig. 10(a) shows the positions where the sagittal sections crossed. Fig. 10(b) shows where the transverse sections crossed. Fig. 10(c) shows where the coronal sections crossed. Fig. 10(d), (e) and (f) are the optimal images for assessing L-HIVD chosen from among the sagittal, transverse and coronal sections, respectively. The sagittal section image suggests mild bulging of the disc in the L4-L5 space while the transverse section reveals space narrowing in the L4-L5 space and compression of the right nerve root. The coronal section provides no diagnostic information.



(a) Positions where sagittal sections crossed



(b) Positions crossed where transverse sections crossed



(c) Positions where coronal sections crossed



(d) A sagittal section

Fig. 10. Multi-axial sections of the spine for an L-HIVD case.

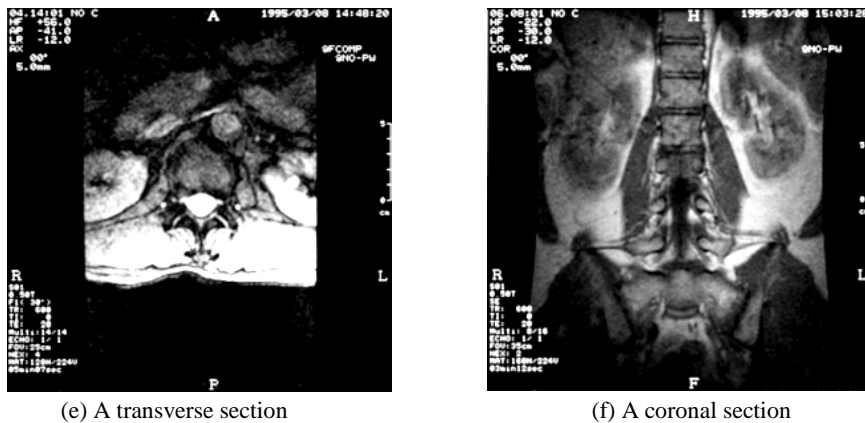


Fig. 10. (Cont'd) Multi-axial sections of the spine for an L-HIVD case.

In this multi-axial (three-axial) volume, the first subvolume is composed of sagittal sections and has  $256 \times 256 \times 6$  voxels. The size of a voxel in this subvolume is  $1.7188\text{mm} \times 1.7188\text{mm} \times 6\text{mm}$ . The second subvolume is by the transverse sections and has  $256 \times 256 \times 14$  voxels. The size of a voxel is  $0.976563\text{mm} \times 0.976563\text{mm} \times 12\text{mm}$ . The third subvolume is by the coronal sections and has  $256 \times 256 \times 11$  voxels. The size of a voxel is  $1.36719\text{mm} \times 1.36719\text{mm} \times 6\text{mm}$ . Of course, the three subvolumes can be considered as three independent one-axial volumes, and any two subvolumes can constitute a two-axial volume. Here, the boundaries of the spinal cord and roots, the veteral body and the disc spaces are manually defined in all the sections.

Figs. 11 and 12 show the rendering results of the one-axial and multi-axial volumes, respectively. Fig. 11 and Figs. 12(a), (b) and (c) display lateral views of the left side of the body. Fig. 12(d) displays a frontal view. Because clinicians mainly observe the relation between the disc spaces and the spinal cord and roots, we augment the veteral body to ease observation. In these figures, the red areas indicate the spinal cord and roots while the green areas represent the disc spaces. The solid arrows denote the central disc with extrusion of L4-5 and nerve root compression at L4. The hollow arrows denote FLD, protrusion at L5-S1 and nerve root compression at L5.

In Figs. 11(a) and (b), from the forth top to down, the disc spaces are L2-L3 (second lumbar spine-third lumbar spine), L3-L4, L4-L5, L5-S1 and S1-S2 (second sarcum spine). Figs. 11(a) and (b) show the results obtained from the quadratic isosurfaces and polygon isosurfaces, respectively. Comparing Figs. 11(a) and (b) reveals that the 3D image approximated by the quadratic isosurfaces is more realistic than that approximated by polygonal isosurfaces. The more realistic image also facilitates diagnosis. Although the 3D image of Fig. 11(b) is reconstructed from the one-axial volume of only six sagittal sections, the 3D geometry of the spine is still depicted. In addition, more solid and realistic information can be obtained than is possible from 2D sections as shown in Fig. 10(d). However, the limited resolution of the image precludes definite diagnosis of herniation at the L4-L5 and L5-S1 spaces. Fig. 11(c) shows good image quality of the spinal cord and roots but poor quality for the disc spaces. Fig. 11(d) shows poorer resolution for the disc space compared to that by the sagittal sections, and poorer resolution of the spinal cord

and roots compared to that by the transverse sections. Because neither the nerve root compression at L4-L5 nor that at L5-S1 is found in Figs. 11(c) and (d), the 3D images of the one-axial volumes of the transverse and coronal sections fail to demonstrate lesions by means of FLD.

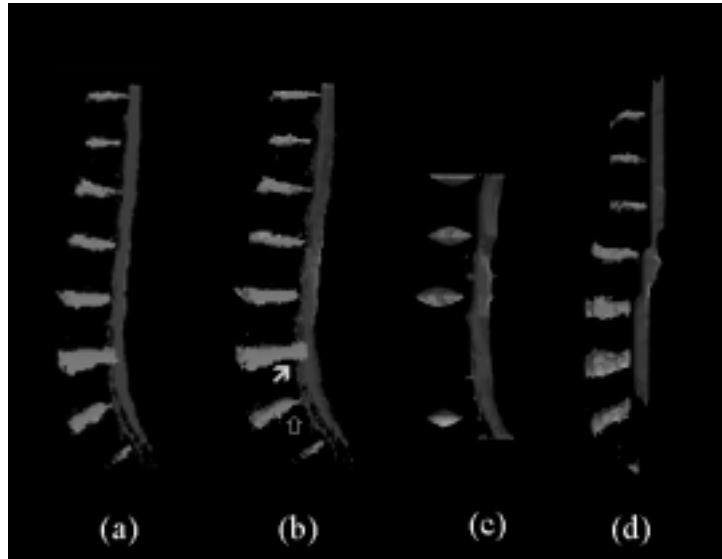


Fig. 11. Rendering results of one-axial volumes: (a) sagittal sections (polygonal approximation), (b) sagittal sections, (c) transverse sections, (d) coronal sections

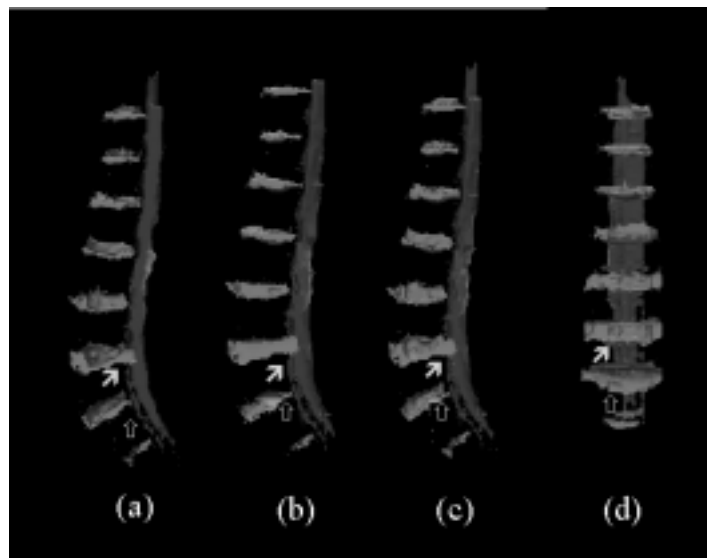


Fig. 12. Rendering results of multi-axial volumes : (a) sagittal and coronal sections, (b) sagittal and transverse sections, (c) sagittal, transverse and coronal sections, (d) sagittal, transverse and coronal sections (frontal view)

Fig. 12(a) shows the result for the two-axial volume composed of the sagittal and coronal sections. Fig. 12(b) shows the result for the two-axial volume composed of the sagittal and transverse sections. Figs. 12(c) and (d) show the results for the three-axis volume composed of the sagittal, transverse and coronal sections. The images (Figs. 12(a) and (b)) from the two-axial volumes provide a more complete picture of the structure of the disc spaces than does that (Fig. 11(b)) from the sagittal sections alone. These images also provide a more complete picture of the spinal cord and roots than does that (Fig. 11(c)) from the transverse sections alone. Moreover, the images (Figs. 12(c) and (d)) from the three-axial volume provide better resolution of the disc spaces than do those (Figs. 12(a) and (b)) from the two-axial volumes.

The clear image of full bulging of the disc space at L4-L5 in Fig. 12(a) and Fig. 12(b) allows easy diagnosis of the central disc, extrusion at L4-5 and nerve root compression at L4. The bulging in the upper portion of the disc at L5-S1 seen in Figs. 12(a) and 12(c) suggests that the spinal roots are pushed away, and that the spinal cord is compressed when the patient walks. The frontal view (Fig. 12(d)) clearly shows the FLD protrusion L5-S1 and nerve root compression at L5 which can not be observed in the frontal view of the 3D image from any of the two-axial volumes. Therefore, optimal diagnoses of the central disc with the extrusion of L4-5, nerve root compression at L4, the FLD protrusion at L5-S1, and nerve root compression at L5 can be achieved by using the three-axial volume. Good diagnoses can be achieved by using either one of the two-axial volumes. Fairly accurate diagnoses can also be achieved by using the one-axial (sagittal) volume. Notably, poor diagnoses may occur if the one-axial volume from either the transverse or coronal sections is used.

## 5. CONCLUSIONS

This study has addressed surface rendering problems for multi-axial cross sections. We have adopted the discrete ray tracing algorithm to improve ray traversing efficiency and to search for hit voxels in multi-axial volumes, and have used  $3 \times 3$  sample points to reconstruct a quadratic isosurface to generate realistic images. Because all the resolution information in cross sections of different axes can be used to approximate the surface of an anatomic structure, 3D images obtained using our proposed discrete ray tracing algorithm can provide the clearest resolution information about the structure. Furthermore, owing to the good quadratic approximation, satisfactory image quality can be expected even when there are few sections and bordered boundaries. Therefore, the diagnostic rate can be improved by using our methods.

Although the proposed methods are quite appropriate for surface rendering of multi-axial volumes, our algorithms are restricted in some ways that may reduce the power of our system. For example, no data structure has been prepared for quadratic isosurfaces. Isosurfaces must be reconstructed again if perspectives are changed. Furthermore, although the hit voxels of a ray from different subvolumes are inside the extent of an increment step, they do not necessarily overlap. This finding suggests that our discrete ray tracing algorithm for multi-axial volumes does not achieve the best efficiency. These restrictions will be dealt with in future works to make our methods more effective.

## ACKNOWLEDGMENT

The authors would like to thank National Science Council for financially supporting this research under Contract No. NSC-85-2213-E033-030.

## REFERENCES

1. B. A. Payne and A. W. Toga, "Surface reconstruction by multiaxial triangulation," *IEEE Computer Graphics and Applications*, Vol. 14, No. 6, 1994, pp. 28-35.
2. W. E. Lorensen and H. E. Cline, "Marching cubes: a high resolution 3D surface construction algorithm," *ACM Computer Graphics*, Vol. 21, No. 4, 1987, pp. 163-169.
3. A. Gueziec and R. Hummel, "Exploiting triangulated surface extraction using tetrahedral decomposition," *IEEE Transaction on Visualization and Computer Graphics*, Vol. 1, No. 4, 1995, pp. 328-342.
4. J. Amanatides and A. Woo, "A fast voxel traversal algorithm for ray tracing," in *Proceedings of Eurographics '87*, 1987, pp. 3-9.
5. D. Cohen and A. Kaufman, "3D line voxelization and connectivity control," *IEEE Computer Graphics and Applications*, Vol. 17, No. 6, 1997, pp. 80-87.
6. A. Fujimoto, T. Tanata, and K. Iwaba, "ATRS: accelerated ray-tracing system," *IEEE Computer Graphics and Applications*, Vol. 6, No. 4, 1986, pp.16-26.
7. R. Yangel, D. Cohen, and A. Kaufman, "Discrete ray tracing," *IEEE Computer Graphics and Applications*, Vol. 12, No. 5, 1992, pp.19-29.
8. R.E. Webber, "Ray tracing voxel data via biquadratic local surface interpolation," *Visual Computer*, Vol. 6, No. 1, 1992, pp. 8-15.
9. A. Kaufman, D. Cohen, and R. Yagel, "Volume graphics," *Computer*, Vol. 26, No. 7, 1993, pp.51-64.
10. D. Cohen and A. Kaufman, "Fundamentals of surface voxelization," *Graphics Models and Image Processing*, Vol. 56, No. 6, 1995, pp. 453-461.
11. S. Wang and A. Kaufman, "Volume-sampled 3D modeling," *IEEE Computer Graphics and Applications*, Vol. 14, No. 5, 1994, pp. 26-32.
12. J. R. Van Aken and M. Novak, "Curve-drawing algorithms for raster displays," *ACM Transaction on Graphics*, Vol. 4, No. 2, 1985, pp.147-169.
13. M.D. McIlroy, "Best approximate circles on integer grids," *ACM Transaction on Graphics*, Vol. 2, No. 4, 1983, pp. 237-263.
14. A. Wallin, "Constructing isosurfaces from CT data," *IEEE Computer Graphics and Applications*. Vol. 11, No. 6, 1991, pp. 28-33.
15. M. S. Hsieh and M. D. Tsai, "Diagnosis of herniated intervertebral disc assisted by 3-dimensional, multiaxial, magnetic resonance imaging," *Journal of the Formosan Medical Association*, Vol. 98, No. 5, 1999, pp. 347-355.
16. E. J. Augtuaio, J. B. Holder, and W. C. Boop, "Computed tomography discography in the evaluation of extreme lateral disc herniation," *Neurosurgery*, Vol. 14, No. 3, 1984, pp. 350-351.
17. M. Kornberg, "Extreme lateral lumbar disc herniations," *Spine*, Vol. 12, 1987, pp. 586-589.

**Ming-Dar Tsai (蔡明達)** was born in Yunlin, Taiwan, R.O.C., in 1960. He received the BS degree in mechanical engineering from National Taiwan University, Taipei, Taiwan, in 1983, and the MS and Ph.D. degrees in machinery precision engineering from the University of Tokyo, Tokyo, Japan, in 1988 and 1991, respectively.

Since 1991, he has been on the faculty of the Department of Information and Computer Engineering, Chung Yuan Christian University, Chungli, Taiwan, where he is currently an associate professor. His research interests include computer graphics, virtual reality, scientific visualization and computer use in medical applications.

**Ming-Shium Hsieh (謝明勳)** was born in Yunlin, Taiwan, R.O.C., in 1948. He received the M.D. degree in Medicine from Taipei Medical College, Taiwan, in 1974, and the Ph.D. degree in Orthopedic Surgery from Essen University, Essen, Germany, in 1982.

Since 1986, he has been on the faculty of the Department of Orthopedics at Taipei Medical College, Taipei, Taiwan, where he is currently the chairman and an associate professor. His research interests include image studies, spine surgery, arthroplasty and sport medicine.