MVA '92 IAPR Workshop on Machine Vision Applications Dec. 7-9,1992, Tokyo

Interactive Surface Display for 3D Medical Images using Multilayer Range Images

Hideyuki Ban, Akihide Hashizume
Ryuuichi Suzuki*

Central Research Laboratory, Hitachi, Ltd.
1-280, Higashi-koigakubo, Kokubunji, Tokyo 185, Japan
*System Development Laboratory, Hitachi, Ltd.
1099, Ohzenji, Asao-ku, Kawasaki, Kanagawa 215, Japan

ABSTRACT

A high-speed display algorithm for surface images from volume data is proposed. This algorithm uses a new data structure called *multilayer range image* (MLRI) data. It displays a surface image within 0.13 seconds from 256x256x256 volume data without any special-purpose hardware such as a graphics accelerator. In addition to the algorithm, we propose a rotation operation method for interactive manipulation. This method can be used to rotate the surface in the way as a real object.

1. INTRODUCTION

Three-dimensional (3D) medical images are widely used in clinical diagnosis, surgical planing and radiation therapy. Because of the huge volume of data obtained from medical imaging systems such as MRI (Magnetic Resonance Imaging), it is difficult to interactively manipulate and display such 3D images. Previous work in this area solved the problem by using special-purpose hardware accelerators [1]. Recently, general-purpose hardware has been progressing more rapidly. For example, the performance of engineering workstations will soon reach 100 million instructions per second (MIPS) [2]. We have therefore developed a high-speed algorithm that can be implemented on general-purpose workstation.

2. HIGH-SPEED DISPLAY METHOD

2.1 Bottleneck of High-Speed Display

Figure 1 shows a typical display algorithm for surface images [3]. In the processes in Figure 1, the object recognition and the parallel projection process require more time than other processes because of the 3D volume data manipulation. However, the object recognition process is executed only one time for each object and is not necessary when the operator changes to a different viewpoint. Thus, it is little need to improve this process for interactive displays. However, the parallel projection process is executed whenever the operator changes viewpoint. Thus, this process causes a bottleneck for interactive or high-speed displays, so it is important to increase the process speed. Hence, we studied a high-speed algorithm for the parallel projection process.

2.2 Reduction with Projection Data

The relationship between the object and the projection plane is shown in Figure 2. In a typical display algorithm, the parallel projection process transforms the all surface coordinates in the binary volume data from the object axis (x, y, z) to the projection plane axis (u, v, w). The number of transformed coordinates (projection data) is about several tens of thousands, so a lot of time is required. However, about a half of the projection data are for the surface hidden from the viewpoint and therefore unnecessary data. In any case, these data are removed by the hidden surface removal process. Consequently, we have developed a method of reducing these unnecessary data.

This reduction method generates the range image from any viewpoint by combining (or interpolating) one, two, or three range images projected on the x, y, or z axes as shown in Figure 3. For simplicity, Figure 3 describes only the xy-plane (at z=0) of the object shown in Figure 2. We shall describe each range image on the x, y, or z axes by the axis name and direction (plus or minus) such as X+

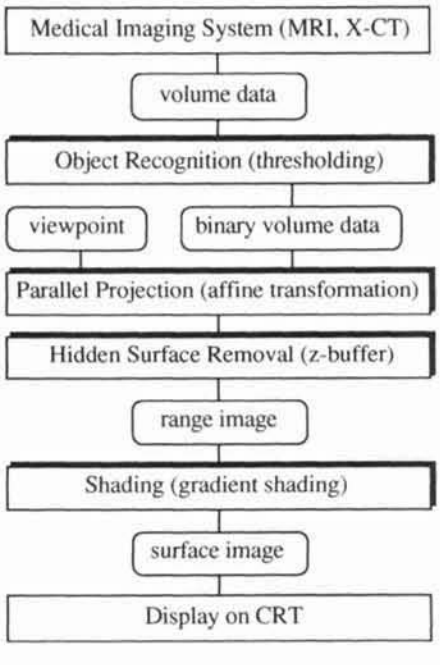


Fig. 1 Typical Display Algorithm

and Y-. Each range image includes the distance from the visible surface to the virtual projection plane in the volume data. Thus, each range image contains information about the surface coordinates in the volume data. For example, the range image (X+) contains the coordinates of voxels **a**, **b** and **c**, and the range image (Y+) contains the coordinates of voxels **c**, **d** and **e**.

The range image from any viewpoint can be generated from, at most, three items of the range image on the x, y, or z axes as shown in Figure 4. In this figure, the range images (X+) and (Y+) provide all the coordinate information about the voxels that are visible from this viewpoint. Therefore, the range image from this viewpoint is generated by affine transformation and z-buffer algorithm from the voxel coordinates in the range images (X+) and (Y+). The appropriate range images on the x, y, or z axes are selected in accordance with the direction of the viewpoint.

This method has the advantage of the high-speed display for the following reasons.

1. Reduction of projection data for the hidden surface.
2. Generation from two-dimensional range images on the x, y, or z axes (smaller than 3D volume data).

2.3 MLRI Data Structure

The above method has the problem that the range image of a hollow object cannot be fully generated at the viewpoint. In the case of Figure 4, the at $\mathbf{u}_{13}$ is zero because there is no coordinate for voxel **f** in any of the range images on the x, y, or z axes. To solve this problem, we propose a new data structure, *multilayer range image* (MLRI), which contains information about these voxels.

The range image shows the distance from a visible surface in the volume data to the projection plane. The MLRI data provides an augmented range image that includes information about the distance, in the volume data, from an invisible surface to a projection plane. In the case of Figure 3 and Figure 4, the distance for voxel **f** is contained in the augmented range image (Y+), that is MLRI (Y+). Therefore, the above method generates a complete range image from the viewpoint by using MLRI data instead of the range images on the x, y, or z axes.

Figure 5 is an example of MLRI data (X+). It contains the distance **s** from the invisible surface that is shown as voxel **S** in Figure 2, in addition to the distance **r** from visible surface that is shown as voxel **R** in Figure 2. Thus, the MLRI data contains all the information about the voxel's position, i.e. the surface shape of the object. All information about the surface voxel positions is converted to at least one of the six MLRI data (X+, Y+, Z+, X-, Y-, Z-).

3. PROPOSED ALGORITHM

3.1 Algorithm Outline

The proposed algorithm generates a range image from the viewpoint using MLRI data. This algorithm consists of three steps as following.

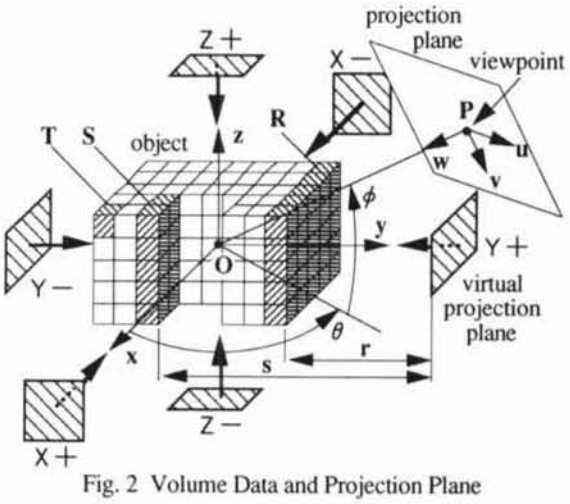


Fig. 2 Volume Data and Projection Plane

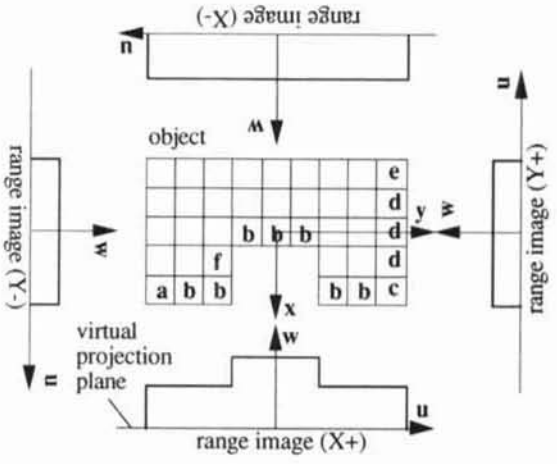


Fig. 3 Range Image on x and y axes

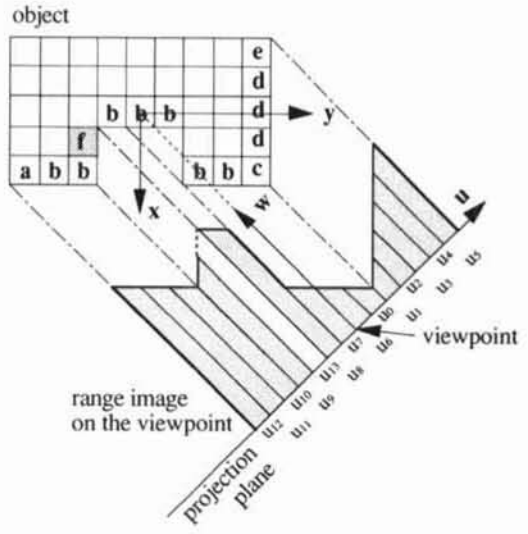


Fig. 4 Range Image generation

Step 1. Transformation to MLRI data

The first step generates binary volume data by thresholding and converts it to MLRI data for the six directions of the virtual projection plane, as shown in Figure 2. First, this step searches the volume data from the virtual projection plane to the object in parallel with the axis. If the value of the voxel is larger than the threshold level, the voxel is recognized as the surface and the coordinate of the voxel is stored into MLRI data. The six items of MLRI data are generated by searching in six direction (X+, Y+, Z+, X-, Y-, Z-).

Step 2. Create Range Image

In this step, we select, at most, three items of MLRI data from the six in accordance with the direction of the viewpoint and create the range image on the viewpoint using the selected MLRI data. This selection uses the component of the vector that is from the origin of the object (**O** in Figure 2) to the viewpoint (**P** in Figure 2), as shown in Table 1. We select MLRI data for each component. For example, if the sign of the component (x, y, z) is (plus, minus, zero), MLRI data X+ and Y- are selected and MLRI data about the z-axis is not selected.

MLRI data contains the information about a hollow part of the object. So, the selected MLRI data contains all information about the voxels which are visible from the viewpoint. Therefore, the range image on the viewpoint is generated completely by affine transformation and z-buffer algorithm about the voxel's coordinates in the selected MLRI data.

Step 3. Shading and Display

In this step, we generate the surface image from the range image by gradient shading algorithm and display it. This step may use the ordinal generation algorithms [3].

3.2 Optimization of MLRI Data

Now, we aim at the corner of the object, such as voxel **T** in Figure 2. Because surface voxels on the corner can be visible from two or three directions of the x, y, or z axes, the coordinate of the voxels is contained in two or three items of MLRI data. In the case of voxel **T** in Figure 2, three items of MLRI data (X+,Y-,Z+) contain the coordinate of **T**. Therefore, the above algorithm executes affine transformation and z-buffer process two or three times about the same voxel's coordinate as at Step 2. To avoid these unnecessary executions, we optimize MLRI data by the removal of this overlapping information.

A special case is when one or two items of MLRI data are selected at Step 2. In this case, the viewpoint is just on the xy-plane, yz-plane or zx-plane. Consequently, it is no problem that we suppose three items of MLRI data are always selected. Then, we compress the selected three items of MLRI data and remove the overlapping information. We call this, "data optimized-MLRI data." The optimized-MLRI data consists of eight items (OPT1-OPT8), as shown in Table 2. When we generate the range image on the viewpoint, one item of the optimized-MLRI data is selected using the component of the same vector defined at Step 2. This selection logic is also described in Table 2.

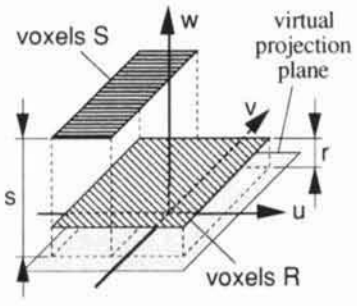


Fig. 5 Multilayer Range Image
(The case of Y+ shown in Fig. 2.)

Table. 1 The selection logic of MLRI data

(a) about x component

sign of component	selected MLRI data
plus	**X+**
zero	not selected about x
minus	**X-**

(b) about y component

sign of component	selected MLRI data
plus	**Y+**
zero	not selected about y
minus	**Y-**

(c) about z component

sign of component	selected MLRI data
plus	**Z+**
zero	not selected about z
minus	**Z-**

The components are for the vector from the origin of the object to the viewpoint (vector $\overrightarrow{OP}$ in Fig. 2).

Table. 2 Optimized-MLRI data

item name	source of MLRI data	sign of component as selected (x,y,z)*
OPT1	**X+, Y+, Z**	(plus, plus, plus)
OPT2	**X-, Y+, Z**	(minus, plus, plus)
OPT3	**X+, Y-, Z**	(plus, minus, plus)
OPT4	**X-, Y-, Z**	(minus, minus, plus)
OPT5	**X+, Y+, Z**	(plus, plus, minus)
OPT6	**X-, Y+, Z**	(minus, plus, minus)
OPT7	**X+, Y-, Z**	(plus, minus, minus)
OPT8	**X-, Y-, Z**	(minus, minus, minus)

The components are for the vector from the origin of the object to the viewpoint (vector $\overrightarrow{OP}$ in Fig. 2).

*: The plus sign includes zero in this table.

3.3 Rotation Operation

When we observe the point of interest of the object, we usually rotate the object to display this point at the center of the screen. Therefore, in order to manipulate the surface image in the same way, the authors propose a rotation operation method using a two-dimensional pointing device such as a mouse. First, the operator moves the pointer to a point on the surface image. Next, he or she presses the mouse button and drags the pointer in a certain direction. Then, the surface image rotates to follow the pointer's movement. Finally, the mouse button is released for the desired view. An example of the proposed rotation operation is shown in Figure 6. With this "Object direct manipulation," the operator can easily change the object direction as desired.

4. IMPLEMENTATION AND ESTIMATION

We implemented the proposed algorithm and operation method on a 76-MIPS general-purpose workstation (Hewlett-Packard HP9000/730CRX) and estimated the processing time using 256x256x256 volume data generated by MRI. The result of this estimation is shown in Figure 7 and Table 3.

We executed z-buffer and gradient shading processes for each pixel that exists the surface image on the screen. The number of these necessary pixels depends on the direction of the viewpoint. Therefore, the processing time changes when the operator moves the viewpoint. As shown in Table 3, the proposed algorithm displays the surface image within 0.13 seconds. This algorithm is over ten times faster than the ordinal generation algorithm that takes about 5 seconds [4].

The proposed rotation operation method can rotate and display the surface image interactively because real-time response is achieved.

5. CONCLUSION AND REMARKS

To achieve an interactive surface display, the authors proposed a high-speed display algorithm using MLRI, a new data structure. We showed that a 76-MIPS general-purpose workstation can display a surface image within 0.13 seconds from 256x256x256 volume data when using the proposed algorithm. The algorithm has the advantage of eliminating the need for any special-purpose hardware such as a graphics accelerator. In addition to the algorithm, we proposed an interactive rotation operation method with object direct manipulation. We showed that the proposed method is simple and effective.

The main areas for future research are as follows.

1. Advanced functions, such as multi-surface images, a combination with oblique plane images, and so on.
2. A quantitative estimation of the proposed rotation operation method.

REFERENCES

[1] Kaufman, A. and Bakalash, R., "Memory and Processing Architecture for 3D Voxel-Based Imagery," IEEE Computer Graphics and Application, **8**, 6 (November 1988), pp. 10-23

[2] Horning, R. et al, "System Design for Low Cost PA-RISC Desktop Workstation", COMPCON spring 91 (February 1991), pp. 208-213

[3] Kaufman, A., "Volume Visualization," IEEE Computer Society Press (1991)

[4] Furuhata, K. and Suto, Y., "Application of Graphics Technology to Medical Field - surgical simulation," Information Processing Society of Japan Technical Report, Graphics and CAD 43-4 (February 1990), pp.1-9 (in Japanese)

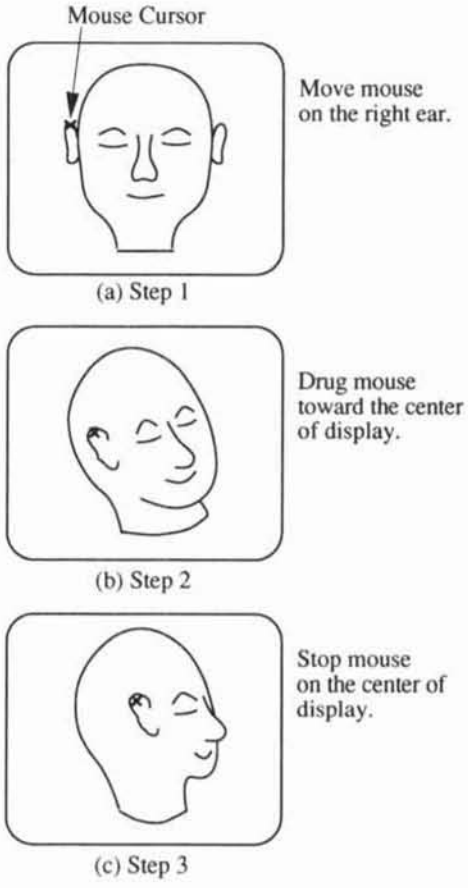
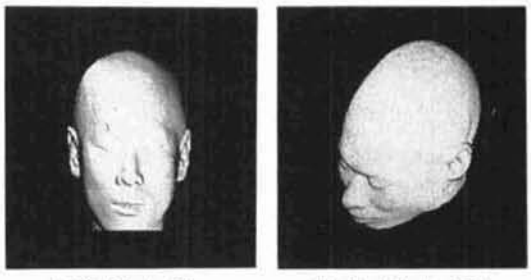


Fig. 6 Object Direct Manipulation
(Rotate the head to observe the right ear.)



(a) $\theta=0°, \phi=0°$ (b) $\theta=45°, \phi=45°$

Fig. 7 Examples of Surface Display

Table. 3 Estimation of the processing time

direction of viewpoint	processing time (sec.)
$\theta = 0°, \phi = 0°$	0.10
$\theta = 45°, \phi = 45°$	0.13