A High Performance Direct Volume Rendering Pipeline

Marcelo Knörich Zuffo
Roseli de Deus Lopes

Grupo de Computação Gráfica
Laboratório de Sistemas Integráveis
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, n.158
05508-900 -- São Paulo, SP, Brazil
email: mkzuffo@lsi.usp.br, roseli@lsi.usp.br

Abstract A high performance direct volume rendering pipeline is proposed. This proposal has as its main qualities a high computational performance, an accurate gradient approximation and an avoidance of large intermediary data structures.

The performance of the new pipeline is shown using an analytical model and experimental results. Two implementations were done for experimental performance evaluation: one based on the proposed pipeline and another based on Levoy's ray casting method.

Keywords: direct volume rendering, high performance visualization.

1. Introduction

In the last few years, the scientific visualization field has grown with the several technical and algorithmic developments in volume visualization [5] and the availability of faster computers. Meanwhile, from a computational point of view, volume visualization usually requires high computer power and large amounts of memory [5, 25].

Big efforts have been made in algorithmic optimizations in high performance computers to make real time volume visualization possible [2, 3, 7, 15, 16, 18, 20, 24]. The majority of them concentrates in mapping the existent algorithms to match the architecture of the high performance computer used.

One of the most disseminated volume rendering algorithm uses the ray-casting method for direct volume rendering. It was proposed by Levoy [11, 12], and it is referred here as **Levoy's direct rendering pipeline.** Due to the high image quality obtained, this algorithm opened many possibilities of use in several applications.

This paper presents an alternative pipeline, aiming at reduction of computation time and intermediary data structures, by means a reordering of the pipeline stages. In this pipeline the resampling step is performed on the original data space. It uses a faster and more accurate way to evaluate the gradient vector, resulting in faster generation of higher quality images.

2. Levoy's direct rendering pipeline

The original Levoy's pipeline is presented in [11, 12] and some improvements can be found in [13, 14]. We briefly describe it here. Figure 1 shows a sketch of Levoy's direct rendering pipeline.

The first step is the conversion of a scalar data volume to an optical data volume (RGBO volume). This is usually done using look-up-tables to determine the initial opacity and color of each voxel.

Good quality images can be obtained choosing appropriate classification functions for displaying any kind of material (opaque or semi-transparent).

The voxel opacity is adjusted using a local gradient magnitude estimation, according to the following equation:

$$sample\_opacity = |\nabla f(x,y,z)| \cdot o[f(x,y,z)] \qquad \textit{(eq. 1)}$$

where:
f(x,y,z) = scalar value of the original volume data;
∇ = gradient operator;
o = opacity classification function.

The voxel color is adjusted applying an illumination model that uses the local gradient estimation as an approximation of the voxel normal vector.

A precise gradient estimation is the key for shading and opacity evaluation.

Then after that, the resampling of the optical data volume (RGBO volume) is done using the ray-casting paradigm. Each final image pixel is computed by accumulating the contribution of opacity and color of each resampled point along the correspondent ray.

Interpolation methods are used to determine each resampled point using information of its neighborhood. The most common methods are the nearest neighbor and the trilinear interpolation. Higher interpolation schemes such as triquadratic and tricubic can also be used at the expense of a dramatic increase in computational time.

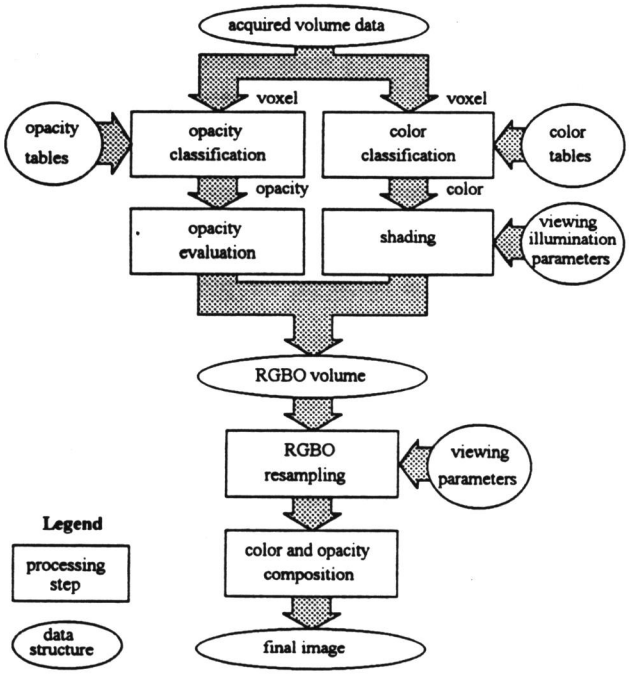

Figure 1 -- Levoy's direct rendering pipeline

From a performance point of view, the main disadvantages of this approach are:

- the necessity of storing an intermediary data structure -- the RGBO volume -- with the same dimension as the original data volume [12],
- the necessity of opacity (O) and color (R, G, B) interpolation during resampling.

Levoy suggests several optimizations to improve the performance, such as:

- stopping the traverse along a ray when accumulating more samples will not change the color of the pixel significantly [11, 12];
- hierarchical spatial enumeration that avoids traversing parts of the volume which contains no data [14];
- adaptative refinement [13].

2.1. Gradient evaluation

The local gradient is calculated by the following equation [10]:

$$\nabla f(x,y,z) = \left(\frac{\partial}{\partial x} f(x,y,z), \frac{\partial}{\partial y} f(x,y,z), \frac{\partial}{\partial z} f(x,y,z)\right) \qquad (eq.\ 2)$$

Levoy uses a 6-neighborhood central difference approximation for the local gradient estimation [12], as shown in figure 2. Using 6-neighborhood central difference approximation, the local gradient is estimated as:

$$\frac{\partial}{\partial x} f(x,y,z) \cong \frac{\Delta f(x,y,z)}{\Delta x} = \frac{f(x+1,y,z) - f(x-1,y,z)}{2}$$
$$\frac{\partial}{\partial y} f(x,y,z) \cong \frac{\Delta f(x,y,z)}{\Delta y} = \frac{f(x,y+1,z) - f(x,y-1,z)}{2} \qquad (eq.3)$$
$$\frac{\partial}{\partial z} f(x,y,z) \cong \frac{\Delta f(x,y,z)}{\Delta z} = \frac{f(x,y,z+1) - f(x,y,z-1)}{2}$$

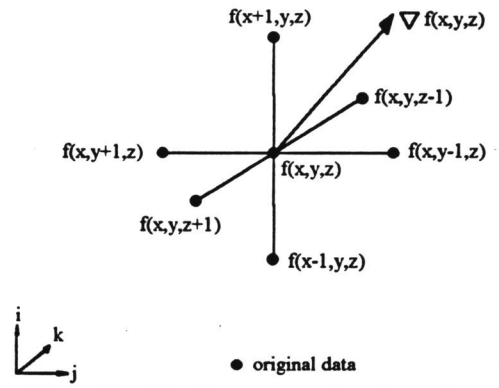

Figure 2 -- Gradient approximation using 6-neighborhood

In Levoy's approach, the gradient is estimated for each voxel of the original volume data (figure 3). Then for each voxel an illumination equation is applied and at resampling step the RGBO voxel is interpolated.

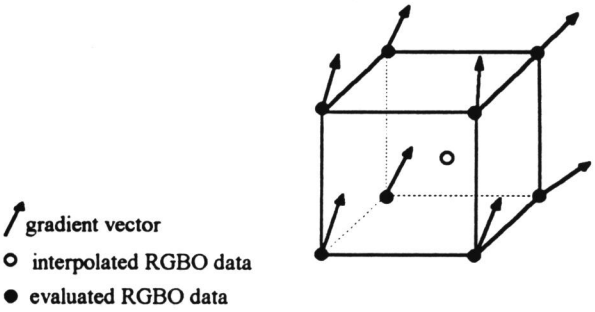

Figure 3 -- Gradient evaluation at Levoy's approach

2.2. Trilinear interpolation

The trilinear interpolation establishes a compromise between computational complexity and image quality.

Given a unitary cube with eight vertices each one corresponding to one voxel, the trilinear interpolation function that defines a scalar field among these eight voxels is done by equation 4:

$$f(x,y,z) = Ax + By + Cz + Dxy + Exz + Fyz + Gxyz + H \quad (eq.\ 4)$$

The interpolation function coefficients (A, B, C, D, E, F, G and H) can be easily determined from the eight voxels belonging to the unitary cube.

In conventional implementations, where high execution performance is necessary, the trilinear interpolation can be evaluated as shown in figure 4.

In this case, the numerical complexity (only 7 linear interpolations) is less than it is by solving the correspondent linear system.

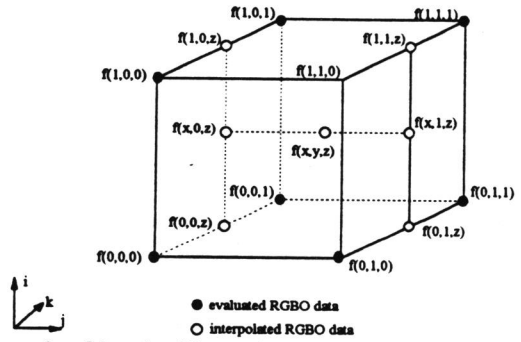

Figure 4 -- Voxel trilinear interpolation in RGBO space

In Levoy's approach, 7 linear interpolations are necessary for each optical property (R, G, B and O). Then for each RGBO voxel evaluation, 28 linear interpolations are needed.

3. A high performance direct volume rendering pipeline

This proposal aims at reducing the mentioned disadvantages of Levoy's approach.

Its main idea is the resampling anticipation (figure 5) which is applied over the original volume data. This reorganization suppresses the necessity of the four resamplings (R, G, B and O) and the intermediary RGBO volume that Levoy's approach need.

In this case, the resampling is done over the original data volume. The opacity classification, color classification, and shading steps are equal to Levoy's approach, but they are only applied on resampled voxels.

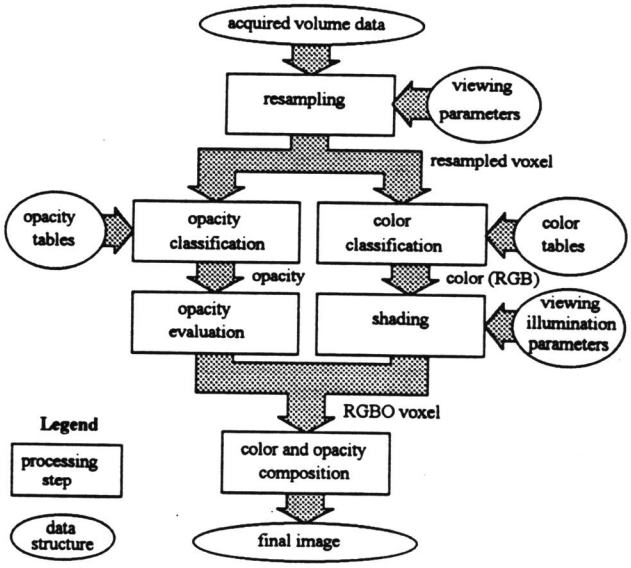

Figure 5 -- Proposed pipeline

3.1. Interpolation and gradient evaluation

This section discusses a method to calculate both the interpolated voxel and the gradient vector.

The interpolation can be done using the same strategy adopted in Levoy's approach. The difference here is that the interpolated data is a resampled voxel of the original volume data instead of a RGBO voxel of the RGBO volume.

The gradient evaluation is a key point of this proposal. While in Levoy's approach the gradient was calculated for each voxel of the original volume data, now it is evaluated directly on the interpolated sample (figure 6).

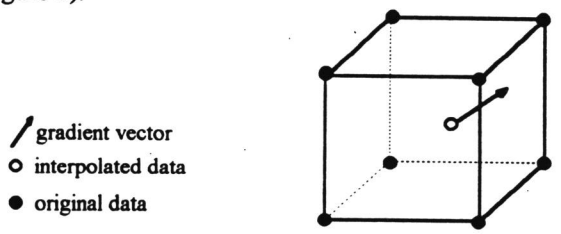

Figure 6 -- Interpolation and Gradient evaluation in the proposed approach

An efficient and accurate way to evaluate the gradient is suggested. The gradient operator, when applied to *equation 4*, yields:

$$\frac{\partial}{\partial x} f(x,y,z) = A + D \cdot y + E \cdot z + G \cdot y \cdot z$$
$$\frac{\partial}{\partial y} f(x,y,z) = B + D \cdot x + F \cdot z + G \cdot x \cdot z \qquad \textit{(eq. 5)}$$
$$\frac{\partial}{\partial z} f(x,y,z) = C + E \cdot x + F \cdot y + G \cdot x \cdot y$$

It is straightforward to prove that:

$$\frac{\partial}{\partial x} f(x,y,z) = f(1,y,z) - f(0,y,z)$$
$$\frac{\partial}{\partial y} f(x,y,z) = f(x,1,z) - f(x,0,z) \qquad \textit{(eq.6)}$$
$$\frac{\partial}{\partial z} f(x,y,z) = f(x,y,1) - f(x,y,0)$$

The above equations determine the gradient by subtracting the projected interpolations in each axis. This allows an efficient gradient calculation as show in figure 7.

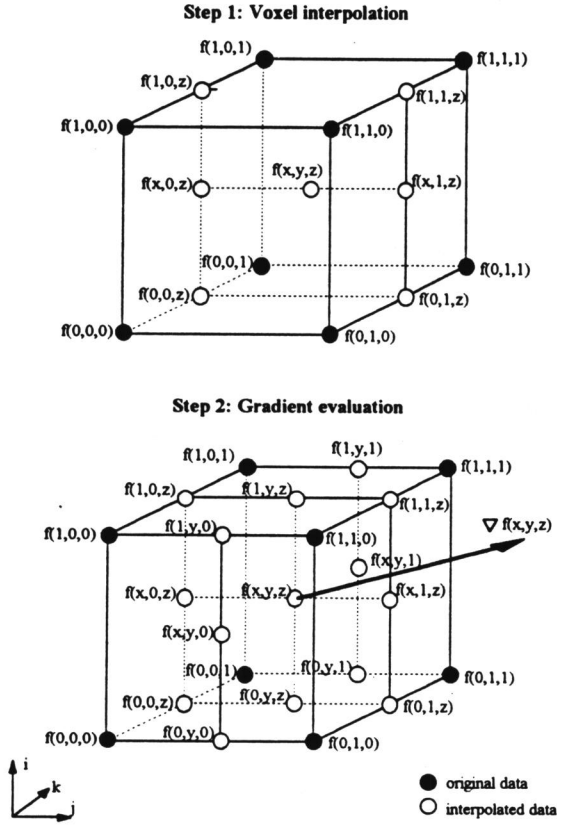

Figure 7 -- Trilinear voxel interpolation and gradient evaluation in the original data space

In this case, 15 linear interpolations are necessary to calculate the interpolated voxel and its gradient vector. Comparing with Levoy's approach, we need here nearly 50% less linear interpolations.

3.2. Optimizations

A simple and efficient optimization is done by testing if the classified opacity of the voxel is zero or smaller than a pre-defined threshold. In affirmative case, it is not necessary to do the rest of the interpolations for gradient evaluation, shading and color composition.

Also, all the other optimizations proposed in the literature [11, 12, 13, 14] such as adaptative refinement, hierarchical spatial enumeration, and early termination, can be applied to the proposed pipeline.

4. Current implementation

Both pipelines were implemented and incorporated to RTV visualization package [27] for experimental evaluation.

RTV is a volume visualization package that has tools for segmentation, classification and visualization of medical images. It was implemented using "C" language on the following platforms: SUN, APOLLO, SGI and Kendall Square.

RTV has being developed by a joint effort between the Laboratório de Sistemas Integráveis - Escola Politécnica - Universidade de São Paulo (Brazil) and the Computer Graphics Unit - Manchester University (U.K.).

Figure 8 shows the data and control flows of a sequential implementation of Levoy's pipeline.

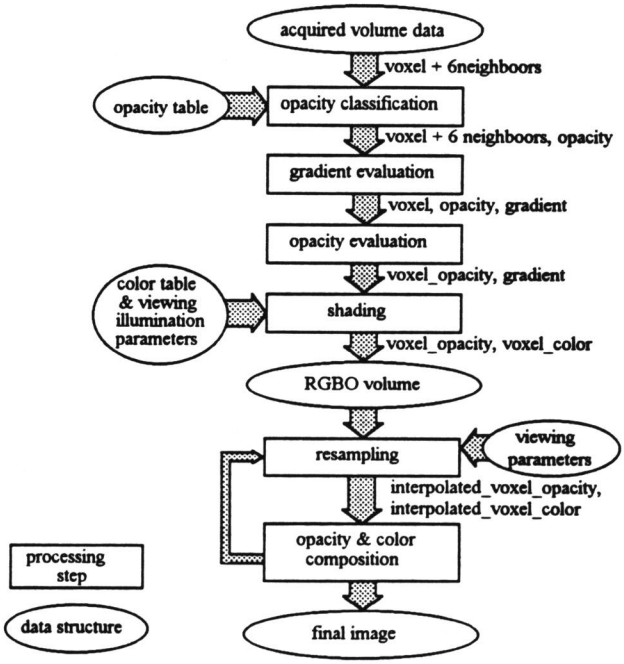

Figure 8 -- Sequential implementation of Levoy's pipeline (data and control flows)

Figure 9 shows the implementation of the proposed pipeline.

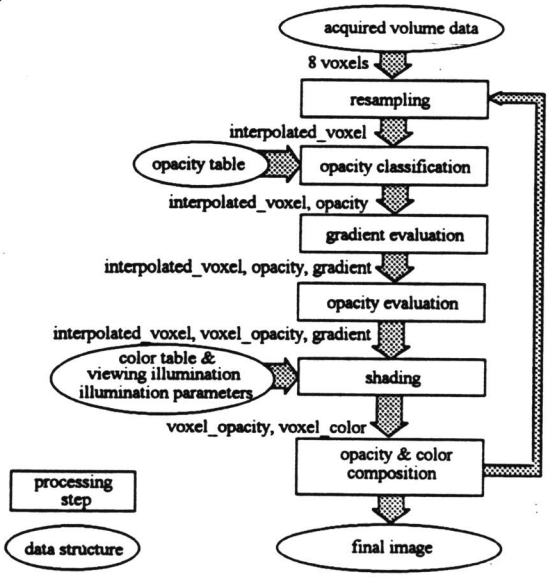

Figure 9 -- Sequential implementation of the proposed pipeline (data and control flows)

The optimization proposed in 3.2 item was done in both implementations.

5. Performance evaluation

In this section, analytical and experimental evaluations are done of Levoy's and the proposed pipelines.

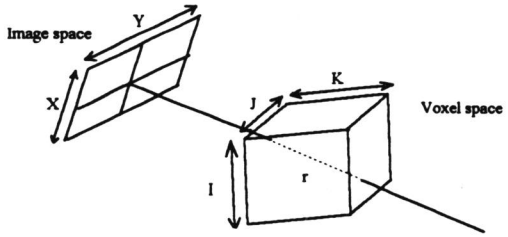

Figure 10 -- Generic scene

Considering a scene such as shown in figure 10, it is possible to do some estimates using the following notations:

T_{Levoy} = *execution time for Levoy's pipeline;*
$T_{proposed}$ = *execution time for proposed pipeline;*
I, J, K = *volume data dimensions;*
X, Y = *image dimensions;*
SR = *sampling rate;*
$\bar{r}$ = *ray average length;*
t_o = *time for opacity evaluation (includes classification);*
t_{grad} = *time for gradient evaluation;*
t_{sh} = *time for shading (includes classification);*
$t_{i(rgbo)}$ = *time for one interpolation (for R, G, B and O);*
t_{comp} = *time for opacity and color composition;*
$t_{i(v+g)}$ = *time for one voxel interpolation and gradient evaluation.*

In the case of Levoy's pipeline, considering a sequential implementation without optimizations, the execution time can be estimated by the following equation:

$$T_{Levoy} = I \cdot J \cdot K \cdot \left(t_{grad} + t_o + t_{sh}\right) + X \cdot Y \cdot \frac{1}{SR} \cdot \bar{r} \cdot \left(t_{i(rgbo)} + t_{comp}\right) \quad (eq.7)$$

Therefore, the execution time is the sum of a constant time (to generate the RGBO volume) plus a variable time (proportional to the final image resolution). Other parameters have influence on this variable time too, such as the average ray length that depends on the observer's position and the opacities evaluated through the ray.

In the case of the proposed pipeline, the execution time can be estimated by the equation:

$$T_{proposed} = X \cdot Y \cdot \frac{1}{SR} \cdot \bar{r} \cdot \left(t_{i(v+g)} + t_o + t_{sh} + t_{comp}\right) \quad (eq.\ 8)$$

For the experimental evaluation, three time measurements have a special importance: T_1, T_2 and T_3, defined as follows.

The time for generating RGBO volume, T_1, is given by the equation:

$$T_1 = I \cdot J \cdot K \cdot \left(t_{grad} + t_o + t_{sh}\right) \quad (eq.\ 9)$$

The time for resampling a voxel (using the trilinear interpolation method) in RGBO volume plus the time for composition, T_2, is given by:

$$T_2 = t_{i(rgbo)} + tcomp \quad (eq.\ 10)$$

The time for resampling a voxel from the acquired volume data (using trilinear interpolation method) and one voxel gradient calculation plus the time for opacity and color classification and shading and the time for composition, T_3, is given by:

$$T_3 = t_{i(v+g)} + t_o + t_{sh} + t_{comp} \quad (eq.\ 11)$$

The first advantage of the proposed pipeline is the elimination of the constant time T_1, used for the generation of the intermediary RGBO volume. Resulting only in a time that is proportional to the final image resolution.

In both cases, the execution time is proportional to the resolution and to the average ray length. The average ray length can be assumed constant for a given image proportion, viewer point and sampling rate.

The comparison of T_2 and T_3, leads to three possible situations:

- case 1: $T_2 < T_3$, the execution time is better only for resolutions lower than the equal performance point resolution (figure 11);
- case 2: $T_2 = T_3$, the execution time is better for any resolution, and the execution time differs by a constant T_1 (figure 12);
- case 3: $T_2 > T_3$, in this case, for any resolution the execution time is better (figure 13).

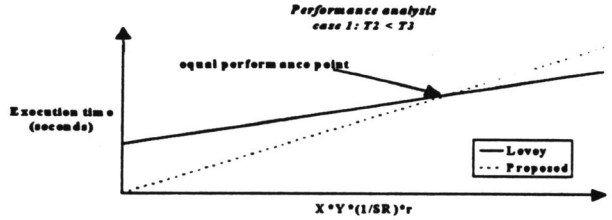

Figure 11 -- Performance analysis for $T_2 < T_3$

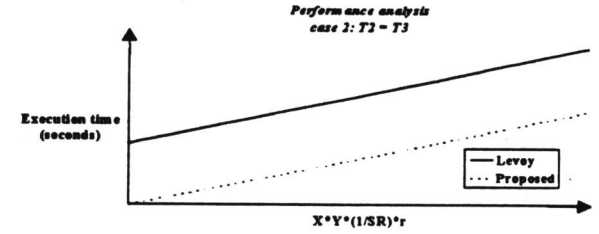

Figure 12 -- Performance analysis for $T_2 = T_3$

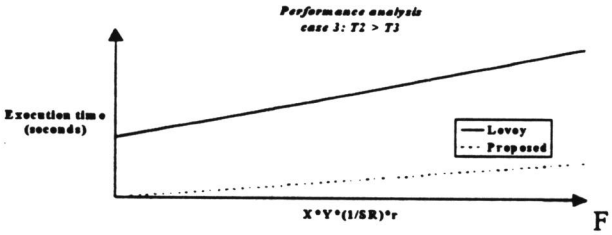

Figure 13 -- Performance analysis for $T_2 > T_3$

X pixels	Y pixels	$X.Y.\ (1/SR).10^3$	T_{Levoy} (seconds)	T_{New} (seconds)
0	0	0	22	1
141	142	100	34	10
200	200	200	41	18
245	245	300	51	27
283	283	400	60	36
316	317	500	70	45
346	347	600	79	54
374	374	700	89	62
400	400	800	100	71
424	424	900	108	80
447	447	1000	118	89
469	469	1100	129	98
490	490	1200	137	106
510	510	1300	148	115
529	529	1400	156	125
547	548	1500	166	133

Table 1 -- Performance results for SR = 0.2

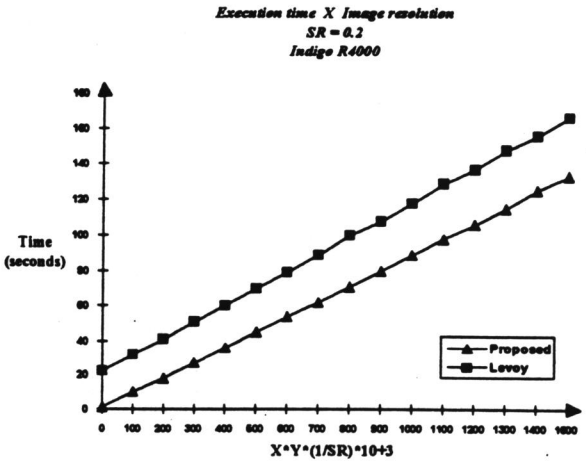

Figure 14 -- Performance evaluation for SR = 0.2

The experimental evaluation was done in a SGI Indigo R4000 [9] graphics workstation with 64 Mbytes of memory using the "C" compiler supplied by Silicon Graphics. The image generated in the evaluation is shown in figure 16.

The experimental results obtained for Sampling Rate (SR) = 0.2 are presented in table 1 and the experimental results obtained for Sampling Rate (SR) = 0.8 are presented in table 2.

X pixels	Y pixels	$X.Y.\ (1/SR).10^3$	T_{Levoy} (seconds)	T_{New} (seconds)
0	0	0	23	1
282	283	100	44	18
400	400	200	64	35
489	490	300	85	50
560	560	400	107	66
632	633	500	126	84
692	693	600	147	98
748	749	700	170	115
800	800	800	190	131
848	849	900	211	147
894	895	1000	231	163
938	939	1100	251	180
979	980	1200	274	197
1019	1020	1300	292	211
1058	1059	1400	312	228
1095	1096	1500	334	246

Table 2 -- Performance results for SR = 0.8

Figures 14 and 15 show that for this implementation the proposed pipeline had experimentally a better performance than Levoy's approach.

Comparing the experimental results with the analytical model, it can be observed that it was reached case 3, $T_2 > T_3$.

Comparing the results showed in figures 14 and 15, it can be observed a performance dependence from the

Sampling Rate parameter. It is easy to demonstrate that more opacity and shading calculations are needed when the sampling rate is smaller. Then the choose of the right sampling rate is very important for faster image generation.

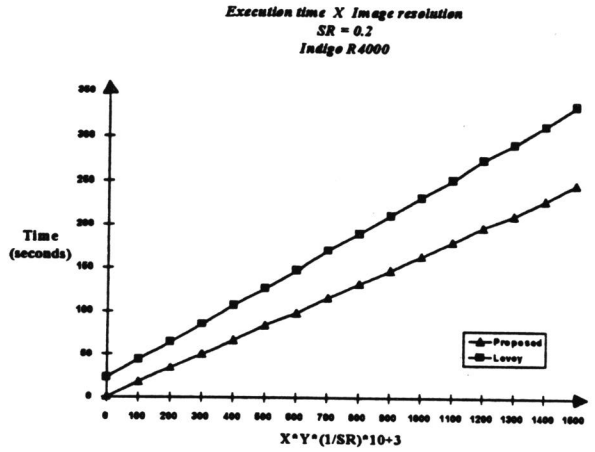

Figure 15 -- Performance evaluation for SR = 0.8

The analytical model presented can be used for a more accurate evaluation of each step of both pipelines.

6. Summary and conclusion

This paper presented a high performance direct volume rendering pipeline.

Besides its advantage of saving memory and execution time, the proposed pipeline could provide better shading quality (due to its more accurate gradient approximation), resulting in better final images.

The elimination of the intermediary volume RGBO has considerable importance for real time visualization. The use of one byte for each optical component (R, G, B and O) in 256x256x256 volume data, for example, would require 64 Mbytes.

Also, this approach is very well suited to be implemented as a volume rendering dedicated circuit in VLSI.

The performance of this proposal is tightly dependent on the implementation of the illumination equation and on the gradient evaluation (used for opacity calculation and for shading). Faster methods for this can be found in [27].

Some directions on future work are: a more careful study of the influence of the new gradient evaluation method over shading and final image quality; an implementation of this pipeline in a high performance parallel computer; and, finally a VLSI chip design.

7. Acknowledgments

We would like to thank Prof. Eric Hoffmann from Iowa University, for the courtesy of allowing us to use the volume data of a human head.

We would like to thank all people from Laboratório de Sistemas Integráveis, for the many suggestions and revisions done to this paper.

8. References

[1] ARVO, J. "Graphics gems", vol. 2, Academic Press, 1992.

[2] BADOUEL, D. *et al.* "Ray tracing on distributed memory parallel computers: strategies for distributing computations and data", course notes, SIGGRAPH'90, vol. 28, August 1990.

[3] DEW, P.M.; EARNSHAW, R.A. & HEYWOOD, T.R. "Parallel processing for computer vision and display", Addison-Wesley Publishing Company, 1989.

[4] DREBIN, R.A. *et al.* "Volume rendering", Computer Graphics, vol. 22, n. 4, August 1988.

[5] ELVINS, T. T. "A survey of algorithms for volume visualization", Computer Graphics, vol. 26, n. 3, August 1992.

[6] FUCHS, H. *et al.* "Interactive visualization of 3D medical data", IEEE Computer, vol., n. 8, August 1989.

[7] GRANT, A. & ZUFFO, M. K. "Approaches to direct volume rendering on distributed memory machines", Workshop on Parallel Processing for Scientific Visualization, University of Edinburgh, May 1993.

[8] HOHNE, K.H. *et al.* "3D visualization of tomographic data using the generalized voxel model", The Visual Computer, n. 6, 1990.

[9] KANE, G. & Hennrich, J. "MIPS RISC archicteture", Prentice Hall, 1992.

[10] KAPLAN, W. "Advanced calculus", Addison-Wesley Publishing Co., Inc., 1959/71.

[11] LEVOY, M. "Display of surfaces from volume data", IEEE Computer Graphics and Applications, vol. 8, n. 3, May 1988.

[12] LEVOY, M. "Display of surfaces from volume data", Ph.D. Dissertation, University of North Carolina at Chapel Hill, 1989.

[13] LEVOY, M. "Volume rendering by adaptative refinement", The Visual Computer, vol. 6, n. 1, February 1990.

[14] LEVOY, M. "Efficient ray tracing of volume data", ACM Transactions on Graphics, vol. 9, n.3, July 1990.

[15] MONTANI, C.; PEREGO, R. & SCOPIGNO, R. "Parallel volume visualization on a hipercube architecture", Workshop on volume Visualization, Boston, MA, 1992.

[16] NIEH, J. & LEVOY, M. "Volume rendering on scaleable shared memory mimd archictetures", Workshop on Volume Visualization, Boston, MA, 1992.

[17] NIELSEN, G. & HAMANN, B. "Techniques for the interactive visualization of volumetric data", 1st. IEEE Conference on Visualization, Visualization'90, San Francisco, CA, USA, October 1990.

[18] PADDON, D.J. & GREEN, S.A. "Parallel ray-tracing", Dept. of Computer Science, University of Bristol, 1989.

[19] PORTER, T. & DUFF, T. "Composing digital images", Computer Graphics, vol. 18, n. 3, July 1984.

[20] SCHRÖDER, P. & SALEM, B. J. "Fast rotation of volume data on data parallel machines", Proc. Visualization, 1991.

[21] UPSON, C. & KEELER, M. "VBUFFER: visible volume rendering", Computer Graphics, vol. 22, n. 4, August 1988.

[22] WEBBER, Robert E. "Ray tracing voxel data via biquadratic local surface interpolation". The Visual Computer, n. 6, 1990.

[23] WESTOVER, L. "Footprint evaluation for volume rendering", Computer Graphics, vol. 24, n. 4, August 1990.

[24] WHITMANN, Scott. "Parallel algorithms and architectures for 3D image generation", course notes, vol. 28, Dallas, USA, ACM SIGGRAPH.

[25] WILHELMS, J. "Decisions on volume rendering", in State of Art in Volume Visualization, course notes, SIGGRAPH'91, n. 8, ACM Press, August 1991.

[26] YOO, T. et alii "Direct visualization from volume data", IEEE Computer Graphics & Applications, July 1992.

[27] ZUFFO, M. K. & GRANT, A. & LOPES, R. D. "RTV -- a tridimensional visualization package for medicine" (in Portuguese), IV Brazilian Conference in Computer Graphics and Image Processing (IV Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, IV SIBGRAPI), Recife, Brazil, 1993.

Figure 16 -- Image generated with the proposed pipeline which was used for performance evaluation

Image credits: Courtesy of Prof. Eric Hoffman, Department of Radiology, College of Medicine, Iowa University. This a MRI volume from a human head with 128x128x60 voxels.