R. Mehaboobathunnisa*, A.A. Haseena Thasneem and M. Mohamed Sathik

Fuzzy Mutual Information-Based Intraslice Grouped Ray Casting

<https://doi.org/10.1515/jisys-2016-0263>

Received October 24, 2016; previously published online November 9, 2017.

Abstract: The traditional ray casting algorithm has the capability to render three-dimensional volume data in the viewable two-dimensional form by sampling the color data along the rays. The speed of the technique relies on the computation incurred by the huge volume of rays. The objective of the paper is to reduce the computations made over the rays by eventually reducing the number of samples being processed throughout the volume data. The proposed algorithm incorporates the grouping strategy based on fuzzy mutual information (FMI) over a group of voxels in the conventional ray casting to achieve the reduction. For the data group, with FMI in a desirable range, a single primary ray is cast into the group as a whole. As data are grouped before casting rays, the proposed algorithm reduces the interpolation calculation and thereby runs with lesser complexity, preserving the image quality.

Keywords: Direct volume rendering, fuzzy mutual information, intra-slice data grouping, ray casting.

1 Introduction

Imaging of medical data has become essential for an efficient diagnosis and treatment of diseases. Prior analysis of infections or abnormalities gives a better idea of the disease of the patients, and therefore physicians can be more accurate. Medical image processing has been a vast and very much focused area of research during the past decades. Researchers pay attention to segmentation, enhancement, and visualization of the medical image data obtained by various modalities like computed tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET). The CT/MRI/PET slices provide information only in the corresponding axes location. In order to improvise the visualization, the slices from the modalities are rendered together in three-dimensional (3D) form. The volume rendering algorithms are broadly classified into two categories, namely the indirect volume rendering (IVR) and the direct volume rendering (DVR) methods. This categorization is made out of the difference in the way the methods process the input 2D data.

The IVR algorithms generate an intermediate surface from the actual input; it is then transformed to a 3D image. Marching cube is the most common and widely used IVR algorithm. The DVR algorithm works through the input directly without any intermediate surfaces and renders the volume. Examples of DVR techniques are ray casting and shear warp. The basic concept to render the volume data by casting rays into it was proposed by Levoy in his paper [7]. The rendering pipeline of a volume follows the sequence as shown in Figure 1.

*Corresponding author: R. Mehaboobathunnisa, Sadakathullah Appa College, Tirunelveli, India, e-mail: nisadiq8797@gmail.com
A.A. Haseena Thasneem and M. Mohamed Sathik: Sadakathullah Appa College, Tirunelveli, India

Figure 1: Rendering Pipeline of Ray Casting.

The ray casting algorithm is a direct volume rendering technique that casts rays directly into the volume, along which samples are taken and the interpolated sampling points are composited into a single color value using Eqs. (1) and (2):

$$C_{\text{out}} = C_{\text{now}}(1 - \alpha_{\text{in}}) + C_{\text{in}}, \quad (1)$$

$$\alpha_{\text{out}} = \alpha_{\text{now}}(1 - \alpha_{\text{in}}) + \alpha_{\text{in}}, \quad (2)$$

where C_{out} is the output color, C_{in} is the accumulated color, C_{now} is the current voxel's color, and α is the opacity factor. It is this single value that represents all the corresponding voxels of the volume data in the image plane, as shown in Figure 2.

Figure 2: Rays Traversing a Set of Voxels Along Its Path.

The rendering procedure depicted in Figure 1 broadly comes under two main levels:

1. Data preparation level – sampling and interpolation
2. Appearance generation level – gradient estimation, classification, shading, composition

Each level incurs its own complexity. The color of a location in the resultant output image is determined by re-sampling the samples taken at regular intervals. The unknown samples are computed using tri-linear interpolation, which performs 17 additions and 16 multiplications [8] for a single computation. Thus, the total complexity of the rendering algorithm is directly proportional to the total number of interpolations made. The motivation of this paper is to bring down the rendering complexity of the ray casting algorithm by grouping the rays originating from relevant neighboring voxels and then to cast them. This way of optimizing the ray casting technique avoids interpolation of similar unknown voxels and also does not compromise the rendering quality. In addition to the other optimizations that also aim for the same as discussed in the following section, this work provides a different aspect in casting the rays, achieving the reduction in the complexity. The organization of the paper is as follows. Section 1 is the introduction. Section 2 reviews the works that have been proposed for the improvement of the traditional ray casting and those works that have used fuzzy mutual information (FMI). Section 3 reveals the proposed improvements in ray casting. Section 4 discusses the results, and Section 5 is the conclusion.

2 Related Works

The quality of the conventional ray casting algorithm has been improved in several ways, which include the improvisation of rendering speed, minimizing the artifacts, and improving the quality of rendering. The objective of the paper is to come out with a faster ray casting. Hence, we limit our survey with works of the same concern. Empty space leaping is done in Ref. [10] by separating the target object from the background using an axis aligned bounding box. Hence, whenever a ray is cast into the volume, a check is made to confirm if it passes through the bounding box and interpolation is carried out only for those samples lying within it. This scheme uses a fast look-up algorithm to identify non-empty cubes and thus avoids blind assignment on the optical properties with a reduction in the computation time. The volume data is cut using the bounding box and is compressed using the octree data structure in Ref. [1]. The object space obtained by cutting the volume data forms the root of the linear octree. The eight children nodes are assigned one of the following states – (i) all voxels having same color (full, F), (ii) some voxels having the same color (partial, P), and (iii) empty nodes (empty, E). This way of state assignment helps not only to skip empty voxels but also to avoid duplication in sampling, i.e. voxels in F state are also skipped. This reduces the ray composition time by three times. Kreeger et al. in their work [4] improved the rendering speed of the conventional ray casting by adaptively increasing the distance of sampling and the number of rays. It begins with a single ray, and the rays get split at each level of sampling and at a larger sampling distance. They used effective filters on their input to improve the quality of the rendered image. The scheme in Ref. [8] performs sampling in three steps – determination of intersection points and the corresponding plane, determination of optical property of all intersection points by linear interpolation, and finally determination of optical property of intersection points between adjacent intersection points using definite proportion and separated points. This reduced one-third of the total computation. The work in Ref. [13] separated the foreground voxels from the background and incorporated the space leaping technique to render the foreground. It also used a multi-factor in the place of simple transfer functions to determine the opacity. This improvement had a tradeoff between speed and image quality. The work described in Ref. [9] fragmented the rays as per their vertical extent in the volume. This avoids unnecessary interpolation in homogeneous regions and modifies the fundamental color and opacity, calculating equations by including an opacity summation term that adds the opacity of all the samples within a fragment. Another accelerated ray casting algorithm proposed in Ref. [12] skips the voxels not contributing to the rendering. These voxels are referred to as the empty voxels. Whenever an empty voxel is encountered, the algorithm makes a skip by an incrementing step value until a non-empty voxel is reached.

Figure 4: Rays Cast into Groups – An Illustration.

Prior to casting a single ray onto a group, the relevance among the group members is measured to confirm if it is reasonable to limit with that single ray. This test is performed by determining the FMI, which is the sum of the FMIs of all available classes, calculated as in Eqs. (3) to (6):

$$\text{FMI} = \sum \text{FMI}_c, \quad (3)$$

$$\text{FMI}_c = -D \log D, \quad (4)$$

$$D = \frac{\sum_{x \in c} u_c(x)}{\sum_{x \in c} u_c(x)}, \quad (5)$$

$$u_c = \sum_{j=1}^c \left(\frac{\text{dist}(x - v_i)}{\text{dist}(x - v_j)} \right)^{-2/m-1} \quad \text{for all } i; m > 1, \quad (6)$$

where u_c is the membership of each variable x to a cluster c , D is the actual match degree of the data to the cluster c , and FMI_c is the dependence value among the members of cluster c . An optimum fuzzy threshold is fixed and the calculated FMI value is compared with the threshold. A single primary ray is hit to a group of $(m \times m)$ voxels if the computed FMI is greater than the threshold and one voxel from the group serves as the reference. On the other case, when the FMI lies within the optimum threshold, the voxels are treated as individuals and rays are cast to each of them, thus splitting the primary ray into $(m \times m)$ number of rays. This concept is illustrated in Figure 4. The figure shows three corresponding voxel groups from the consecutive slices. A 3×3 group from the first slice is assumed to have FMI within the threshold, and hence a single ray is cast to it. The reference voxel of the set is shown as a solid cube. The corresponding set from the second slice has a greater FMI, and so limiting to cast a single ray from it has the risk of information loss. Therefore, rays are shot to all of its voxels independently. This process extends throughout the volume.

3.1 FMI Mask

FMI-based data grouping acquires the use of an additional data structure – an FMI mask as shown in Figure 5. The FMI mask stores the details of whether the voxels are grouped or not; in the case of maintaining the

Figure 5: FMI Mask in Real and Virtual Slices – An Illustration.

groups, this mask holds the information about the reference voxel. A look-up table is created to save the interpolated values of the virtual voxels. The interpolated value of the reference voxels is used for the members of its group.

Whenever a voxel group is begun to be processed, its FMI mask is looked up to get its reference voxel, which is identified as the voxel with mask value 1. In general, the top-left-corner voxel is registered as the reference voxel, as in Figure 5. Hence, it is sufficient to interpolate only the reference voxel, and that value is assigned as the data value for its group members. This way of grouping avoids performing interpolation for closely related and relevant samples, and so reduces the computational complexity. In short, as rays proceed along the data, all samples are traversed but not all are processed.

3.2 Pseudocode

Input: 2D CT/MRI slices, sampling distance SD, direction vector DV, fuzzy threshold FT, group size ($GR \times GC$)
Output: 2D resultant image with 3D details
Steps:
for each slice
Initialize groups ($GR \times GC$), $group_1$ to $group_n$, $processed_flag$ { $p_flag = 0$ for virtual slices; $p_flag = 1$ for real slices}
for each group_i
 Compute FMI
 if FMI is greater than FT //relevant voxels
 for $[i,j]=[GR,GC]$
 fuzzy_mask($group_i$)[1,1] = 1
 fuzzy_mask($group_i$)[i,j] = 0
 else //independent voxels
 for $[i,j] = [GR,GC]$
 fuzzy_mask($group_i$)[i,j] = 1
 Cast rays into the volume
 If fuzzy_mask(sample) = 1
 Tri-linearly interpolate the sample
 Else
 if $p_flag = 0$
 Look up for the corresponding reference voxel
 else
 Retrieve the value
 Compute the accumulated intensity

4 Experimental Results

The ray casting and the proposed ray casting techniques are executed with two different data sets (a $150 \times 150 \times 52$ CT head volume and a $150 \times 150 \times 100$ tibia fibula CT) in a system with 4-GB RAM and 4-GHz processor, and the results are shown below (Figures 6 and 7). The results are registered for varied sampling distance values and the rendering time for multiple views using classical ray casting and the proposed optimization, and are tabulated in Tables 1–4.

The measures used in the analysis are the total number of interpolated samples, data computation time (DCT), and multi-view rendering time (MVRT). MVRT is determined as in Eq. (7):

Figure 6: CT Head Dataset.

Rendered using (A) ray casting and (B) proposed approach at sampling distance = 0.5, viewpoint = 90° .

Figure 7: CT Tibia Fibula Dataset.

Rendered using (A) ray casting and (B) proposed approach at sampling distance = 0.5, viewpoint = 45°.

Table 1: Analysis of Ray Casting vs. Proposed Approach for CT Head Dataset.

Method	Data orientation	Viewpoint	Samp. Dist.	Total no. of samples (×10 ⁵)	Interpolated samples (×10 ⁵)	DCT (s)
Ray casting	90°	90°	0.5	23.175	11.475	6.972
			0.25	46.125	34.425	20.674
			0.2	57.6	45.9	27.562
			0.1	114.975	103.275	48.3635
			0.5	23.175	11.475	6.8335
Proposed approach	45°	90°	0.5	16.19276	8.0155	4.8299
	90°	45°	0.5	23.175	6.36748	3.0505
	90°	90°	0.5	46.125	18.62507	9.9702
			0.2	57.6	22.79684	12.5471
			0.1	114.975	43.5932	25.6046
			0.5	23.175	7.37596	4.6525
	90°	45°	0.5	16.19276	4.21677	3.0995

Table 2: MVRT for CT Head Dataset Using Ray Casting and Proposed Approach.

Method	GT (s)	Σ DCT (s)	MVRT = GT + Σ DCT (s)
Ray casting	0	115.2349	115.2349
Proposed approach	25.5676	58.9244	84.4914

$$\text{MVRT} = \text{GT} + \sum \text{DCT}, \quad (7)$$

where GT is the time taken to perform the grouping of data throughout the volume. This time includes the time to check the data for relevance and implement the grouping. In general, the computational complexity of classical ray casting is given as in Eq. (8):

$$\text{CC} = V\delta, \quad (8)$$

where V is the total number of virtual samples traversed for rendering and δ is the perspective weight ($0 < \delta \leq 1$) to signify the variation in the sample count when the viewpoint varies. It is the variable V that influences the complexity of the algorithm. The value of V for ray casting and the proposed approach is given in Eqs. (9) and (10), respectively:

Table 3: Analysis of Ray Casting vs. Proposed Approach for CT Tibia Fibula Dataset.

Method	Data orientation	Viewpoint	Samp. Dist.	Total no. of samples ($\times 10^5$)	Interpolated samples ($\times 10^5$)	DCT (s)
Ray casting	90°	90°	0.5	44.775	22.275	13.7004
			0.25	89.325	66.825	40.5139
			0.2	111.6	89.1	55.625
			0.1	222.975	150.75	92.3033
			0.5	44.775	22.275	13.6688
Proposed approach	45°	90°	0.5	21.741	10.8075	6.6899
	90°	45°	0.5	44.775	10.5962	6.5113
	90°	90°	0.5	89.325	21.13915	15.9597
	45°	90°	0.2	111.6	26.4106	17.8641
			0.1	222.975	52.76801	28.32
			0.5	44.775	10.5962	6.5062
			0.5	21.741	5.1451	3.1014

Table 4: MVRT for CT Tibia Fibula Dataset using Ray Casting and Proposed Approach.

Method	GT (s)	$\sum$ DCT (s)	MVRT = GT + $\sum$ DCT (s)
Ray casting	0	222.5013	222.5013
Proposed approach	64.7543	78.2627	143.017

$$V_{\text{Ray casting}} = \sum_{i=1}^P v_i, \quad (9)$$

$$V_{\text{Proposed}} = \sum_{i=1}^P v_i(I), \quad (10)$$

where v_i is the number of virtual voxels in each plane i and $v_i(I)$ is the number of independent virtual voxels, i.e. ungrouped virtual voxels in each plane i .

5 Conclusion

The ray casting algorithm incorporated with grouping based on FMI reduces the computations incurred by interpolation. It works under the assumption that medical data do not vary much in successive voxels and in successive slices, and hence grouping becomes reasonable. As the work presented in the paper interpolates only one voxel on behalf of all the members of a group, the number of interpolation calculations gets significantly reduced, and hence the computational complexity as well. Moreover, as the check for relevance among the group members is performed on every subsequent sample locations, it is assured that variations in data are also detected, if any. The sampling distance can be selected as per user requirement. A small sampling distance value generates more accurate and smoother result than the higher values.

The huge fall in the interpolation count is not reflected in the rendering time because of the time incurred by the grouping strategy. In spite of the advantage of reduction in the overall rendering time for multiple views, the result generated by this algorithm shows some block artifacts as a result of grouping, and hence the work is to be extended to reduce the block artifacts and to produce smoother results.

Bibliography

- [1] F. Gong and H. Wang, An accelerative ray casting algorithm based on crossing-area technique, in: *International Conference on Machine Vision and Human-Machine Interface*, China, 2010.
- [2] J. Grande, M. d. R. Suarez and J. R. Villar, A feature selection method using a fuzzy mutual information measure, in: *Innovations in Hybrid Information Systems*, Springer, Berlin, 2007.
- [3] B. Kasko, Fuzzy entropy and conditioning, *Inf. Sci.* **40** (1986), 165–174.
- [4] K. Kreeger, I. Bitter, F. Dachille, B. Chen and A. Kaufman, Adaptive perspective ray casting, in: *Symposium on Volume Visualization*, 1998.
- [5] R. Kushaba, A. Al-Jumeliy and A. Al-Ani, Novel feature extraction method based on fuzzy entropy and wavelet packet transform for myoelectric control, in: *International Symposium on Communications and Information Technologies (ISCIT 2007)*, 2007.
- [6] B. Lee, J. Yun and J. Seo, Fast high quality ray casting with virtual samplings, *IEEE Trans. Vis. Comput. Graph.* **16** (2010), 1525–1532.
- [7] M. Levoy, Display of surfaces from volume data, *IEEE Comput. Graph. Appl.* **8** (1988), 29–37.
- [8] L. Lin, S. Chen, Y. Shao and Z. Gu, Plane-based sampling for ray casting algorithm in sequential medical images, *Comput. Math. Methods Med.* **2013** (2013).
- [9] T. Ling and Q. Zhi-Yu, An improved fast ray casting volume rendering algorithm of medical images, in: *4th International Conference on Biomedical Engineering and Informatics*, IEEE, China, 2011.
- [10] D. Qing, J. Chen and Z. Wang, An improved ray-casting algorithm based on AABB, in: *IEEE International Conference on Audio Language and Image Processing*, China, 2010.
- [11] L. Sánchez, R. Suárez, J. R. Villar and I. Couso, *Some Results About Mutual Information Based Feature Selection and Fuzzy Discretization of Vague Data*, IEEE, London, 2007.
- [12] Y. Shicai, W. Qianjun and L. Rong, *A High Efficient and Speed Algorithm of Ray Casting in Volume Rendering*, IEEE, China, 2011.
- [13] Y. Tian, M.-Q. Zhou and Z.-K. Wu, A rendering algorithm based on ray-casting for medical images, in: *Proceedings of International Conference on Machine Learning and Cybernetics*, China, 2008.
- [14] Y. S. Tsai, U.-C. Yang, I.-F. Chung and C.-D. Huang, A comparison of mutual and fuzzy-mutual information-based feature selection strategies, in: *International Conference on Fuzzy Systems*, IEEE, India, July 2013.