THE 11th INTERNATIONAL SYMPOSIUM ON ADVANCED TOPICS IN ELECTRICAL ENGINEERING
March 28-30, 2019
Bucharest, Romania

A WEB Platform for Rendring and Viewing MRI Volumes using Real-Time Raytracing Principles

Felix-Constantin Adochiei[1], Radu-Ion Ciucu[1], Ioana-Raluca Adochiei[2], Sorin Dan Grigorescu[1], George Călin Serițan[1], Miron Casian[3]
[1]University Politehnica of Bucharest, Romania
[2]Military Technical Academy "Ferdinand I", Bucharest, Romania
[3]"Gheorghe Asachi" Technical University of Iasi, Romania
felix.adochiei@upb.ro, radu.ciucu@gmail.com, edu_ioana_raluca@yahoo.com, sorin.grigorescu@upb.ro, george.sertian@upb.ro, casian_miron@yahoo.com,

***Abstract*: In this paper, we present a web platform for volume rendering of MRI data. Our platform are designed for mobile and desktop devices and VR applications. Unlike a classical desktop application which is platform dependent, we developed a series of software services in order to optimize and deliver fast and efficient volumetric data. The proposed platform is the summary of our initial efforts to implement raytracing algorithms in practical situations. We consider our platform and implementation a viable solution for volumetric rendering of MRI data in web environments.**

***Keywords*: Raytracing, MRI, DVR, Volume, CT**

I. Introduction

Direct volume rendering (DVR) is a computer graphics technique used to process and calculate the density and visual aspects of sparse volumes and voxel domains. In the medical field, it is used to 3D reconstruct magnetic resonance imaging (MRI) and computed tomography (CT) data in order to emphasize the shape, depth and features of anatomical structures. Furthermore by employing image processing techniques specific patterns can be highlighted in a three-dimensional overview. Recent advances in GPU hardware have made possible the acceleration of raytracing algorithms in order to achieve a real-time visualization of imaging data such as CT and MRI [1-3].

Traditional ways of interacting with a volume renderer are by adjusting optical characteristics of a given rendered scene trough a mathematical transfer function that affects the color space and the opacity [4, 5].

A modern rendering pipeline must contain tools that provide an outline for each volumetric feature inside MRI data, as well as an intuitive approach towards researching and discovering new details and structures. Methods such as ray casting, segmentation and texture-based interpolation tend to have a limited range of support for the objectives presented in our paper [6-8].

Our paper presents a web platform for real-time visualization of MRI volumes focused on fast interaction and web virtual reality (WebVR) integration. Volumetric rendering solutions started to gain traction in the scientific and medical industry in the past 10 years [18-20]. Most platforms built are desktop based and require extensive hardware upgrades mainly on the GPU side. Our proposed solution aims to be highly accessible and independent of a runtime platform (OS) by using a web medium in order to process, transfer and visualize medical data. The main components of our platform are the rendering pipeline which uses a biased ray-tracing algorithm with an adaptive threshold stage and real-time intractable front-end for mobile and WebVR.

The first section is dedicated to the ray-tracing principles and volumetric data processing. We will also discuss the principle of bounding volume hierarchy (BVH) which was implemented to transfer the voxel from the backend pipeline to the frontend in order to be rendered and viewed. Moreover we will present the software resources used in the implementation of our platform.

The results section will feature the characteristics of our platform from a software architecture standpoint and will include performance-related information. We will emphasize on the specific characteristics of the represented data along side the digital imaging and communications in medicine (DICOM) original files and impact of the adaptive threshold system we implemented.

The paper will be concluded by discussing the main difficulties we faced in implementing a raytracing renderer and the decision to split our pipeline. We will also highlight the future development directions and the planned new features and improvements for our platform.

II. Materials

A. *Volume Raytracing principles*

Research in volume rendering spans in multiple directions covering areas such as: Transmission rendering, Feature exploration, accelerated volume rendering, and machine learning applications to volume rendering.

Raytracing is a stochasting method of approximating virtual surfaces, sparse domains and objects. The most common way of raytracing objects is by using a MonteCarlo raytracer[6 - 7].

978-1-7281-0101-9/19/$31.00 ©2019 IEEE

Volume rendering relies heavily on dedicated GPU(graphical processing units) hardware to load medical imaging data into the texture memory units.

The 3D representation reconstrution is built in multiple stages of sampling, domain classification and compositing of the given proxy geomtsry.

Look-up tables (LUT) are employed for the direct classfication of tissue types, most commonly one dimensional LUTs for CT scan data, where the contrast allows for direct detection of each individual tissue present in the imaging data.

On the other hand bidimensional techniques target the medical imaging slices along with the spacial data embbeded in the DICOM file by interpolating along a virtual height axis.[8]

Almost all methods require the usage of texture memory to store the data. Common issues are digital artifacts resulted from low sampling rates and compression artifacts inherited from the original data [9]

Common techniques to improve the viewing quality of the rendered volume are: bilinear filtering and trilineal interpolation which are widely used in CGI(computer generated image) industry and VR(virtual reality) applications. [10]

Another method for reconstrucing imaging data is by placing objects(spheres, cubes etc) with the following parameters: size, surface shading(diffuse, glossy) and transmittance along a projected perspective[11]. Then the proxy geometry is loaded in the memory with the global illumination and shading parameters calulated.

Scalar data is used for calculating gradients and orientations of vectors in the dataset. The gradient obtained helps in the reconstruction and surface shading of the volume domain.[12] Scalar data specific values are:

- Magnitude of the scalar field
- Vector orientation
- Normal mapping

B. *Voxel data and DVR*

Volumetric rendering quality in MCRT is directly tied to the number of steps measured inside the domain in a singe pass. This is achieved by implementing a computational model that accounts for light traversal trough semi-permeable domains of scalar fields.

Relation 1 showcases the computational aspect of volume rendering in the form of an integral. The calculation is applied for each ray that passes trough a given domain.

The coeficients present in *realtion 1* are:

- Coordinates *(x;y)* of the output image *I*
- Viewport postition *a* which is the location of the ray starting to trace the volume.
- Ray exit is directly tied to *b*

$$I(x,y) = \int_a^b c(s) e^{-\int_a^s K(u)du} ds \qquad (1)$$

- C is the domain influence over the ray intensity.

A number of steps can be given in order to measure how much the ray is attenuated in distance between the ray entrance a and ray exit b. The number of steps affects both performance and the accuracy with which the volume is reconstructed.

The measurement is directly tied to the light transmission parameters, and overall density. In the case of MRI volumes which are built using images stacked on one axis the information used to render can contain information about the density of the tissue. From an exploratory standpoint voxel space can be optimized by reducing and interpolating the source images inside a three-dimensional array of voxels that inherit the density values from the grayscale MRI images.

A voxel representation of the stack of images is generated and raytraced. The transmission function acts as a discriminator between the tissue density and the opacity of the specific voxel. Such a function has direct bias towards the grayscale value of the bidimensional image, by controlling the bias value and the number of steps the ray is measured a physically accurate and photorealistic render can be achieved fig 1.

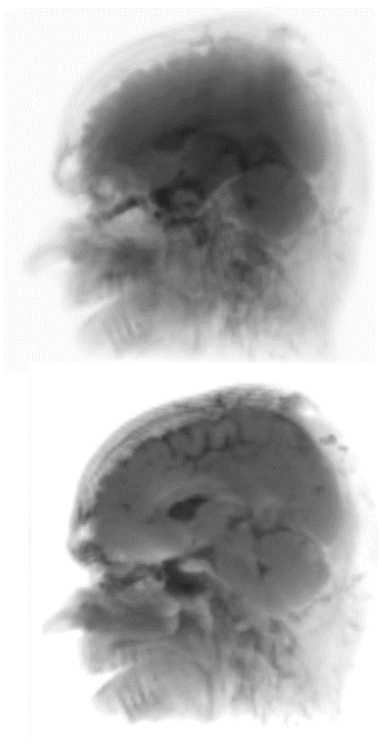

Fig. 1. Volume density as affected by the transmission function

C. *Adaptive Binarization thresholds*

In order to highlight specific details within the MRI data the following an adaptive binarization threshold discussed in previous works was implemented [14 - 16].

The determination of the threshold index value which usually sits between the second highest peak value and the previous peak value. The procedure is applied to the grayscale image histogram of the MRI data with smoothing filter that has a window dimension of 10% of the histogram vector length. In fig 2. is showcased how the adaptive threshold affects the rendering aspects.

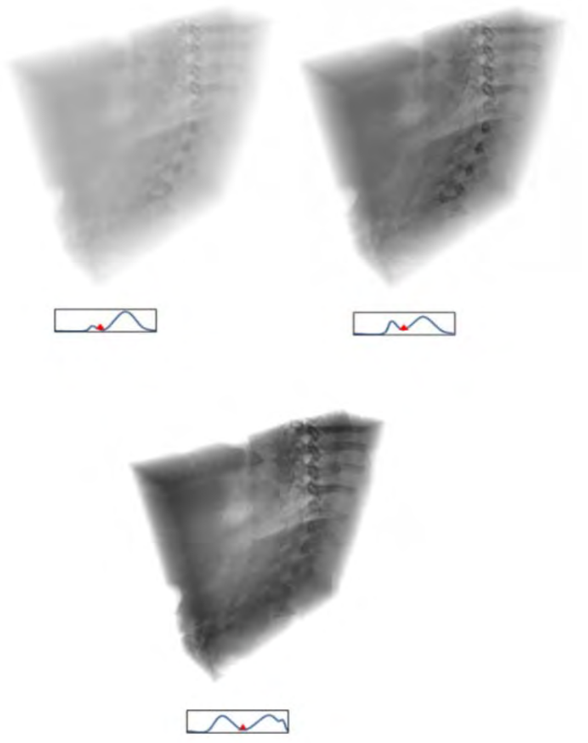

Fig. 2. Volume density as affected by the threshold implementation with a simplified threshold curve and index pointer

III. Results

A. *Webgl Implementation*

Our WebGL implementation is developed to be as independent as it can given the browser environtment, The transfer function and volume data are supplied directly by our backend solution. The renedering pipeline can be directly adjusted by using the transmission and HSL(hue, saturation, luminace) functions at any stage of our frontend implementation.

Our platform uses a backend-frontend software architecture in which we process, compress and pack the volume data in scalable server environment and use a webgl-enabled browser to render the volume in real time using a voxel cache.

All procedures regarding processing, compressing and transferring the DICOM MRI data happened on the server side. The front interprets the voxel data and has a simple post processing stage with Hue, Saturation and Luminance (HSL).

B. *Back-end*

We implemented the image processing stage using a python based micro-service. The DICOM data is extracted and processed in PNG format with a depth of 8Bits. Based on DICOM vector coordinates the exact spacing between each slice is set arbitrarily . The whole dataset is transferred using a classic BVH algorithm as voxel data and sent to the client (webgl enabled browser) via a websocket connection. The backend operations are showcased in figure 3.

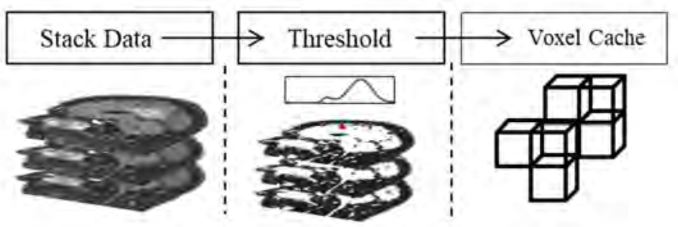

Fig. 3. Image processing stages trough the software backend

C. *Front-end*

The frontend is comprised of a HTML5 canvas which displays the render volume and the HSL controls we implemented in order to post-process the rendered frames and allow arbitrary adjustments.

Our render supports stereo rendering since we load the voxel data locally and is stored in the memory of the client. The stereo rendering can be used for VR applications. The raytracing renderer is implemented using webgl standard scripting GLSL, and called within the front-end using JavaScript routines and workers(fig 4).

The web browser environtment relies on a number of elements to be parametrized. The scene in which our MRI data is loaded allows adjustments to the transmittance of our voxel domain. The raytracer is directly dependant to the camera position and bounding box of the rended MRI. Ray discrimination is calculated in the following manner: direct raycasting are used to sample the volume and and indirect rays are used to sample the environment and dynamic lighting.

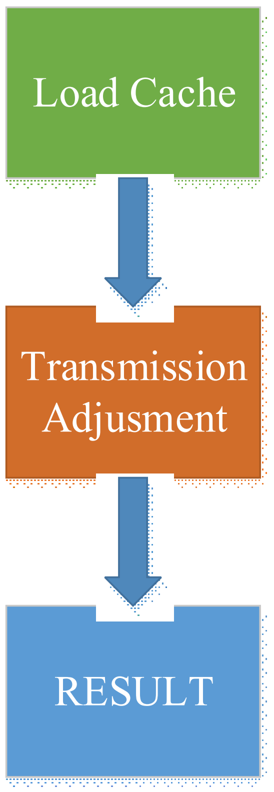

Fig 4. Frontend stages in webgl

Conclusions

In this paper we presented a web platform capable of visualizing and rendering 3D volumetric data from MRI datasets. The results presented feature an acceptable framerate of 60FPS and easily distinguishable anatomical characteristics of the source data. We discussed our approach and focused on the main difficulties of rendering volumes inside a web browser. Further more we presented the main method for computing the distance between slices by using DICOM geometry operators. We consider our platform and implementation a viable solution for volumetric rendering of MRI data in web environments.

The following aspects need to be addressed: In order achieve a 60FPS framerate on a desktop device, we pushed our software pipeline towards a dedicated GPU. On the mobile devices in order to render at an acceptable framerate the step count of the raytracing renderer was visibly reduced, thus the necessity for adaptive threshold algorithm was justified in order to scale down the voxel information and highlight only specific parts of the MRI dataset (fig5).

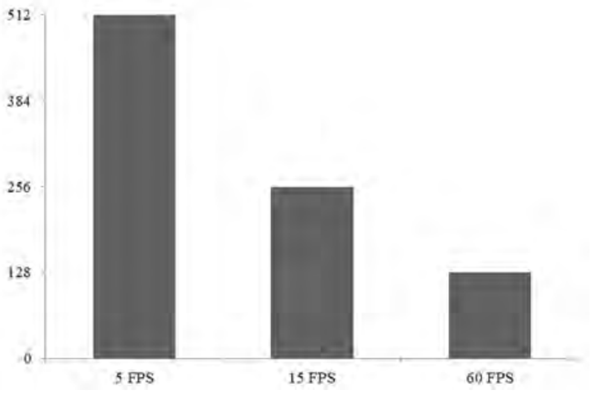

Fig. 5. Overall performance on a desktop device where Y-axis represents the voxel array and X-axis represents the frame-per second of the WebGL canvas

Currently the hardware industry is shifting towards dedicated raytracing GPU modules which would greatly decrease processing and rendering times.

Shortly, we would like to make the following improvements to our platform:

- Implementing multidimensional transfer functions
- Adaptive sampling and step count
- More post-processing options
- Global illumination implementation
- AI based tissue density processing

Acknowledgement

This paper was developed within the program National Project: GNaC 2018 ARUT no. 19/06.02.2018 from Politehnica University of Bucharest.

References

[1] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The contour spectrum. In Proceedings IEEE Visualization 1997, pages 167–173, 1997.

[2] H. Costin, "A Fuzzy Rules-Based Segmentation Method for Medical Images Analysis", Int. Journal of Computers, Communications & Control (IJCCC), vol. 8, No. 2, 2013, pp. 196-205, ISSN 1841-9836, 2013

[3] L. D. Bergman, B. E. Rogowitz, and L. A. Treinish. A rule-based tool for assisting colormap selection. In IEEE Proceedings Visualization 1995, pages 118–125, October 1995.

[4] B. Cabral, N. Cam, and J. Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In ACM Symposium On Volume Visualization, 1994.

[5] D. B. Conner, S. S. Snibbe, K. P. Herndon, D. C. Robbins, R. C. Zeleznik, and A. van Dam. Three-dimensional widgets. In Proceedings of the 1992 Symposium on Interactive 3D Graphics, pages 183–188, 1992.

[6] Curta C., Nicu A.,Plesa M., Valcu M., Ciupa R. (2009) Positioning System for Transcranial Magnetic Stimulation. IFMBE Proceedings, vol 26. Springer, Berlin, Heidelberg, DOI: 10.1007/978-3-642-04292-8_32

[7] H. Costin, C. Rotariu, "Medical image analysis and representation using a fuzzy and rule-based hybrid approach", International Journal of Computers Communications & Control (IJCCC), Oradea, Romania, Vol. 1, Supplement: S, pp. 156-162, 2006

[8] S. Di Zenzo. A note on the gradient of a multiimage. Computer Vision, Graphics, and Image Processing, 33(1):116–125, 1986.

[9] D. Ebert, C. Morris, P. Rheingans, and T. Yoo. Designing effective transfer functions for volume rendering from photographic volumes. IEEE TVCG (to appear), 2002.

[10] K. Engel, M. Kraus, and T. Ertl. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In Siggraph/ Eurographics Workshop on Graphics Hardware 2001, 2001.

[11] A. Van Gelder and K. Kim. Direct volume rendering with shading via three-dimensional textures. In ACM Symposium On Volume Visualization, pages 23–30, 1996. 12.

[12] C. Marincaş, A. Nicu,C. Csutak, R.V. Ciupa, Evaluating SAR on the human head (during MRI examination) as function of frequency, MEDITECH 2014, Cluj-Napoca, Romania, ISBN 978-3-642-04291-1;

[13] C. Rotariu, A. Pasarica, G. Andruseac, H. Costin, D. Nemescu, "Automatic Analysis of the Fetal Heart Rate Variability and Uterine Contractions", Proc. of EPE 2014 (IEEE Int. Conf. and Exposition on Electrical and Power Eng.), Iasi, Romania, 16-18 Oct. 2014, pp. 553-556, 2014.

[14] J. Kniss, G. Kindlmann, and C. Hansen, "Multidimensional transfer functions for interactive volume rendering," IEEE Transactions on visualization and computer graphics, vol. 8, no. 3, pp. 270–285, 2002.

[15] C. Correa and K.-L. Ma, "Size-based transfer functions: A new volume exploration technique," IEEE Transactions on Visualization and Computer Graphics, vol. 14, no. 6, pp. 1380–1387, 2008.

[16] Various, "The occlusion spectrum for volume classification and visualization," IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 6, pp. 1465–1472, 2009.

[17] C. D. Correa and K.-L. Ma, "Visibility histograms and visibility-driven transfer functions," IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 2, pp. 192–204, 2011.

[18] I. Wald, G. P. Johnson, J. Amstutz, C. Brownlee, A. Knoll, J. Jeffers, J. Gunther, and P. Navratil, "OSPRay-A CPU ray tracing framework ¨for scientific visualization," IEEE Transactions on Visualization and Computer Graphics, vol. 23, no. 1, pp. 931–940, 2017.

[19] Ciprian Larco, Radu Pahonie, Mihai Mihaila-Andres, "Experimental study on mode I fracture of fibredux unidirectional prepeg", AIP Conference Proceedings 1836, 020037 (2017); doi:http://dx.doi.org/10.1063/1.4981977

[20] D. Jonsson and A. Ynnerman, "Correlated photon mapping for interactive ¨global illumination of time-varying volumetric data," IEEE Transactions on Visualization and Computer Graphics, vol. 23, no. 1, pp. 901–910, 2017.

[21] C. Marincaş, A. Nicu,C. Csutak, R.V. Ciupa, Evaluation of thermal distribution in the head following exposure to an electromagnetic field generated by a MRI, MEDITECH 2014, Romania, ISBN 978-3-642-04291-1, ISSN 1680-0737, DOI 10.1007/978-3-319-07653-9, Springer