Creative Interaction for Plasma Physics

Han Sol Shin[1], Jee Ho Song[1], Tae Jun Yu[2], and Kun Lee[3](✉)

[1] Department of Information and Communication,
Handong Global University, Pohang, Republic of Korea
hansol@handong.edu, vanadate.kr@gmail.com

[2] Department of Advanced Green Energy and Environment,
Handong Global University, Pohang, Republic of Korea
taejun.yu@gmail.com

[3] School of Computer Science and Electronic Engineering,
Handong Global University, Pohang, Republic of Korea
kunlee@handong.edu

Abstract. Particle accelerator has mainly used in nuclear field only because of the large scale of facility. However, since laser-plasma particle accelerator, which has smaller size and spends less cost, developed, the availability of this accelerator is expended to various research fields such as industrial and medical. In these, real-time and interactive applications, accelerating the compute time is a critical problem. We conduct a simulation discussed the method of visualization of multivariate data in three dimensional space by using colored volume rendering and surface rendering. This system offers real-time 3D images via convert various file format comes from PIC simulation into OpenGL texture type to analyze and modify plasma data. We also suggest new method to visualize plasma data by using alpha shape to achieve a different level of detail for the given plasma based particle acceleration data. Many scientists are not familiar with the concepts of programming languages. Therefore, we employ the visual- scripting engine to define the rendering context of plasma physics.

Keywords: Plasma · Data visualization · Visualization toolkit

1 Introduction

Plasma is one of the states of matter, which is consist of ionized electrons and protons. Plasma is electrically conductive, so unlike gas, it responds strongly to electric and magnetic fields or its own nearby particles. Scientists mainly use computer simulation to predict the movement of charged particles of plasma on the influence of electromagnetic fields. For the precise results, most of simulation data is very large. It is impossible to catch the meaning of scientific simulation data, especially high resolution and multi-dimensional data, without visualization. Visualization gives insight to scientists.

Our suggestion includes the method of visualization multi-dimensional data with real-time performance. Real-time performance is important feature because interactive view of visualized data makes it easy for scientists to get some insights. The other feature of our work is the visual-scripting. Visual-scripting helps the scientists who are

A. Marcus (Ed.): DUXU 2016, Part III, LNCS 9748, pp. 214–222, 2016.
DOI: 10.1007/978-3-319-40406-6_20

not familiar with professional programming language to pre-process their data and decide to what to render.

2 Problem Description

Existing visualization toolkits, such as VisIt and ParaView, were developed for general purpose. These toolkits deal with various data format and type. They also rapidly release new version by open source community. However, there are some problems in visualizing plasma-based particle acceleration data. First of all, they require a long configuration time in order to produce desired images due to unnecessary processes incurred by their versatility. Also, they face difficulties in visualizing electric fields and reconstructing surface because of the restricted data pre-processing. Furthermore, their unintuitive interfaces make it harder for unexperienced users to learn them. To solve these problems, we developed a plasma specialized visualization toolkit (Fig. 1).

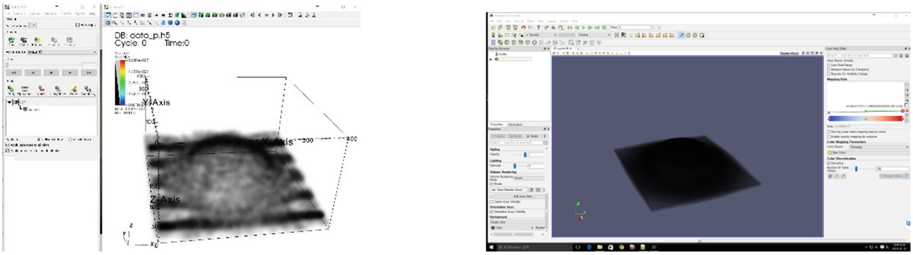


Fig. 1. (a) VisIt and (b) ParaView

3 Simplify the Process of Configuration

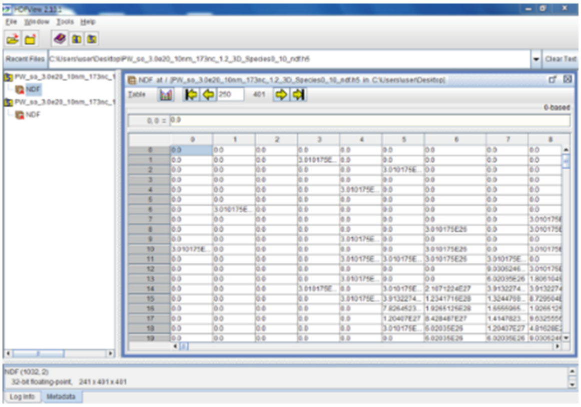
3.1 Existing Method

We benchmark previous visualization toolkits. We researched the processes from the beginning until getting the rendered image of plasma, using texture-based volume rendering method, in each toolkit while using identical data of charge density in the plasma. Our plasma data is HDF5 three dimensional 32-bit floating point grid data. They have 241 × 401 × 401 resolution. HDF5 is the data model, library, and file format used by scientific data simulations. It has been designed flexibly so it is suitable for various datatypes. Figure 2 is electric charge density data represented by three dimensional array (Table 1).

When operated, VisIt and ParaView show the following process (Table 2).

3.2 Proposed Method

In order to simplify the configuration process, we found the need for a reduction in the setting stage and shortening the time to adjust the setting values, and decided to provide default values and real-time feedback.



C:\Users\user\Desktop\PW_so_3.0e20_10nm_173nc_1.2_3D_Species0_10_ndf.h5

NDF at / [PW_so_3.0e20_10nm_173nc_1.2_3D_Species0_10_ndf.h5 in C:\Users\user\Desktop]

0, 0 = 0.0

	0	1	2	3	4	5	6	7	8
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	3.010175E...	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	3.010175E...	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	3.010175E...	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	3.010175E...	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.010175E
8	0.0	0.0	0.0	0.0	0.0	0.0	3.010175E26	0.0	3.010175E
9	0.0	0.0	0.0	0.0	3.010175E...	0.0	0.0	0.0	0.0
10	3.010175E...	0.0	0.0	0.0	0.0	0.0	3.010175E26	0.0	3.010175E
11	0.0	0.0	0.0	0.0	3.010175E...	3.010175E...	3.010175E26	3.010175E...	0.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	9.0305246...	3.010175E
13	0.0	0.0	0.0	0.0	3.010175E...	0.0	0.0	6.02035E26	1.806104E
14	0.0	0.0	0.0	3.010175E...	0.0	3.010175E...	2.1071224E27	3.9132274...	3.913227
15	0.0	0.0	0.0	0.0	3.010175E...	3.9132274...	1.2341716E28	1.3244769...	8.729564E
16	0.0	0.0	0.0	0.0	0.0	7.8264523...	1.9265125E28	1.6555065...	1.926512
17	0.0	0.0	0.0	0.0	0.0	1.20407E27	8.428487E27	1.4147823...	9.632555
18	0.0	0.0	0.0	0.0	0.0	3.010175E...	6.02035E26	1.20407E27	4.81628E
19	0.0	0.0	0.0	0.0	0.0	0.0	6.02035E26	6.02035E26	9.030524

NDF (1032, 2)
32-bit floating-point, 241 x 401 x 401

Fig. 2. Three dimensional grid data of electric charge density

Table 1. Data properties

Type	HDF5 Scalar Dataset
No. of Dimensions	3
Resolution	241 × 401 × 401
Data type	32-bit floating point

Table 2. Process of VisIt and ParaView

Step	VisIt	ParaView
1	Open a "HDF5" file	Write a "XMDF" file to recognize "HDF5" file
2	Add a plot "Volume" with "NDF"	Open the "XMDF" file
3	Draw plots	Select "XMDF Reader"
4	Edit "Volume plot attributes"	Apply
5	Select "3D Texturing" in "Render Options"	Select "Volume" in "Representation"
6	Adjust "Color Data" in "1D transfer function" (*appropriately)	Edit "Color Mapping Data" (*appropriately)
7	Apply	

Default Values. We make these common processes concise through offering default values in the basic setup: data type, cell size, color, rendering options, etc. Moreover, we recommend and provide the interpolation of variation by referring to the distribution of data.

Real-Time Feedback. We chose the texture-based rendering amongst many volume rendering method. In real-time application, volume rendering methods which contain numerous computation, including ray casting method, is not appropriate.

Through the process above, the previous setup process was simplified into only two steps to get the result: open files and click the "volume" button.

4 Methods of Visualization

4.1 Texture-Based Volumetric Rendering

For the realistic rendering, volume ray-casting technique is commonly used. But visualization of scientific data, specifically plasma data, is not required serious physical lighting model because seeing plasma is not natural phenomenon. What we need is conceptual image of it. We use OpenGL's 3D texture and alpha blending technique for the real-time performance.

We calculated the linear interpolation between two colors representing minimum and maximum value respectively on data set. We set the alpha value to size of the value of the three dimensional array's component. Figure 3 is gray-scaled image of alpha values. We stack these images up on three dimensional space, and set blend mode to get final color of volumetric image. In Fig. 4, charge density of plasma is represented by color and transparency.

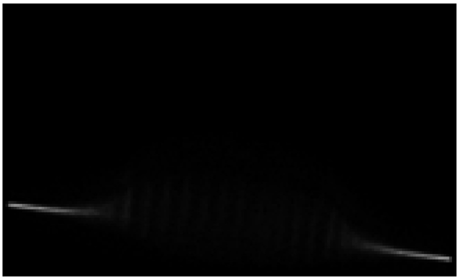


Fig. 3. Slice of generated 3D texture

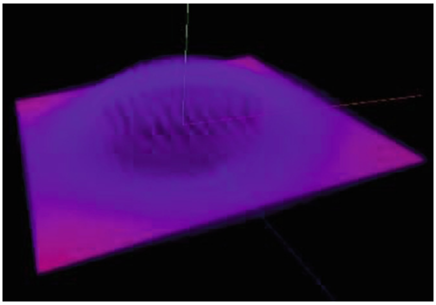


Fig. 4. Visualized image of electric charge density of plasma using OpenGL texture and alpha blending

4.2 Electric Field Computation

It costs high in both memory usage and time consuming to sample the space to compute charge density and calculate electric field on that with large size of plasma data. However, by using convolution theorem, we make O(n^2) problem into O(nlogn) problem. Here are equations we used.

$$V_{net} = \int \rho(\vec{r}) \frac{k}{\left|\vec{R} - \vec{r}\right|} d\,Vol \tag{1}$$

$$h(R') = f(r') * g(R' - r') \tag{2}$$

$$h(R') = F^{-1}(F(f(r')) \times F(g(r'))) \tag{3}$$

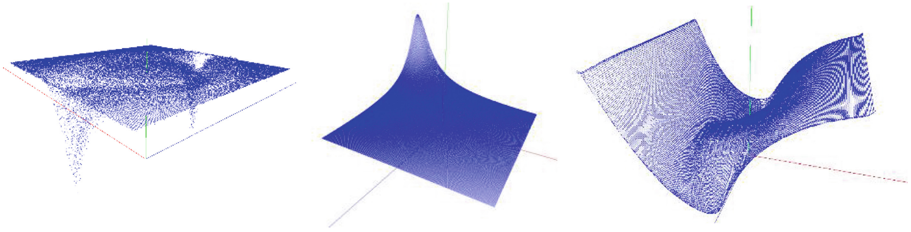
Equation (1) is ordinary equation for the electric potential. Equation (2) is the convolution form of Eq. (1). Equation (3) is application of convolution theorem on Eq. (2). We accelerate the FFT (Fast Fourier Transformation) algorithm by using GPGPU (General Purpose Graphic Processor Unit). Using convolution theorem, it can be unwanted result on margin because of periodicity of discrete Fourier transformation. To prevent this, we need to pad the date with zeros as much as size of kernel. In addition, the size of kernel must be same with of image for the point-wise multiplication. So we need additional memory space, and following is required data size of image.

$$N = \left(N^{fw} + N^{gw} - 1\right) \times \left(N^{fh} + N^{gh} - 1\right) \times \left(N^{fd} + N^{gd} - 1\right)$$

when $N^{fw}, N^{fh}, N^{fd}, N^{gw}, N^{gh}, N^{gd}$ are width, height and depth of image and kernel, respectively.

Figure 5 is visualization of intermediate data while calculate electric field, and Fig. 6 is the vector field of electric field in plasma, final result of calculation.

We compared the compute time between convolution on CPU without FFT and convolution on GPU with FFT. We considered the time to transfer the data form CPU memory to GPU memory. Environment of used system are Intel i7 920 CPU, 8 GB DDR3 RAM, and Nvidia Geforce GTX980 GPU (Table 3).



(a) (b) (c)

Fig. 5. (a) is electric charge density of plasma, (b) is convolution kernel and (c) is calculated electric potential of plasma.

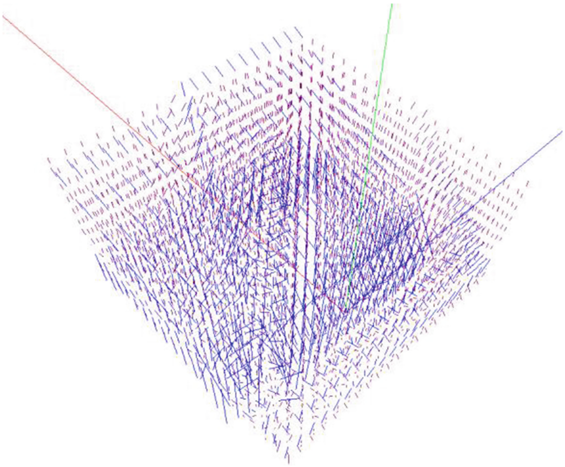


Fig. 6. Line representation of electric field

Table 3. Comparison of the performance for FFT algorithm and conventional convolution algorithm

N	Compute time of convolution on CPU (ms)	Compute time of FFT algorithm on GPU (ms)	Compute time of FFT algorithm on GPU + Time to copy data from main memory to GPU memory (ms)
8k	290	815	815
64k	18.5k	801	803
125k	70k	796	801
5000k	–	1693	1852

4.3 Surface Reconstruction

We offer two method of surface construction. One is iso-surface, the other is alpha shape. We calculated iso-surface using marching cubes algorithm. This algorithm is method to extract triangles that is lying on same value, iso-surface, from a three dimensional grid data. This is mainly used for medical visualization. Figure 8 is rendered image of surface where iso-value is 0, which is generated by the algorithm. Three dimensional charge density data is discrete, therefore this algorithm does not guarantee connectivity of generated surface. That is the reason why we considered alpha shape.

In case of alpha shape, we use the package CGAL (Computational Geometry Algorithm Library). In CGAL, the α-complex of *S* is a subcomplex of this triangulation of *S*, containing the a-exposed *k*-simplices, $0 \leq k \leq d$. A simplex is a -exposed, if there is an open disk (resp. ball) of radius squre root of α through the vertices of the simplex that does not contain any other point of *S*, for the metric used in the computation of the underlying triangulation. That makes convex hall where α is infinite value (Fig. 7).

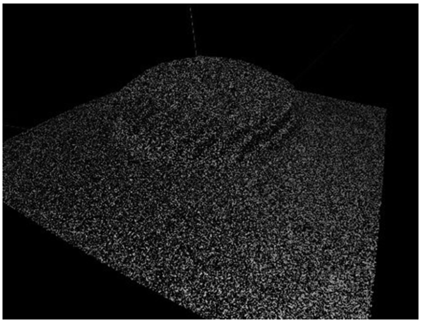
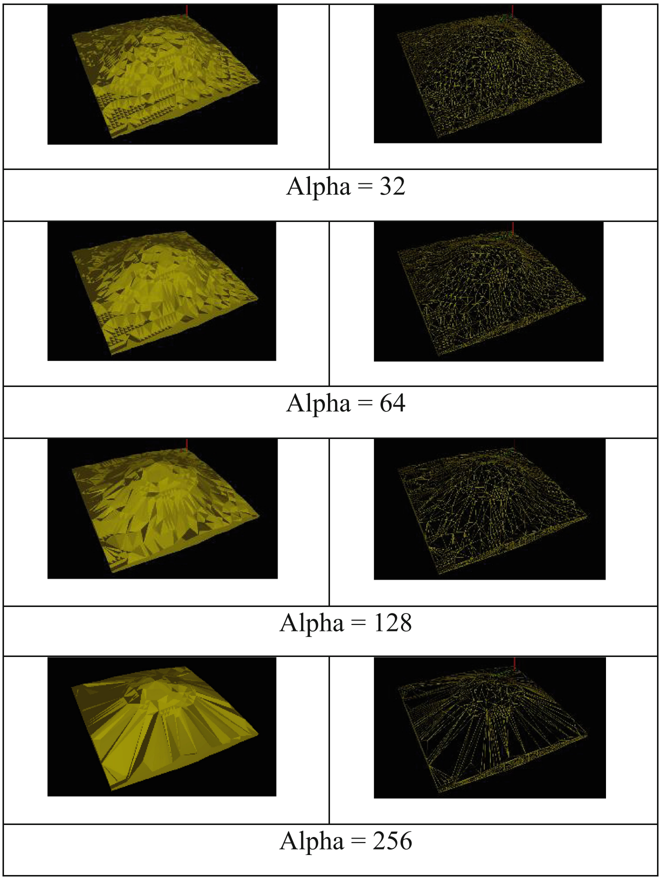


Fig. 7. The plasma's surface constructed by marching cubes



Alpha = 32

Alpha = 64

Alpha = 128

Alpha = 256

Fig. 8. Alpha shape of proton data

5 Visual Scripting

We used Visual scripting to complement the unintuitive interfaces. Visual scripting is based on node-based interface to writing the logic. Each of nodes represents data, branching control, and operation. Visual scripting is suitable for the data-driven paradigm because it is flow of data. In addition, it easily converted to parallelized computation. Script engine is necessary feature of visualization toolkit for the flexible environment to compose the rendering plot and the data preprocessing. However, many scientists are not familiar with programming language. For example, scientists have to remember a lot of command to use Matlab. It is not only hard to learn, but also easy to make mistakes. With the visual-scripting engine, scientists easily make a render context, the plot of rendered scene. Following script is example of the Matlab commands, and Fig. 9 is example of the rendering context built by visual scripting.

```
load mri D                                          % load da-
ta
D = squeeze(D);                                     % remove
singleton dimension
limits = [NaN NaN NaN NaN NaN 10];
[x, y, z, D] = subvolume(D, limits);                % extract
a subset of the volume data

[fo,vo] = isosurface(x,y,z,D,5);                    %
isosurface for the outside of the volume
[fe,ve,ce] = isocaps(x,y,z,D,5);                    % isocaps
for the end caps of the volume

------------ omit -------------

view(-40,24)
daspect([1 1 0.3])                                  % set the
axes aspect ratio
colormap(gray(100))
box on
```

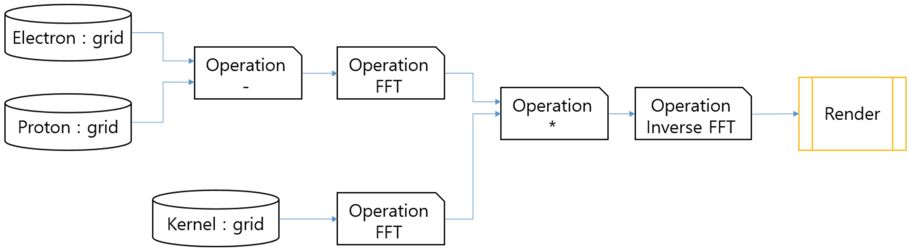


Fig. 9. The rendering context built by visual scripting

6 Conclusions

In this paper, we discuss how to visualize volumetric scattered data. In industrial and medical fields, real-time and interactive applications, accelerating the compute time is a critical problem. We designed our visualization toolkit for interactive and real-time observation of multi-dimensional plasma data. Using texture based volumetric rendering, we achieve both quality and performance. We use GPGPU and convolution theorem to accelerate computing time to calculate electric fields.

To achieve a different level of detail alpha shapes method is applied. Level of detail increases the efficiency of rendering by decreasing the workload. The alpha complex is a sub-complex of the Delaunay triangulation. For a given value of the alpha complex includes all the simplices in the Delaunay triangulation which have an empty circumsphere with squared radius equal or smaller than □. For further research, we are trying to apply this proposed method to the various tri-variate scattered data.

We also suggested visual scripting which have various advantage of representing a rendering plot or context. It is easy to learn even for scientists who have not experience of programming languages.

Acknowledgments. This work was supported by the Industrial Strategic technology development program, 10048964, Development of 125 J•Hz laser system for laser peering funded by Ministry of Trade, Industry & Energy (MI, republic of Korea).

References

1. Rubel, O., Geddes, C.G.R., Chen, M., Cormier-Michel, E., Wes Bethel, E.: Feature-based analysis of plasma-based particle acceleration data. IEEE Trans. Vis. Comput. Graph. **20**(2), 196–210 (2014)
2. Nielson, G.M., Hagen, H., Muller, H.: Scientific Visualization, pp. 477–525. IEEE Computer Society, Washington (1997)
3. Pavel, K., David, S.: Algorithms for Efficient Computation of Convolution, Design and Architectures for Digital Signal Processing. InTech, Rijeka (2013)
4. NVIDIA CUDA Toolkit Documentation v7.0. http://docs.nvidia.com/cuda/index.html. Accessed April 2015
5. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. Comput. Graph. **21**(4), 163–169 (1987)
6. Lee, K.: Multiresolution volumetric medical data modeling based on gaussian curvature by using weighted alpha. J. Next Gener. Inf. Technol. **3**(3), 41–51 (2012)
7. Shin, H., Yu, T., Lee, K.: Visualization of scattered plasma-based particle acceleration data. J. Korea Multimedia Soc. **18**(1), 65–70 (2015)
8. Shin, H., Yu, T., Lee, K.: Visualization of internal electric field on plasma. J. Korea Multimedia Soc. **19**(1), 80–85 (2016)