In Situ Visualization of Laser-Plasma Interaction

P. Valenta [1, 2], M. Danielová [1], O. Klimo [1, 2], S.V. Bulanov [1, 3]

[1] ELI Beamlines, Institute of Physics, Czech Academy of Sciences, Prague, Czech Republic
[2] Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Prague, Czech Republic
[3] Kansai Photon Science Institute, National Institutes for Quantum and Radiological Science and Technology, Kizugawa, Kyoto, Japan

INTRODUCTION

Thorough understanding of ultra-intense laser-plasma interaction may enable new routes in fundamental research as well as a wide range of applications [1]. However, such systems involve collective behavior of particles in self-consistent electromagnetic fields which is, in general, complex and strongly non-linear problem that can be investigated only with the help of numerical simulation.

An exponential increase of computational throughput of supercomputers enables researchers to perform simulations with unprecedented accuracy. Using the conventional post-processing approach of data analysis, such simulations would require extremely large amount of data to be stored on a persistent storage. The storage bandwidth performance, however, has not grown up as rapidly as the computational power. In practice, the data coming from the simulations have to be stored only at several time-steps or at much coarser resolution than the original data, the rest is just discarded. Therefore, a significant part of information may be potentially lost.

The technique where the simulation data are concurrently analyzed and visualized while it is being generated is usually referred to as in situ processing. In situ processing could circumvent the bottleneck of data transfer. By coupling the visualization and simulation together, one may process and analyze the simulation data at high spatial and temporal resolutions without the necessity of involving the storage resources.

Recently, we have instrumented the code *EPOCH* [2] with the *ParaView Catalyst* [3]. *EPOCH* is massively parallel, multi-dimensional plasma physics simulation code based on the particle-in-cell (PIC) method. *ParaView Catalyst* is a library that has been designed for in situ coupling of numerical codes with the state-of-the-art visualization system. Here we present our implementation strategy, performance analyses and demonstrate the in situ capabilities on several large-scale laser-plasma simulations.

IMPLEMENTATION DETAILS

The effect of in situ analysis on PIC codes has been extensively studied before [4]. Within this work, the *EPOCH* code has been coupled with *ParaView Catalyst* via so-called adaptor. Adaptor is a simulation interface, which should be separated from the main code in order not to disturb it and to simplify the build process. The *EPOCH* adaptor uses *Catalyst* C++ application programming interface to implement the following three methods:

The first method, which is called at the initial phase of the simulation, initializes *Catalyst* input channels and loads preconfigured visualization pipelines. There are multiple input channels provided. One channel is reserved for the underlying Cartesian grid, on which all the field quantities are calculated. Other channels are reserved for particles and their number depends on how many particle species the simulation involves. This approach ensures that the *Catalyst* pipelines can output results based on the field information only, on particular particle species information only or on arbitrary combination of inputs.

Visualization pipelines decide what operations to perform on the data and how to output desired results. In these pipelines, one can benefit from all the post-processing capabilities that *ParaView* offers. They may also contain the IP address and the port number of the *ParaView* server if the user wants to enable so-called live visualization. The pipeline may be hardcoded or in the form of a Python script that can be generated using the *ParaView* graphical user interface (GUI).

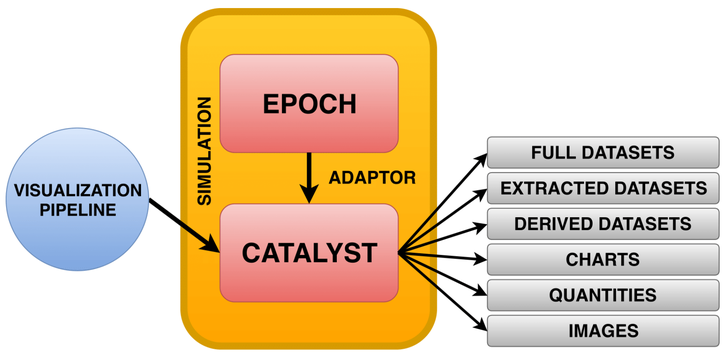

Figure 1: EPOCH workflow for in situ visualization using ParaView Catalyst.

The second method, which is called at each simulation time-step, builds all the necessary data structures and invokes the data processing specified in loaded pipelines. Since the *ParaView* is built on the standard visualization toolkit *VTK* [5], the simulation internal data structures have to be transferred into the *VTK* data structures. The Cartesian computational grid and its decomposition including ghost cells are represented by *vtkMultiBlockDataSet* and *vtkRectilinearGrid*, respectively. Each particle species is represented by distributed *vtkUnstructuredGrid* containing only points and *VTK_VERTEX* cells that index them. *VTK* data structures have to be recalculated only when the local domain changes (e.g. when particle cross the boundary of the domain or using load-balancing).

The Cartesian grid contains three field quantities – the electric field vector (**E**), the magnetic field vector (**B**) and the current density vector (**J**). To avoid deep-copying of arrays from the simulation data structures, the adaptor takes advantage of the structure-of-arrays memory layout using the *vtkSOADataArrayTemplate* class. This ensures that the adaptor will efficiently reuse the simulation memory and only a negligible amount of additional memory is used in creating *VTK* data structures.

Each particle contains the vector of its position that is linked with one vertex of the corresponding unstructured grid. In addition, the particle contains vector of its momentum and weight, which is a scalar quantity that determines how many real particles it represents.

The last method is used at the end of the simulation to release all the Catalyst resources. All three methods are then incorporated to the main code using preprocessor directives.

PERFORMANCE ANALYSIS

The *EPOCH* code instrumented with *Catalyst* benefits from various new capabilities. It can produce a wide range of outputs (Fig. 1):

- Full datasets
- Extracted datasets (slices, iso-surfaces, etc.)
- Derived datasets (streamlines, vector fields, etc.)
- Charts (plots along a line, histograms, etc.)
- Quantities (minimum or maximum values, etc.)
- Images

All these operations are prescribed in the visualization pipelines and executed only at user defined intervals or in response to specific conditions or events in the simulation. Analysis and visualization is then performed in place and in parallel, which results in much faster time to insight into the investigated phenomena than using traditional approaches.

In addition, the code can be run in two main modes of operation – batch mode and interactive mode. In the batch mode, the code executes the visualization pipelines automatically. In the interactive mode, the user can connect to *Catalyst* using the *ParaView* GUI and control the simulation. Since the user can interactively explore the data as it is being generated, this provides an invaluable tool for scientific insight as well as for debugging of simulation codes. Both two modes can be run simultaneously.

We have analyzed the performance of the instrumented code in terms of compute time and I/O cost for certain analysis operations on a test 3D simulation. Then we have compared the results with the identical simulation run which outputs the data that would have to be stored for the same analysis using conventional post-processing approach. The simulation dumped three field quantities (**E**, **B**, **J**) at relatively high frequency (each tenth iteration out of 2100) and in double precision. All test simulations were executed on 16 nodes of the ELI Beamlines computer cluster. Each node contains 16 Haswell-EP cores and 128 GB DDR4 RAM. Nodes of the cluster are connected by a fully non-blocking fat-tree Infiniband QDR network, with 40 Gbps bandwidth. The storage bandwidth performance is approx. 1.8 GB/s.

The results are shown in Tab. 1. As can be clearly seen, the simulation spends approx. 95 % of the total time on I/O operations and requires around 8.2 TB of disk space in the case of dumping whole datasets for post-processing. On the other hand, the ratio between computation and I/O operations regarding the individual *Catalyst* filters is much more reasonable. Therefore, using the right visualization pipelines to extract the features of interest, one can drastically reduce the I/O and speed-up the simulation.

Type of output	Runtime (hh:mm:ss)	Total size
No I/O	01:46:07	0.0
Full Dataset (SDF)	36:42:52	8.2 TB
Slice (VTM)	02:14:07	19.5 GB
Iso-surface (VTM)	02:27:47	115.1 GB
Plot over line (PVTP)	01:53:12	17.5 MB
Image (PNG) *	01:54:27	13.8 MB

Table 1: Comparison of compute time and I/O cost for several data processing operations. *Images rendered at 1080p using software renderer.

RELATIVISTIC FLYING MIRRORS

In the following, we demonstrate the capabilities of the code *EPOCH* instrumented with *ParaView Catalyst*. The goal was to determine the impact of using in situ diagnostics on the interpretation of scientific results.

We have chosen to model relativistic flying plasma mirrors (RFM), a concept based on the reflection of counter-propagating laser beam from thin dense electron layers traveling with velocities close to the speed of light. Due to the double Doppler effect, the reflected wave is compressed, amplified and its frequency is up-shifted. Generated sources of coherent electromagnetic radiation are of great demand for innovative time-resolved application experiments [6, 7]. Various schemes described in theoretical as well as experimental studies have proven the feasibility of this concept [8].

In this case, the in situ analysis might be particularly useful since the standard PIC simulations require extremely high spatial and temporal resolution to accurately describe a large band of frequencies and, at the end of the day, the main part of information is carried by only a small fraction of reflected radiation.

We have performed several large-scale 2D and 3D simulations using the *EPOCH* code instrumented with *Catalyst* corresponding to the ongoing research of relativistic flying mirrors. Fig. 2 shows certain characteristics that have been obtained in situ using visualization pipelines.

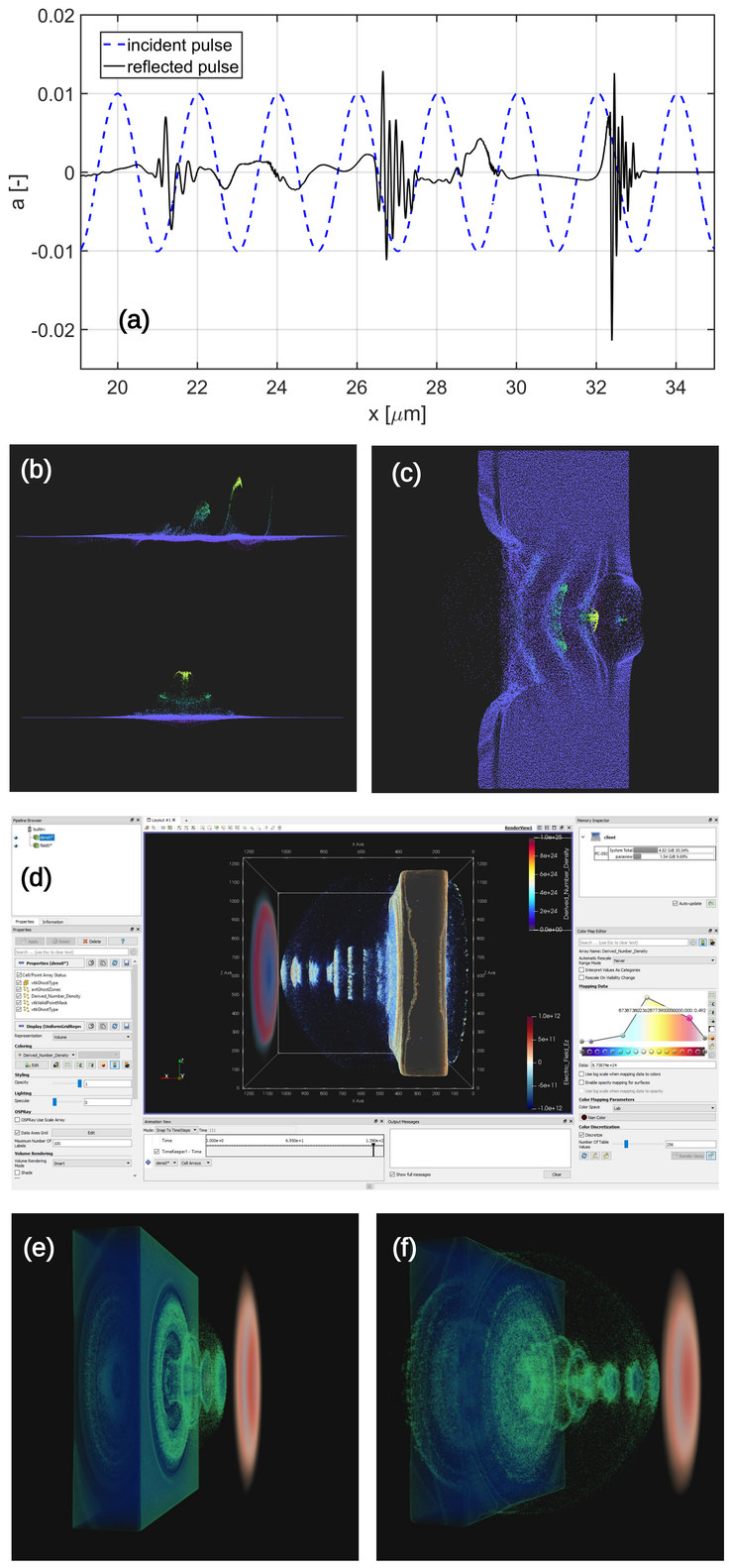

Figure 2: **(a)** Plot over line filter applied on the incident and reflected light. **(b)**, **(c)** Rendered images of the phase space. **(d)** Electron density and laser electric field showing the formation of plasma mirrors in the interactive mode using *ParaView* GUI. **(e)**, **(f)** 3D laser-plasma interaction displayed using volume rendering.

CONCLUSION

Within this work, we have coupled the plasma physics simulation code *EPOCH* with the in situ library *ParaView Catalyst*, we have outlined the implementation strategy and carried out performance analyses. Then we have used the instrumented code for several large-scale simulations of relativistic flying plasma mirrors.

In situ visualization is expected to enable a wide range of new interactive applications in the near future. In addition, as HPC moves towards the exascale era, in situ approach is widely predicted to become an indispensable tool for speed-up of large-scale simulations and more efficient use of modern super-computing resources [9].

ACKNOWLEDGEMENTS

- This work was supported by the project High Field Initiative (CZ.02.1.01/0.0/0.0/15 003/0000449) from the European Regional Development Fund.
- Computational resources were provided by the ECLIPSE cluster of the ELI Beamlines.
- The development of the EPOCH code was funded in part by the UK EPSRC grants EP/G054950/1, EP/G056803/1, EP/G055165/1 and EP/M022463/1.

REFERENCES

[1] G. A. Mourou, T. Tajima and S.V. Bulanov, *Rev. Mod. Phys.* **78**, 309 (2006)

[2] T. D. Arber et al., *Plasma Phys. Control. Fusion* **57**, 11 (2015)

[3] A. C. Bauer, B. Geveci and W. Schroeder, *ParaView Catalyst User's Guide*, Kitware, Inc. (2017)

[4] O. Rubel et al., *IEEE Computer Graphics and Applications* **36**, 3 (2016)

[5] W. Schroeder, K. Martin and B. Lorensen, *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*, 4th Edition, Kitware, Inc. (2006)

[6] F. Krausz and M. Ivanov, *Rev. Mod. Phys.* **81**, 1 (2009)

[7] R. Neutze et al., *Nature* **406**, 752 (2000)

[8] S. V. Bulanov et al., *Physics - Uspekhi* **56**, 429 (2013)

[9] E. W. Bethel, H. Childs, C. Hansen, *High Performance Visualization: Enabling Extreme-Scale Scientific Insight*, Chapman & Hall/CRC (2012)