PARALLEL DATA VISUALIZATION

Jean M. Favre
Swiss National Supercomputing Center, Switzerland

Challenges in large data visualization

Parallel numerical solvers in engineering are ubiquitous. Simulation engineering worldwide enjoy the availability of large to massive supercomputer power, and the datasets produced in fluid flow simulations most often exceed the capacity of a desktop to handle the post-processing analysis tasks. Numerical results are found in grids of size greater than a billion cells, with transient solutions over thousands of time steps. Without surprise, scientific visualization softwares have also matured in the realm of parallel architectures in order to enable the processing of such massive results. Of paramount importance to most practitioners, is the idea that datasets should not be moved away from the large computing environments where they have been generated. In fact, at least in the academic environment, it is not practical and economically feasible for a researcher to run simulations on a large and shared compute cluster, and then to move the results to an under-sized institute desktop and storage area.

We normally use scientific visualization in at least two ways. Interactively, in a mode where the scientist can explore and query the datasets in search of relevant features, applying visual paradigms leading to the insight and understanding of the simulation. And in batch-oriented fashion, where a given scenario for the feature extraction, representation and animation scripts must be replayed, with different parameters, different time steps, etc. We are thus motivated to use an environment enabling, rapid access to the data, feature extraction (particle traces, vortex cores, cutting planes, isosurfaces, etc.), image generation, and replay of previously saved scenarios. When datasets reach the tera-scale and beyond, parallel processing is the only practical approach and can leverage on several decades of practice in High-Performance-Computing (HPC).

Today, several data visualizations softwares enable parallel data analysis. We will cite the most prominent in the commercial world, EnSight Gold [1], and FieldView [2]. Alternatively, there exist also some excellent open-source softwares enabling parallel visualization. VisIt [3] and ParaView [4; 5] enjoy the most publicity. For the reminder of this text, our presentation will refer to the open-source ParaView, in order to give attendees of the von Karman Institute for Fluid Dynamics' Lecture Series the motivation to deploy and practice parallel visualization in their own environment and at no cost. Lessons learned will however be applicable both to ParaView and VisIt.

Large data visualization methods

Before engaging in the discussion of parallel data visualization, it is instructive to consider an alternate approach, enabling the visualization of very large data even on a small size desktop.

Data Streaming

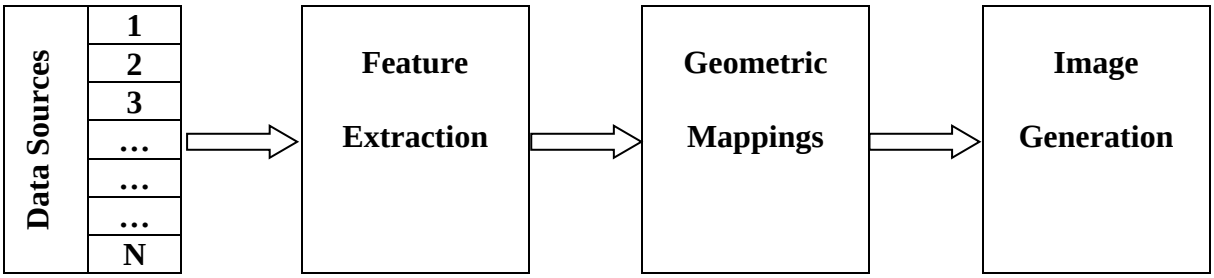

Figure 1: The Visualization Pipeline

We generally speak of the Visualization Pipeline (Figure 1) as a metaphor for a series of filters acting on the data; reading the data results, transforming them into some derived quantities, creating geometric representations of them, and generating a pixel image. The visualization pipeline will require processing power and memory storage proportional to the input datasets. Attempting to visualize datasets bigger than the physical size of the host system would clearly be unfeasible, since each stage of the pipeline consumes more memory and processing. Using *Data Streaming* [6], an oversized dataset could be processed if the data can be handled in a piecewise manner, breaking it into smaller pieces, processing each piece individually, and combining the results into the final image. Data Streaming derives its name from the now familiar concept of watching video streams. A very large film can be watched without the need to hold a copy of the full video file. Backward stepping is however not possible. The video stream is watched once, and then discarded. In data-streaming visualization, one can execute a visualization pipeline for a very large dataset under the following conditions:

- The data must be separable. That is to say that the dataset can be broken up (read from disk files) in N pieces, one piece at a time, in a simple and efficient fashion and each piece should be coherent in geometry, topology, and data structures.
- The data must be mappable. In order to decide how many pieces to use and to size up the overall memory consumption of the pipeline, we should know what portion of the input data is required to generate a given portion of the output.
- The result should be independent of the piece-wise sub-division. The final image should be independent of the number of pieces and of the order of execution.

Data partitioning methods will vary with the nature and structure of the data. A large three-dimensional structured grid as used in a Direct Numerical Simulation is easily broken up along the I, J, K axes of its computational domain. A multi-block structured can be broken up, one block at a time. A fully unstructured simulation will be broken up following the geometric partitioning used by the flow solver itself. Examples of these three modes of sub-division are shown below (Figure 2).