

Visualize This: Lessons from the Front-lines of High Performance Visualization

Ayat Mohammed
Virginia Tech
3030 Torgersen Hall
Blacksburg, Virginia 24060
maaayat@vt.edu

Nicholas F. Polys
Virginia Tech
3030 Torgersen Hall
Blacksburg, Virginia 24060
npolys@vt.edu

Duncan Farrah
Virginia Tech
119 Robeson Hall
Blacksburg, Virginia 24060
farrah@vt.edu

ABSTRACT

This paper presents a comprehensive workflow to address two major factors in multivariate multidimensional (MVMD) scientific visualization: the scalability of rendering and the scalability of representation (for perception). Our workflow integrates the metrics of scientific computing and visualization across different STEM domains to deliver perceivable visualizations that meet scientists' expectations. Our approach attempts to balance the performance of MVMD visualizations using techniques such as sub-sampling, domain decomposition, and parallel rendering. When mapping data to visual form we considered: the nature of the data (dimensionality, type, and distribution), the computing power (serial or parallel), and the rendering power (rendering mechanism, format, and display spectrum). We used HPC clusters to perform remote parallel processing and visualization of large-scale data sets such as 3D point clouds, galaxy catalogs, and airflow simulations. Our workflow brings these considerations into a structured form to guide the decisions of visualization designers who deal with large heterogeneous data sets.

CCS CONCEPTS

• **Human-centered computing** → **Scientific visualization**; **Visual analytics**; **Empirical studies in visualization**;

KEYWORDS

Scalable visualization workflow, Remote rendering, 3D point cloud, Galaxy catalogs

ACM Reference format:

Ayat Mohammed, Nicholas F. Polys, and Duncan Farrah. 2018. Visualize This: Lessons from the Front-lines of High Performance Visualization. In *Proceedings of PEARC, Pittsburgh, PA USA, July 2018 (PEARC18)*, 8 pages. DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Extremely large scientific data sets require scalable paradigm/taxonomy that handle the variability and the dimensionality representation in the scientific visualization. These paradigms should enable big data processing and visualization with a reasonable computational cost and an acceptable cognitive (including perception) level. In an effort to identify a concrete research agenda to advance natural user interfaces for scientific visualization, (Keefe and Isenberg 2013) explained a set of six specific challenges, one of the challenges was

"Collaboration, immersive environments, and high-performance computing". They emphasized on developing new methods for using natural user interfaces with visualization to deal effectively with high performance computing and with big data with a great focus on collaboration.

Wijk considered visualization from three perspectives. First, considering visualization from a technological point of view, targeting proven effectiveness and efficiency. Second, to think of visualization as an art that itself is interesting to seek elegance and beauty. The third perspective is to consider visualization as an empirical science uses generic laws with predictive power. The author emphasized that these different views are strongly related and can mutually affect each other, but each one has its own unique role, (Van Wijk 2006). The proposed paradigm considers visualization as an art combined with empirical science.

To map dataset variables to visual forms we should follow The best practices in data visualization recommendations, (Ware 2013):

- (1) Mapping quantitative data to 'ordered' visual elements, such as position, size, and brightness;
- (2) Mapping qualitative data, such as names and categories, to 'unordered' visual elements, such as shape and color.

Based on the visualization guidelines suggested by Card et al. (1999); Ware (2000, 2013) and the generic model for visualization developed by (Van Wijk 2006), we propose a scientific visualization paradigm to address the scalability of rendering and representation of multivariate multidimensional scientific data. Furthermore, this paper discuss the application of the proposed paradigm on four case studies that addresses two main research questions regarding user's perception of scientific visualization:

- (1) What are users' sensitivities to different screen management approaches, can we improve comparison tasks by composition (overlay) or juxtaposition (side-by-side) approaches?
- (2) How does immersion or collaboration affect these dynamic of perception and performance? Can visual differencing be improved with extra technology or collaborators?

This paper describes scientific visualization solutions for different case studies that explore these research questions. Each case study came from a different research domain with its own requirements and subsequent visualization design. These visualization procedures were designed and implemented according to the cutting-edge visualization practices and to meet the domain experts' requirements. The first and the second case studies (Aerodynamics Visualization and LiDAR 3D Point Cloud) provided us with insights to design a detailed workflow for scalable scientific visualization, which we

present in Section 5. Our final case study demonstrates the application of the suggested workflow to a novel domain with unique datasets (Galaxy catalogs visualization).

2 AERODYNAMICS VISUALIZATION

Urness (Urness 2003) proposed a new method called 'Color weaving', which uses Hue and Saturation, to allow multiple colors to be closely interwoven (instead of being blended) by the assignment of distinct separate hues to individual streamlines. They varied the Saturation of the color at each point according to the value in the corresponding scalar distribution, which allowed each color to encode multiple values in a continuous distribution. Their method assigned color indices to streamlines in an alternating manner that depends on the order in which they are encountered in a deterministic walk through the pixel grid. The result is a multicolored line integral convolution (LIC) image that resembles a tapestry woven with different colored threads. The authors also defined a 'Texture stitching' technique, which allowed faithful preservation of region boundaries in multi-frequency LIC through the use of a post-LIC merging of selected adjacent regions. They obtained separate LIC textures based on correlated high and low frequency noise input patterns, then combined the results using a binary mask to force adherence to pre-defined boundary curves (Urness 2003) and (Hagh-Shenas et al. 2007). Our method is similar to the color weaving method with one difference: that we have categorical data assigned to the hues and quantitative data assigned to Saturation.

2.1 Dataset Description

This visualization was built for HVAC system using different graphical representations; mainly colors and glyphs. The model has different time steps for a 3D point data set. The data set attributes are describing temperature, humidity and air flow velocity.

This data set has four dimensions(independent variables), the space forms three dimensions (x, y, z) and time is the fourth dimension. In addition, the data set attributes are considered as the dependent variables which are temperature, humidity and air flow velocity.

2.2 Processing and Rendering

The velocity vector was calculated from the scalars (V_x , V_y , and V_z) to represent the airflow as a stream tracer, and then a tube filter was applied to enhance the aesthetic streamline representation of the airflow. An arrow glyph was used to represent the direction of the flow.

The temperature values were mapped to the tubes' color that transit from cool to hot color map. The humidity values were represented as colored wireframe of the room boundaries that vary between a gradient of two colors. The scene was designed using ParaView, in which each variable was assigned glyphs and colors according to the suggested representation to embrace the role of all variables in the scene and the aesthetic representation. That was carried out by adjusting ParaView' color transfer functions to represent the

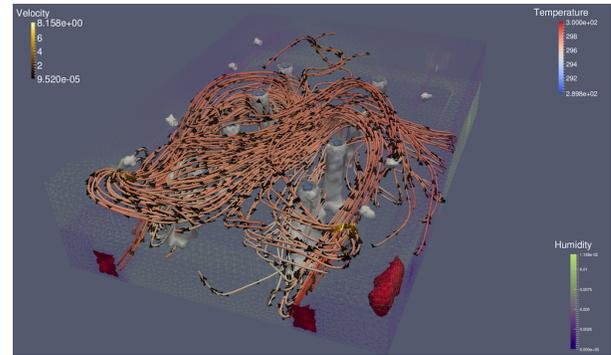


Figure 1: One time step scene for the HVAC system.

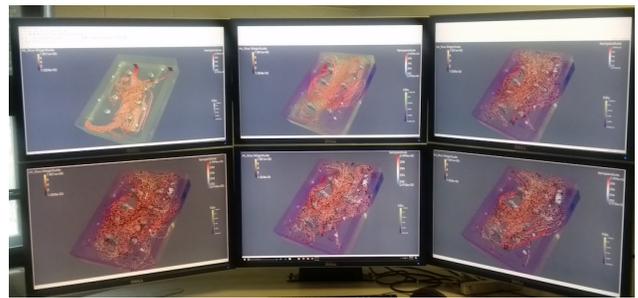


Figure 2: HVAC system: Six time steps are displayed simultaneously on tiled display

variable data type as quantitative, ordinal or qualitative/categorical.

2.3 Output modality and display

As a result, 3-dimensions 3-variables data set were encoded in one scene with less occlusion and less crowding, figure 1. Each scene is representing one time step of the simulation, where time is the fourth dimension in this data set. The most prominent way to represent the time here is through producing animation for the entire simulation. This data set is a time series of 250 time step, a movie of 250 time frames was produced. But to effectively visually discriminate between two or more time steps, the animation was not the candidate visualization.

To overcome that challenge of enabling the analysts to visually discriminate between time steps, the juxtaposition (side-by-side) approach was applied to different scenes of the time series. Using the linked views feature in ParaView, scenes of six time steps were displayed as a 2x3 matrix on six HD mounted displays as in (Fig.2). We linked the cameras across all time steps so that user navigation would be identical across any view the user explored.

Immersive collaborative environments provide analysts with more visual data exploration tasks and enhanced visual discrimination than the desktop displays especially for MDMV visualization, (Keefe and Isenberg 2013). An X3D scene was generated for different time steps in the data set, with an adjusted camera view to be properly displayed in a CAVE environment. The immersion enabled analysts to get more insights and identify numerous distinctive values in

the data set attributes. The only interaction with the scene in this case was changing views/navigating in the 3D space.

3 LIDAR 3D POINT CLOUD

3.1 Dataset description

Light Detection and Ranging or Laser Imaging, Detection and Ranging (LiDAR) is a remote-sensing technique that uses laser light to sample the surface of the earth, generating highly dense accurate x,y,z georeferenced data set. LiDAR, primarily used in airborne laser mapping applications, is cost-effective compared to traditional surveying techniques such as photogrammetry. There are numerous fields that use LiDAR data such as agriculture, urban planning, forestry, 3D archeological reconstruction, hydrologic modeling, terrain analysis, and infrastructure design. Some hundred megabytes (several millions of points) sizes clouds can be feasibly visualized using standard tools on a single powerful machine. However, scans of even small environments can easily lead to massive size datasets consisting of billions of points. Rendering these massive datasets with standard visualization tools often exceed the machine's main and graphics memory. In addition, such big data needs to be stored on a central server and can only be accessed remotely from client machines (de Haan 2009).

For bench marking we used a massive LiDAR scan of five billion 3D point cloud for a building that contains motion sensors in different locations of the building. The main purpose of the visualization is to detect the location of these motion sensors across the building by interactively navigating the dataset using an immersive CAVE environment.

3.2 Methodology

In this case study we developed a High Performance Visualization Pipeline (HPVP) using Python and ParaView to render, visualize and interactively navigate massive LiDAR data sets. The HPVP consists of four main stages: a) unstructured points flat file preprocessing, b) unstructured grid partitioning, c) remote rendering, and d) navigating the data in the immersive environment. The pipeline was developed using parallel Python scripts and ParaView that can run in server-client mode. We converted the LiDAR flat files to a ParaView readable VTK file format using Parallel Python.

To enable ParaView parallel rendering, we applied the D3 filter that partitions the unstructured VTK files to smaller files. The number of partitions generated equal to the number of processors that run the partitioning algorithm. The partitioned files were remotely rendered to the client ParaView session by running the ParaView pvserver on the server. Different ParaView filters were applied on the data set to enable the users to locate the sensor patterns in the building. The last stage of the pipeline is navigating the LiDAR point cloud in the CAVE by generating a 3D scene or by using the immersive ParaView mode.

3.2.1 Preprocessing. Preprocessing a point cloud of five billion points (270 GB) with intensity information stored as RGB values requires parallelism enabled using high-performance computing resources. We requested to obtain the dataset of each floor of the building separately (five floors) which will ease validation of the building visualization. The flat files contain a record of a 3D point

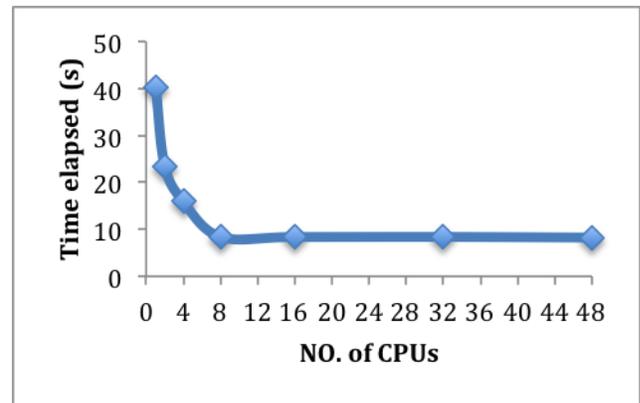


Figure 3: Flat files to VTK format conversion using Parallel Python

per line, and a count of points each other n lines. The points counter messes up the file format, so it should be excluded from all files. The cleaning process was carried out using 2 of our cluster's nodes with 2 Intel Xeon E5-2680v3 2.5GHz (Haswell), 24 cores, and 128 GB memory per node. With a recorded average completion time of 1554.8s (25.91 m) for a flat file with an average of 762,572,939 points and an average size of 38 GB. After finishing the cleaning task, the dataset was divided among the computing nodes to allow parallelism using a custom Python script which also converted the flat file format to a VTK format.

For the sake of benchmarking, the completion times of the file format conversion task using a varying number of cores is reported here to convert a 17,852,062 points file to a VTK file. Figure 3 shows scalability of the conversion process using our cluster's computing nodes. The performance increases with the no. of processors but it plateaus after the number of processors exceeds 8.

3.2.2 File Partitioning. A group of five VTK files that contain 17,852,062 points are still too big to be loaded and rendered using one processor's memory using Paraview. So, in order to run Paraview in parallel mode, the D3 filter is applied (Moreland 2013). D3 filter re-partitions a data set into load-balanced spatially convex regions as an unstructured grid that can be loaded and rendered in parallel from the server to the client machine.

3.2.3 Remote Rendering. The data set was rendered to a client machine using two computing nodes with 24 CPUs and 2 GPUs (Nvidia Tesla K40m or K80) each to scale up the performance. Using Paraview pvserver which is a parallel MPI program that handles data processing and rendering in the same parallel job (Squillacote and Ahrens 2007). The data set was represented as points colored by their intensity values over a gray scale color map. Figure 4 shows the chessboard pattern labeled by red circles which is the pattern that represents the location of the motion sensors in the building.

3.2.4 Navigation in the CAVE. The CAVE has a machine contains Intel Xeon E5-2680 v3 @ 2.5GHz with 24 multi-threaded cores and 512 GB memory. The machine streams the graphics through projectors to three 10' by 10' walls and a floor of stereo projection screens with a resolution of 26,214,400 stereo px. the CAVE has



Figure 4: Remote rendering using Paraview pvserver; red circles indicate fiducial targets



Figure 5: Users navigating the point cloud rendered from the server in the CAVE.

InterSense IS900 driving a head and a wand as a tracking system. It provides the strongest sense of immersion, allowing the user to freely walk around models in their virtual world.

Immersive ParaView CAVE mode is used in the CAVE along with Virtual-Reality Peripheral Network (VRPN), which is a library that implements a network-transparent interface between application programs and the set of physical devices such as trackers used in a virtual-reality (VR) system (Hansen 2012). With that CAVE configuration, we were able to load and navigate a subset of around 20 million points. This immersive setup allows multiple users to navigate the same data set in the same space with a single view interaction, figure 5. ParaView also allow multiple client machines to connect to one server, which enables users' remote collaboration to develop a shared scene.

3.3 Exploration and evaluation

The main aim of our study was to find the locations of the sensors distributed across the five levels of the building. our visualization paradigm showed a scalable performance when was applied to the

given LiDAR point cloud using HPC resources and a parallel processing.

We were able to render 1 billion points in one remote interactive session of pvserver using 24 CPUs and 2 GPUs to a desktop display. We could navigate the 3D view and change the points representation and color map, which were point and gray scale color map respectively. In addition, We were able to generate a 3D scene of the LiDAR point cloud to enable the scientist to navigate the building and look for the pattern that represents the sensors in the point cloud. The navigation was carried out in a CAVE using a wand and a head tracker to walk through the model and zoom to a certain area of interest to find the patterns of interest.

4 VISUALIZATION PARADIGM DESIGN

Herein, we are trying to formalize a description of a generalized scientific visualization paradigm (a workflow based on the practical experience we gained from developing visualization for the aforementioned cases). The proposed paradigm depends mainly on the dataset nature and the scientist's purpose of the visualization as shown in figure(6). The governing equation for the whole process can be described as follows:

$$f(N, T) \xrightarrow{C} V$$

Where f represents the visualization development and V is the resulting visualization,

$$C = \{P \cup GA \cup R \cup D\}$$

which describes the computational and rendering configurations needed to develop V . Both $f(N,T)$ and C are using the scientific computing and visualization metrics as parameters:

- (1) $N: N \in \mathbb{Z}_{\geq 2}^+$
- (2) $T: (data)Type ["scalar","vector"]$
- (3) $P: Processing ["serial", "parallel", "local", "in situ", "remote"]$
- (4) $GA: computation of derived Geometric and Appearance properties ["streamlines", "glyphs", "contours", "slices", "colormaps", "texturemaps", "..."]$
- (5) $R: Rendering ["still scenes", "animations", "3D worlds", "auditory", "haptic", "..."]$
- (6) $D: Display ["handheld","desktop", "tiled displays", "immersive", "..."]$

Number of dimensions and variables (N) represents how many independent and dependent variables are included in the visualization. This number can be any integer number greater than 2 represents at least one dimension and one variable (attribute). For example, plotting displacement values across time, the dimension here is the time and the variable is the displacement. Another example, is a multivariate 2D maps which have two dimensions (x and y coordinates) and m variables (such as soil types, slope, water availability, and solar aspect). But most of simulation generated datasets are considered as 4D datasets, where x , y , and z represent the spatial coordinates and the fourth dimension is the time. Usually, those datasets are associated with more than one variable that change with the spatial and temporal dimensions. As the airflow

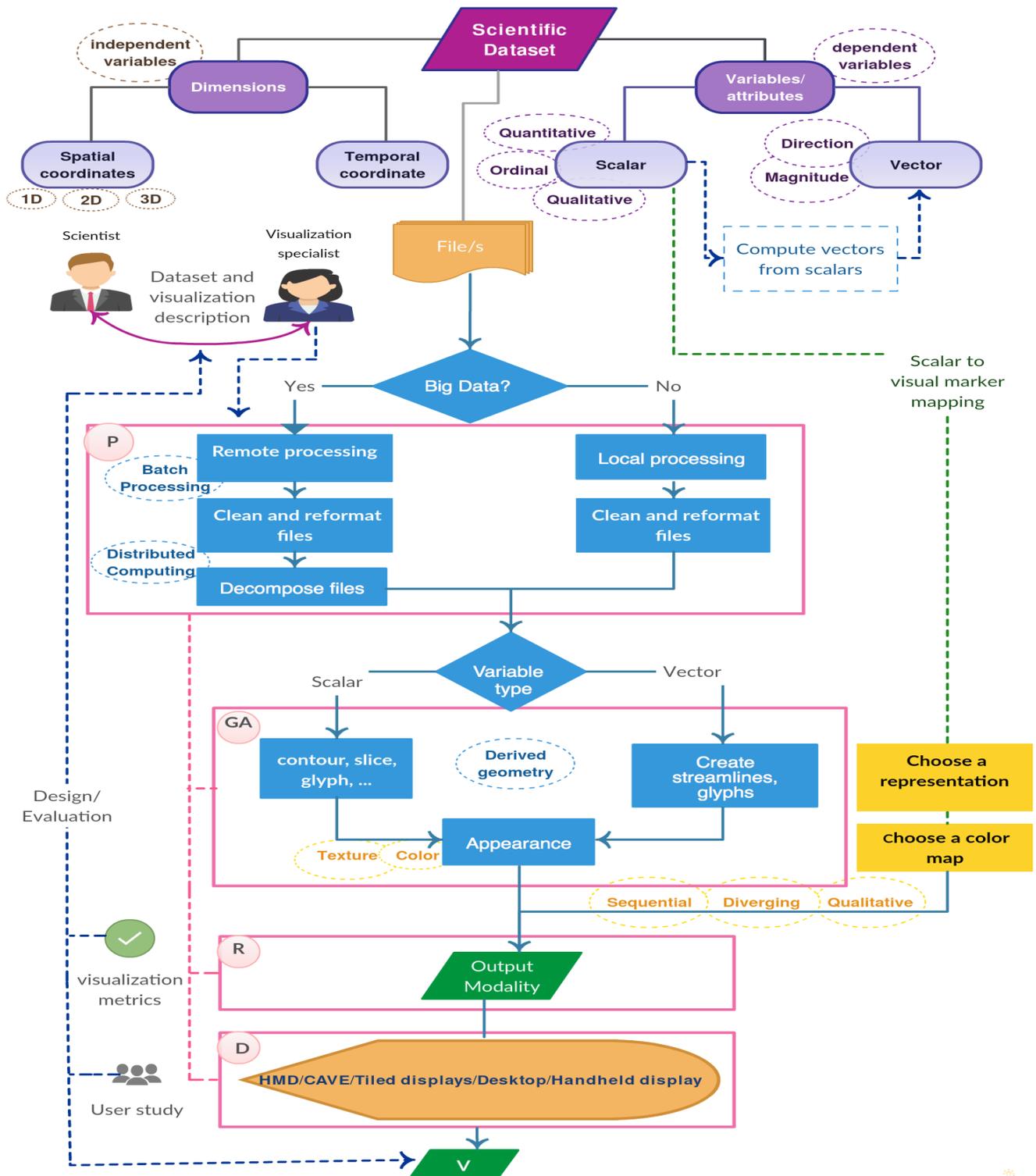


Figure 6: Proposed Scientific visualization paradigm

simulation was discussed earlier in this paper which has 4D and 5 variables.

Dataset variables' Types, (T), has two possible values: scalar or vector. Scalars can represent quantitative, ordinal, or categorical variables, this will affect the selection of the visual component these variables will be mapped to. For example if the color is chosen to represent a quantitative variable then the color map should be designed to represent a continuous range of color such as a gradient of one color or a gradient between two colors. The dataset might contain vector variables (also c can be derived from the scalars) which need to be represented as stream lines to show the direction of the vector, and another visual marker is needed to show its magnitude such as size or color.

Different Scientists have different requirements of their visualizations, depending on their research purpose. There are several scientific visualization tools that can be used to generate the desired visualization with a minimal amount of time and effort, such as Paraview, Meshlab, Qgis, and VMD. Some of these visualization tools are general purpose visualization tools like Paraview, and the others are specific for a certain domain; for example, VMD is for molecular dynamics, and Qgis is for geospatial data visualization. A plethora of open source visualization applications such as Paraview provide a Python interface that enables users to customize their own functions and add new plugins to the tool (Hansen 2012; Henderson et al. 2004). Also, many other toolkits and libraries for GIS data, such as the Geospatial Data Abstraction Library (GDAL) (project 2016) and LAsTools (rapidlasso 2016) come with Python interfaces, this enables fast integration and linking of general 3D visualization applications with GIS specific data formats. We used Python language as suggested by (de Haan 2009; van Oosterom et al. 2015) for data preprocessing and decomposition to enable remote rendering Paraview parallel mode in particular.

Multiprocessing modules were developed for Python to enable true parallel processing with Python on a multiprocessor machine or across machines (Matloff 2011).

We used the Parallel Python (PP) module that provides a simple way to incorporate parallelism into python applications. Internally ppsmp uses processes and IPC (Inter Process Communications) to organize parallel computations. The applications written with PP work in parallel even on many computers connected via local network or Internet. Cross-platform portability and dynamic load balancing allows PP to parallelize computations efficiently even on heterogeneous and multi-platform clusters (Vanovschi 2017).

According to the dataset size and the required visualization design, the Preprocessing (P) is to be determined as serial or parallel and either local, in situ, or remote rendering. (GA) is the computing required for the derivation of derived geometries such as streamlines and glyphs in the case of vector data, and contours, slices and glyphs for scalar fields and their appearances (such as color and texture). For the Rendering output (R) and Display type (D) describe computational components to generate the the visualization output and where it should be shown.

For example, still scenes (such as 2D images and 3D scenes), auditory and haptic feedback can be displayed on a handheld, desktop display, a tiled display, HMD or in a CAVE. Alternatively, an animation can be generated to show the time series of the visualized

dataset or even to show different camera views. Different combinations of P, GA, R, and D are the arguments of the Computing C that is used by $f(N,T)$ to generate the visualization V. The exploration and evaluation of the visualization should be done by the scientist and the visualization specialist. Also the evaluation can be obtained via conducting user study or by calculating visualization metrics such as effectiveness and expressiveness of the visualization as in, (Halim and Muhammad 2017).

5 GALAXY CATALOGS VISUALIZATION

Over the last decade, astronomy has become extraordinarily data intensive. For example, in 2004 the Hubble Space Telescope archive spanned 18Tb while the Sloan Digital Sky Survey (SDSS) archive was of order 50Tb. In comparison, current data archives dwarf these - the SDSS archive now spans several hundred terabytes, and the IRSA facility hosts just under a petabyte. Furthermore, near-future facilities such as the Large Synoptic Survey Telescope (LSST), the Wide-Field Infrared Survey Telescope (WFIRST), LO-FAR, SkyMapper, and the Square Kilometer Array will produce daily data volumes of over a terabyte, and each require data repositories that can host several petabytes. As a result, it is now as much of a challenge to make effective use of astronomy data as it is to acquire it. This increase in data archive size, and the commensurate increase in data volumes used in astronomy investigations, accompanies and motivates a paradigm shift in how astronomers visualize and interact with data. Astronomy is a visual subject, and so images and diagrams are used extensively to conceptualize ideas and explain results. However, high-dimensional data visualization is a fundamental, enabling technology for knowledge discovery. In astronomy, visualization has seen surprisingly limited use as an astronomical research tool. Visualization is used in theoretical astrophysics to study cosmological n-body simulations. Such visualizations have provided striking, rapid insights into the simulations themselves. Other uses include (by a few authors) to make plots interactive (Cioc et al. 2013). However, for observational astronomy, visualization has barely been used to explore complex astronomical datasets. In this section, we show the application of the suggested workflow to astronomical (galaxy) catalogs.

5.1 Dataset description

We consider three astronomical catalogs, of increasing size and complexity. The first is a catalog of 166,583 galaxies, or "quasars", obtained from the Sloan Digital Sky Survey. The quasars are found across ten thousands degrees of the sky, and all have known distances, via their redshifts. In addition to position in space, the catalog also includes 139 columns of meta-data for each quasar, including information such as their luminosities, masses, and kinematic properties, all of which are of interest to astronomers working on different projects that use catalogs of this type, (Pâris et al. 2012).

The second is a catalog of 1,000,000 galaxies that were found via infrared imaging observations. The galaxies in this catalog also have eighty columns of meta-data, but are found in six widely separated fields on the sky, rather than one large field. Moreover, while the positions of the galaxies on the sky are known, their distances, via their redshifts, have significant associated uncertainties. The

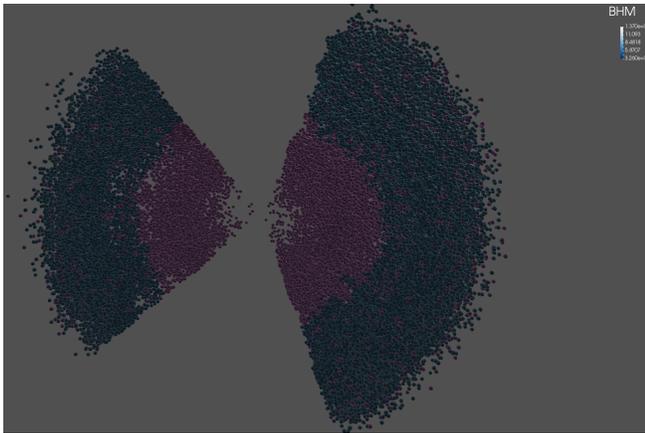


Figure 7: 166,583 galaxies visualization

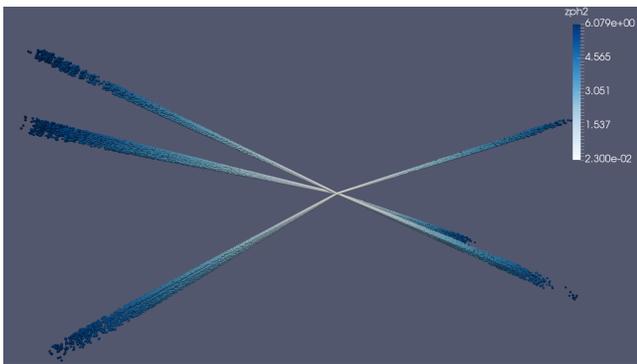


Figure 8: 1,000,000 galaxies visualization

visualization challenges with these data are thus to encode information on each source alongside a representation of the uncertainties on their distances, which are of fundamental importance in using them in a research setting as in figure 8, (Rowan-Robinson et al. 2013).

The third is a catalog of approximately 1,200,000,000 stars that were observed by the Gaia mission. This catalog contains a measure of each star's position on the sky as well as their distance, encoded via parallax. There is less meta-data per source for this catalog, but the challenge is to show the spatial distribution of all the stars together in a visualization, whilst being able to quickly switch from a "birds-eye" view that encompasses the whole catalog, to a view that includes only a few hundred stars, at any point in the stellar distribution, (AGENCY 2016).

The dimensionality of these data sets is 3D and the number of variables are varying but we started with the variables that are needed to calculate the coordinates of each galaxy and other derived quantities such as Black Hole Mass (BHM). There were five sets of equations to calculate the coordinates, therefore, including the other galaxy attributes we have 22 variables of these data set to be visualized. Each astronomical catalog was in a different format that is not native to most of the generic scientific visualization tools,

so we processed the data files and calculated the desired attributes and generated csv files.

5.2 Processing and Rendering

we applied the proposed paradigm to the astronomical catalogs, in which, files contain records for each galaxy per line, we generated Python scripts to read variables and compute the derived quantities. To enable parallel rendering we converted the file to vtu files that can be read by Paraview. The conversion script was carried out using 2 of our cluster's nodes with 2 Intel Xeon E5-2680v3 2.5GHz (Haswell), 24 cores, and 128 GB memory per node. With a recorded average completion time of 4.3s for a flat file with an average of 1000,000 points and an average size of 222 MB. Through an interactive Paraview session using pvserver with 24 CPUs and 2 GPUs, the data set was loaded and rendered remotely. We decided to apply a sphere glyph to each point to represent the galaxy luminosity "iMag" and colored by the BHM. The representation is repeated for every derived coordinates.

5.3 Output modality and display

The output was generated as a 3D scene as well as an animation, the 3D scene was navigable in the CAVE. Also the dataset was loaded and rendered using Paraview CAVE mode, which enabled visualization development and navigation in the same time. The scientist was able to watch the animation on his own desktop display and navigate his dataset in the CAVE as well. He gave a positive feedback about the visualization and proposed other variables to be added to the visualization and admitted the value that this visualization will add to the exploration of the astronomical catalogs.

6 CONCLUSION AND FUTURE WORK

Through our practical work with scientists across several domains, we have distilled a generalized workflow for scientific visualization that includes HPC/HPV considerations. This workflow describes the components of visual mapping and computation based on the volume (size) and variety (types) of data in a dataset. Our experience has shown that this workflow can be useful across several kinds of multi-dimensional and multi-variate data and is not limited to one format or modality (i.e. images, videos, 3D scenes).

We have demonstrated how our workflow can be used across a variety of display modalities and platforms. The display platform and user interface can be a significant factor in the insights and experience of a user. We demonstrated the use of tiled displays to show timseries as well as the use of immersive projection to examine geometric structures of large datasets. Finally, we successfully applied our paradigm to a novel dataset and domain (galaxy catalogs).

Our workflow abstraction also shows highly potent areas for future research- especially to improve the efficiency of the modulating factors in C (computing). For example, in the case of data pre-processing, we might consider new sampling and statistical strategies. In the case of the interactive rendering paradigm, we might consider new parallel algorithms for deriving geometry such as streamlines on distributed dataset, or new load balancing and

visual quality assurance techniques for cluster rendering and collaboration.

Our workflow gives visualization designers and developers specific issues and decisions to consider when creating visual mappings. This contextualizes and supports much of the prior work in scientific visualization techniques toward improving the perceptual power of a representation. As well, prior work on the scalability of High-Performance Visualization (HPV) workloads and performance remains of central importance. Indeed, as our formulation shows, the central challenge of scientific visualization is to maximize the perceptual and cognitive value of a visualization while minimizing its computational cost.

REFERENCES

- EUROPEAN SPACE AGENCY. 2016. GAIA DATA RELEASE 1 GAIA DR1. (2016). <https://www.cosmos.esa.int/web/gaia/dr1>
- S. K. Card, J. D. Mackinlay, and B. 1999. Shneiderman. 1999. *Readings in information visualization: using vision to think*. Morgan Kaufmann.
- Alexandru Cioc, SG Djorgovski, C Donalek, E Lawler, F Sauer, and G Longo. 2013. Data visualization using immersive virtual reality tools. In *American Astronomical Society Meeting Abstracts# 221*, Vol. 221.
- Gerwin de Haan. 2009. Scalable visualization of massive point clouds. *Nederlandse Commissie voor Geodesie KNAW 49* (2009), 59.
- H. Hagh-Shenas, S. Kim, V. Interrante, and C. Healey. 2007. Weaving versus blending: a quantitative assessment of the information carrying capacities of two alternative methods for conveying multivariate data with color. *IEEE Transactions on Visualization and Computer Graphics*. 13, 6 (2007), 2007.
- Zahid Halim and Tufail Muhammad. 2017. Quantifying and optimizing visualization: An evolutionary computing-based approach. *Information Sciences* 385 (2017), 284–313.
- Charles Hansen. 2012. *High Performance Visualization: Enabling Extreme-Scale Scientific Insight*. Chapman and Hall/CRC.
- Amy Henderson, Jim Ahrens, Charles Law, and others. 2004. *The ParaView Guide*. Kitware Clifton Park, NY.
- Daniel F Keefe and Tobias Isenberg. 2013. Reimagining the scientific visualization interaction paradigm. *Computer* 46, 5 (2013), 51–57.
- Norm Matloff. 2011. Programming on parallel machines. *University of California, Davis* (2011).
- Kenneth Moreland. 2013. The paraview tutorial. *Sandia National Laboratories, Tech. Rep. SAND* (2013).
- I. Páris, P. Petitjean, É. Aubourg, S. Bailey, N. P. Ross, A. D. Myers, M. A. Strauss, S. F. Anderson, E. Arnau, J. Bautista, D. Bizyaev, A. S. Bolton, J. Bovy, W. N. Brandt, H. Brewington, J. R. Browstein, N. Busca, D. Capellupo, W. Carithers, R. A. C. Croft, K. Dawson, T. Delubac, G. Ebelke, D. J. Eisenstein, P. Engelke, X. Fan, N. Filiz Ak, H. Finley, A. Font-Ribera, J. Ge, R. R. Gibson, P. B. Hall, F. Hamann, J. F. Hennawi, S. Ho, D. W. Hogg, Ž. Ivezić, L. Jiang, A. E. Kimball, D. Kirkby, J. A. Kirkpatrick, K.-G. Lee, J.-M. Le Goff, B. Lundgren, C. L. MacLeod, E. Malanushenko, V. Malanushenko, C. Maraston, I. D. McGreer, R. G. McMahon, J. Miralda-Escudé, D. Muna, P. Noterdaeme, D. Oravetz, N. Palanque-Delabrouille, K. Pan, I. Perez-Fournon, M. M. Pieri, G. T. Richards, E. Rollinde, E. S. Sheldon, D. J. Chlegel, D. P. Schneider, A. Slosar, A. Shelden, Y. Shen, A. Simmons, S. Snedden, N. Suzuki, J. Tinker, M. Viel, B. A. Weaver, D. H. Weinberg, M. White, W. M. Wood-Vasey, and C. Yèche. 2012. The Sloan Digital Sky Survey quasar catalog: ninth data release. *aap* 548 (2012). DOI: <http://dx.doi.org/10.1051/0004-6361/201220142>
- OsGeo project. 2016. GDAL Geospatial Data Abstraction Library. (2016). <http://www.gdal.org/>
- rapidlasso. 2016. LAStools software for rapid LiDAR processing. (2016). <http://www.cs.unc.edu/~isenburg/lastools/>
- Michael Rowan-Robinson, Eduardo Gonzalez-Solares, Mattia Vaccari, and Lucia Marchetti. 2013. Revised SWIRE photometric redshifts. *Monthly Notices of the Royal Astronomical Society* 428, 3 (2013), 1958–1967.
- Amy Henderson Squillacote and James Ahrens. 2007. *The paraview guide*. Vol. 366. Kitware.
- Timothy Urness. 2003. et al. 2003. In *Effectively visualizing multi-valued flow data using color and texture*. Proceedings of the 14th IEEE Visualization 2003 (VIS'03). IEEE Computer Society.
- Peter van Oosterom, Oscar Martinez-Rubi, Milena Ivanova, Mike Horhammer, Daniel Geringer, Siva Ravada, Theo Tijssen, Martin Kodde, and Romulo Gonçalves. 2015. Massive point cloud data management: Design, implementation and execution of a point cloud benchmark. *Computers & Graphics* 49 (2015), 92–125.
- Jarke J Van Wijk. 2006. Views on visualization. *IEEE transactions on visualization and computer graphics* 12, 4 (2006), 421–432.
- Vitalii Vanovschi. 2017. Parallel Python software. (2017). <http://www.parallelpython.com>
- C Ware. 2000. *Information Visualization: Perception for Design* Morgan Kaufmann. New York (2000).
- Colin Ware. 2013. *Information visualization: perception for design*. Elsevier.