Vapor 3

A modern 3D visualization suite for Geophysical Sciences

Scott Pearse
Computational Information and
Science Laboratory
National Center for Atmospheric
Research
Boulder, CO USA
pearse@ucar.edu

ABSTRACT

Vapor v3.1 is an open-source, cross platform visualization and analysis tool for finite element simulations in the physical sciences. Vapor offers users a set of tools called Renderers that let them depict their simulation data in different ways.

With this most recent release, Vapor has undergone a significant refactorization of its code base which has spanned a 15 year history. Much of Vapor's old code has been discarded and rewritten to improve its ease of use, maintainability, and rendering quality. These improvements aim to help scientists easily demonstrate their simulations in a way that results in superior communication of their research.

The video pertaining to this submission will demonstrate how Vapor is used, new features in the current version, and how things have changed since the previous major release, Vapor 2.6.

KEYWORDS

Visualization, Geophysics, Open Source, WRF, NetCDF, MPAS

1 Supported Data

The first step in visualizing a dataset in Vapor is to make sure that the user's data is in agreement with Vapor's requirements. Vapor 3.1 can currently read the following four data formats:

1) WRF-ARW
2) CF-Compliant NetCDF
3) MPAS
4) VDC (Vapor Data Collection)

Previously in Vapor2.X, users could convert generic NetCDF data into a Vapor-readable format as long as the data existed on a structured grid. However this process of data conversion was often a confusing exercise due to the multitude of different ways NetCDF data could be described. The NetCDF Climate and Forecast (CF) Conventions are a standard that is gaining acceptance in geoscientific and medical fields. Vapor 3.1's users will be able to load data with little effort compared to what was needed in Vapor 2.X, as long as their data contains a hand full of fields specified in the CF Conventions.

As with Vapor 2.X, Vapor 3.1 supports the wavelet based VDC data format. As simulations continue to grow in scale, the resultant data grows in scale as well. The process of rendering these data sets can take a frustratingly long time. The VDC format can improve a users' interaction with Vapor by allowing the option to render a compressed version of their data set. This provides faster rendering times, and a more responsive application. Once a user has found a perspective and color map for their rendering that they like, they can then sample the data at its true resolution for a final rendering.

Vapor 2.X's code base had the disadvantage that it was written over a long period of time, without all requirements being known up front. With Vapor 3.1, developers have had a chance to take a step back and reassess the 15 years of development that took place with Vapor 2.X for refactorization. One of the most important components that's been refactored is the data model. The new implementation allows users to load multiple datasets and render them together. For example, users can now load and visualize data from a CF-Compliant NetCDF ocean model and an atmospheric WRF-ARW model simultaneously.

2 New Graphical User Interface

Another component that has been refactored in Vapor 3.1 is the user interface. Two significant changes have been made to improve the ease of use of Vapor 3.

The first improvement was to divide all Renderer controls into four panels that will be consistently presented among all Renderers. These four panels are:

1) Variables - Where the user will control variable selection and compression parameters (if using VDC)

DOI: 10.1145/3332186.3341299

2) Appearance - Where users will control the color and opacity of their renderer
3) Geometry - Where users will define the region the current renderer should draw within, as well as transforms like scaling, translation, and rotation for that renderer
4) Annotation - Where users can apply a color bar

The second improvement to the GUI is a way to only present renderers that are relevant to what the user needs. In Vapor 2.X, each Renderer was given a top-level tab that the user would need to select in order to enable and change that Renderer's settings. With 12 tabs in total, the interface became cumbersome to use. In Vapor 3.1, when users create instances of their Renderer, that instance gets added to a list that's displayed in the top left corner of the screen. When the user clicks on different instances they've created, the GUI will update according to that Renderer's parameters. Nothing gets displayed outside of what the user has explicitly created.

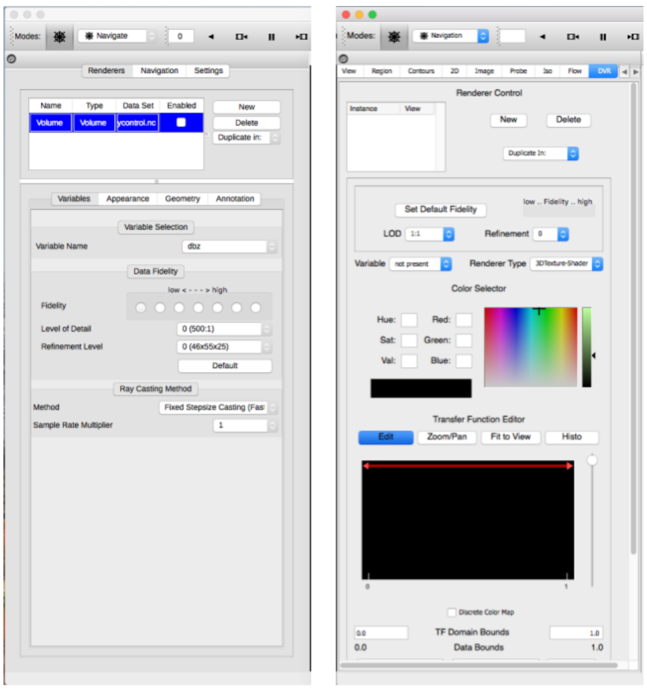

Figure 1: Vapor 3.1's GUI for the Volume Renderer (Left) compared to the legacy Volume Renderer GUI from Vapor 2.6 (Right)

3 Ray Casting Renderers

Vapor 3.1 has two new ray casters for Volume Rendering and Isosurfaces. Our current tests on ray casting rectilinear grids have shown that compute times have improved by a factor of 10x, and have a more accurate visual appearance. New colormap interpolation schemes have improved visualization quality among all Renderers as well.

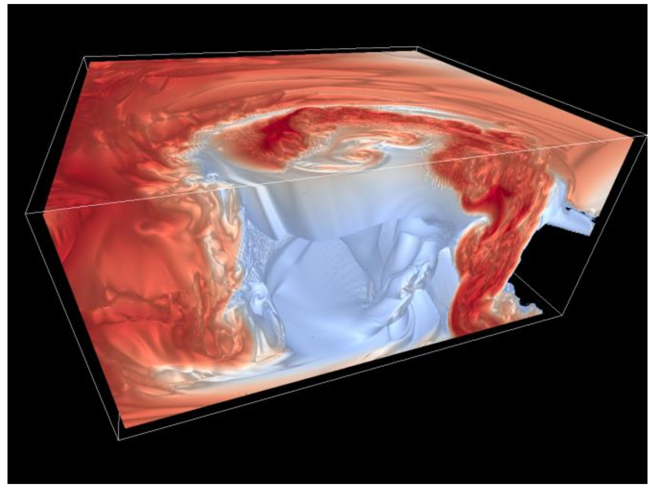

Figure 2: A volume rendering of an F5 tornado simulation, using Vapor 3.1's new ray caster

4 Python Calculation Engine

Vapor's Python Engine is a powerful tool that lets users calculate new variables that have been derived from the native values within their datasets. The NumPy and SciPy modules have been included for convenience. Users can access this tool through the Tools->Python Engine menu. From there variables like wind speed can be calculated from a script.

5 Future Development and Milestones

VAPOR3 Development Roadmap			
Capability	2019 V3.1	2019 V3.2	Future V3.x
VAPOR Version 2 enter EOL	✓		
Direct Volume Rendering (regular grids)*	✓		
Isosurfaces (regular grids)*	✓		
Statistics*	✓		
Line plotting*	✓		
Contour lines*	✓		
2D data*	✓		
Time animation*	✓		
Geo-referenced images*	✓		
Unstructured grids (2D)	✓		
Multi-dataset visualization	✓		
Python calculation engine*	✓		
Direct Volume Rendering (unstructured grids)		✓	
Isosurfaces (unstructured grids)		✓	
Flow visualization*		✓	
Key frame animation*		✓	
OpenGL 4.x support		✓	
Python scripting interface		✓	
Accelerator support		✓	

Parallel (MPI) VDC API		✓	
Parallel (MPI) rendering backend			✓
Progressive data access / compression (unstructured grids)			✓
Advanced wavelet encoders and/or alternate compressors			✓
Web interface			✓

ACKNOWLEDGMENTS

VAPOR is a product of the National Center for Atmospheric Research's Computational and Information Systems Lab. Support for VAPOR is provided by the U.S. National Science Foundation (grants # 03-25934 and 09-06379, ACI-14-40412), and by the Korea Institute of Science and Technology Information