Scientific Visualization with ParaView

Giuseppe Di Bernardo & Markus Rampp

Max Planck Computing and Data Facility (MPCDF)

giuseppe.di-bernardo@mpcdf.mpg.de
markus.rampp@mpcdf.mpg.de

June 13, 2017

ParaView
Parallel Visualization Application

MPCDF Max Planck Computing & Data Facility

Overview

MPCDF Max Planck Computing & Data Facility

What is ParaView?

- ParaView is an **open-source application** and **architecture display** and analysis of scientific datasets;
- ParaView has been demonstrated to process billions of unstructured cells and to process over a trillion structured cells;
- ParaView's parallel framework has run on over 100,000 processing cores;
- ParaView's **key features** are:
  - An open-source scalable, multi-platform for visualizing **2D/3D data** (excels at traditional scientific vis qualitative 3D rendering);
  - An extensible, modular **architecture** based on open standards

    e.g. Custom apps, plugins, `Python` scripting, `ParaViewWeb`, `Catalyst`
  - Support for distributed computation models to process **large data sets** (from notebooks to world's largest supercomputers);
  - An open, extensible, and **intuitive user interface**;

MPCDF Max Planck Computing & Data Facility

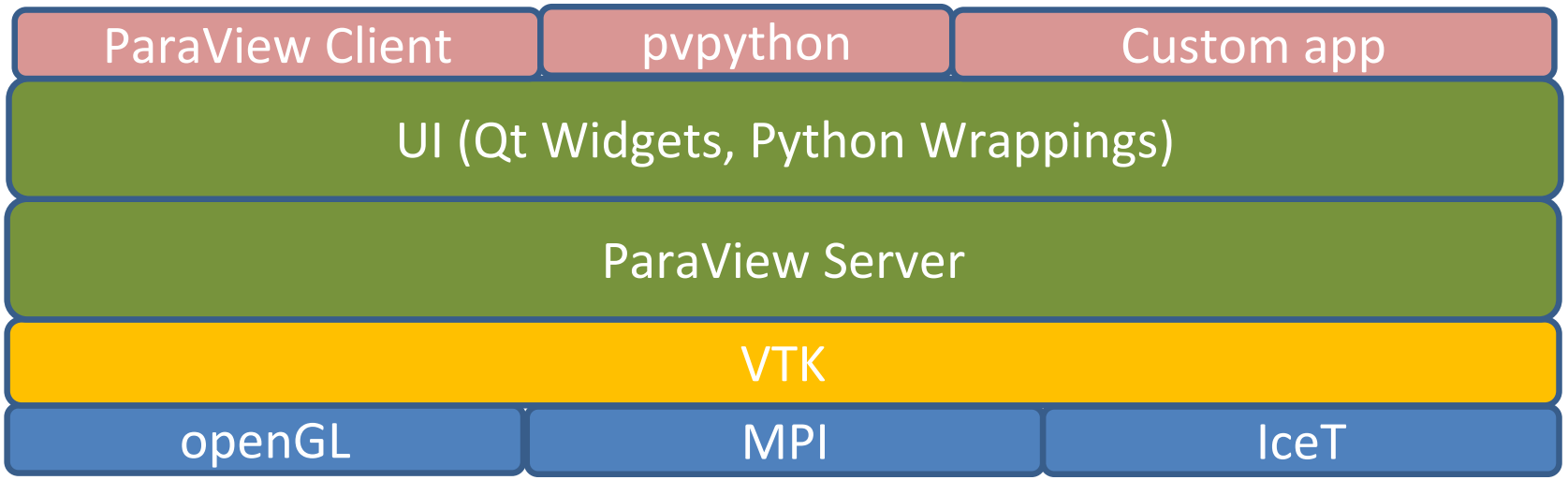
ParaView: a standard de-facto

- ParaView is used by many academic, government, and commercial institutions all over the world;
- ParaView is downloaded roughly 100,000 times every year;
- ParaView also won the HPCwire Readers' Choice Award and HPCwire Editors' Choice Award for Best HPC Visualization Product or Technology

MPCDF Max Planck Computing & Data Facility

ParaView: The architecture

The application most people associate with ParaView is really just a small client application built on top of a tall stack of libraries that provide ParaView with its functionality.



MPCDF Max Planck Computing & Data Facility

ParaView: The big picture

The application most people associate with ParaView is really just a small client application built on top of a tall stack of libraries that provide ParaView with its functionality.

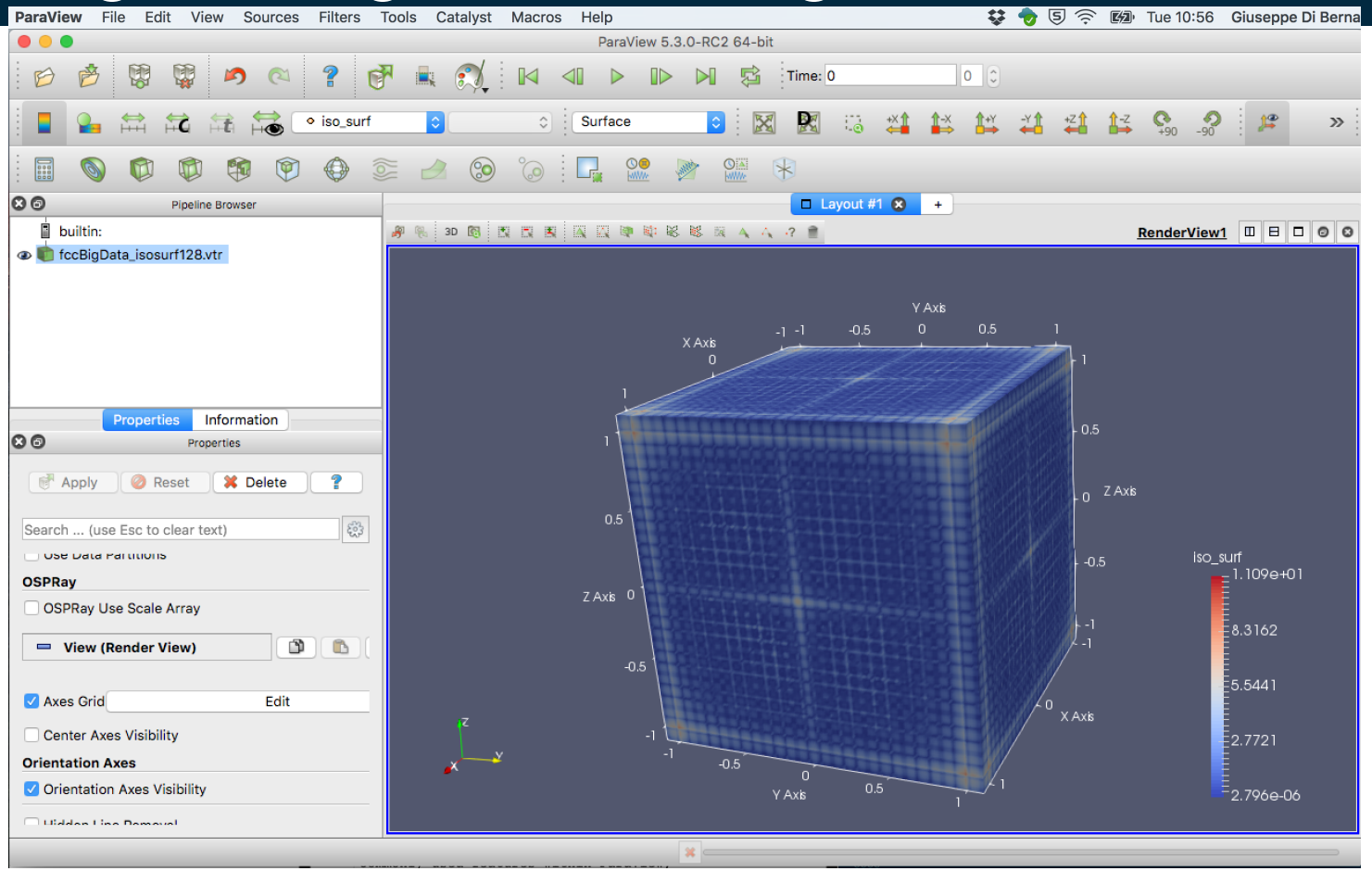
Full Stack

- ParaView comes with a `pvpython` application that allows you to automate the visualization and post-processing with `Python` scripting;
- ParaView Server library provides the abstraction layer necessary for running parallel, interactive visualization. It relieves the client application from most of the issues concerning if and how ParaView is running in parallel;
- The **Visualization Toolkit** (VTK) provides the basic visualization and rendering algorithms.

ParaView: Loading, filtering and rendering

GUI elements

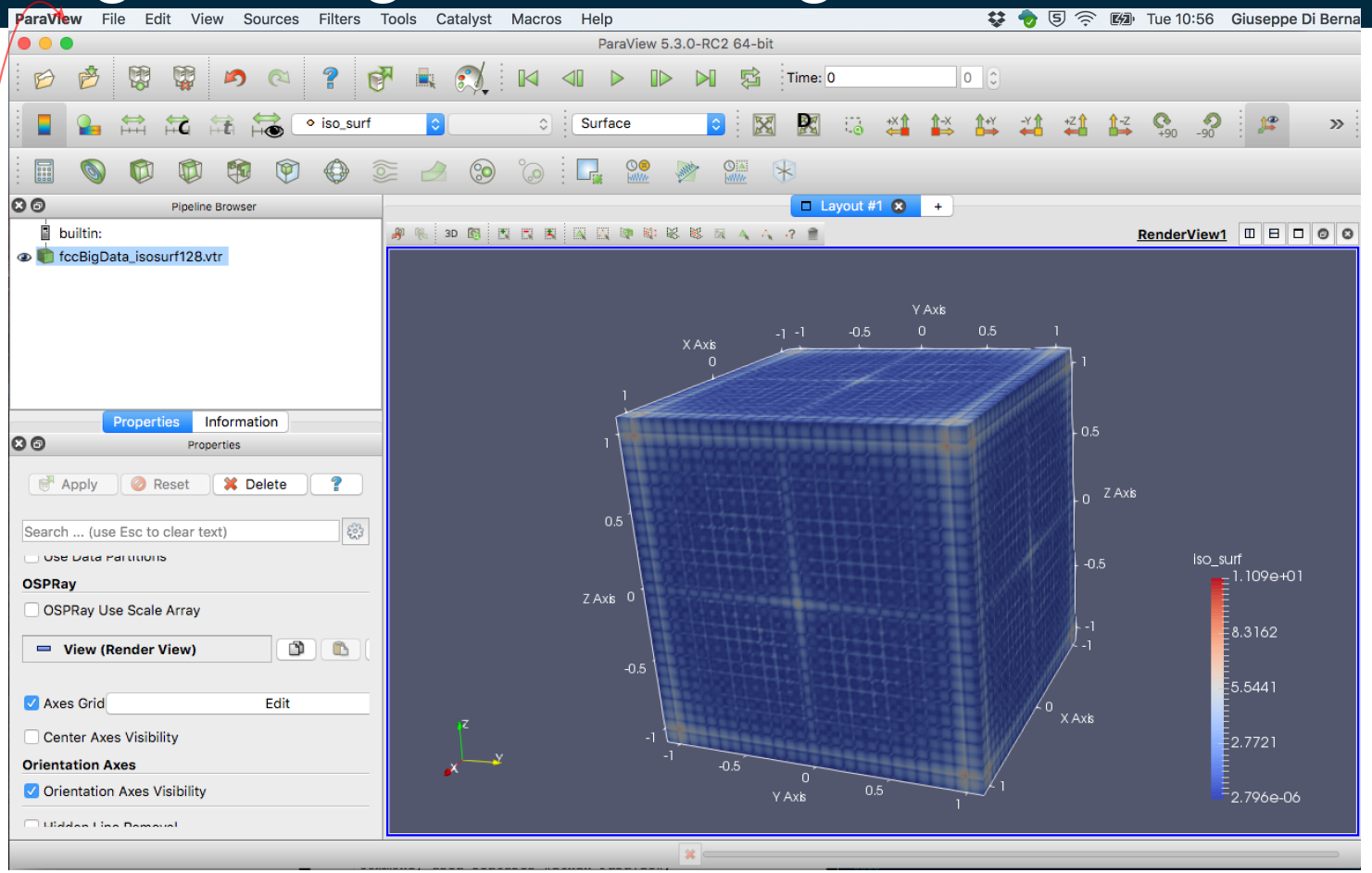
- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



ParaView: Loading, filtering and rendering

GUI elements

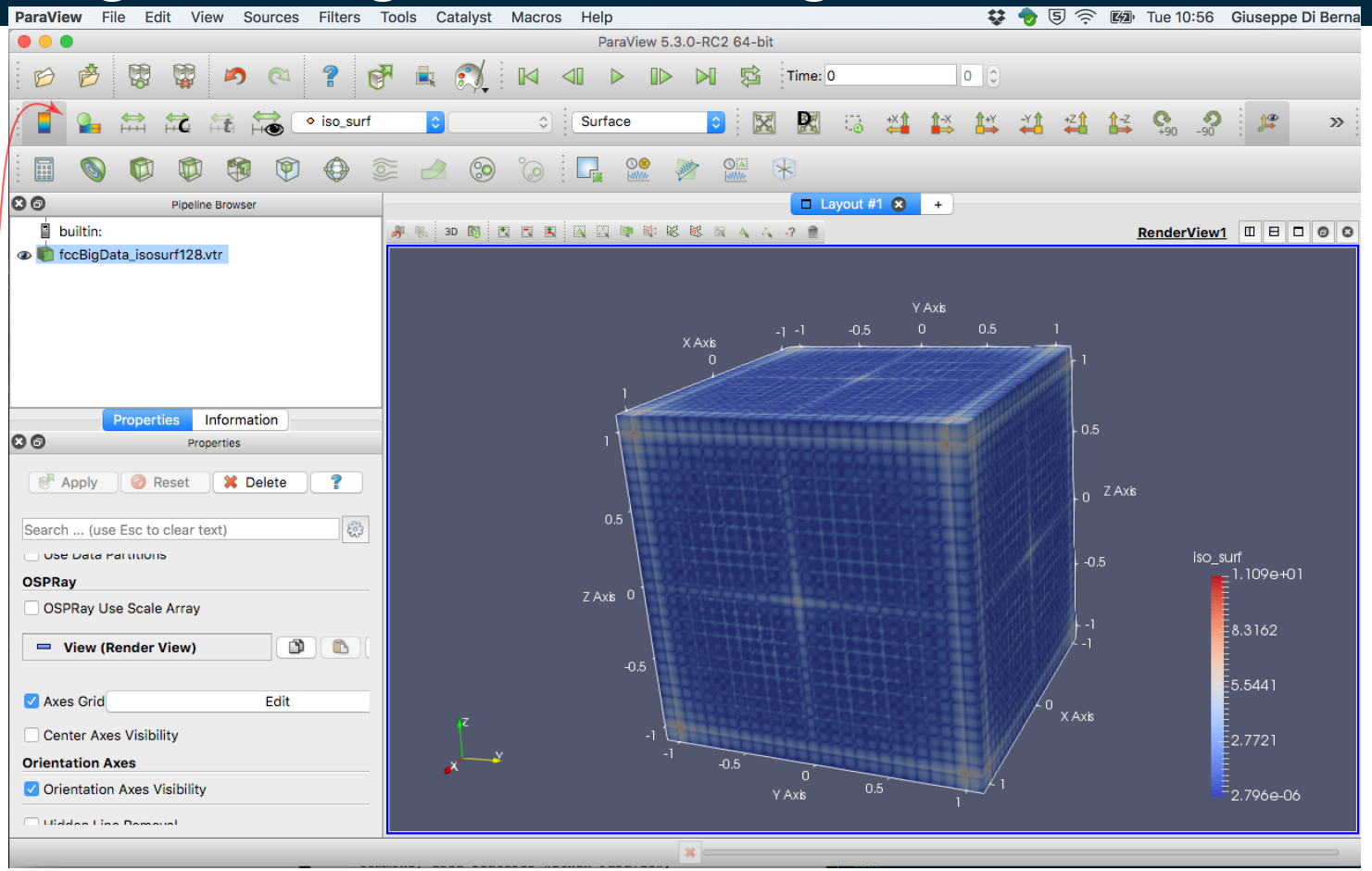
- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



ParaView: Loading, filtering and rendering

GUI elements

- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views

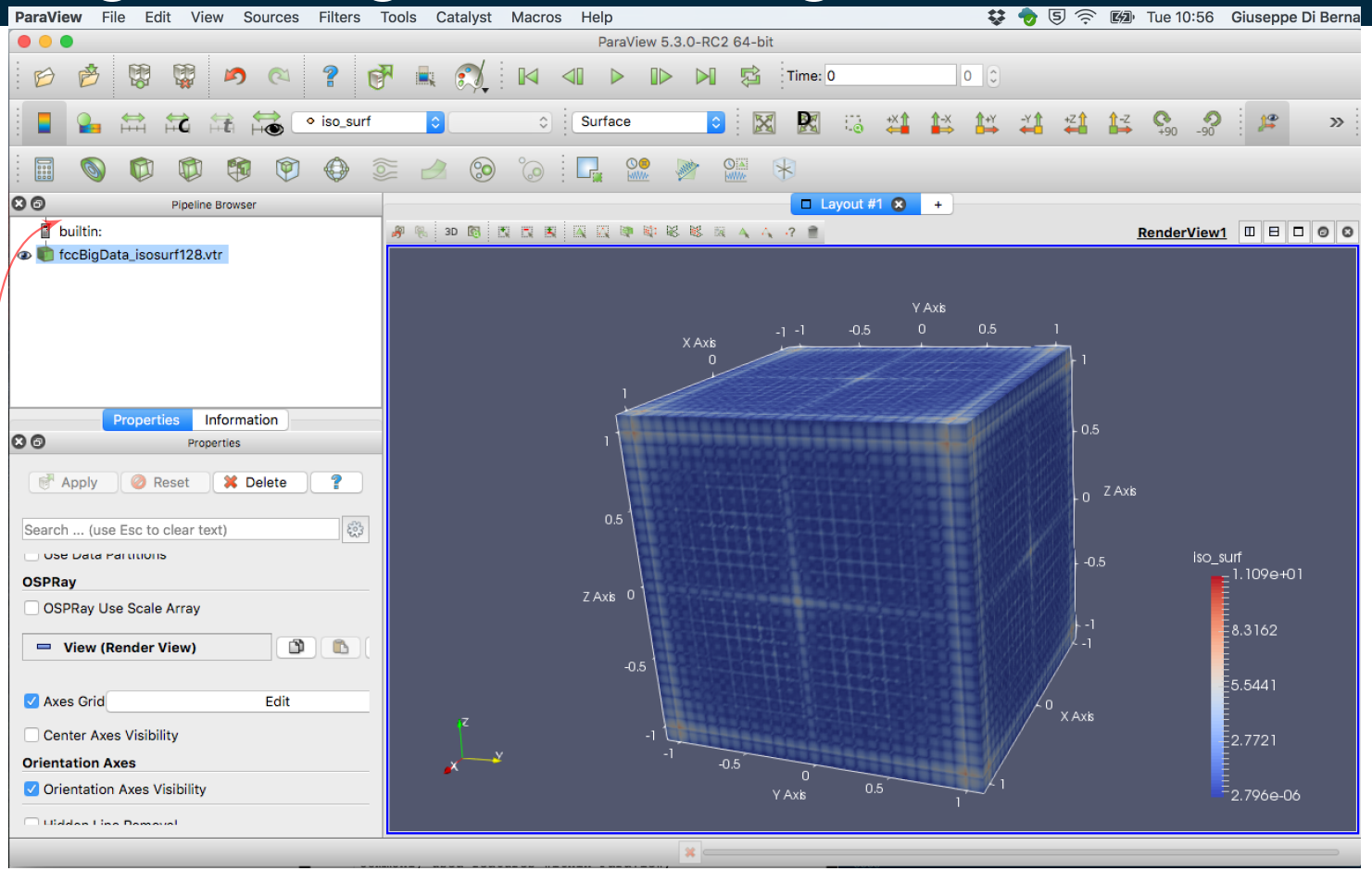


MPCDF Max Planck Computing & Data Facility

ParaView: Loading, filtering and rendering

GUI elements

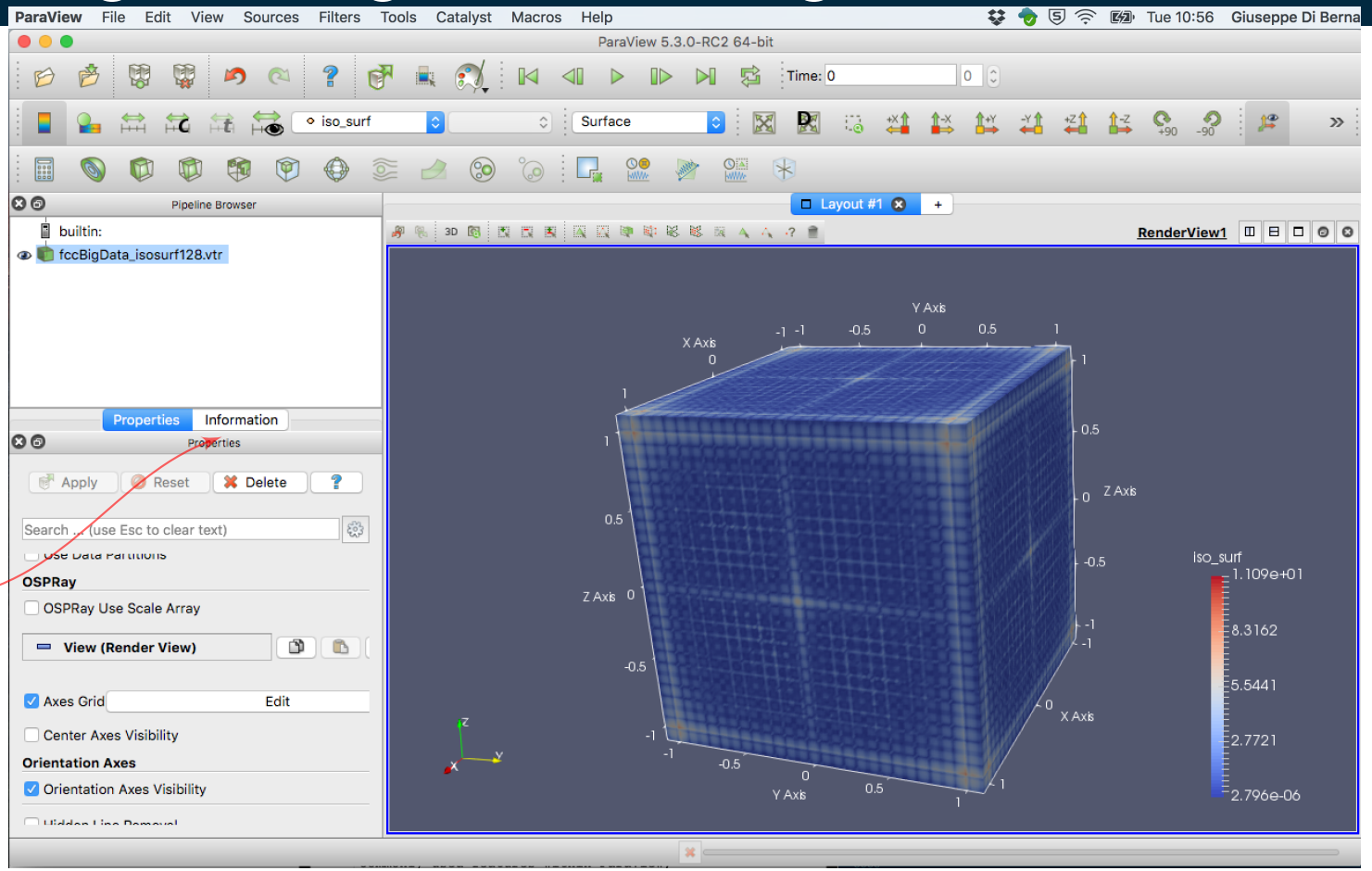
- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



ParaView: Loading, filtering and rendering

GUI elements

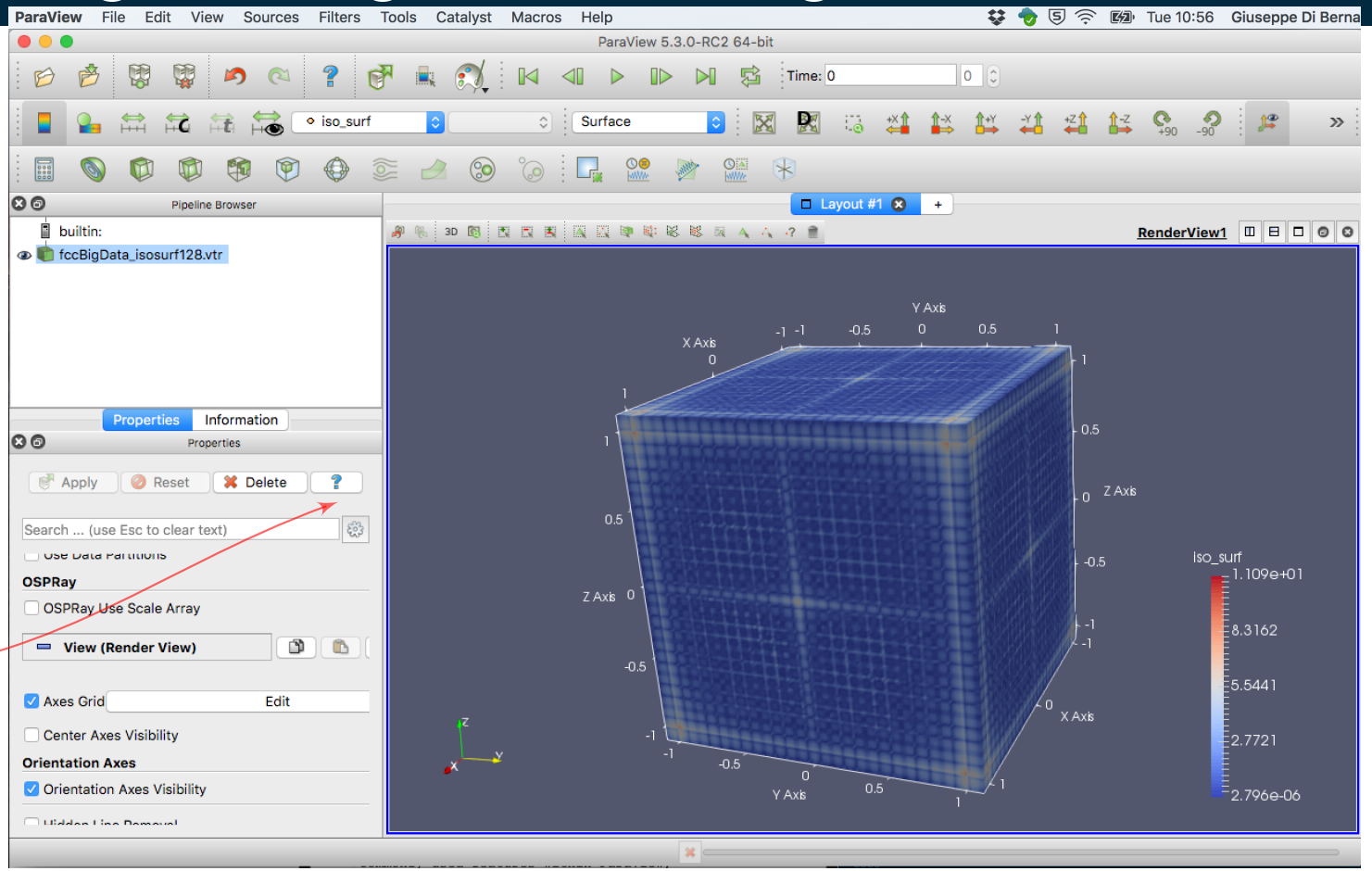
- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



ParaView: Loading, filtering and rendering

GUI elements

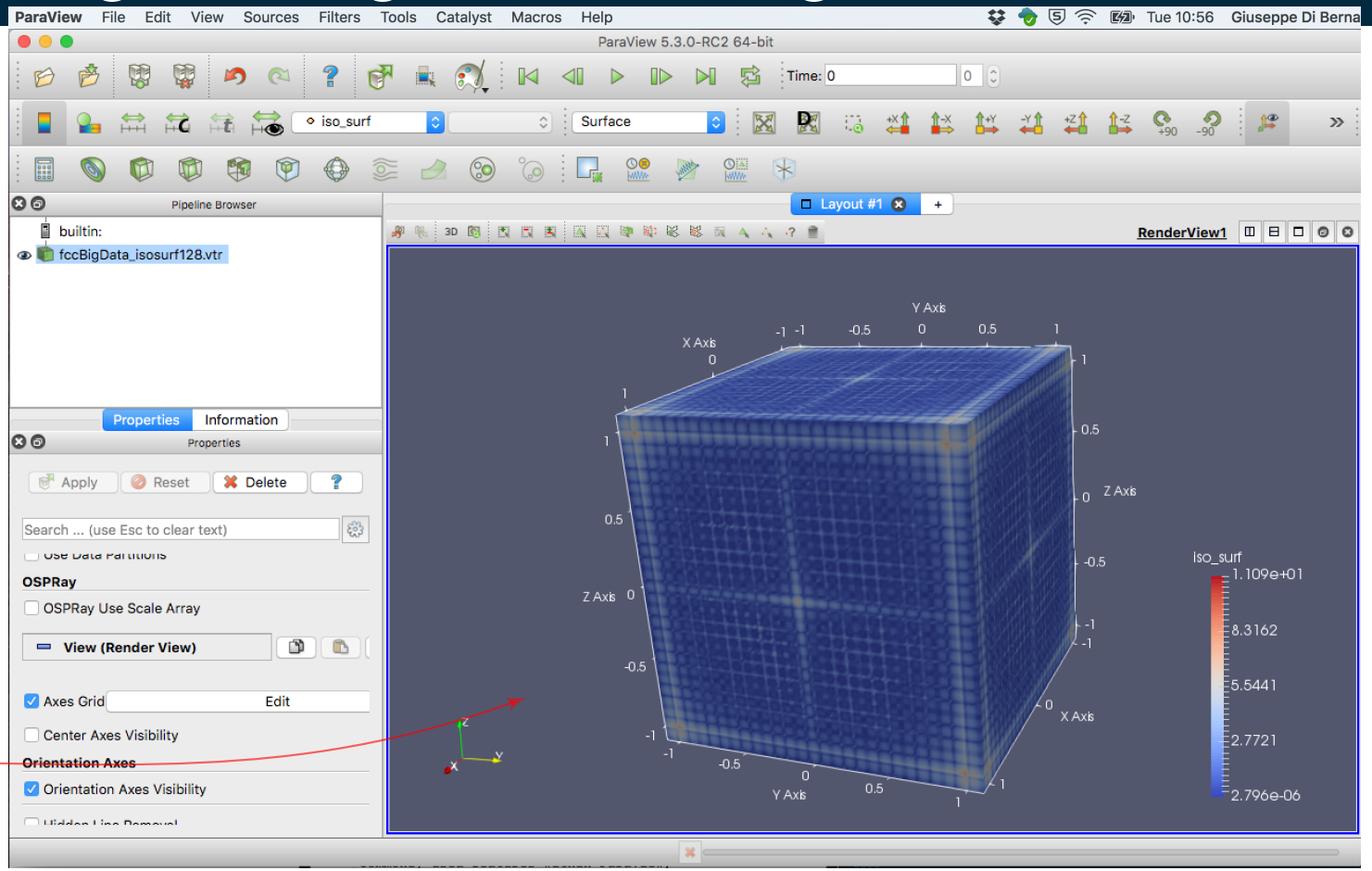
- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



ParaView: Loading, filtering and rendering

GUI elements

- Menu
- Toolbars
- Pipeline
- Inspector
- Help
- Views



GUI elements definition

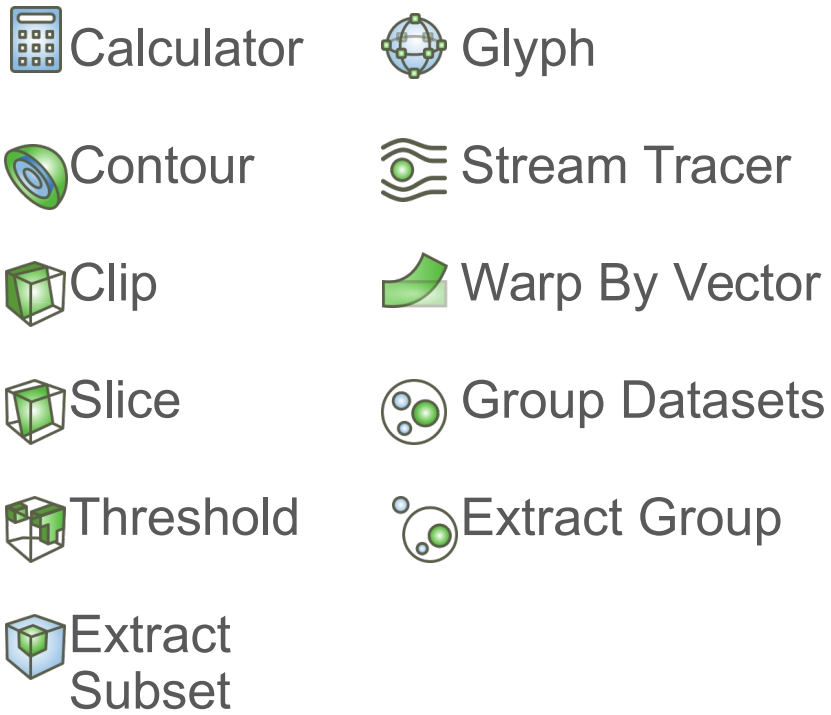
1. **Menu Bar**: allows you to access the majority of features;
2. **Toolbars**: provide quick access to the most commonly used features within ParaView;
3. **Pipeline Browser**: ParaView manages the reading and iterating of data with a pipeline. A convenient list of pipeline objects with an indentation style that shows the pipeline structure;
4. **Properties Panel**: to view and change the parameters of the current pipeline object. The properties are by default coupled with an information tab that shows a basic summary of the data produced by the pipeline object;
5. **3D View**: to present the data so that you may view, interact with, and explore your data.

MPCDF Max Planck Computing & Data Facility

Filters Menu

...some of them

1. **Calculator**: evaluates a user-defined expression on a *per-point* or *per-cell* basis
2. **Contour**: extracts the points, curves, or surfaces where a scalar field is equal to a user-defined value. This surface is often also called *iso-surface*
3. **Threshold**: extracts cells that lie within a specified range of a scalar-field



MPCDF Max Planck Computing & Data Facility

Other Filters

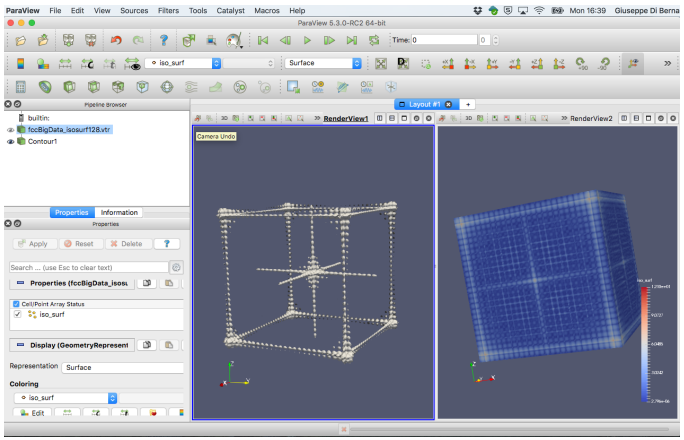
...some of them

- Recent
- AMR (adaptive mesh refinement)
- CTH
- Common (the same list available in the filters toolbars)
- Cosmology

- **Data Analysis**: quantitative values from the data
- Material Analysis
- **Statistics**: descriptive statistics of the data (primarily in tabular form)
- Temporal
- Alphabetical

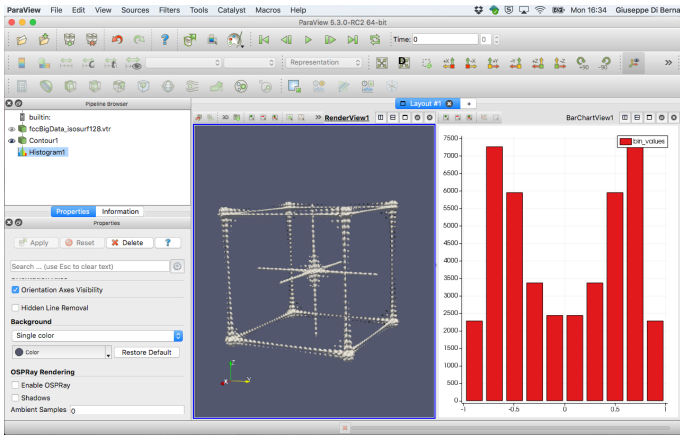
MPCDF Max Planck Computing & Data Facility

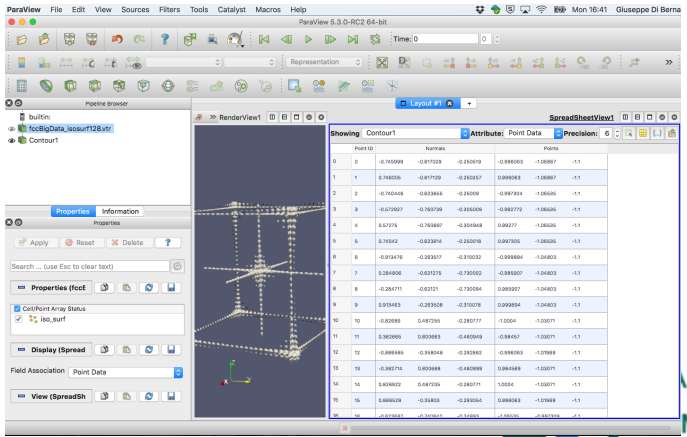
ParaView: Multiple views



Some examples

1. 3d + 3d
2. 3d + Histograms
3. 3d + table



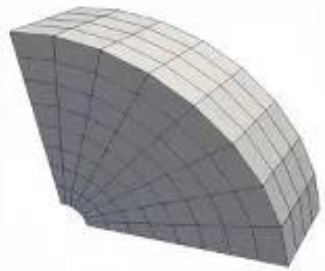


ParaView: The dataset types (sampling structures)

ParaView was designed primarily to handle data with spatial representation. Thus the **data types** used in ParaView are meshes

Uniform Rectilinear
(vtkImageData)

Non-Uniform Rectilinear
(vtkRectilinearData)



Curvilinear
(vtkStructuredData)

Polygonal
(vtkPolyData)

Unstructured Grid
(vtkUnstructuredGrid)

Multi-block

Hierarchical Adaptive Mesh Refinement (AMR)

Hierarchical Uniform AMR

Octree

MPCDF Max Planck Computing & Data Facility

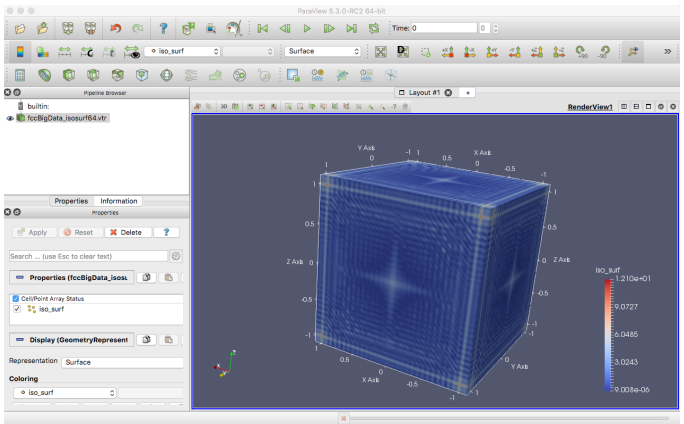
Data Selection

The goal of visualization is often to find the importance details within a large body of information. ParaView's selection abstraction is an important simplification of this process.
Selection is the act of identifying a subset of some dataset. More specifically the subset identifies particular select points, cells, or blocks within any single data set.
In ParaView, selection can take place at any time, and the program maintains a current selected set that is linked between all views.

1. The most direct means to create a selection is via the `Find Data` dialog 
2. Another way of creating a selection is to pick elements right inside the 3D view: **rubber-band selection**

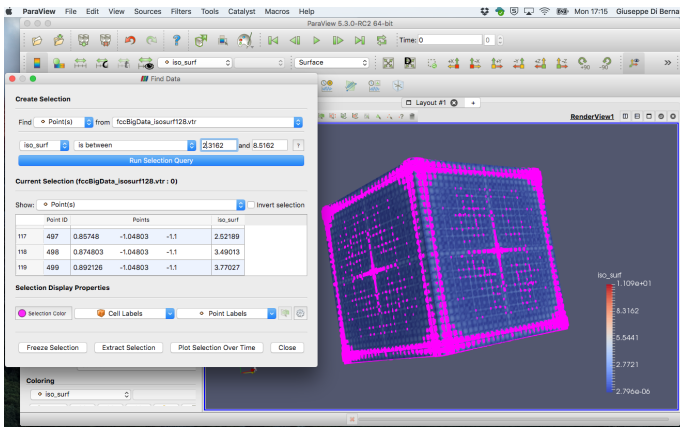
MPCDF Max Planck Computing & Data Facility

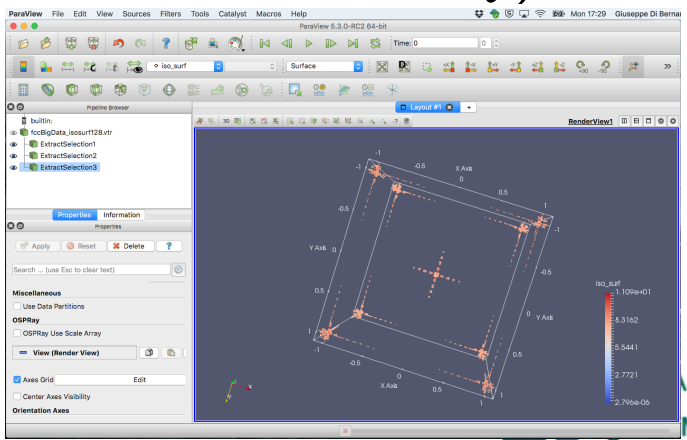
Query Data by Attribute Values - Find Data dialog



Some examples

1. load, filter and display your volume data   
2. find data matching your criteria 
3. extract selection (`Run Selection Query`)





Query Data Spatially - Selection

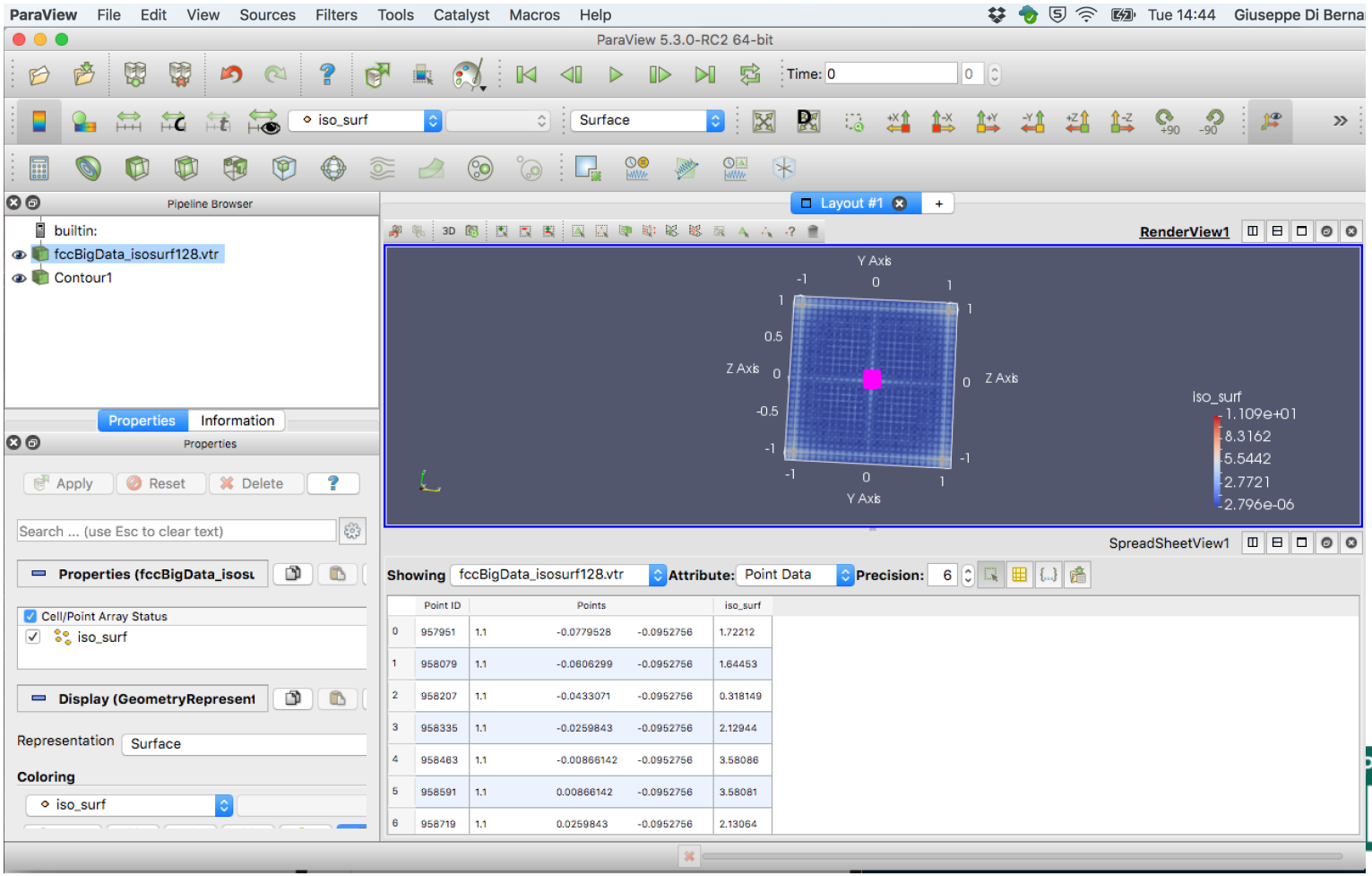
...by clicking and dragging the mouse in the 3D view

- **Select Cells On (Surface)**: cells that are visible in the view under a rubber band
- **Select Points On (Surface)**: points that are visible in the view under a rubber band
- **Select Cells Through (Frustum)**: cells that exist under a rubber band
- **Select Points Through (Frustum)**
- **Select Cells with Polygon**: you draw a polygon
- **Select Points with Polygon**
- **Select Blocks**: Select blocks in a multiblock dataset
- **Int. Select Cells**
- **Int. Select Points**

MPCDF Max Planck Computing & Data Facility

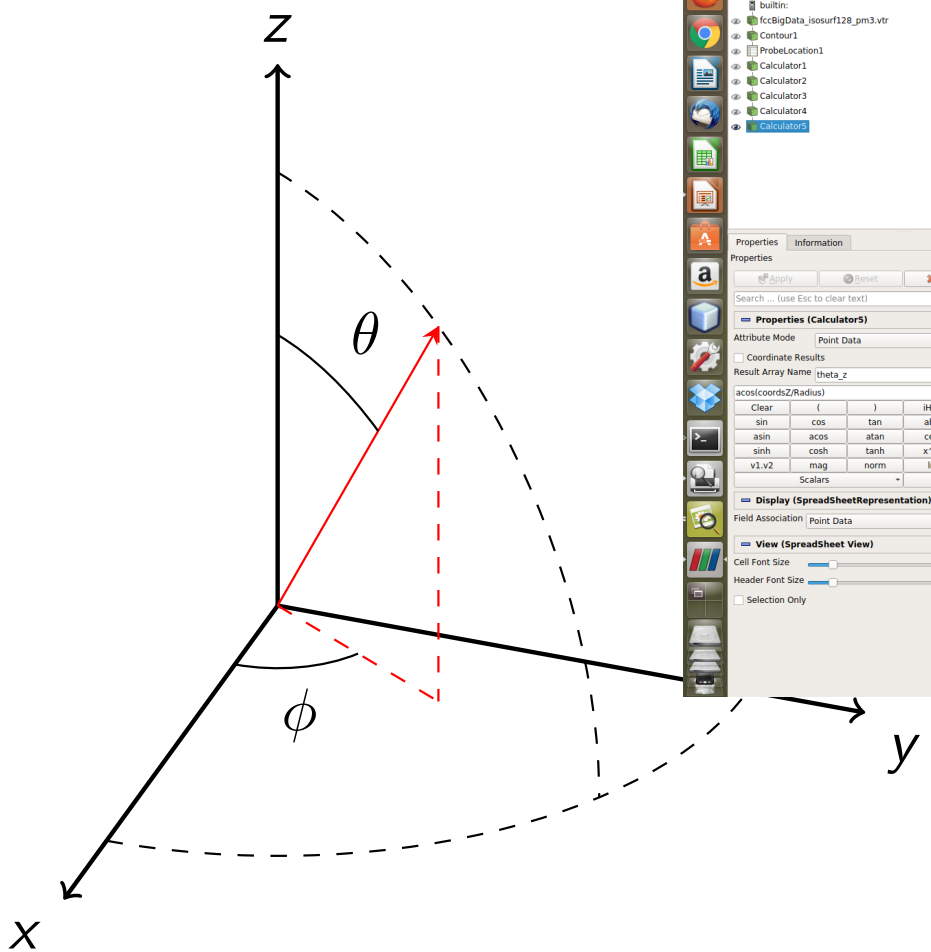
Query Data Spatially - Selection

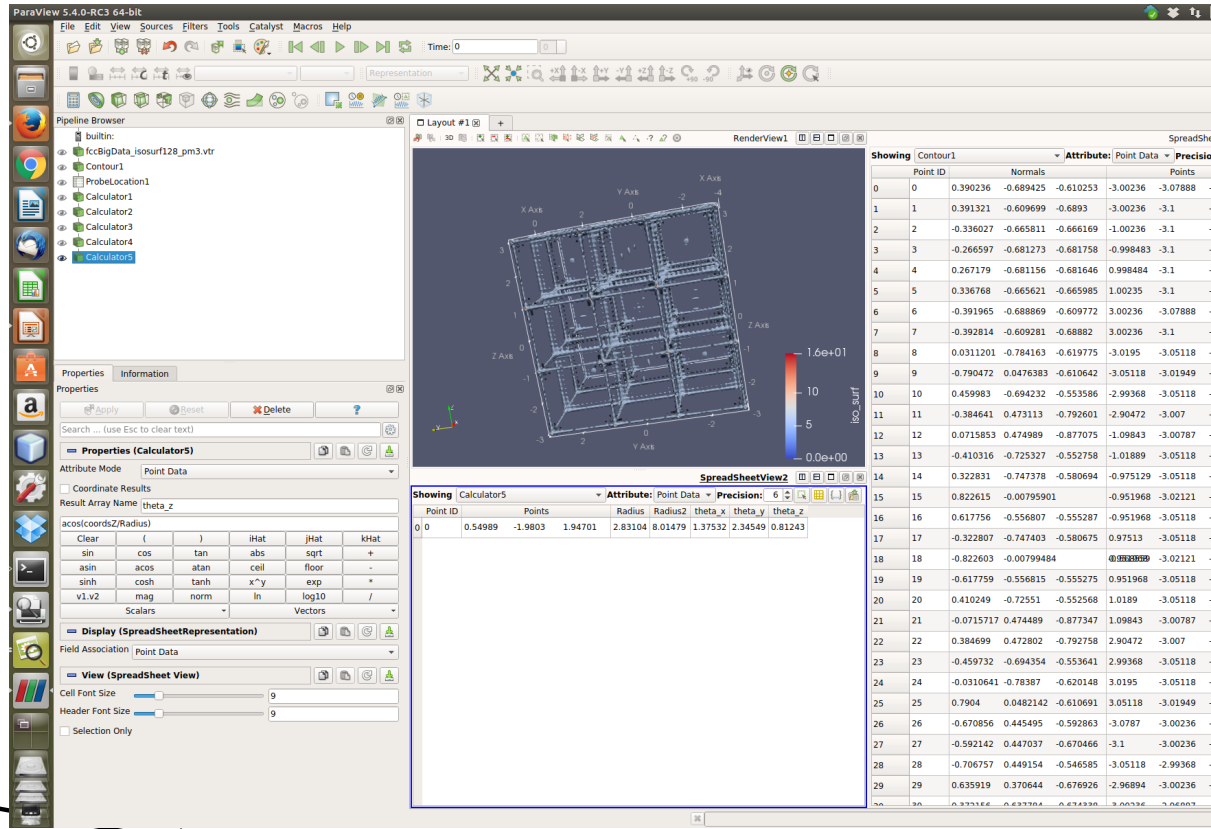
...by clicking and dragging the mouse in the 3D view



MAX PLANCK COMPUTING & DATA FACILITY

Hands-On: extract the tuple of radius and angles





The End

MPCDF Max Planck Computing & Data Facility