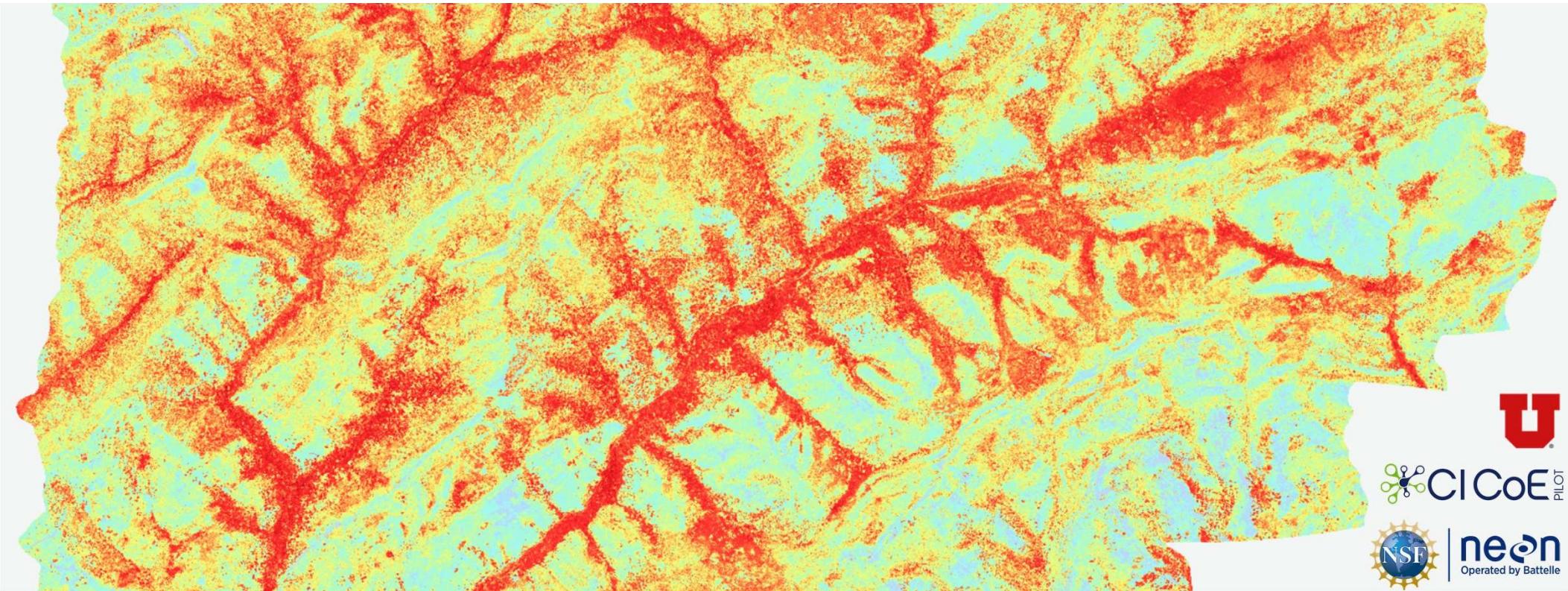


CI/CS WORKSHOP

THE COMMUNITY TOGETHER





CICoE PILOT



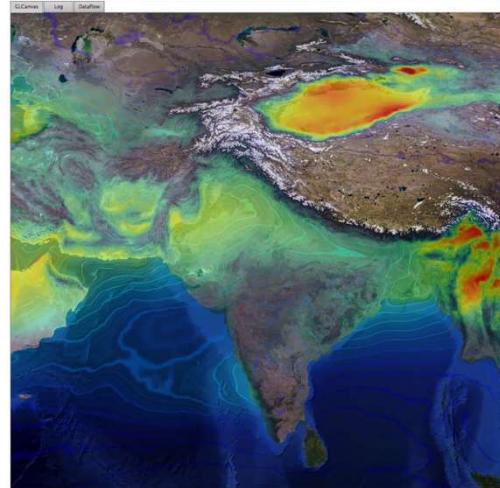
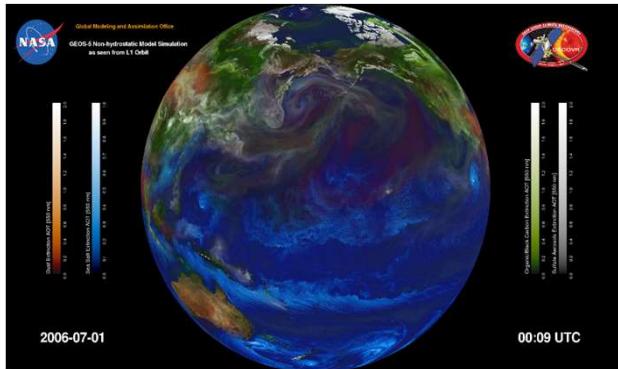
Low-cost, interactive access
and visualization of large
scale scientific data

Steve Petruzza,
Giorgio Scorzelli,
Valerio Pascucci

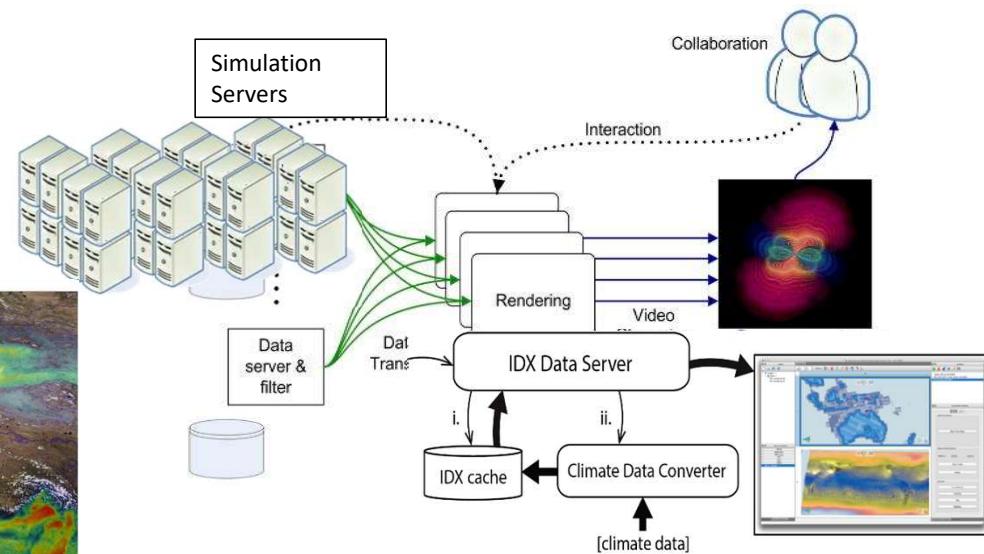
Scalable Deployment: Exploration of 3.5TB of Weather/Climate Data in Real Time

Workflow

- Data creation
- Processing
- Analysis
- Visualization
- Data Management



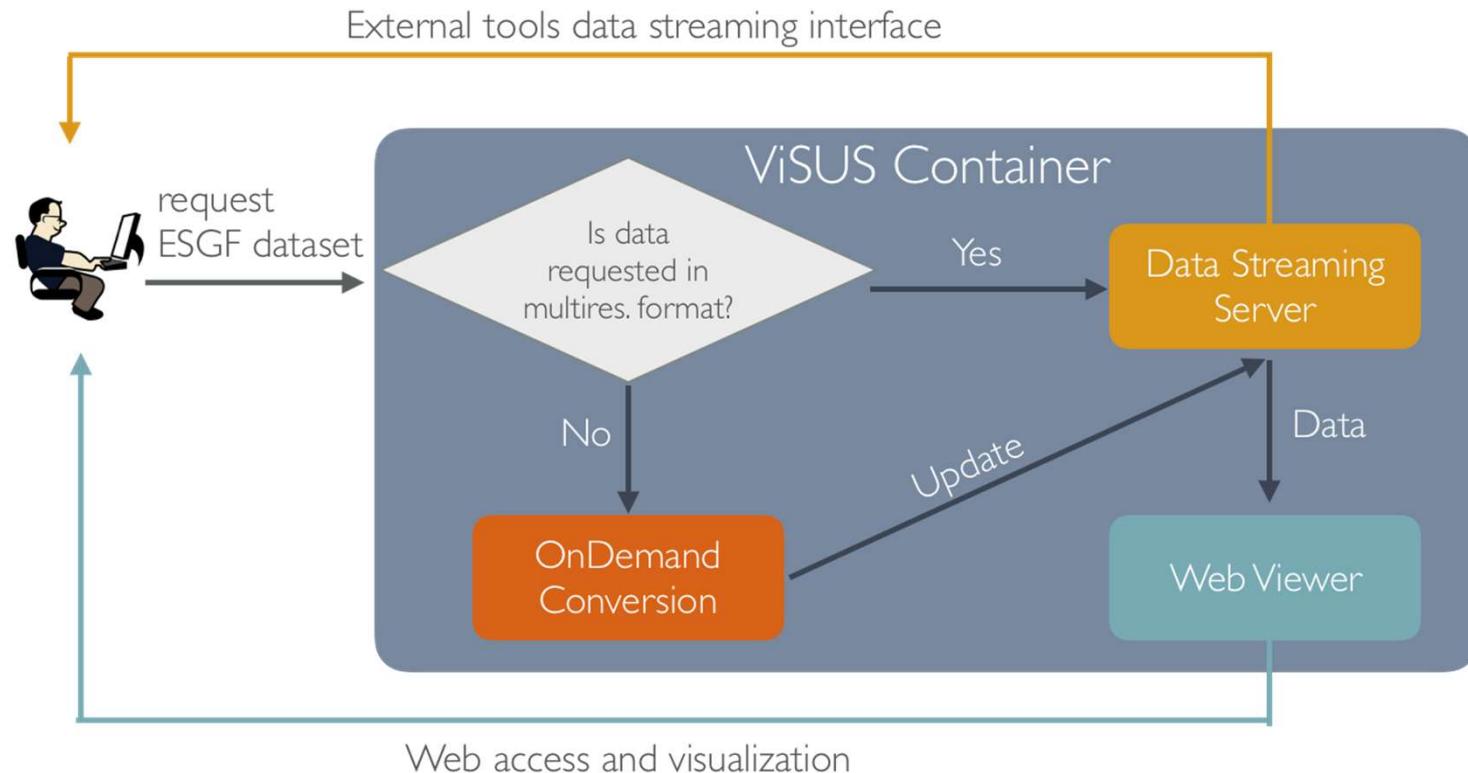
- 7km GEOS-5 “Nature Run”
- 1 dataset, 3.5 PB
- theoretically: openly accessible
- practically: precomputed pics



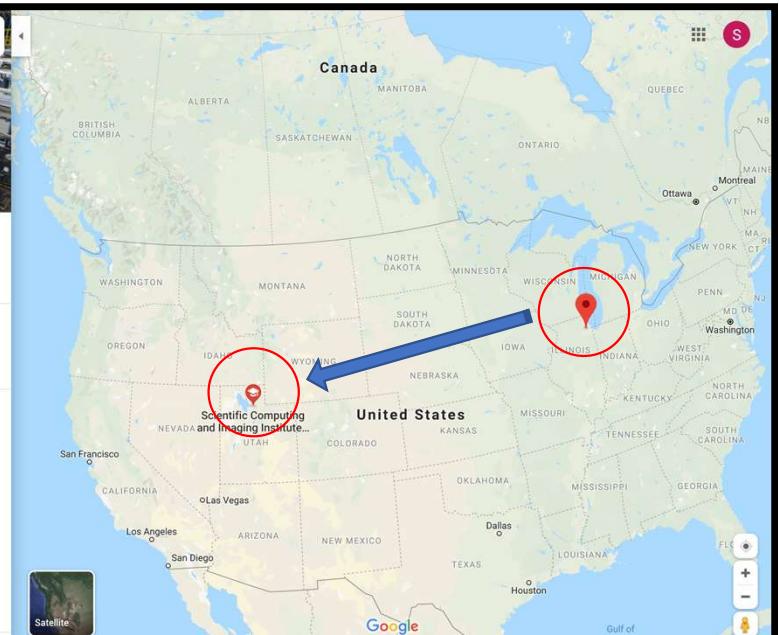
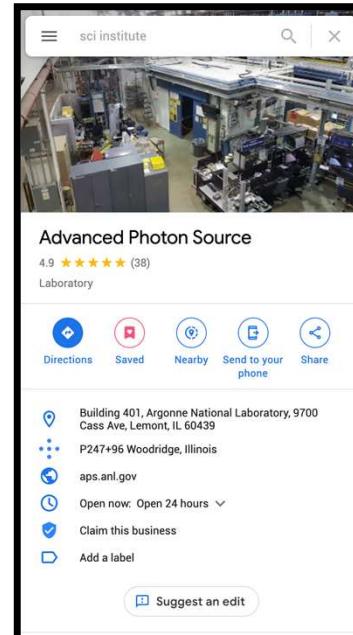
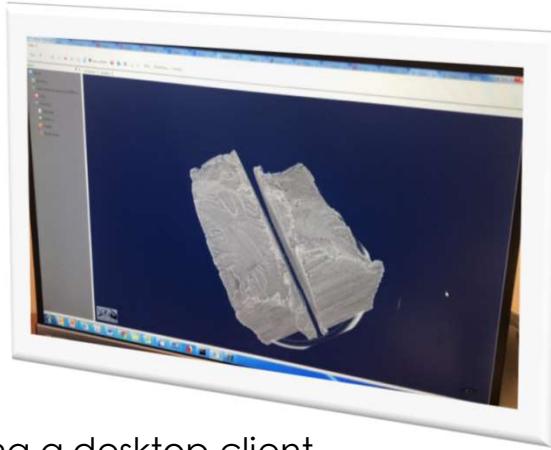
Distributed Resources

- 3.5 PB of *data store in NASA*
- *Primary ViSUS server in LLNL*
- *Secondary ViSUS server in Utah*
- *Clients connect remotely*
- *Work without additional HPC resources*

Containerized on-demand conversion and streaming service (for ESGF data)



High Performance Data Movements for Real-Time Access to Large Scale Experimental Data (Dockerized server)



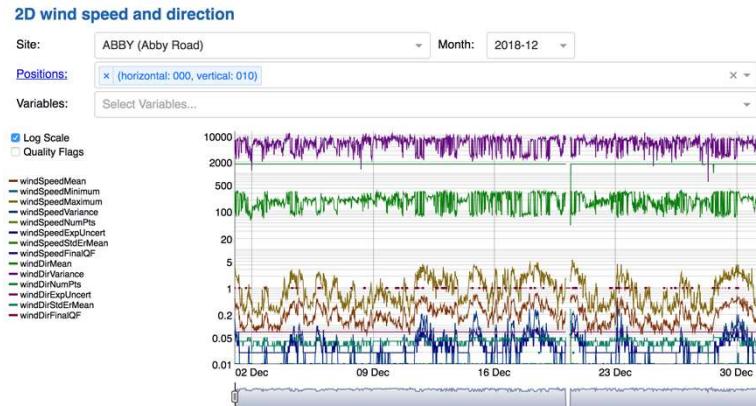
- Using a desktop client (or a webviewer)

Prof. Ashley Spears was able to see the data being acquired at APS from her office at UoU

- [Webviewer Demo:](#)
Aluminum Foam
of similar size

NEON AOP data access

- NEON has a large amount of data that is shared with the community through their **data portal**
- There exist **APIs** to download those data in bulk
(per site, per year, per data product, now also by area)
- For some data, such as sensor measurements, the portal provides an **interactive** navigation system
- For others, like **Airborne Observation Platforms data**, there is a long list of image files...
- There is a need to present all AOP data interactively, where the users can preview, navigate, and select/access/download the data they need



Include	Filename	Site	Month	Size
✓	2017_ABBY_1_546000_5060000_image.tif	ABBY	2017-06	13.61 MB
✓	2017_ABBY_1_546000_5061000_image.tif	ABBY	2017-06	21.09 MB
✓	2017_ABBY_1_546000_5062000_image.tif	ABBY	2017-06	32.95 MB
✓	2017_ABBY_1_546000_5063000_image.tif	ABBY	2017-06	30.23 MB
✓	2017_ABBY_1_546000_5064000_image.tif	ABBY	2017-06	32.88 MB
✓	2017_ABBY_1_546000_5065000_image.tif	ABBY	2017-06	34.83 MB
✓	2017_ABBY_1_546000_5066000_image.tif	ABBY	2017-06	34.44 MB
✓	2017_ABBY_1_546000_5067000_image.tif	ABBY	2017-06	40.91 MB
✓	2017_ABBY_1_546000_5068000_image.tif	ABBY	2017-06	38.67 MB
✓	2017_ABBY_1_546000_5069000_image.tif	ABBY	2017-06	35.13 MB
✓	2017_ABBY_1_546000_5070000_image.tif	ABBY	2017-06	29.52 MB
✓	2017_ABBY_1_546000_5071000_image.tif	ABBY	2017-06	29.74 MB
✓	2017_ABBY_1_546000_5072000_image.tif	ABBY	2017-06	32.44 MB
✓	2017_ABBY_1_546000_5073000_image.tif	ABBY	2017-06	27.54 MB
✓	2017_ABBY_1_546000_5074000_image.tif	ABBY	2017-06	6.68 MB
✓	2017_ABBY_1_547000_5059000_image.tif	ABBY	2017-06	19.35 MB
✓	2017_ABBY_1_547000_5060000_image.tif	ABBY	2017-06	57.84 MB

Showing 1 to 100 of 20,850 entries

AOP data

CiCOE data access/visualization/management efforts

- AOP data ingestion and publication
- Multiresolution streaming data access (via javascript, python, C++)
- Experimented with time series and hyperspectral data
- Experimented with mixed tile sources (Google Earth+AOP data)
- NEON endpoint for data discovery and viewer embedding
- Deployment experiments on CloudLab

Data ingestion and publication

- R scripts to download AOP dataset (byFileAOP), can we do better?
- Data processing and management:
 - extract the bounding boxes information from GeoTIFF and generate scripts to convert each dataset
 - Query NEON APIs to derive “month” value from file paths
 - Populate Utah endpoint database
 - Update the streaming server with the new datasets to make available

Streaming server/data portal

- Apache module
- Provides streaming access to data hosted locally or remotely
- Available within a Docker container or standalone installation
- Web UI to manage and ingest new datasets

The screenshot shows the ViSUS DataPortal homepage. At the top right are logos for NSF, neon (Operated by Battelle), and a red stylized 'U'. The navigation bar includes links for Home, Configure Server, Manage Data, Explore Data, and Login. The main title is '>ViSUS DataPortal' with a subtitle 'Ciao, you are working on the host **dataportal**, from here you can:'. Below this are three main buttons: 'Configure' (with a grid icon), 'Manage Data' (with a camera icon), and 'Explore Data' (with a map icon). Each button has a description and a blue 'View' button. A section titled 'List of datasets on this server' shows three entries: 'DP3.30010.001-D07-2016_ORNL_2-L3-Camera-Mosaic-V1', 'DP3.30010.001-D08-2019_TALL_5-L3-Camera-Mosaic', and 'DP3.30010.001-D08-2019_DELA_5-L3-Camera-Mosaic', each with a 'View' button.

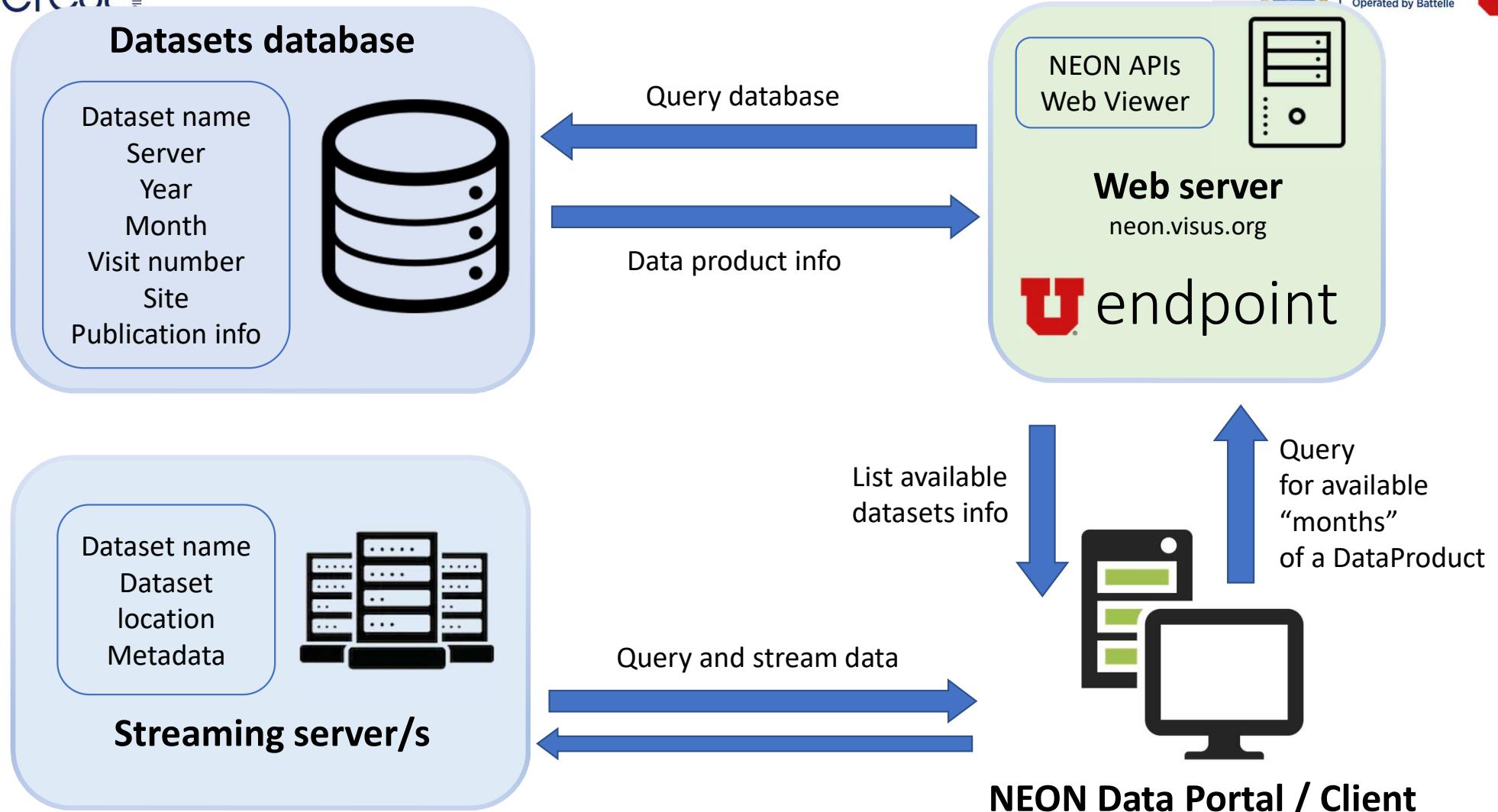
Utah NEON APIs endpoint

- Handles requests to `/neonapi/products/{productCode}`
- Follows NEON APIs syntax
- Provide configuration strings to access a specific dataset
- Datasets could be relocated to other streaming servers transparently

```
// https://neon.visus.org/neonapi/products.php/DP3.30010.001

{
  "data": {
    "productCode": "DP3.30010.001",
    "siteCodes": [
      {
        "siteCode": "ABBY",
        "availableMonths": [
          "2018-07",
          "2017-06"
        ],
        "availableDataUrls": [
          "server=https%3A%2F%2Fdataportal.sci.utah.edu%2Fmod_visus%3F&dataset=DP3.30010.001-D16-2018_ABBY_2-L3-Camera-Mosaic-V01",
          "server=https%3A%2F%2Fdataportal.sci.utah.edu%2Fmod_visus%3F&dataset=DP3.30010.001-D16-2017_ABBY_1-L3-Camera-Mosaic-V01"
        ]
      },
      {
        "siteCode": "ARIK",
        "availableMonths": [
          "2017-05"
        ],
        "availableDataUrls": [
          "server=https%3A%2F%2Fdataportal.sci.utah.edu%2Fmod_visus%3F&dataset=DP3.30010.001-D10-2017_ARIK_1-L3-Camera-Mosaic-V01"
        ]
      }
    ]
  }
}
```





First integration

- Component embedded as an iframe
- Datasets and time navigation
- The parent window pass the dataset settings (retrieved from the endpoint) to the iframe (GET) which provides the rest of the functionalities

NEON integration (live)
[Basic RGB](#)
[Vegetation indices \(hyperspectral\)](#)

University of Florida deep learning tree classification (live):
[This is a link to the viewer with the trees](#)
[Detailed view](#)
[Link to their main project](#)

Visualizations

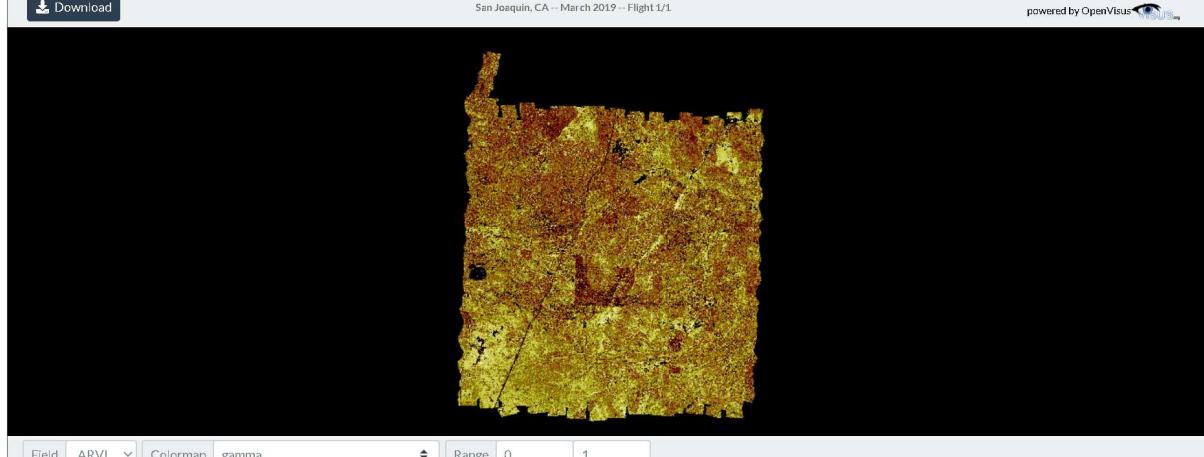
AOP Data Viewer

This viewer allows for interactive exploration of remotely sensed data from the Airborne Observation Platform (AOP). Change the field site and flight for this data product using the tools below to stream different data into view. Pan and zoom in the view to stream higher resolution imagery. This pilot data viewer is provided through a collaboration with the [Visus Project at the University of Utah](#) and more updates are planned for the future.

Site SJER Year 2019 Flight 1/1 (March)

2017 2018 2019

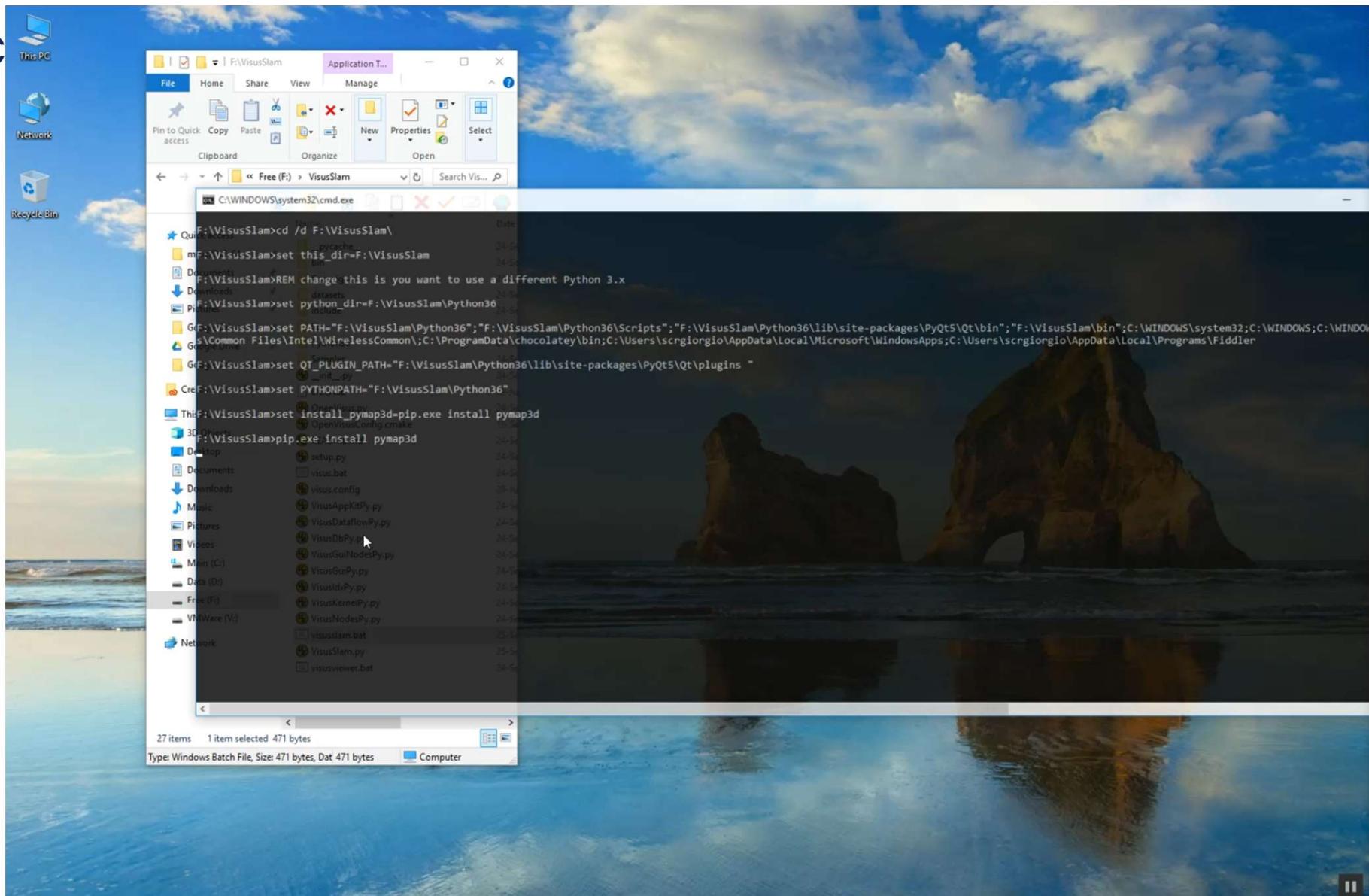
[Download](#) San Joaquin, CA -- March 2019 -- Flight 1/1
powered by OpenVisus





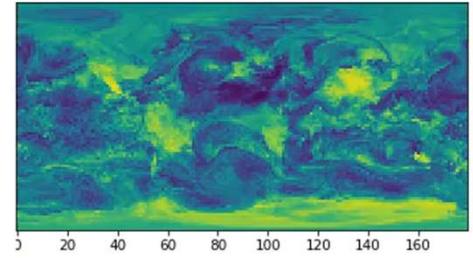
Interoperability

- Explored solutions to integrate in the same visualization multiple “tile” sources
- Proof of concept of use AOP data and Google Earth
- New version of data format and server will allow to visualize AOP data in their geographical context

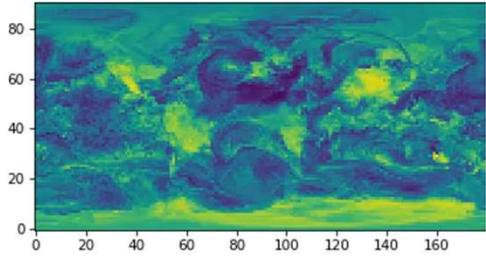


Progressive vs Linear Computation of Time Averages for Climate Simulations

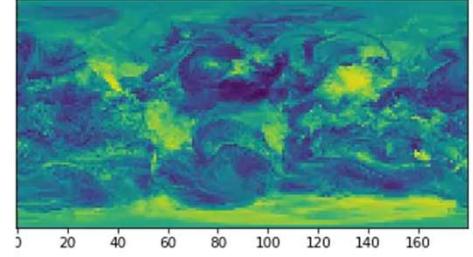
Linear data access



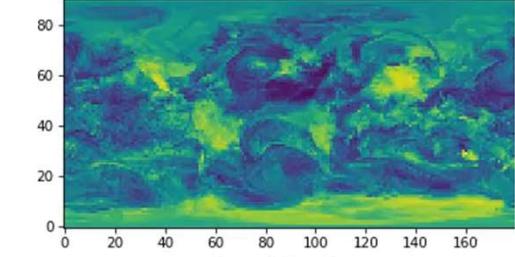
Progressive data access



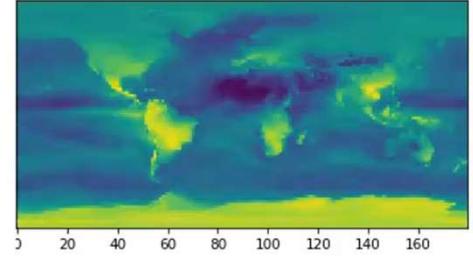
Linear average computation



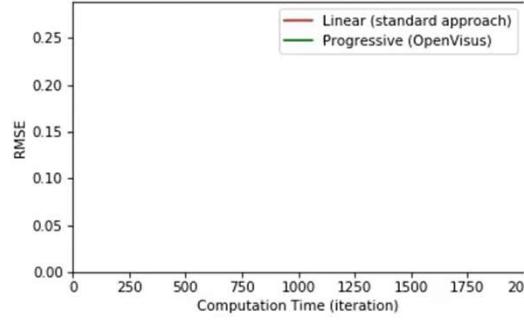
Progressive average computation



Target result



Computation Error
(red=traditional,green=progressive)



jupyter openvisus-progressive8 LastCheckpoint 9d1f20d8 [submitted]

File Edit View Insert Cell Kernel Widgets Help

File Edit View Insert Cell Kernel Widgets Help

OpenVISUS: read from a remote dataset and create and use a local cache

```
In [1]: %matplotlib inline
display.HTML("style.css").container( width:100% (important); )</style>")
```

```
In [2]: %matplotlib notebook
import os
import math
import matplotlib.pyplot as plt
import numpy as np
import time
from ipywidgets import *
from OpenVisus import *

## Note: In this example we use a local cache in the folder "./nature"
## If you don't want to use caching remove the cache_dir="./nature" when you use readData() or set this to ""

In [3]: %%DModule.attach()
ipythonengine is working fine
```

```
In [4]: %%DModule
# Load a dataset
dataset=natureDataset("https://mmlnuya.sci.utah.edu/red_visus/datasets/nature_2007_seri1_hourly")
```

```
In [5]: %%DModule
def readData():
    timedataset.getDefaultsTime(),
    fielddataset.getDefaults(),
    logdataset.getLogs(),
    resolutionresolution, cache_dir="")

    if(cache_dir!=""):
        access_config = StringTree.FromString("<access name='Multiplex' type='multiplex'><access name='cache' type='disk' chmod=rw url='<cache_dir>/visus.idx' /><access name='source' type='network' chmod=rw compression=zlib /></access>")
    else:
        access_config = StringTree()

    accessdataset.createAccess(access_config)

    querybyQuery(dataset, field, time, ord('r'))
    query_end_resolution.push_back(resolution)
    dataset.beginQuery(query)
    dataset.endQuery(query)

    dataset.queryCache(query)
    dataset.cache(query, cache_dir)

    return dataset
```

```
def showData(data):
    fig=plt.figure(figsize=(10,5))
    ax = fig.add_subplot(1,1)
    ax.set_data(data, origin='lower')
    plt.show()
```

```
In [6]: %%DModule
compute_err = False
compute_err2 = True
time.sleep(20)

for t in range(0, nT):
    print("t",t,"linear", "progressive", "err", "err2")
    print("t",t,"linear to ",linear_t[t], "progressive to ",linear_t[progressive_t[t]], "err", "err2")
    f = np.double(readData(t=linear_t[t]), resolutionres, fielddataset.getFieldByName("TOTANGSTR"), cache_dir=cache_directory)
    last_output = output
    output = np.double(np.zeros((f.shape[0],1)))
    output[0] = last_output[0]
    output[1] = last_output[1]
    last_output = output
    f0 = np.double(readData(t=linear_t[t+1]), resolutionres, fielddataset.getFieldByName("TOTANGSTR"), cache_dir=cache_directory)
    last_output0 = output
    output = np.double(np.zeros((f0.shape[0],1)))
    output[0] = last_output0[0]
    output[1] = last_output0[1]
    last_output0 = output
    ba.imshow(f, origin='lower')
    ba.imshow(f0, origin='lower')
    ba.imshow(last_output, origin='lower')
    errors.append(rmse(output, result))
    errors2.append(rmse(output, result))
    err2.append(rmse(output, result))

    if interval > 1:
        for i in range(1, (interval-1)):
            errors2.append(errors[i])
            errors2.append(errors[i+1])
        f0 = np.double(readData(t=linear_t[t+1]), resolutionres, fielddataset.getFieldByName("TOTANGSTR"))
        ba.imshow(f0, origin='lower')
        ba.imshow(last_output, origin='lower')
        errors.append(rmse(f0, result))
        errors2.append(rmse(f0, result))
        err2.append(rmse(f0, result))

    plt.show()
    print("time ",t,"err", "err2")
    fig.canvas.draw()
    time.sleep(1)
    j+=1

print("")

output.tofile(cache_directory+"result", str(output.shape[0])+"x"+str(output.shape[1])+"x"+str(output.shape[2]))
output_directory="result"+str(output.shape[0])+"x"+str(output.shape[1])+"x"+str(output.shape[2])
print("differences", "metadatas", output_directory)
print("output", "type", "shape", output_directory)
print("")
```

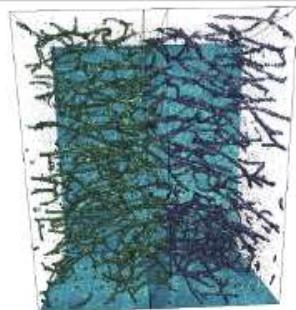
```
time 22 linear to 220 progressive to 220
```

Integrated Data Acquisition, Management and Computation for Neuroscience

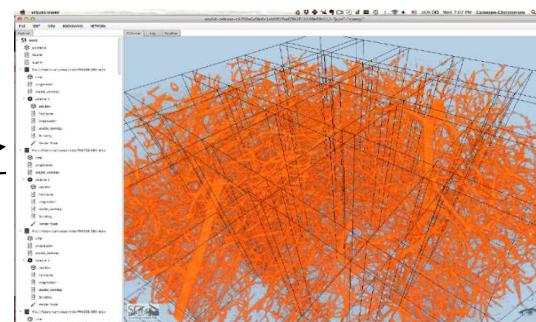
(1) Data Source



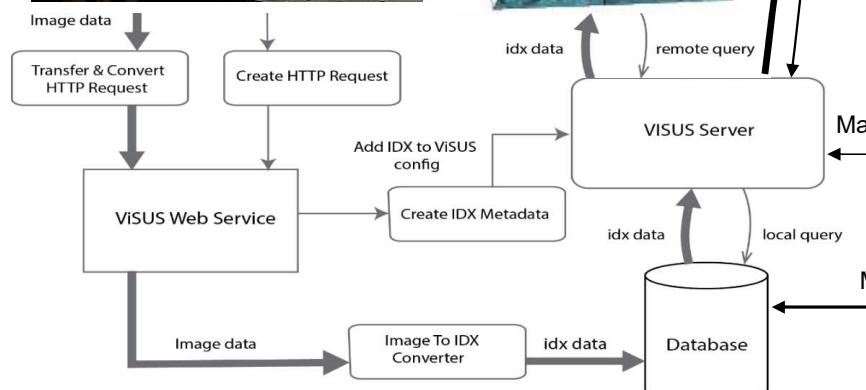
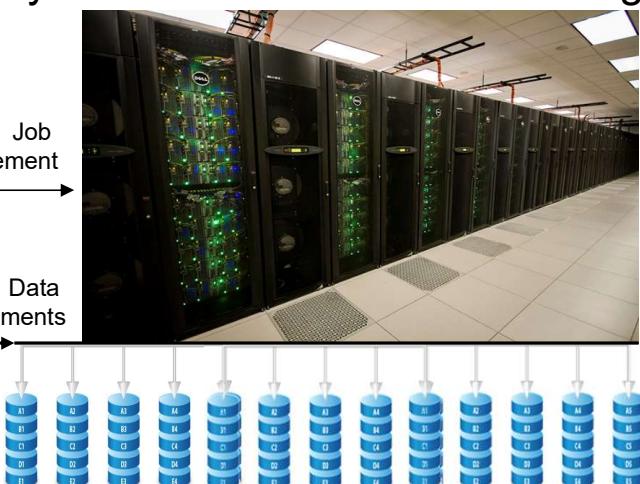
(2) Preliminary Interactive Analytics



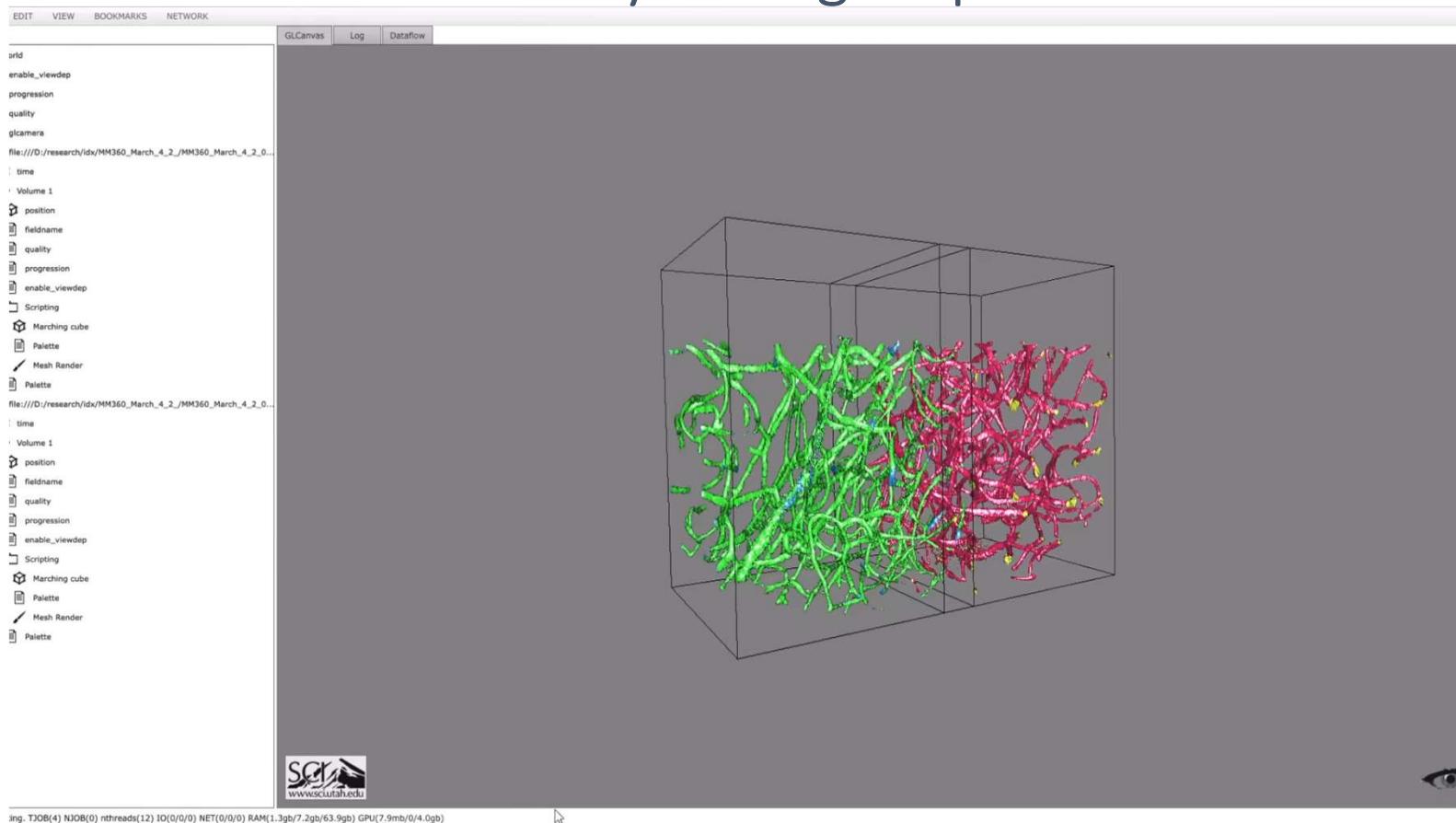
(4) Interactive, Exploratory Assessment and Feedback



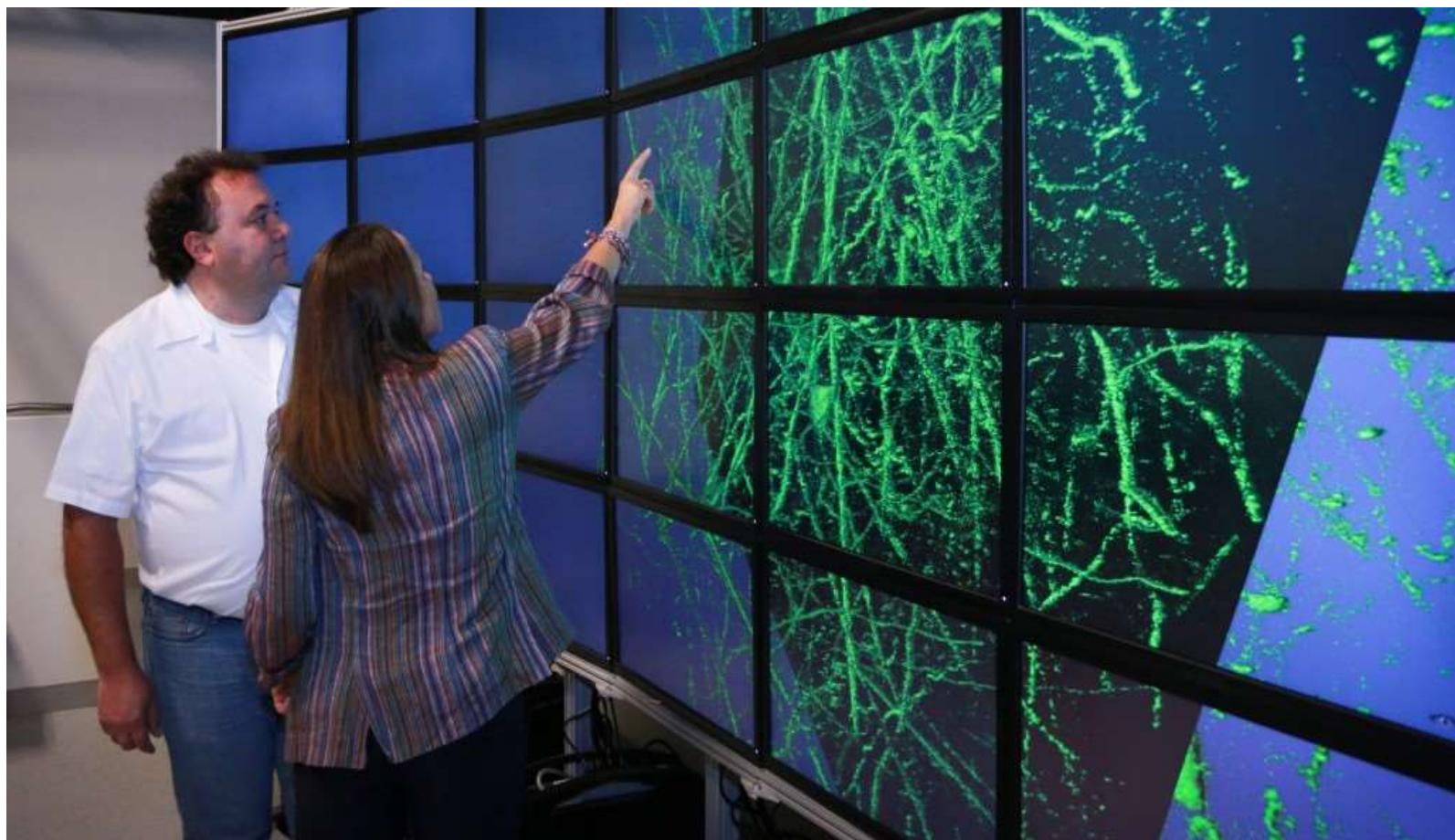
(3) Asynchronous Parallel Processing



Remote Monitoring of Data Quality During Acquisition



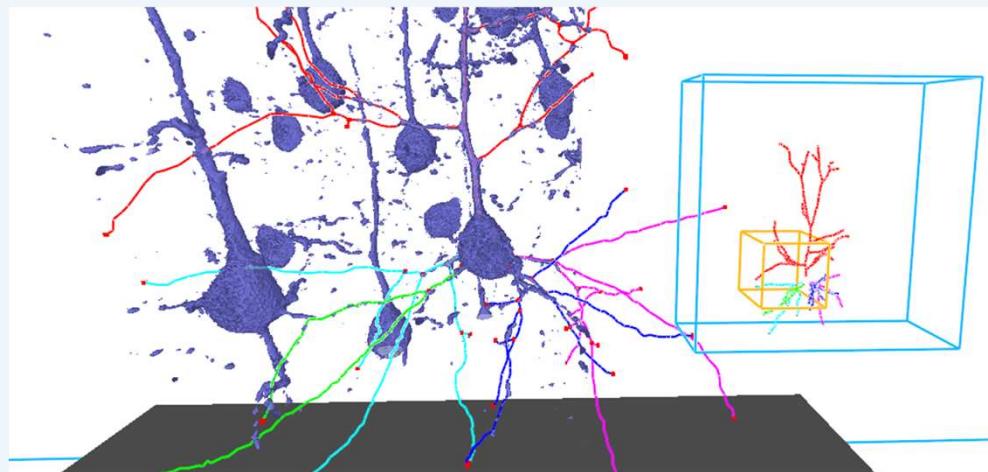
Online Acquisition and Interactive Visualization of Terascale Microscopy



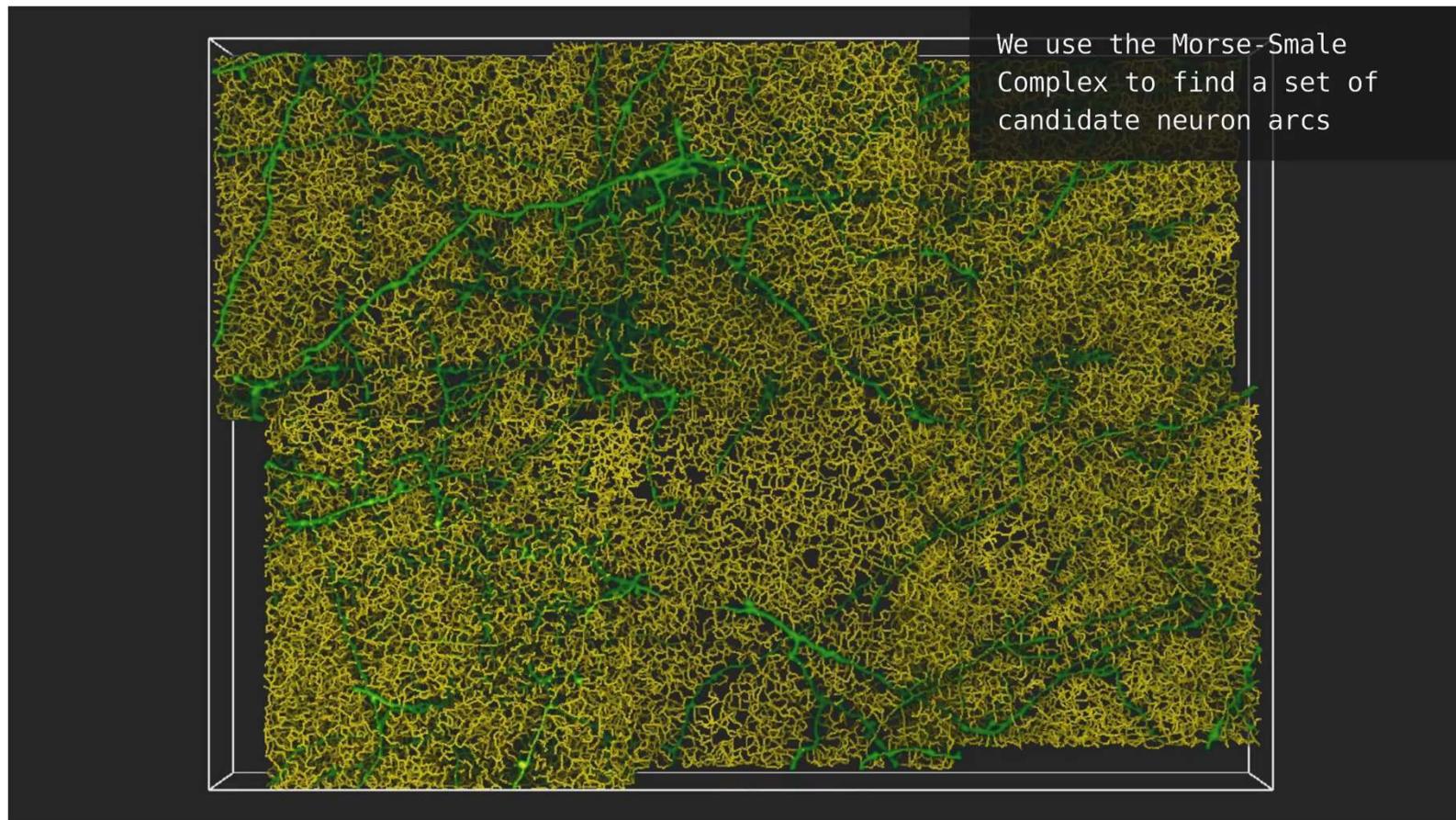
A Virtual Reality Visualization Tool for Neuron Tracing (VRNT)

Conducted a design study with neuroanatomists at the Moran Eye Center to develop a new tool for manual neuron tracing in VR

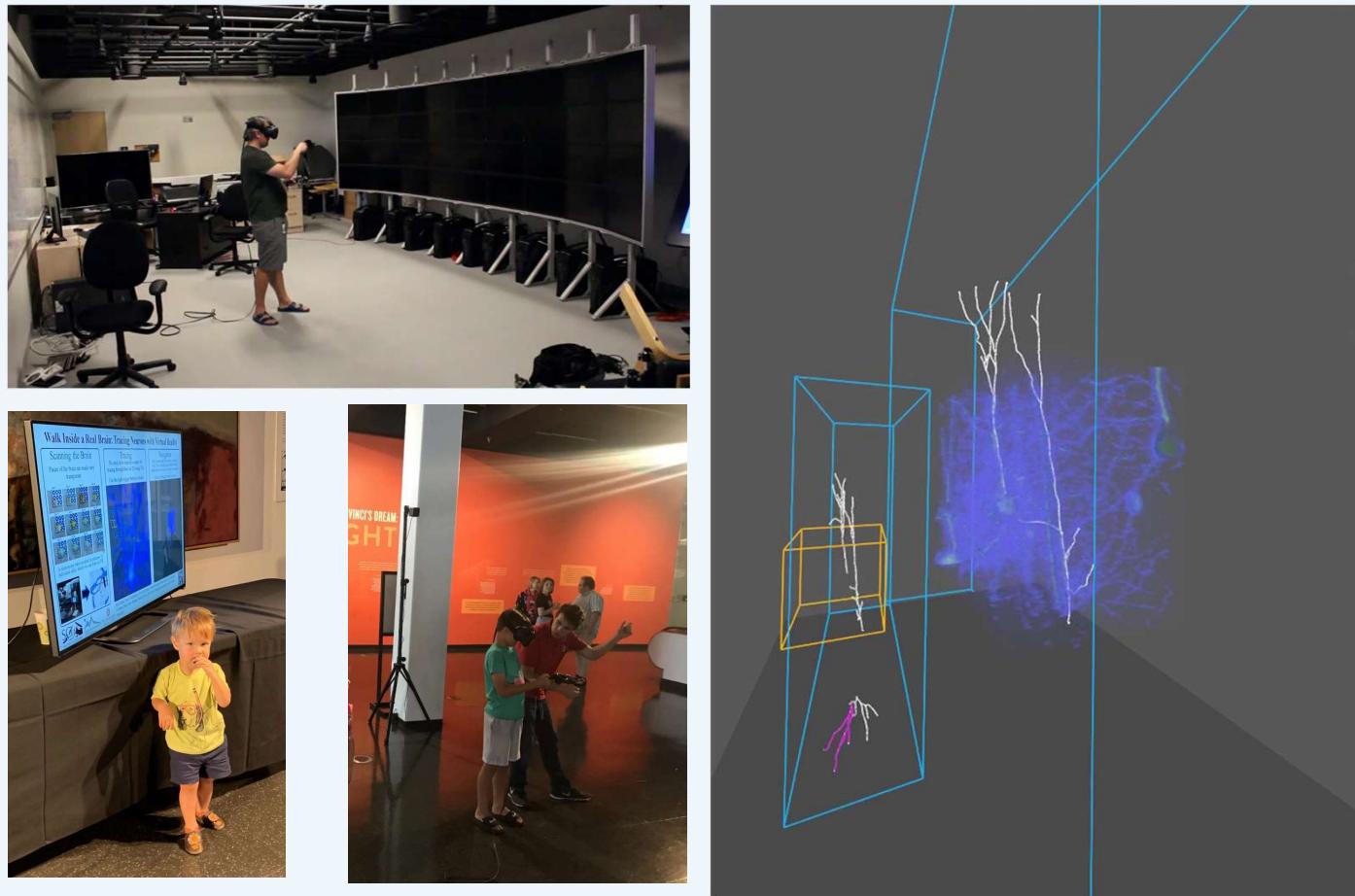
Tracing works similar to tilt brush, faster and more intuitive than desktop software



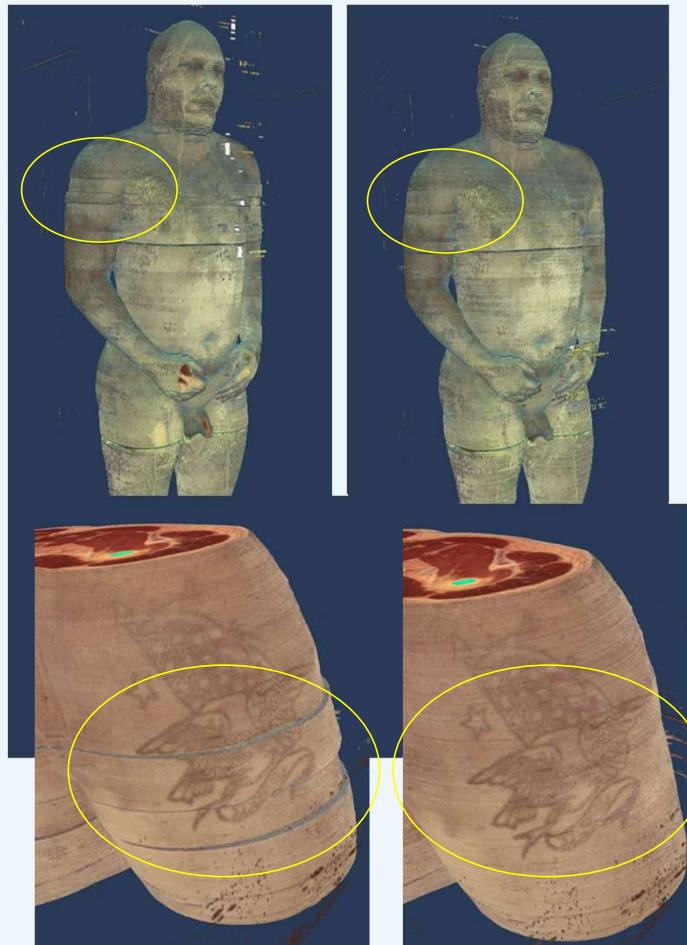
Topology Aided Neuron Tracing in a Virtual Environment



Evaluation in a neuroscience lab and outreach in a science museum



Python (Jupyter) scripting for local/remote data processing and visualization on demand



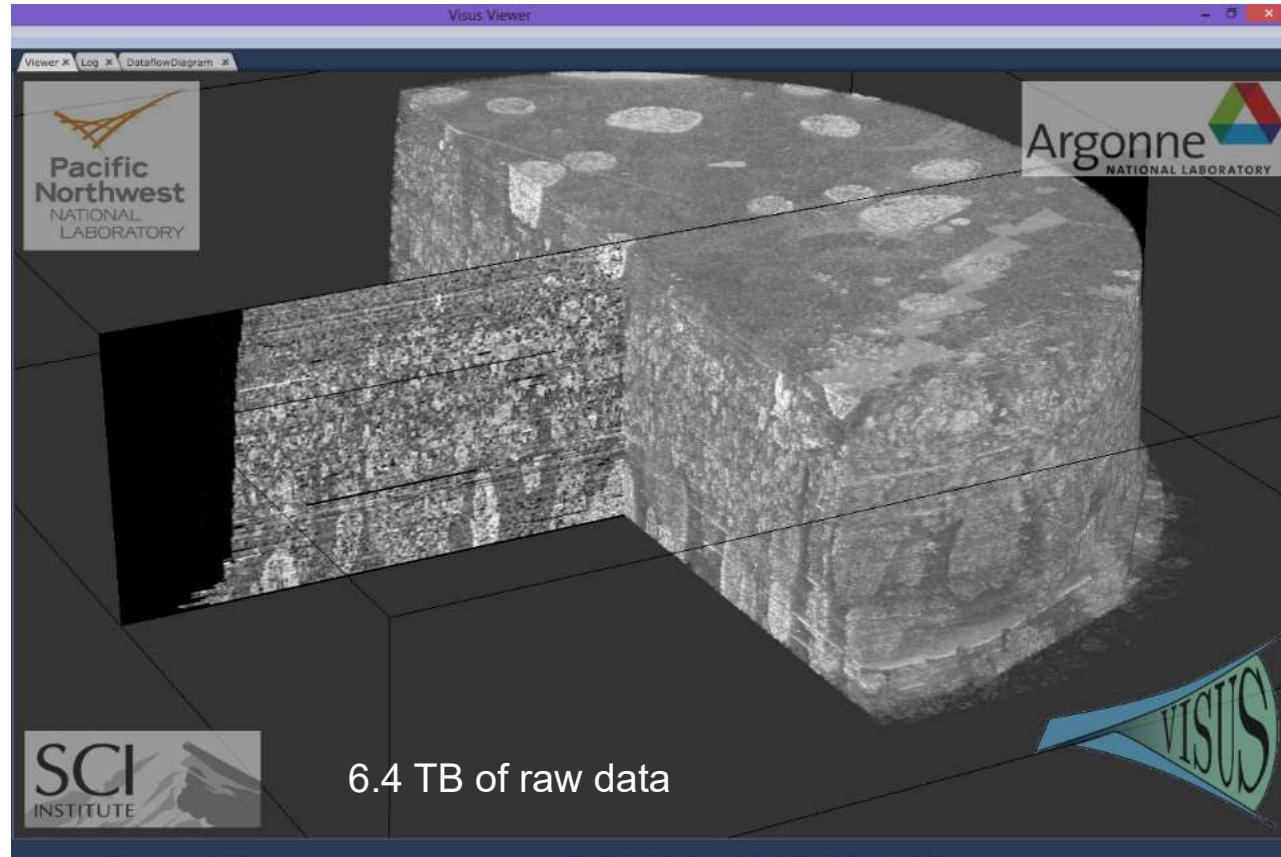
PyDataset and **PyViewer** to simplify data exploration

```
region=(0.04, 0.95, 0.05,0.73,  
0.15,0.15+0.1)  
dataset=LoadDatasetPy("visus.idx")  
RGB, bounds=dataset.readData(region,-6)  
  
...  
viewer=PyViewer()  
viewer.addVolumeRender(RGBA, bounds)
```



Demo: Interactive Remote Analysis and Visualization of 6TB Imaging Data

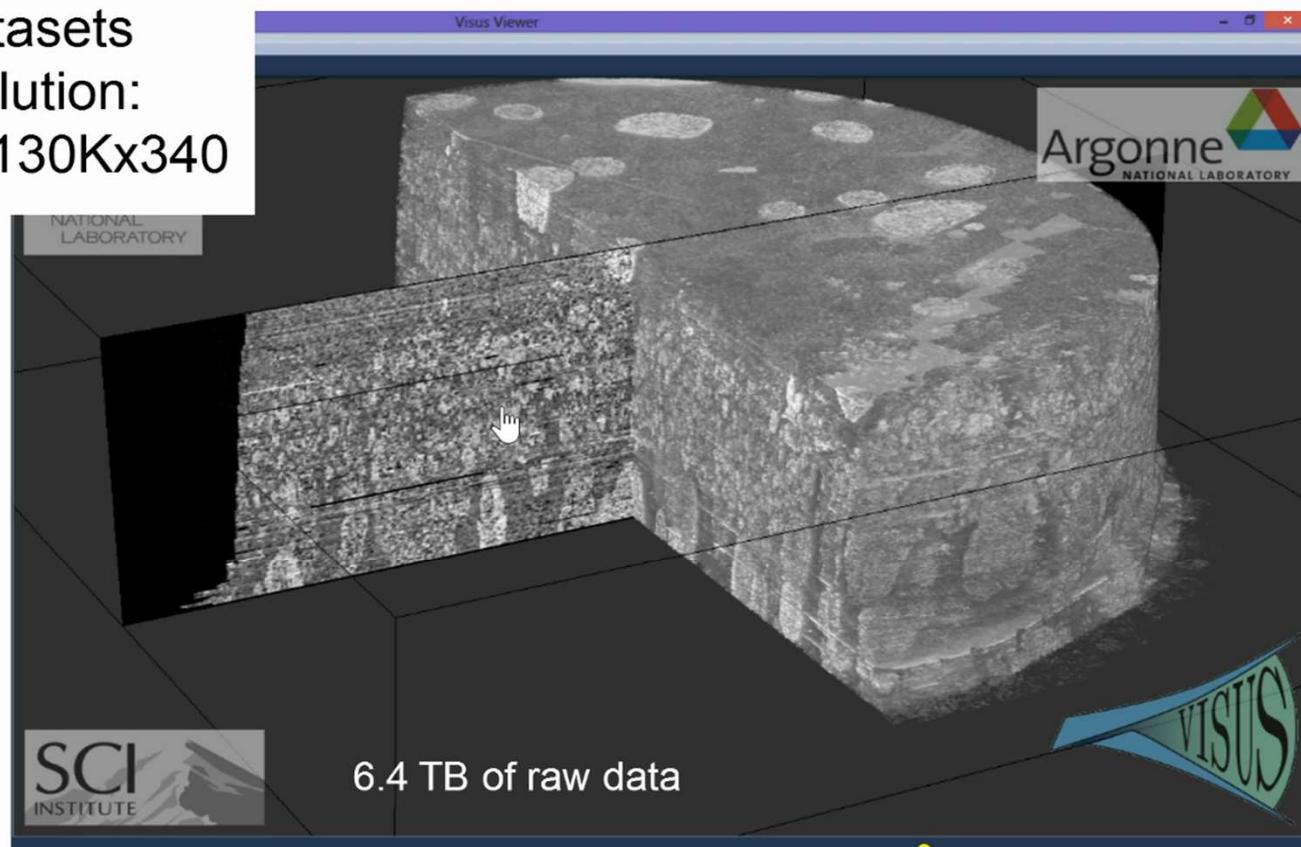
- EM datasets of resolution: 130Kx130Kx340



Web Server

Demo: Interactive Remote Analysis and Visualization of 6TB Imaging Data

- EM datasets of resolution: 130Kx130Kx340



Web Server

Pascucci-23



Deployment targets and extensions

Live addition to NEON website

- Identify data products
- Complete data ingestion pipeline
- Dedicated data portal
- Management of geospatial coordinate system
- Interoperability with google maps for context
- Implement testing procedures
- Basic enhancements of visualization and navigation
- Improve embedding (e.g., share)

Continuing support and advanced features

- Scaling of and addition of new data products
- Improve color blending of orthomosaics
- Python support (for download and scripting)
- Collaborative user interface
- Support for LIDAR data
- Support for user/NEON data upload

References

- Utah endpoint
<https://neon.visus.org/neonapi/products/{productCode}>
- Endpoint and web viewer source code repository
<https://github.com/sci-visus/neon-visus>
- Documentation and other use cases
www.visus.org