

# ***3D Visualization with ParaView***

**Adam Lyon, SCD/Muon g-2**

**Computing Techniques Seminar November 3, 2015**

# Why we're here

---

- o **Why we're (g-2, JimK & I) are looking at ParaView?**
- o **What makes ParaView unique?**
- o **The niche it may fill**
- o **Questions you should ask yourself before considering**
- o **Just DEMO it already (OK - will go through the above quickly)!**

## Why do I have anything to say on this?

**I'm a Scientist II on Muon g-2 & Head of SCD SSA Quadrant;**

**In 1998 I arrived at Cornell as a CLEO/U of Rochester Postdoc to find everyone only used black and white monitors (ick!). Wrote a python based event display in order to get a color one (wouldn't you like to see that track in purple?)**

# The problems

---

**Muon g-2 was facing several problems where visualization could be very helpful**

- o **Validation of our Geant geometry**

**We had hints of incorrect positions in the geometry**

- o **Debugging of Magnetic fields in Geant**

**We have some complicated and time-varying fields (kicker magnets) – needed verifying**

- o **Debugging Tracking (the usual stuff)**

**Comparison of reconstructed hits & tracks to truth hits & trajectories – needs to run post-grant**

# No easy solutions

---

- o Validation of our Geant geometry

*We seemed to reach the performance limit of Geant 3D viewers  
Geant/OpenGL hard to control, HepRApp & JAS3/Wired4 painful,  
Didn't try Geant/Qt since wouldn't help with track debugging*

- o Debugging of magnetic fields in Geant

*Not clear how to visualize the fields on top of geometry with  
Geant/Root tools*

- o Debugging Tracking

*Need to superimpose reconstruction & Geant views  
for comparison (view Geant info without running Geant)  
Root/Eve could not faithfully display our Geant-generated GDML  
Not many other options here*

**Bigger Problem: The SCD no longer has expertise in visualization. All experiments writing their own viz apps - no one solution**

# Simple requirements

---

*This visualization niche: Aid for validation, debugging, development*

Requirements (or maybe expectations, perhaps realized after the fact):

**Fast image manipulation (rotate, pan, zoom)**

**Ability to hide objects (e.g. certain detector components)**

**Ability to ingest data from simple text or CSV files (for debugging)**

**[e.g. how do you debug stuck Geant when you can't get the regular output?]**

**Ability to superimpose data from different sources**

**[e.g. see 3D Model from CAD on top of Geant geometry]**

**I didn't want to have to write a lot of code, nor generate code that would be hard to maintain (e.g. I don't want to change Geant)**

**Ability to make pretty pictures/movies doesn't hurt (but not a specific requirement)**

# ParaView

Jim Kowalkowski had mentioned and briefly played with ParaView – thinking about it to fill visualization needs of art users

So I starting looking at it and liked what I saw

Written with 3D performance as top priority

Based on very mature VTK toolkit

Note the logos →



# Features of ParaView

---

**Applies principles from the visualization & supercomputing communities**

**Extremely responsive scene manipulation (rotate, pan, zoom)**

**Image quality is degraded (Level of Detail) during manipulation for speed. Configurable**

**Many tools for visualization of scalar and vector fields**

**Arrows, streamlines, heat maps, volume rendering (opacity)**

**Easy slicing, cutting, thresholds, “warping”**

**Can make 2D plots of aspects of the 3D data**

**Animation**

**Pipeline metaphor – sources, filters, and sinks**

**Very easy to overlay data using multiple sources (easy transformation if necessary - cm to mm)**

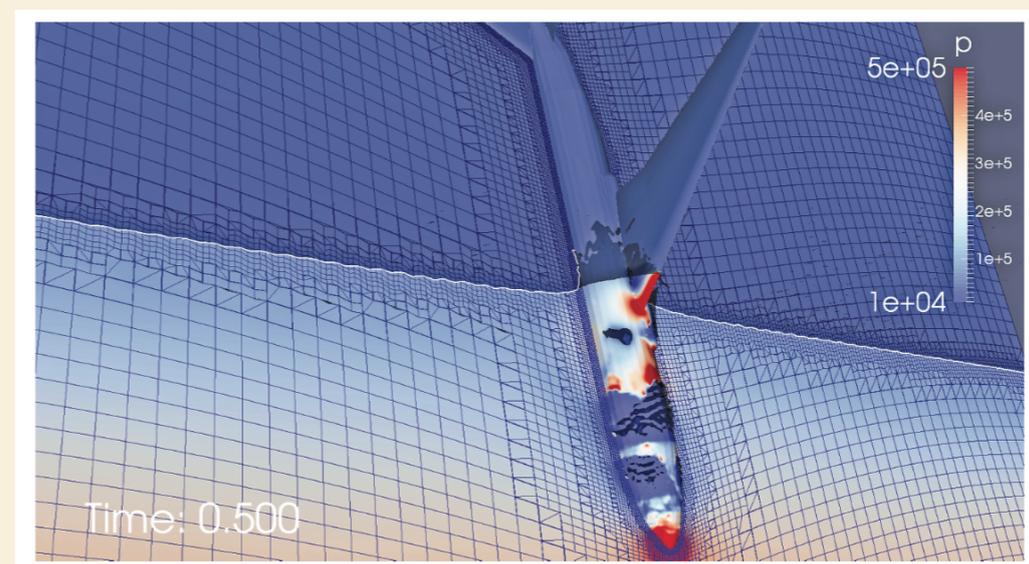
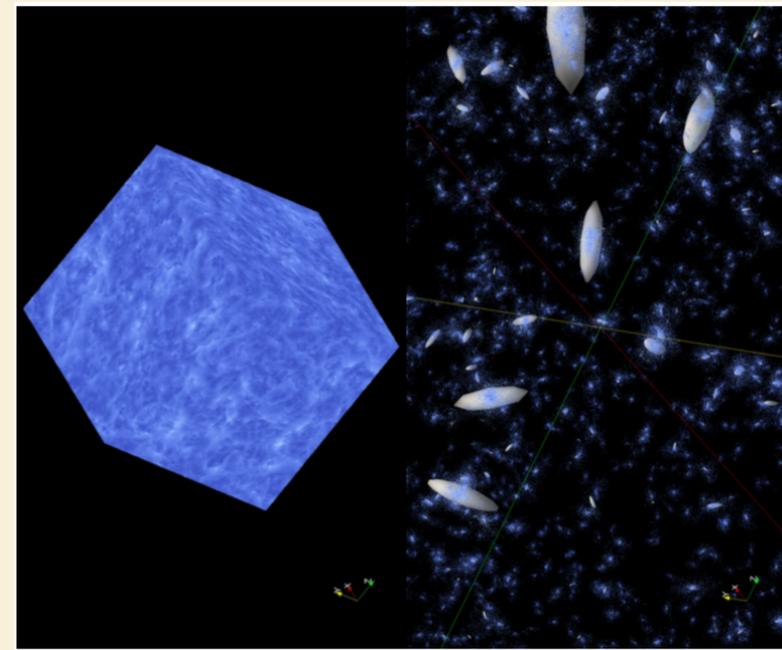
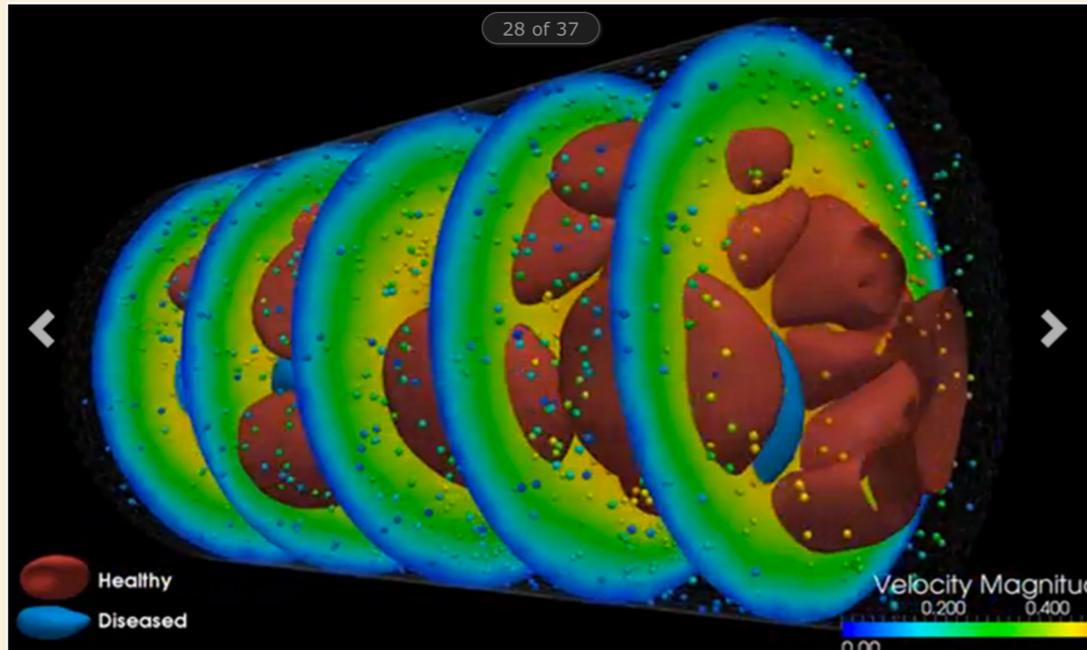
**Can write your own sources and filters in Python with built in numpy & matplotlib**

**Deep automation with Python**

**Less domain specific (certainly not tuned to our domain)**

# Who uses ParaView?

**Mesh based simulations, especially computational fluid dynamics (CFD) @ Supercomputer Centers**



See <http://www.paraview.org/gallery/>

Chen, et. al., <http://dx.doi.org/10.1090/noti1236>

# ParaView Support

---

**HEP simulations are not usually mesh-based:**

- o We're pushing on a tool that's not our domain
- o But this tool seems to be flexible enough to work in many cases
- o There's a lot of interest in adding HEP users

**I've found the support from Kitware to be excellent and they are interested in adding HEP science to their portfolio – Enabling Science is important**

**Every Supercomputing center has a visualization group, and they do ParaView (+ other applications [VisIt])**

**The Argonne ALCF Visualization group (Joe Insley) is interested – and they have a big visualization system**

# How do we get our data into ParaView?

---

**ParaView speaks VTK!**

**Many readers for file formats we don't use**

**BUT, it's not hard to make VTK objects (in C++ & Python) and integrate directly in ParaView**

- It's a pain to translate (but not hard after learning)**
- + Make better use of ParaView/VTK's features when you speak its language (my opinion)**

# Readers

- ParaView Data (.pvd)
- VTK (.vtp, .vtu, .vti, .vts, .vtr)
- VTK Legacy (.vtk)
- VTK Multi Block (.vtm, .vtmb, .vtmg, .vthd, .vthb)
- Partitioned VTK (.pvtu, .pvti, .pvts, .pvtr)
- ADAPT (.nc, .cdf, .elev, .ncd)
- ANALYZE (.img, .hdr)
- ANSYS (.inp)
- AVS UCD (.inp)
- BOV (.bov)
- BYU (.g)
- CCSM MTSD (.nc, .cdf, .elev, .ncd)
- CCSM STSD (.nc, .cdf, .elev, .ncd)
- CEAcud (.ucd, .inp)
- CMAT (.cmat)
- CTRL (.ctrl)
- Chombo (.hdf5, .h5)
- Claw (.claw)
- Comma Separated Values (.csv)
- Cosmology Files (.cosmo, .gadget2)
- Curve2D (.curve, .ultra, .ult, .u)
- DDCMD (.ddcmd)
- Digital Elevation Map (.dem)
- Dyna3D(.dyn)
- EnSight (.case, .sos)
- Enzo boundary and hierarchy
- ExodusII (.g, .e, .exe, .ex2, .ex2v., etc)
- ExtrudedVol (.exvol)
- FVCOM (MTMD, MTSD, Particle, STSD)
- Facet Polygonal Data
- Flash multiblock files
- Fluent Case Files (.cas)
- GGCM (.3df, .mer)
- GTC (.h5)
- GULP (.trg)
- Gadget (.gadget)
- Gaussian Cube File (.cube)
- JPG Image (.jpg, .jpeg)
- LAMPPS Dump (.dump)
- LAMPPS Structure Files
- LODI (.nc, .cdf, .elev, .ncd)
- LODI Particle (.nc, .cdf, .elev, .ncd)
- LS-DYNA (.k, .lsdyna, .d3plot, d3plot)
- M3DCI (.h5)
- MFIX Unstructured Grid (.RES)
- MM5 (.mm5)
- MPAS NetCDF (.nc)
- Meta Image (.mhd, .mha)
- Miranda (.mir, .raw)
- Multilevel 3d Plasma (.m3d, .h5)
- NASTRAN (.nas, .f06)
- Nek5000 Files
- Nrrd Raw Image (.nrrd, .nhdr)
- OpenFOAM Files (.foam)
- PATRAN (.neu)
- PFLOTRAN (.h5)
- PLOT2D (.p2d)
- PLOT3D (.xyz, .q, .x, .vp3d)
- PLY Polygonal File Format
- PNG Image Files
- POP Ocean Files
- ParaDIS Files
- Phasta Files (.pht)
- Pixie Files (.h5)
- ProSTAR (.cel, .vrt)
- Protein Data Bank (.pdb, .ent, .pdb)
- Raw Image Files
- Raw NRRD image files (.nrrd)
- SAMRAI (.samrai)
- SAR (.SAR, .sar)
- SAS (.sasgeom, .sas, .sasdata)
- SESAME Tables
- SLAC netCDF mesh and mode data
- SLAC netCDF particle data
- Silo (.silo, .pdb)
- Spherical (.spherical, .sv)
- SpyPlot CTH
- Spy Plot (.case)
- Stereo Lithography (.stl)
- TFT Files
- TIFF Image Files
- Tsurf Files
- Tecplot ASCII (.tec, .tp)
- Tecplot Binary (.plt)
- Tetrad (.hdf5, .h5)
- UNIC (.h5)
- VASP CHGCA (.CHG)
- VASP OUT (.OUT)
- VASP POSTCAR (.POS)
- VPIC (.vpc)
- VRML (.wrl)
- Velodyne (.vld, .rst)
- VizSchema (.h5, .vsh5)
- Wavefront Polygonal Data (.obj)
- WindBlade (.wind)
- XDMF and hdf5 (.xmf, .xdmf)
- XMol Molecule

From Bill Sherman, IU, SC14 Workshop

# Ingesting Data

---

**Need to convert your data to VTK objects**

**So far I've done**

- o **Geant's HepRepXML format via my GeantToVTK Python ParaView Plugin**
- o **Text [csv] files via external python scripts (pvpython) - I have a generic script**
- o **Direct translation of art objects within art via C++/Catalyst [A real event display]**
- o **Specialty "Python Programmable source" within ParaView (e.g. Read numpy file)**

**Translation from Root files would probably not be hard (if you build ParaView and Root with the same Python, ParaView could do "import ROOT")**

**Dumb text files can be crucial (e.g. stuck Geant use case)**

o **Can connect ParaView to a running process supplying VTK objects (in-situ processing/Catalyst) - but communication back to process is very limited**

o **ParaView can run in parallel mode, using multiple CPUs and GPUs on a remote cluster displaying locally (e.g. our Lattice QCD GPU cluster). I've tried 4 CPU's/GPU's and had mixed results (probably don't have a big enough problem yet) – don't underestimate the power of the GPU in your laptop.**

# Questions for you

---

**What are your specific visualization needs?**

**Where does viz fit in to your work?**

- o Perhaps many places
- o g-2's specific use cases have been a good guide

**What niche(s) are you trying to fill?**

- o The “one event display to rule them all” remains elusive
- o While ParaView is very useful & flexible, I don't think it'll fit every use case
- o E.g. Probably not good for highly interactive tracking debugging with real-time algorithm communication
- o E.g. Probably not good for display kiosks operated by public

**What's a demo that would help you decide?**

**[Requirements? Expectations?]**

# Next steps

---

**On g-2 we've been pleasantly surprised/amazed by ParaView. Extremely useful for debugging Geant**

**There's a lot more to visualization than we know**

**Careful - it's fun** (and a bit addictive, you'll want to use it everywhere)

**SCD Visualization Strategy may involve ParaView**

[attractive: Mature system with broad outside support, aligns us with HPC/ASCR community]

**Documenting things so far**

<http://lyon-fnal.github.io/paraviewForHep/>

**Just DEMO it already...**

# Demo 1 - General ParaView

---

- o Download it yourself from [www.paraview.org](http://www.paraview.org)
- o See Manual at <http://www.paraview.org/paraview-guide/>
- o Note pipeline
- o Make Wavelet (see help); Apply Button
- o Info - *Scalar field*
- o Coloring; Surface; Volume (slow) - change opacity
- o Camera movement (Camera Undo); Note LOD
- o Filters - slice (one, multiple) - delete
- o Clip (eyeballs) - delete
- o Contour - then slice

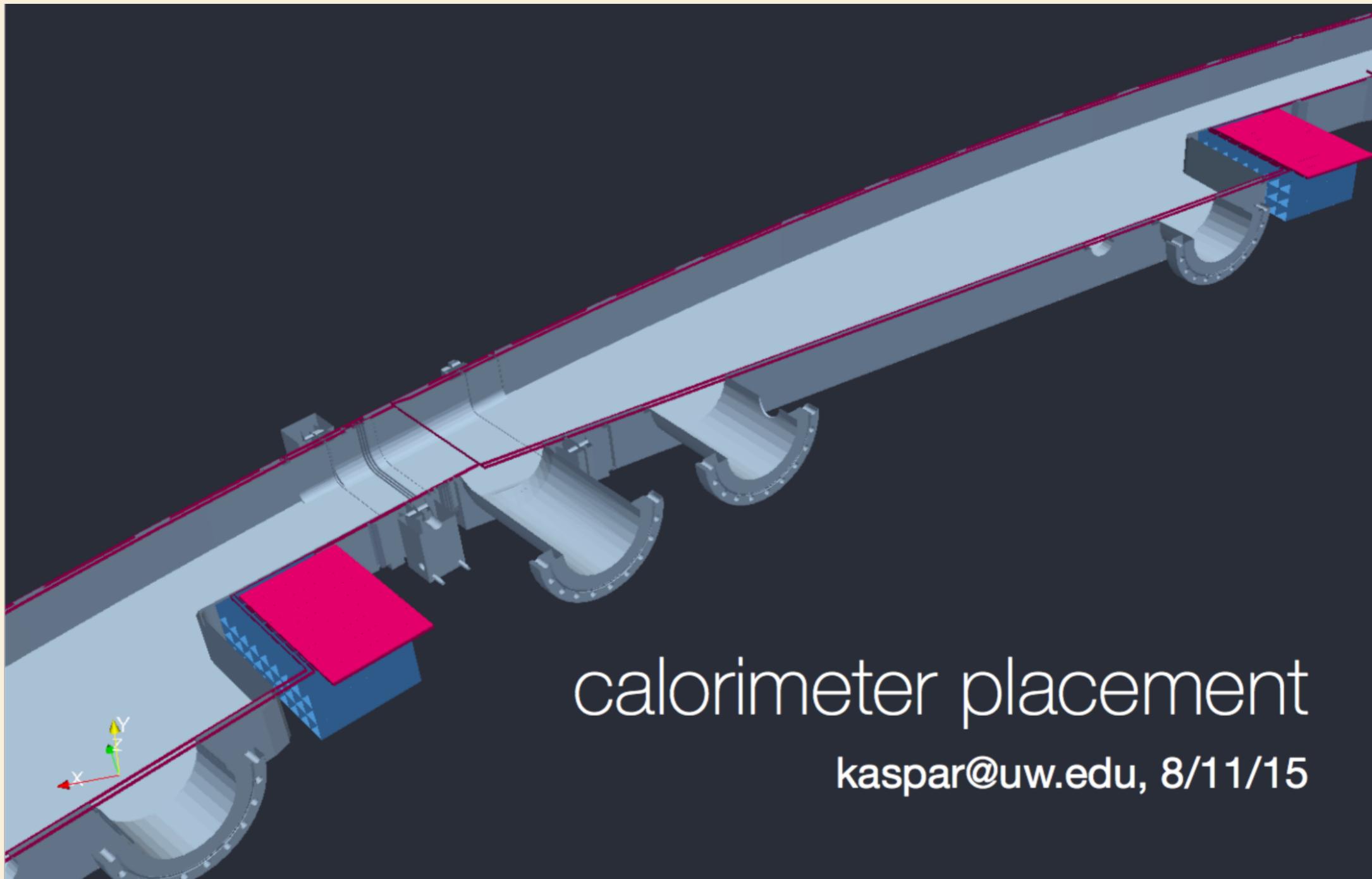
# Demo 2 - g-2 Ring

---

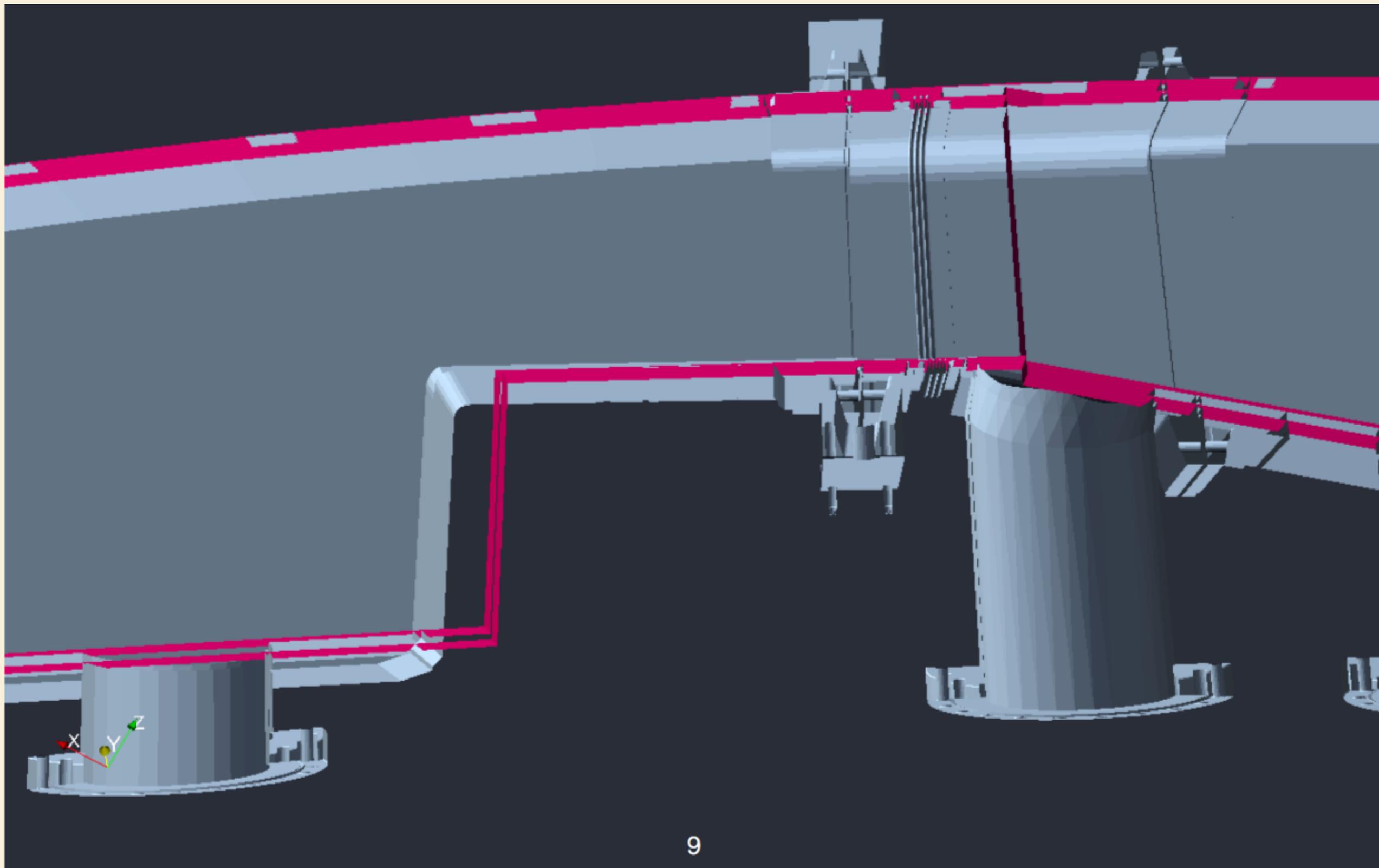
- o show file, unzip -l
- o Load g-2 ring from HepRepXML file; note options
- o -Y view, change colors
- o Multi-block inspector
- o Front face culling
- o Right click on inflector green circle thing
- o Color by material
- o Change center of rotation
- o Wireframe/surface
- o save state; restart; reload
- o Load event (single); -SUPERIMPOSING OF VIZs- Rename sources
- o multi block inspector on events (note e is very short)
- o Event particle table
- o Color by PDG
- o VCR Controls
- o On event (time) 4, select block, See in field data spreadsheet
- o Select muons only from Event source
- o Look at event 23
- o Remember - these are reading in from a file
- o Remove event; Load overlay events 0:5

# The problem

Calorimeter Placement in gm2ringsim (Jarek / [GM2-doc-3032](#) ) - using Inventor and ParaView

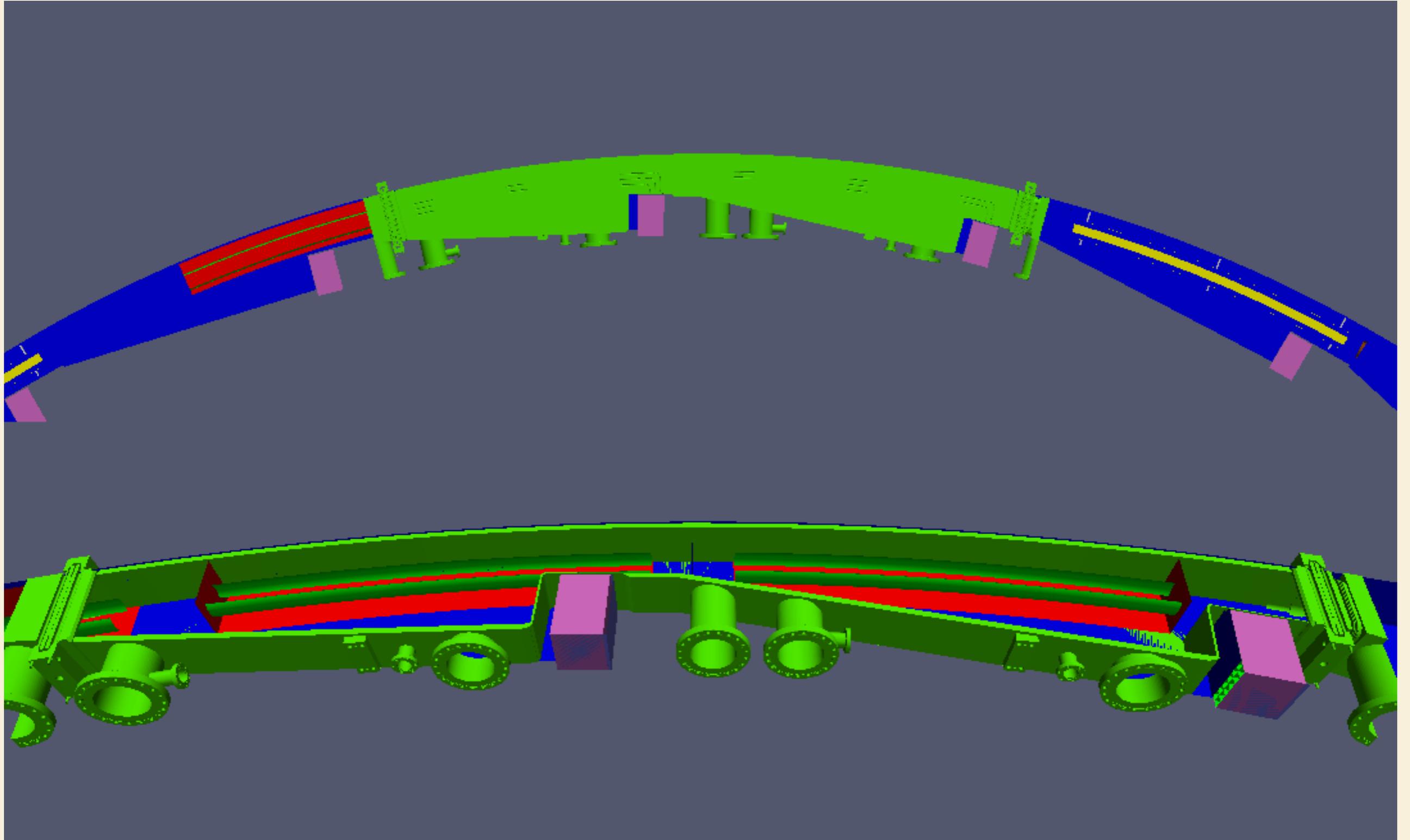


# The problem



9

# The problem



# Demo 3 - Live art

---

- o **Start ParaView with CatalystLiveButtons**
- o **Note that anything to do with time (animation) will crash ParaView**
- o **Flip through some events**
- o **Save an event for later**
- o **Make a normalized time  $\text{globalTime}/\text{max}(\text{globalTime})$**
- o **Try event 8**
- o **Overlay ring**
- o **Color by pdg**
- o **Make line width 4**
- o **Threshold by global time**
- o **Can really study the events**
- o **Show making a movie**
- o **Show movies**
  
- o **Go change FCL to stop on calorimeter hits**
- o **Show making spheres out of hits**

# Demo 4 - Kicker field

---

- o Look at files in demo3/kickerdata
- o Load geometry (choosing \[03\], fix colors, cull front face)
- o Load field scan (note multi-timesteps), show info (29M points)
- o Try surface
- o Try volume - VOLUME only works on scalar field, use calculator, By is what matters
- o Try slices - y, camera normal
- o Try stream tracer on point -1358, 0, 6851, 10000 points, radius 1000
- o Glyphs - arrows - scale by vector, factor 120, 100000 points
- o Maybe try a contour

## Show animation

### Try front animation

- o Load geometry (wireframe), fields
- o Set 2D
- o Slice in along camera
- o Set colors to 1.4, 1.5 (hit upper gear for annotation),
- o Need two calculators - one for By and another for y
- o Threshold
- o plot over line in front of calorimeter (use 2d, just show slice), left axis custom range
- o Add time annotation
- o Do animation - 10 frame/s, 1 per frame

# Demo 5 - LArTPC fun!

---

- o **Load wv3d\_clean.pvsm**
- o **Show histograms, warps (noisy first)**
  
- o **Load wv3d\_add.pvsm**
- o **Try volume rendering**
  
- o **Load wv3d\_match.pvsm**
- o **Histogram**
- o **Threshold at 50 - all scalars off!**
  
- o **Maybe try others**