

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

**Adam Lyon, SCD/Muon g-2  
CS Liaison Meeting 2015-10-14**

# The problems

---

**Muon g-2 was facing several problems where visualization would be important**

- o **Validation of our Geant geometry**

**We had hints of detectors at wrong positions**

- o **Debugging of Geant fields**

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

- o **Debugging Tracking**

**Comparison of reconstructed hits & tracks to truth hits & trajectories**

# No easy solutions

---

- o Validation of our Geant geometry

*We seemed to reach the performance limit of standard Geant 3D viewers (did not try Geant with Qt)*

*Geant/OpenGL hard to control, HepRApp & JAS3/Wired4 painful*

- o Debugging of Geant magnetic fields

*Not clear how to visualize the fields 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*

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

# **My simple-minded requirements**

---

**[Perhaps 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)**

**Ability to superimpose data from different sources**

**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). I just don't have time to do that**

# 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**



# ParaView

---

[www.paraview.org](http://www.paraview.org)

**Built on top of VTK (Visualization Tool Kit)**

**C++ libraries (began in 1993 by GE) & *Python* wrapper**

**Authors started Kitware 1998**

**Strong early support at LANL and Sandia**

**Mainly supported by Kitware**

- o **Open source development company**
- o **Lots of Federal grants (DOE/ASCR & NIH)**
- o **Big efforts in medical applications (3DSlicer)**
- o **Big players in DOE High Performance Computing**
- o **Extremely responsive support**

# Features of ParaView

---

**Embodiment of what the visualization field has learned over the years + the kitchen sink (e.g. collaboration tools from a SLAC SBIR)**

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

**Image quality is degraded during manipulation for speed. Highly 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**

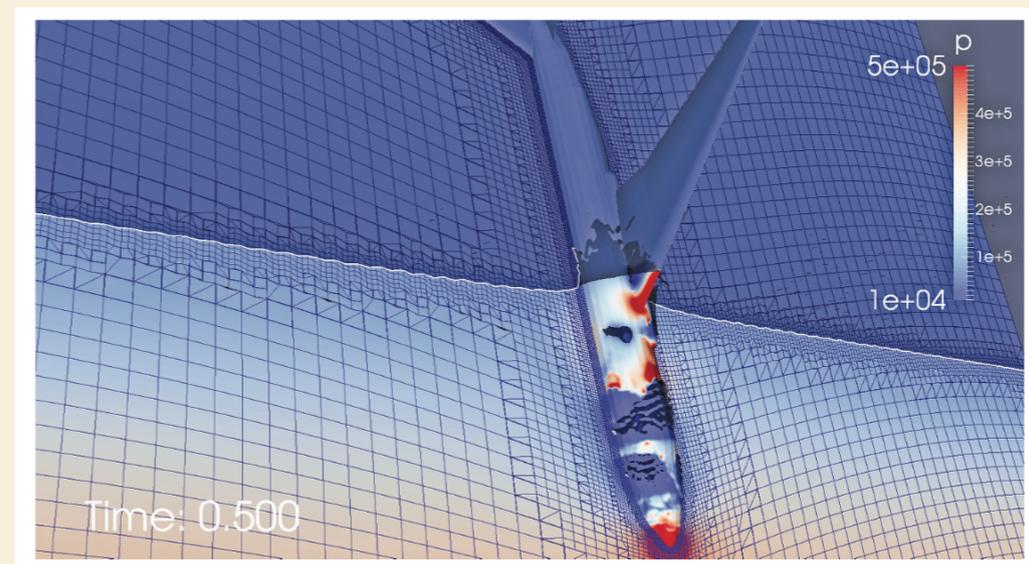
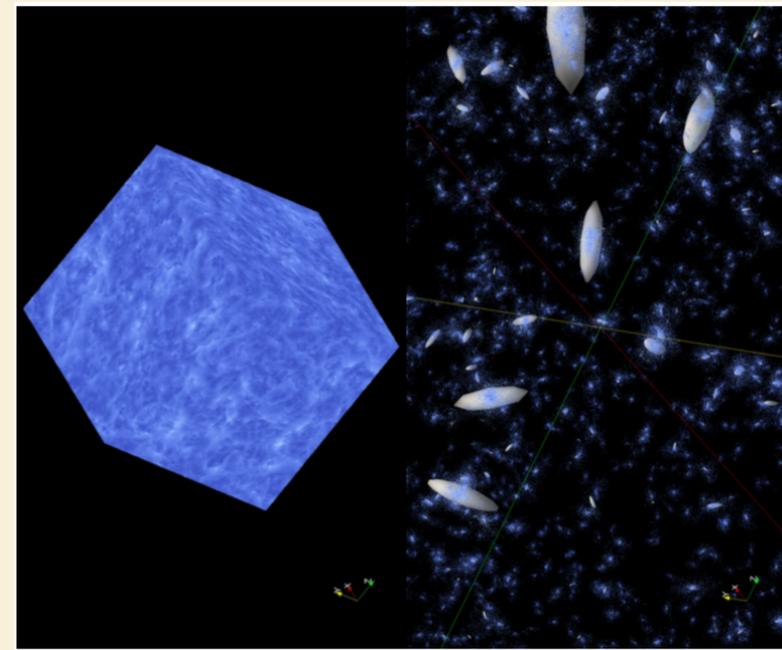
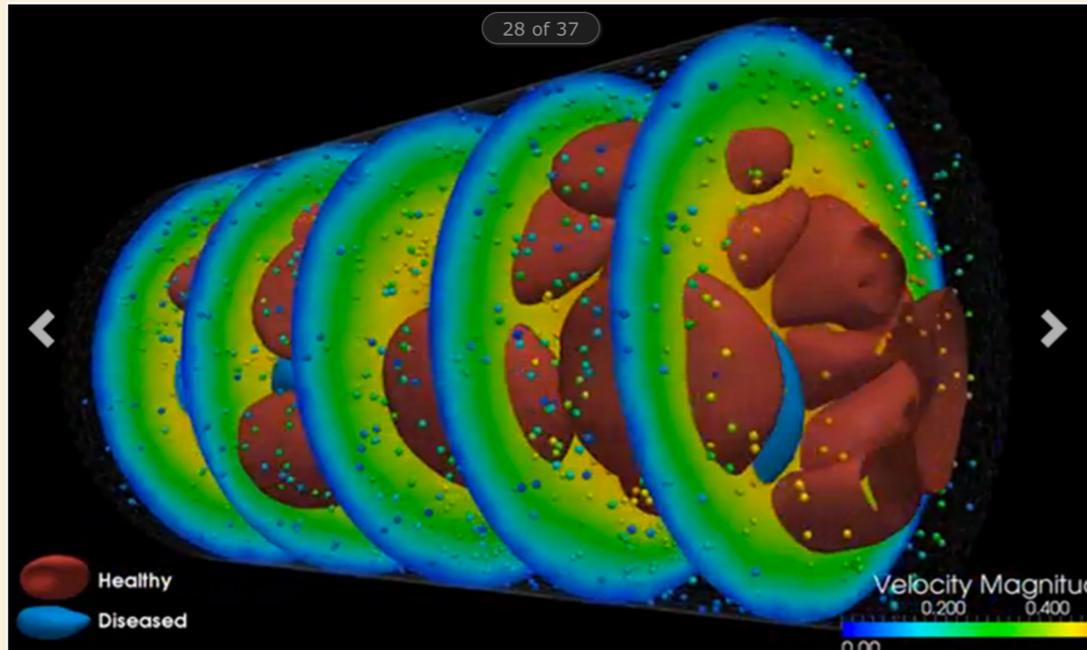
**Very easy to overlay data using multiple sources (easy transformation if necessary - cm to mm)**

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

**Deep automation with Python**

# 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, but ParaView is still suitable**

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

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

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

# ParaView reading Geant data

---

**Tried Geant VRML output. ParaView has reader**

**But, information in VRML was limited**

- o Monolithic**
- o Color info was incomplete**
- o Did not honor visibility**

**Showed promise (e.g. speed of manipulation)**

**VRML is kinda old anyway (replaced by x3d)**

# ParaView and HepRepXML

---

**HepRepXML (v2 HepRep): XML based output format with visualization information for JAS3/WIRED4 viewer**

**Complicated XML, but lots of meta-data**

- o **Geometry hierarchy with names**
- o **Color and visibility**
- o **Material information**
- o **Geant solid information**
- o **Geometry shortcuts (e.g. make cylinder instead of polygons)**
  
- o **Event information as well (momentum, energy, PDG #)**
- o **Trajectories and hits**
  
- o **Efficient - one file for geometry, many files for events, packaged in a zip file**

# GeantToVTK

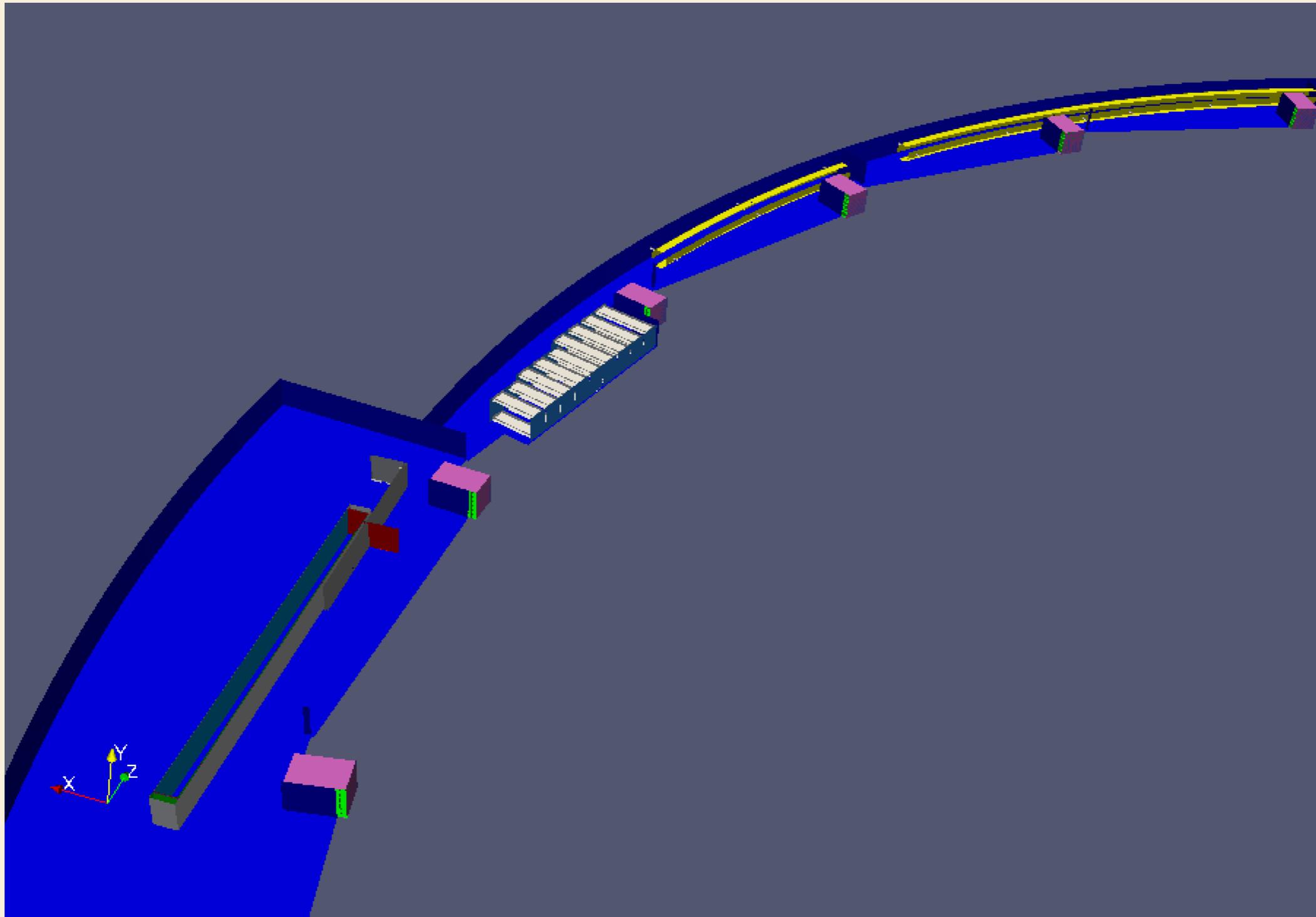
---

**My Python scripts to convert HepRepXML into VTK objects (vtkMultiblocks)**

**A Plugin for ParaView with Readers and Filters**

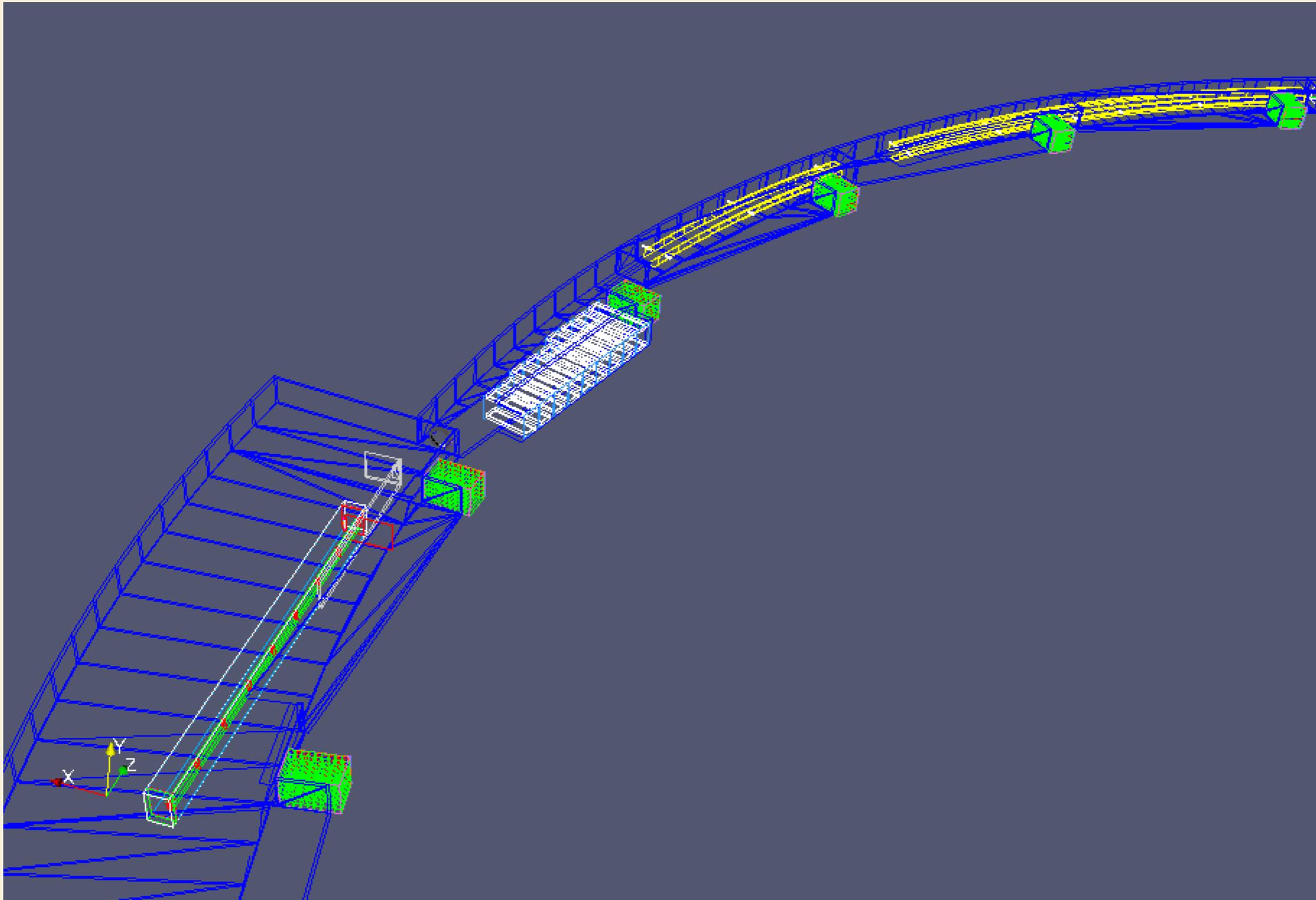
**See <https://cdcvs.fnal.gov/redmine/projects/geanttovtk/wiki/Wiki>**

# Muon g-2 Ring in ParaView

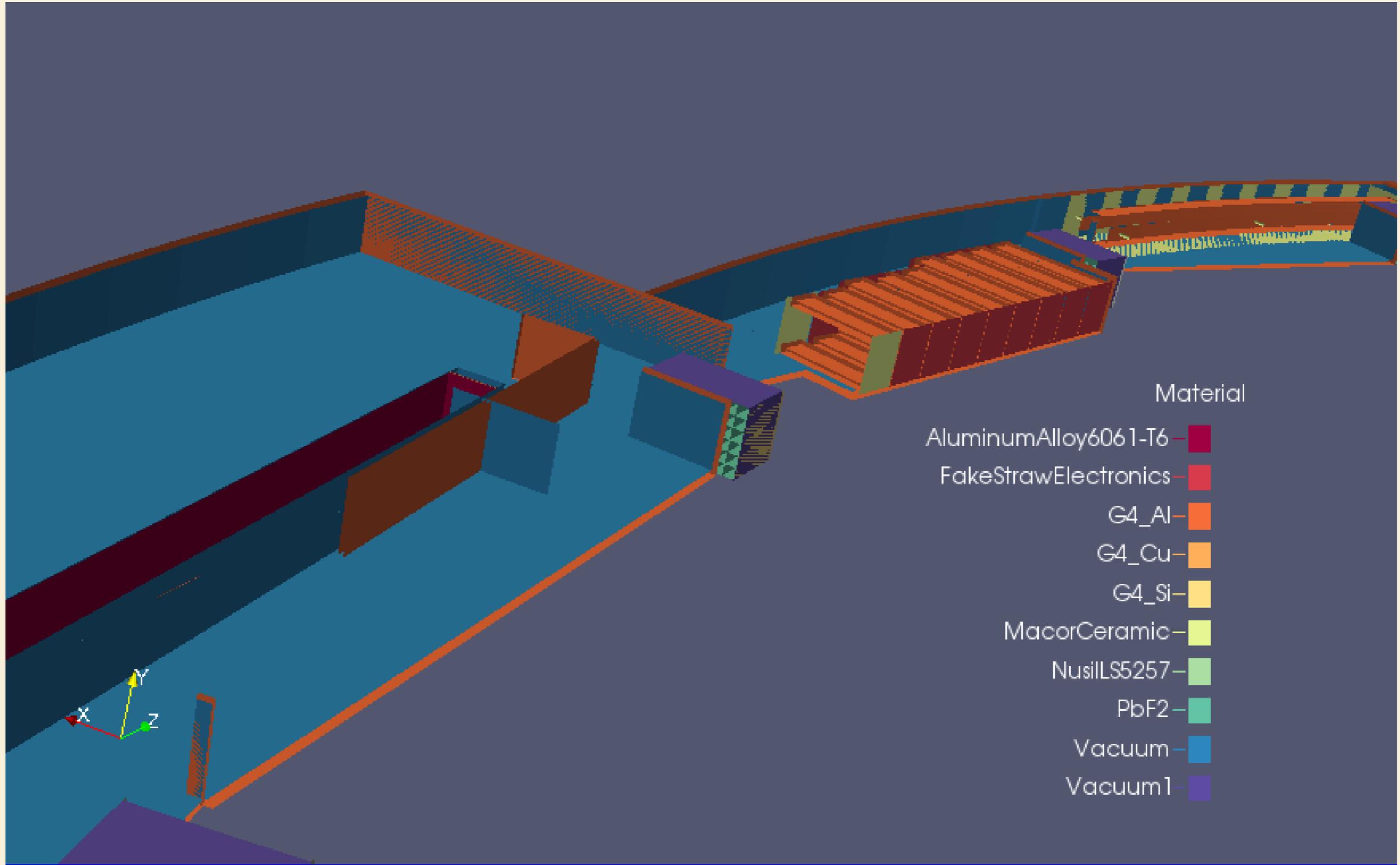


**Example of  
“Front face  
culling”**

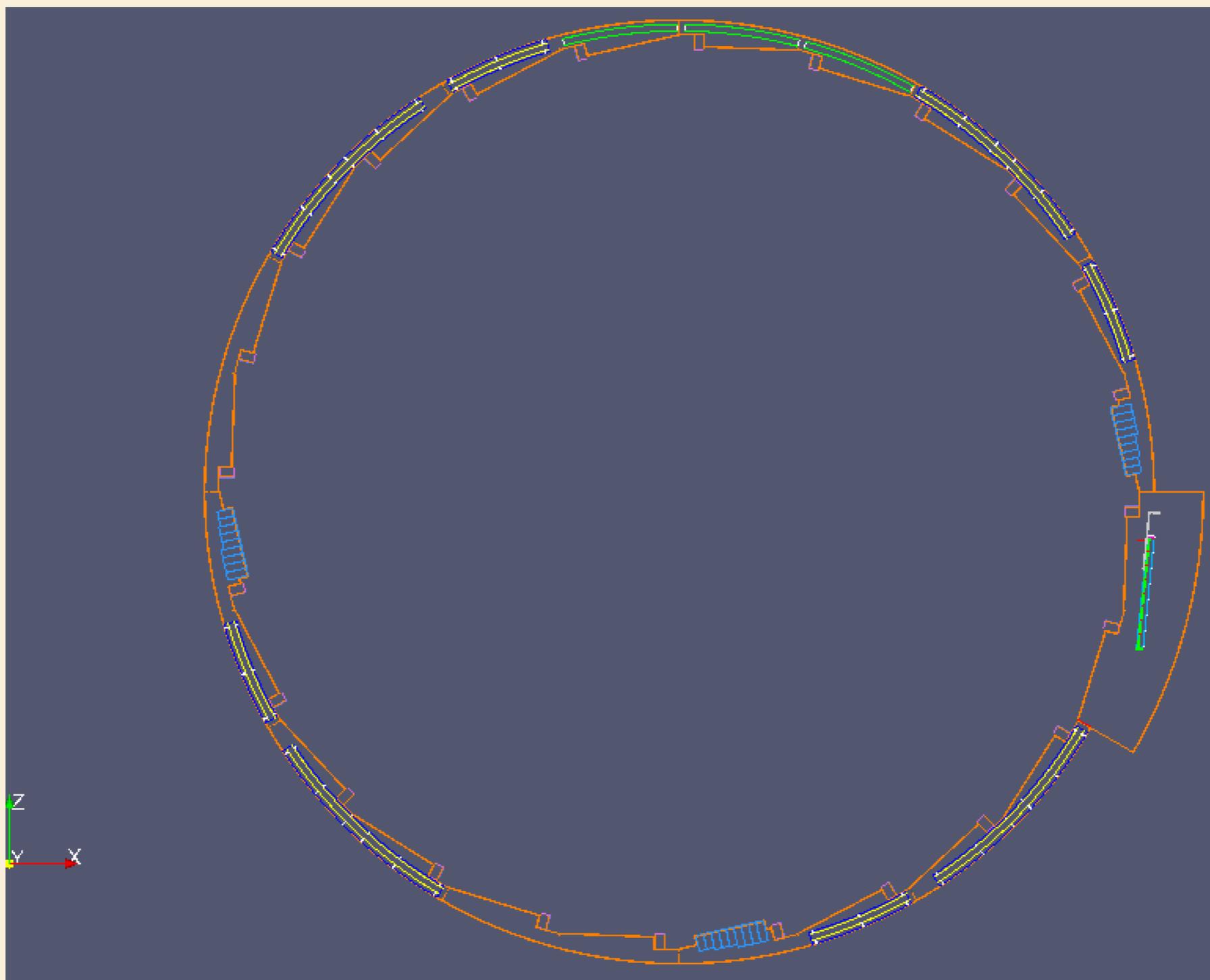
# Wireframe view



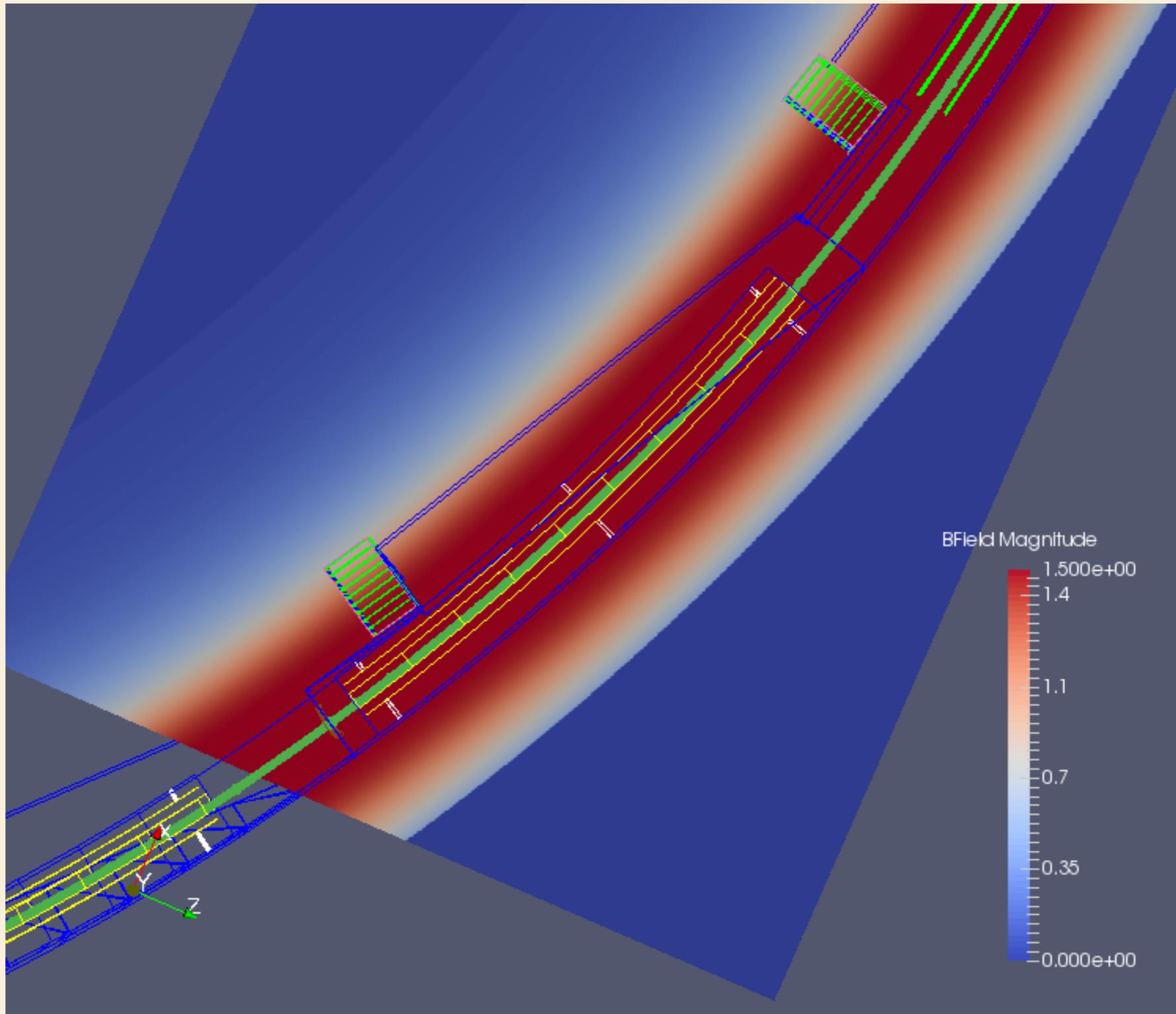
# Metadata



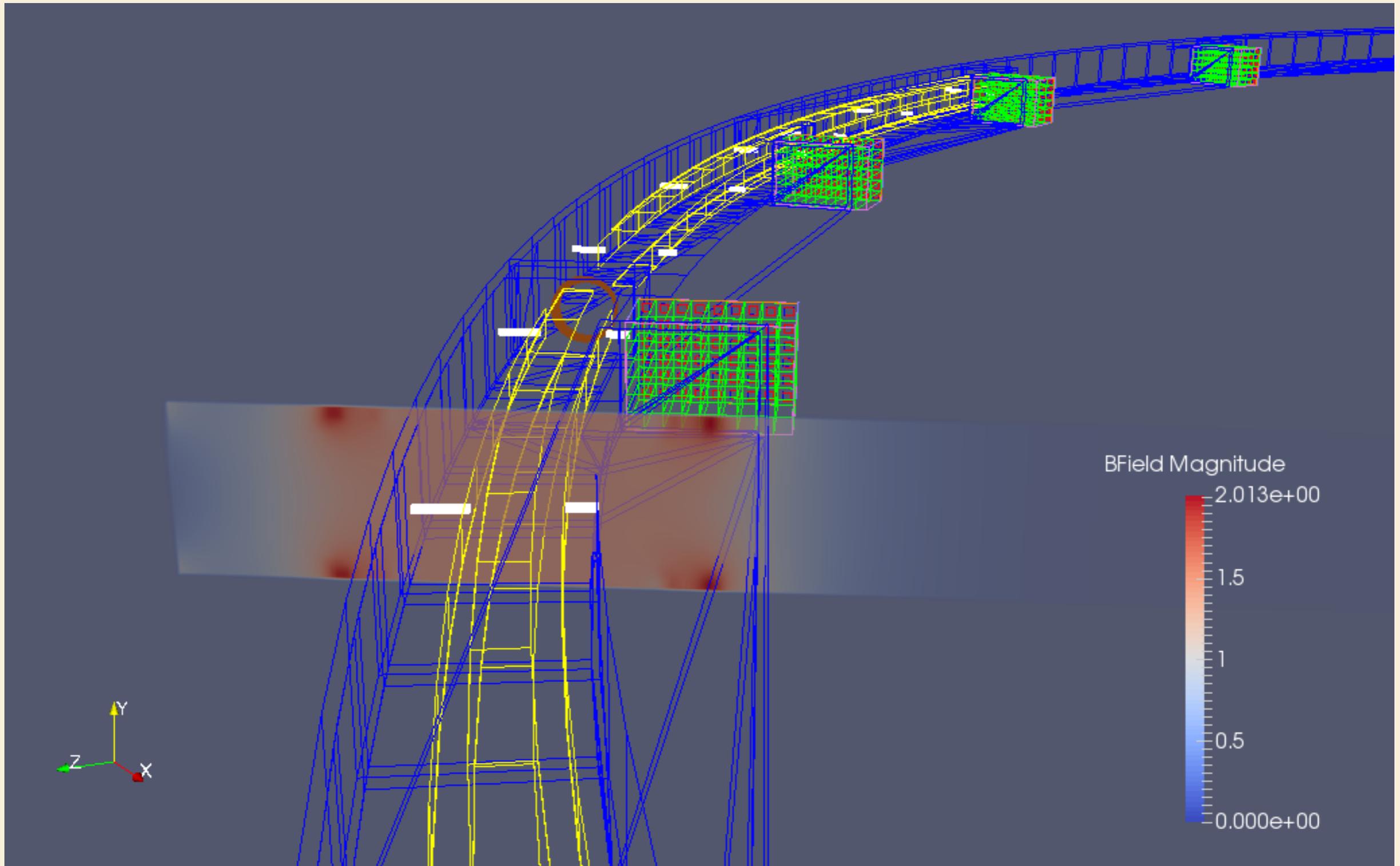
# Thin slice about $y=0$



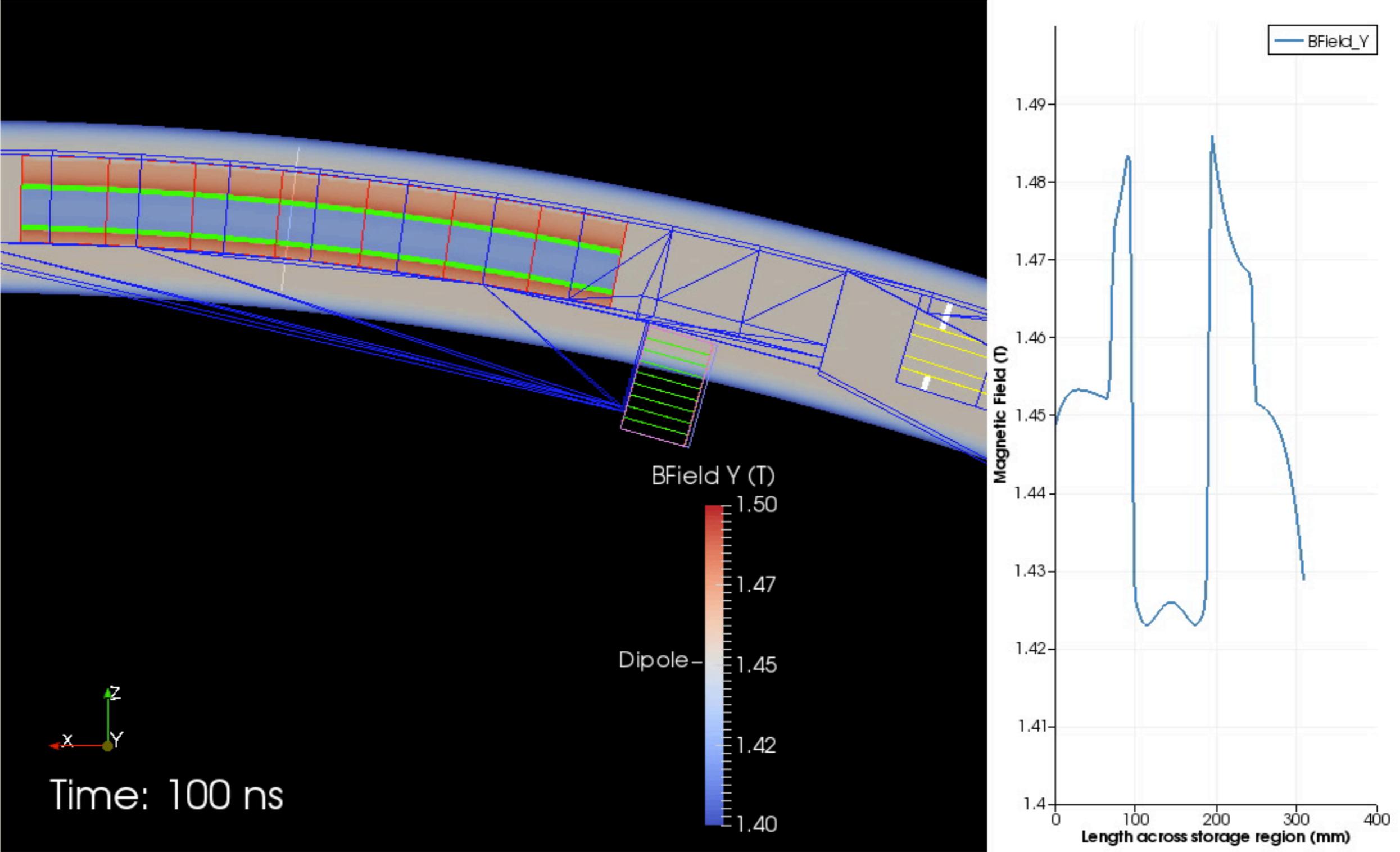
# Arcs & Quads



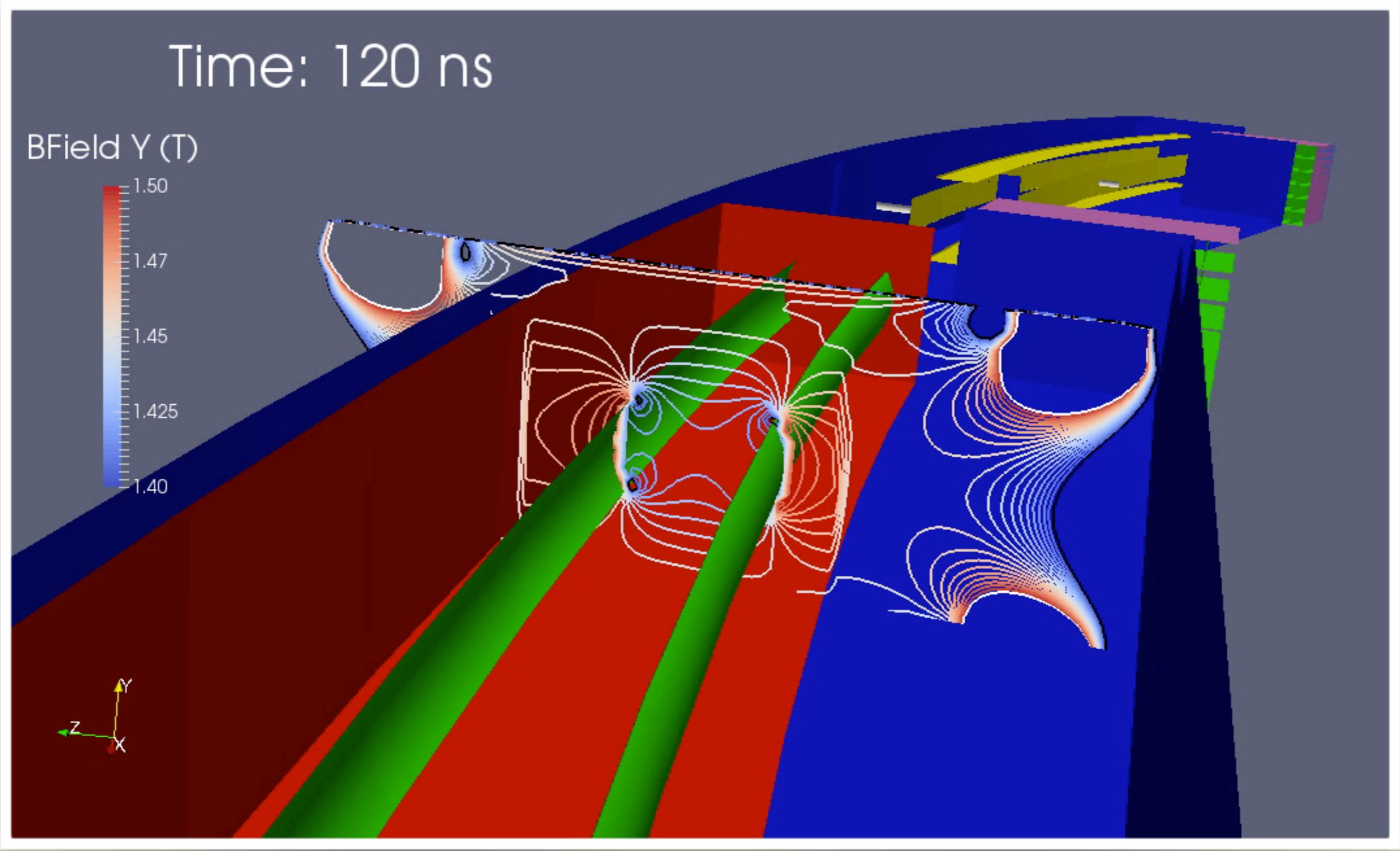
# Arcs & Quads



# Kickers

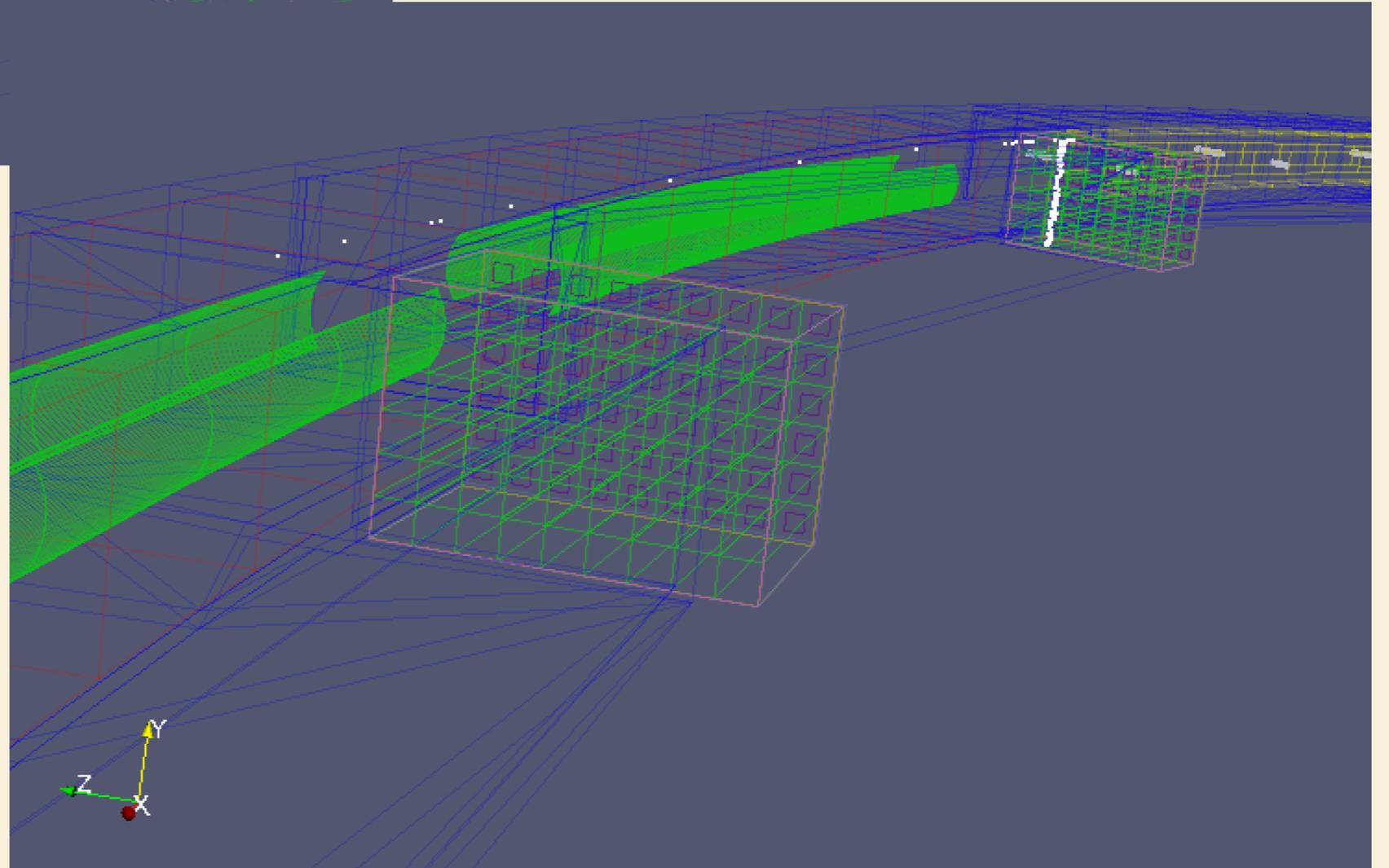
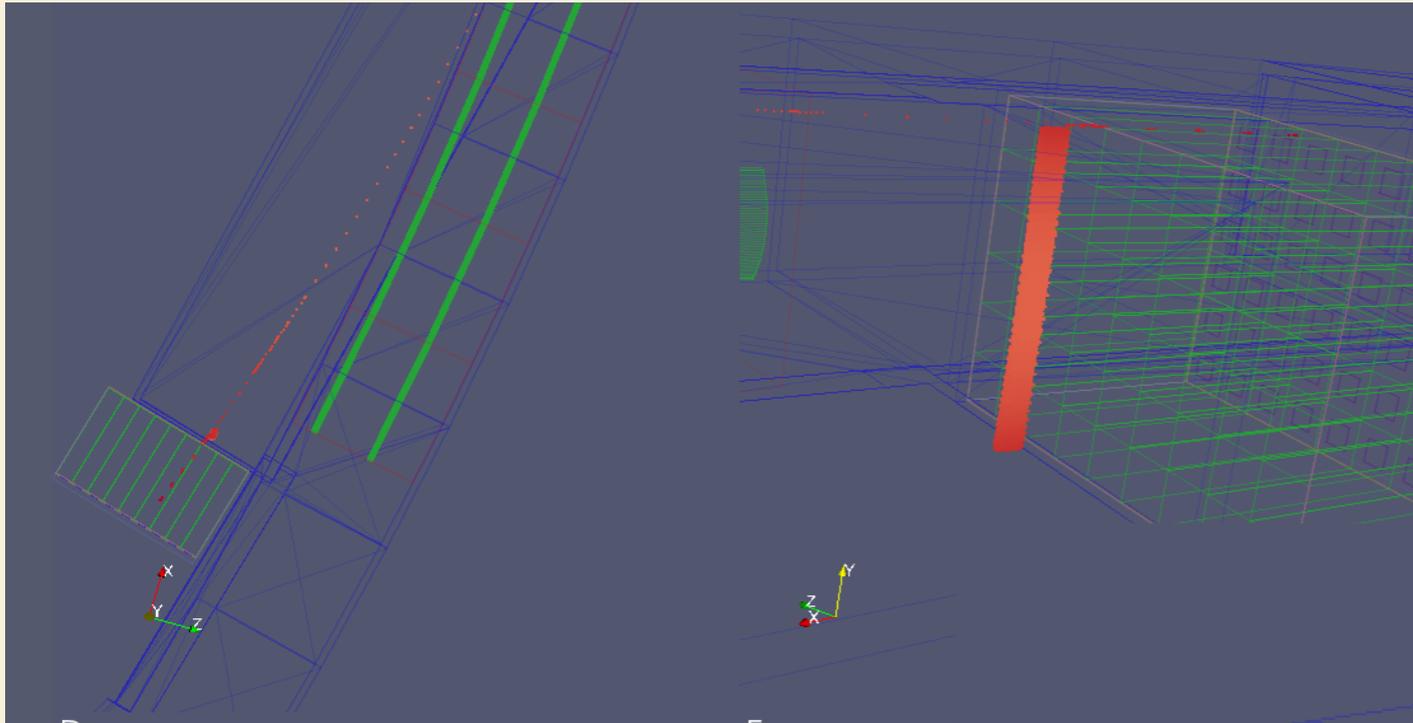


# Kickers



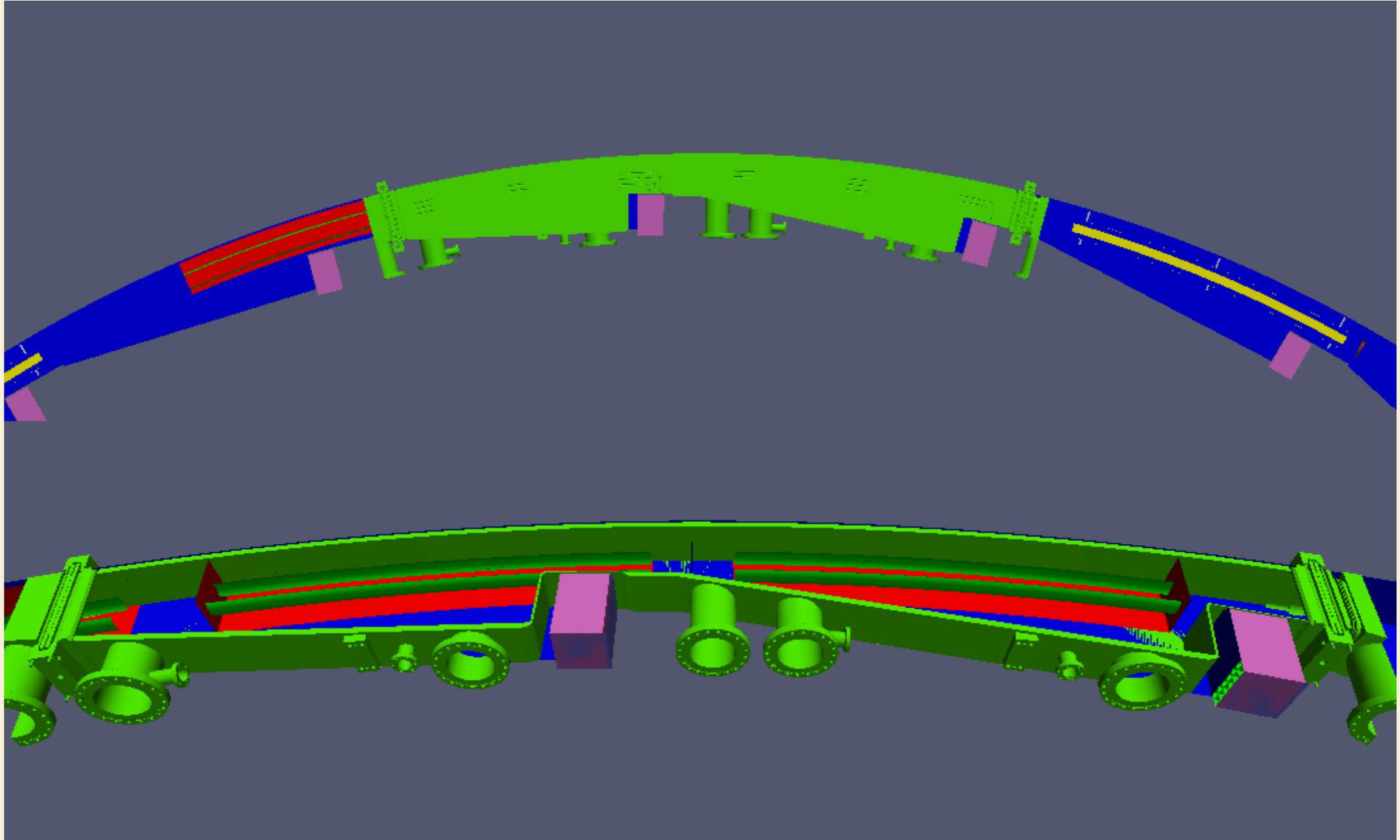
# Debugging - stuck event

**Write debugging information when Geant itself could not**

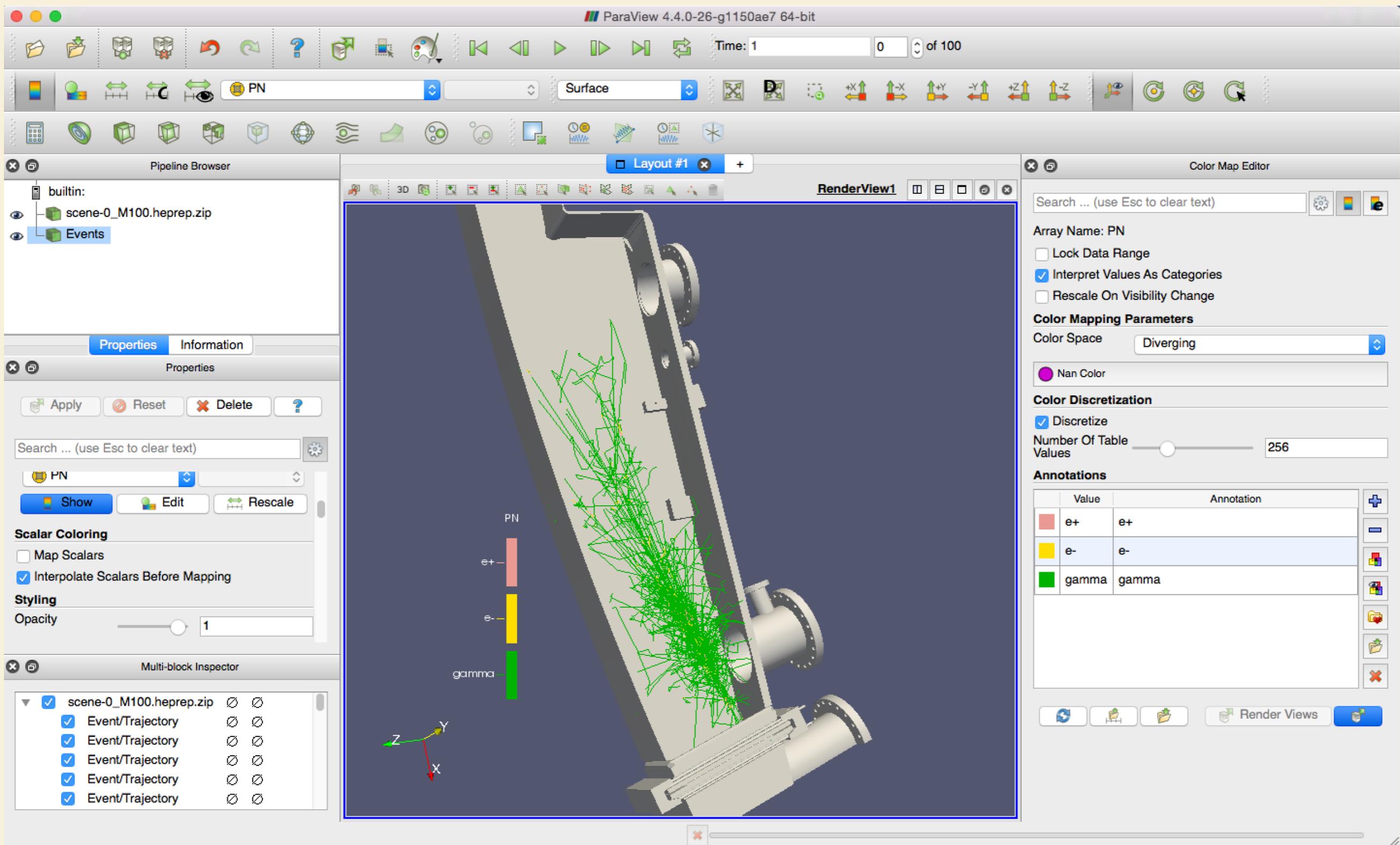


# Comparing geometry

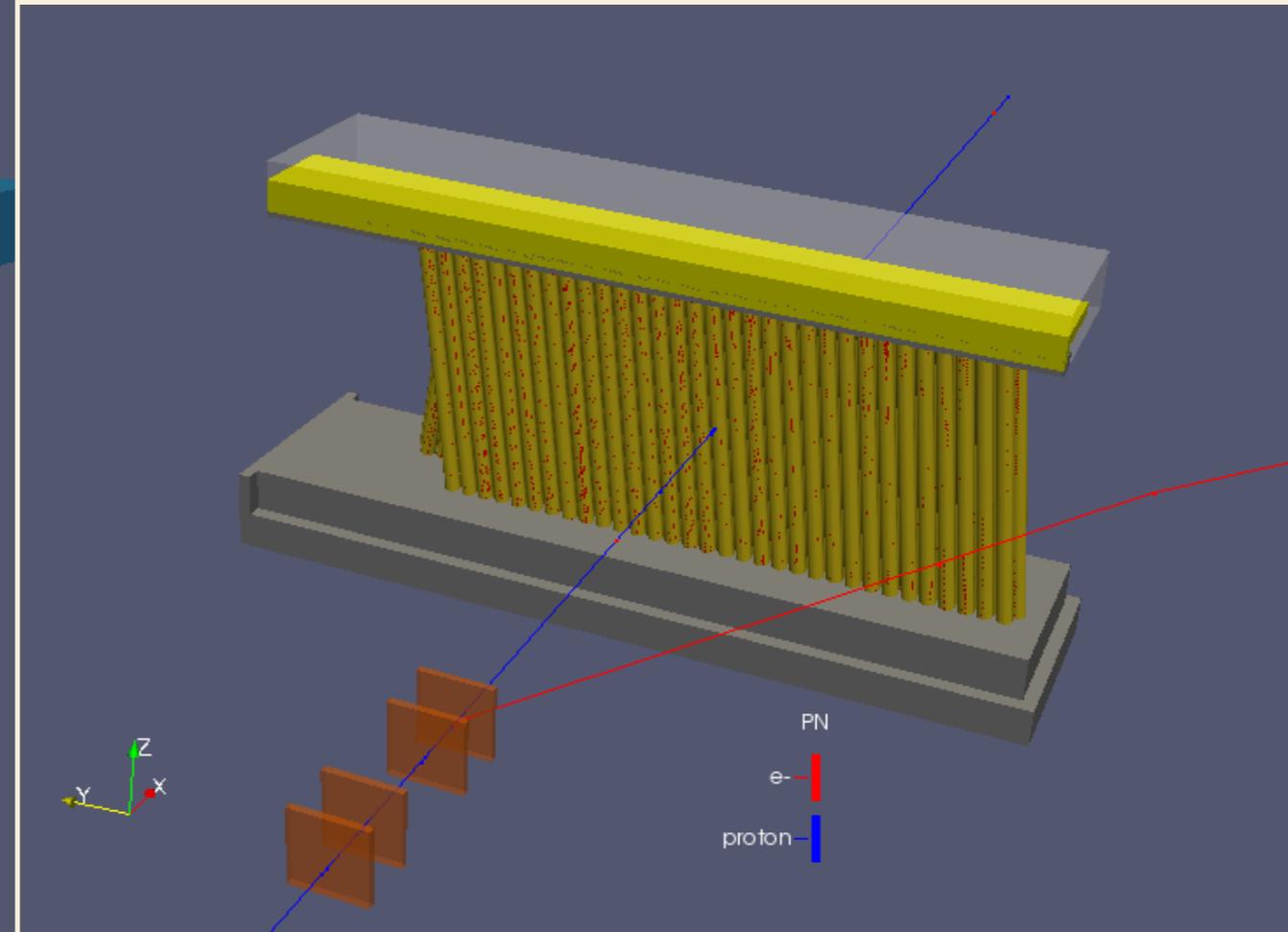
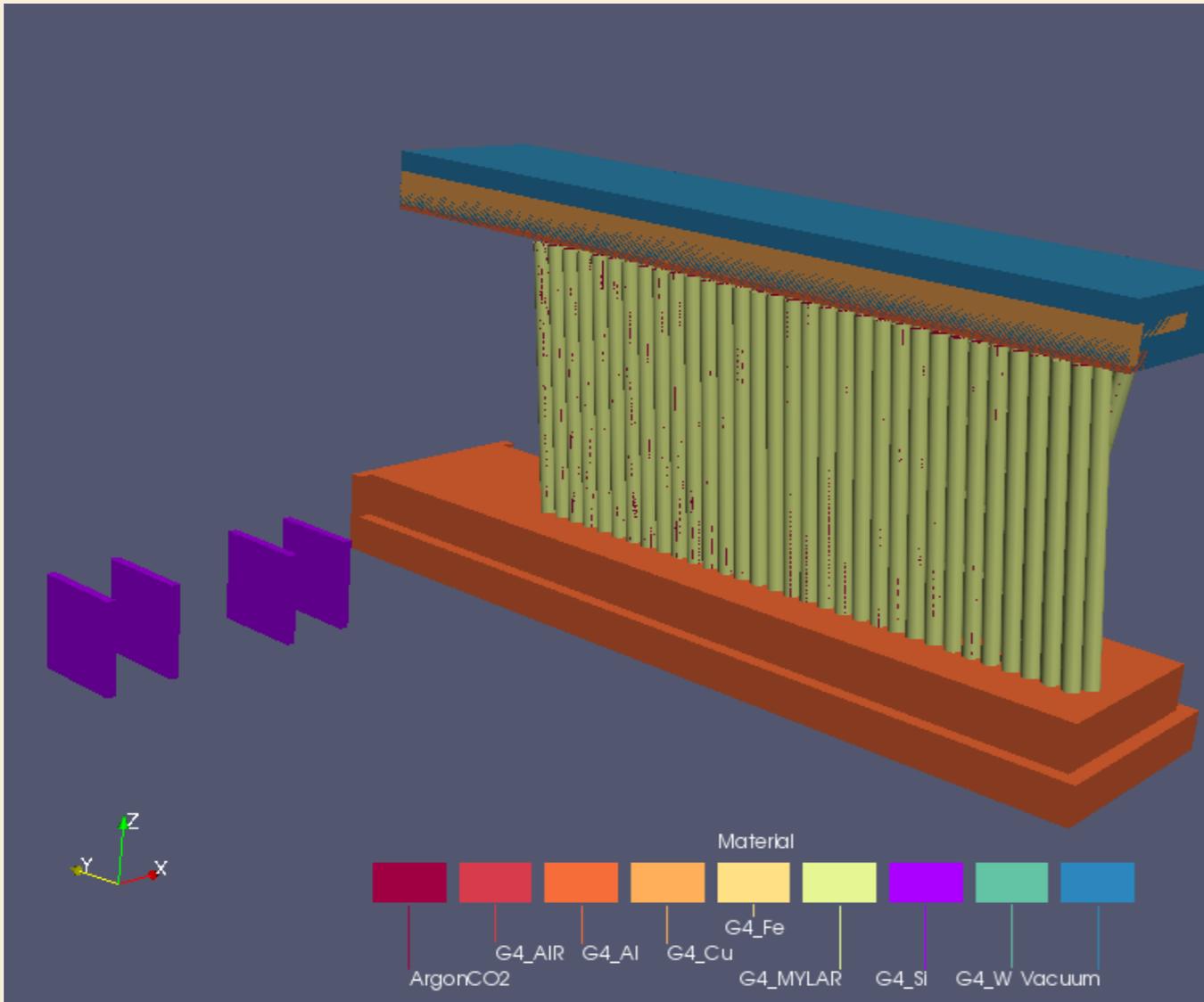
## Overlay Geant with CAD model (STL file)



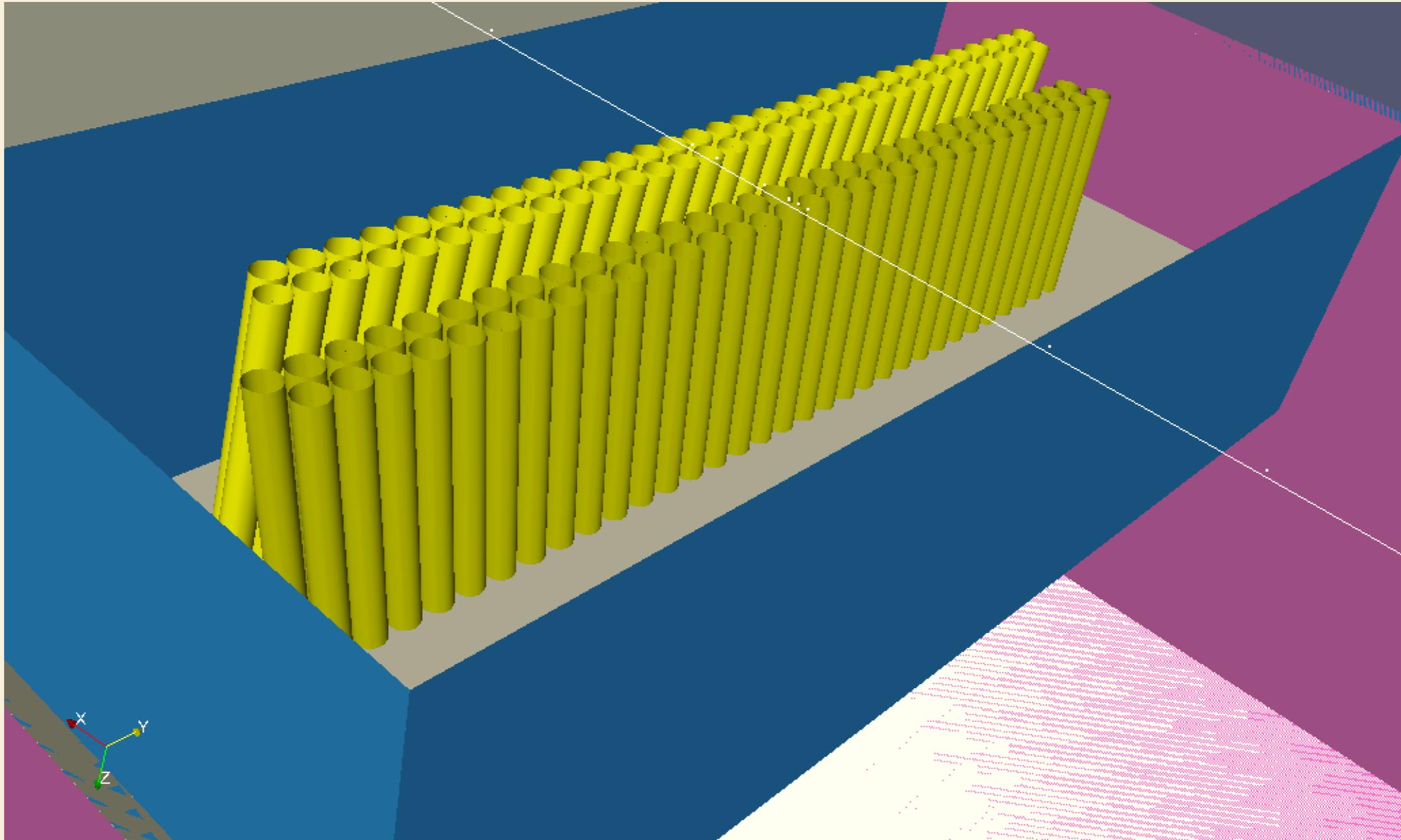
# Debugging



# Test beam



# Clipping



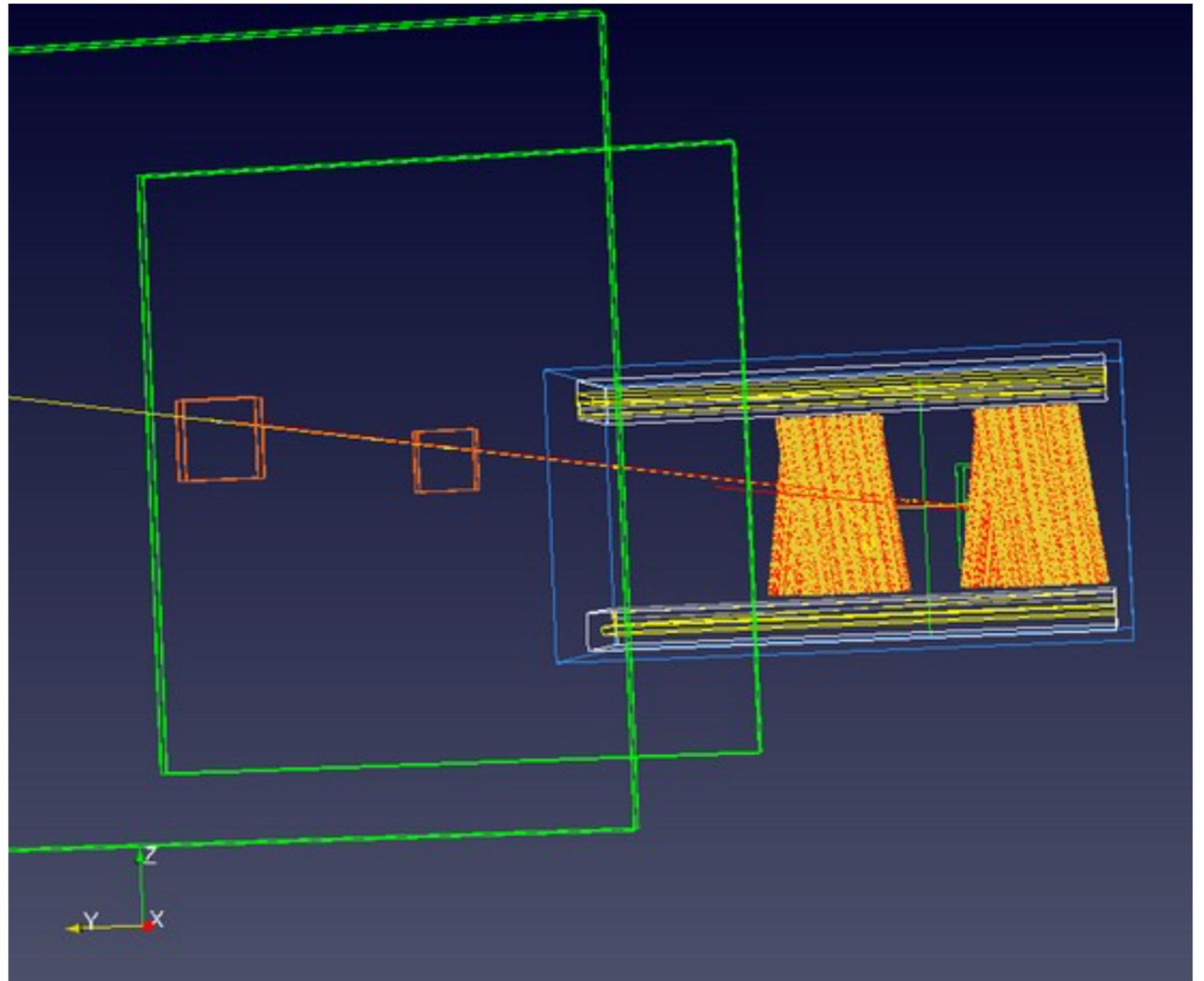
# Tracker Test Beam



Becky, Joe, Saskia,  
Tom

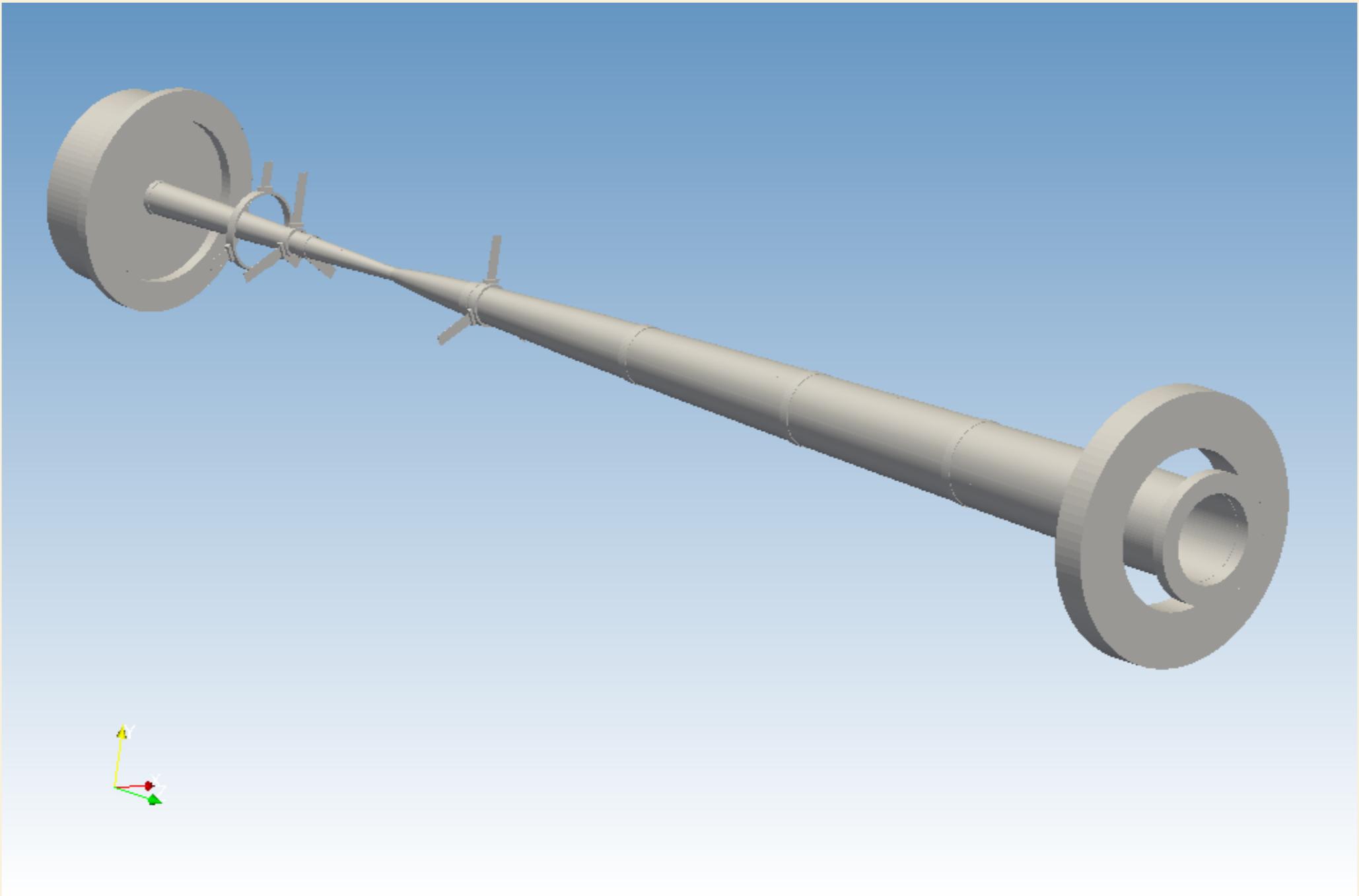
Sim truth track

Reconstructed track

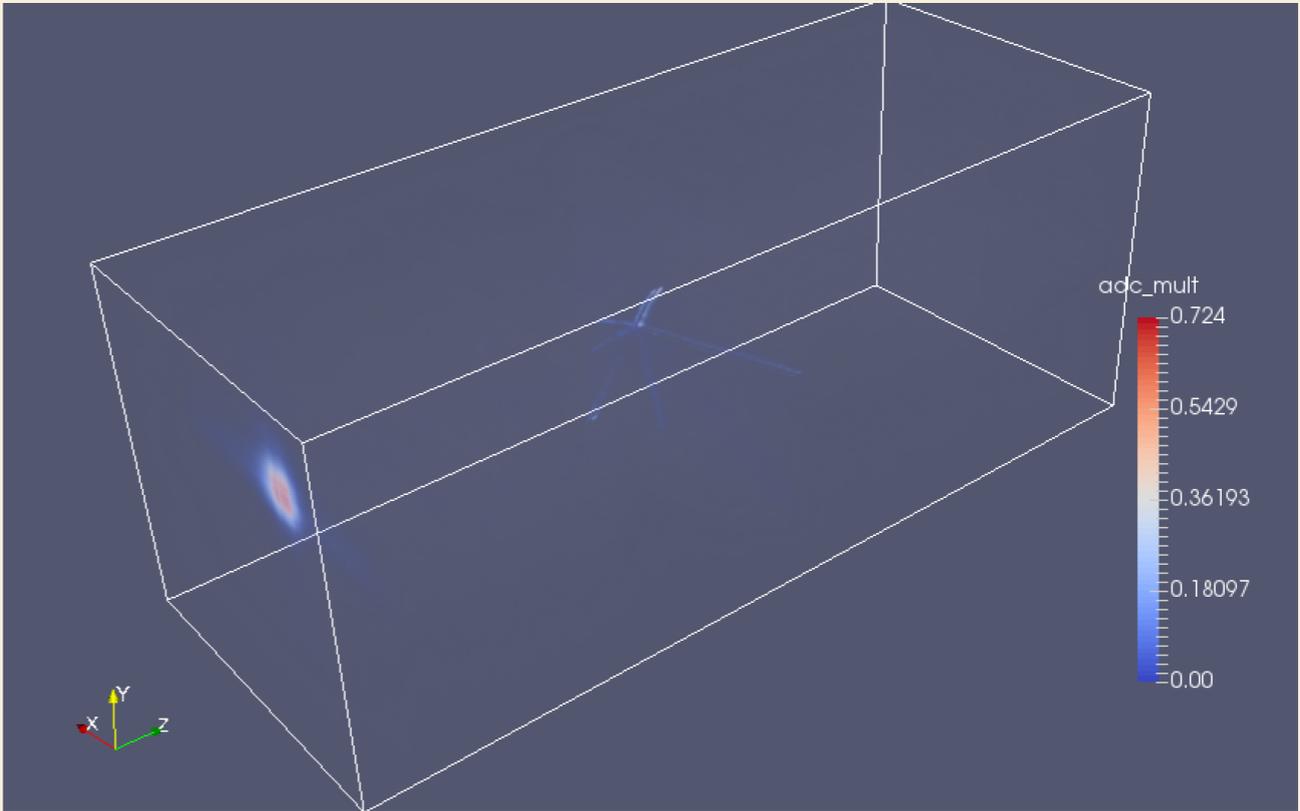
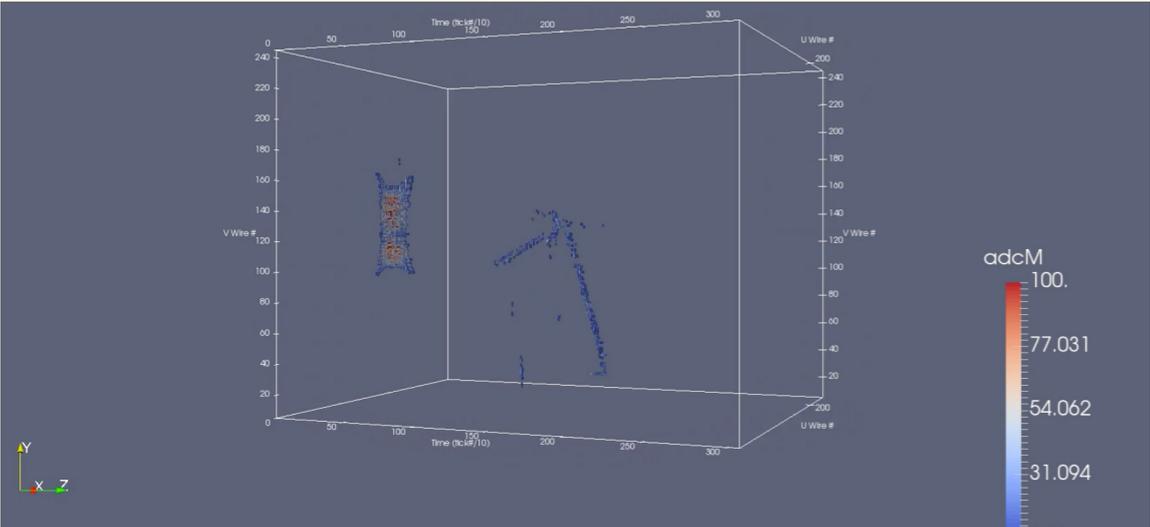
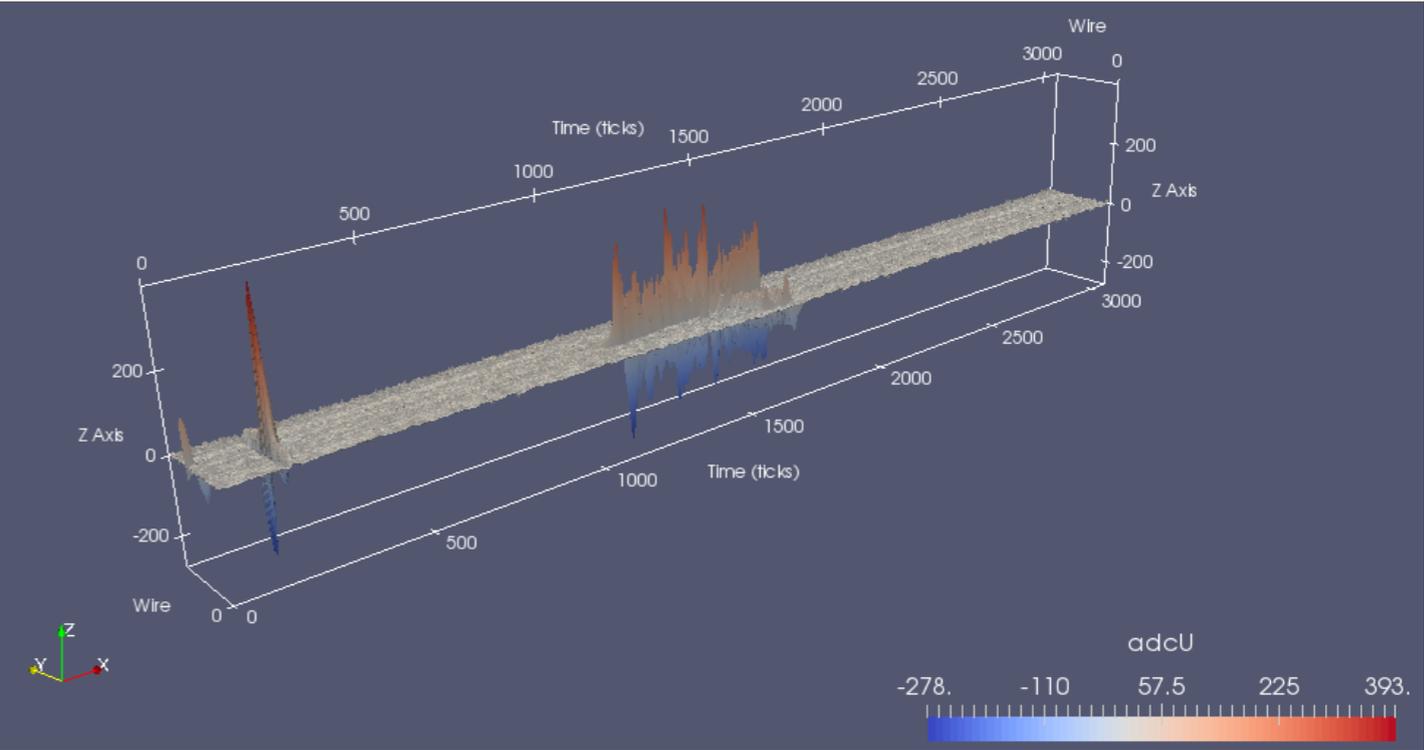
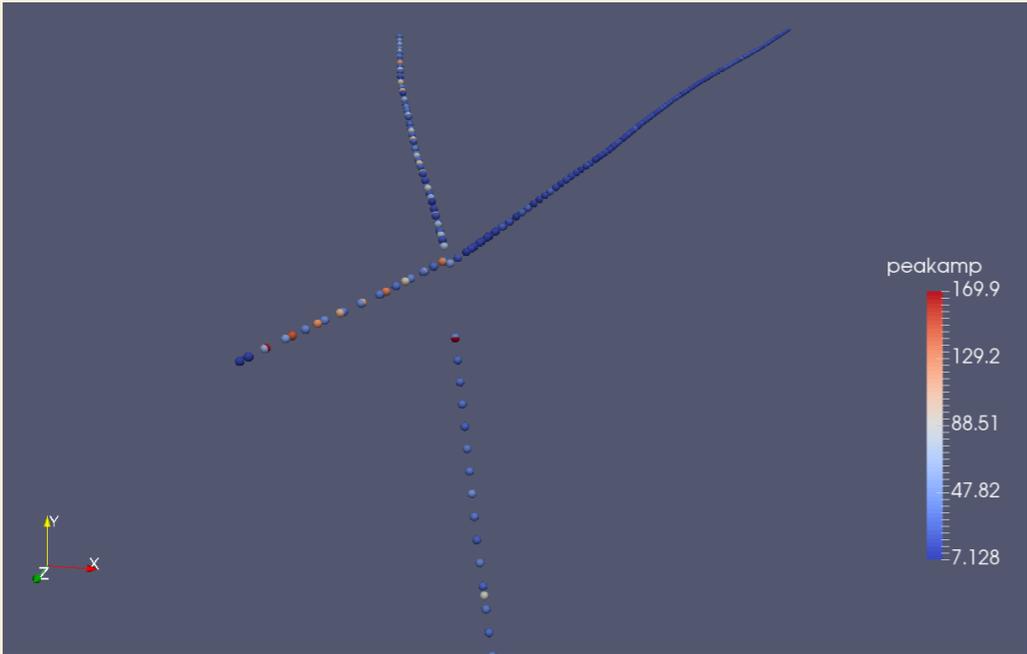


# Others

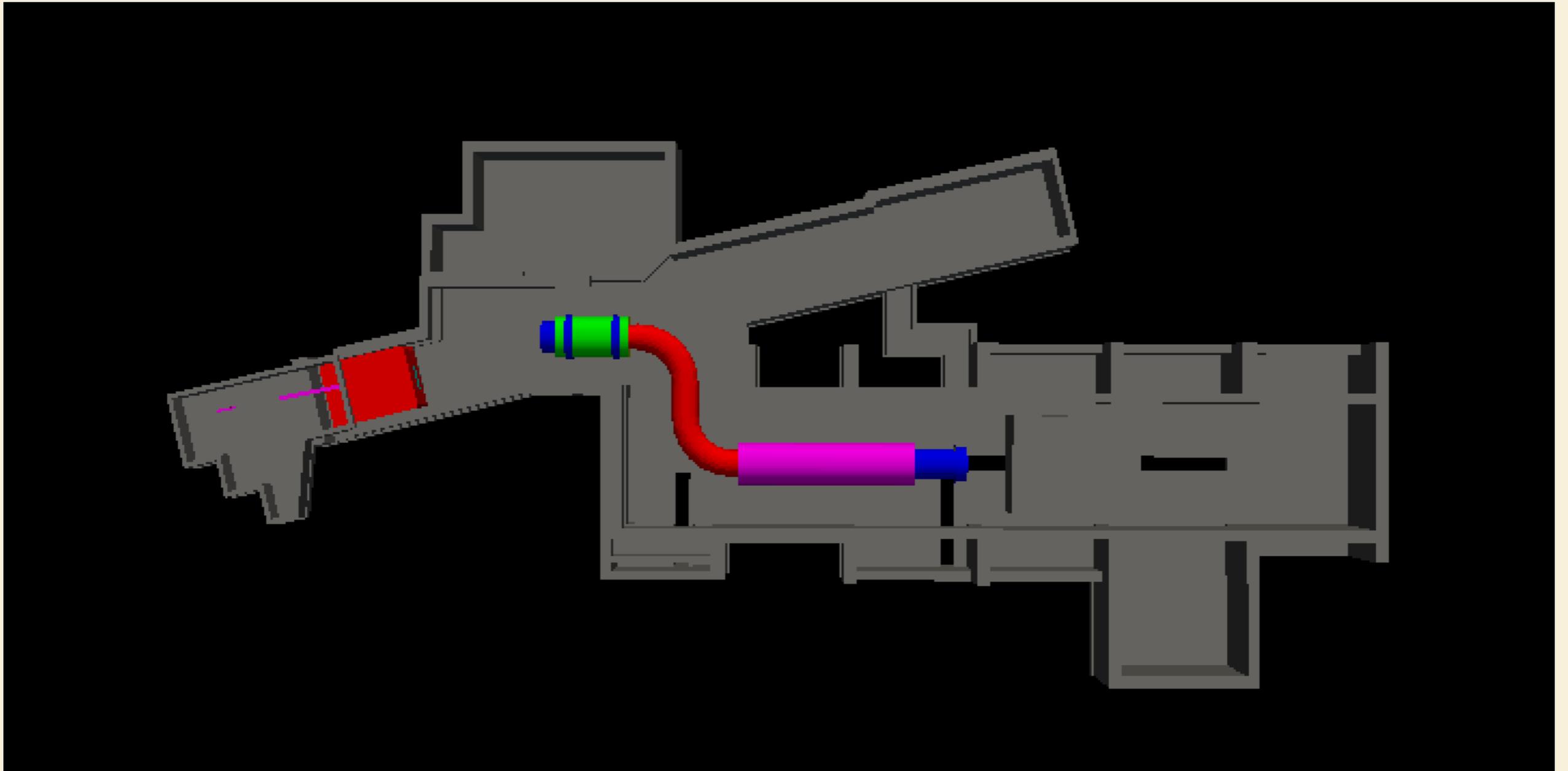
## Dune Horn (Laura Fields)



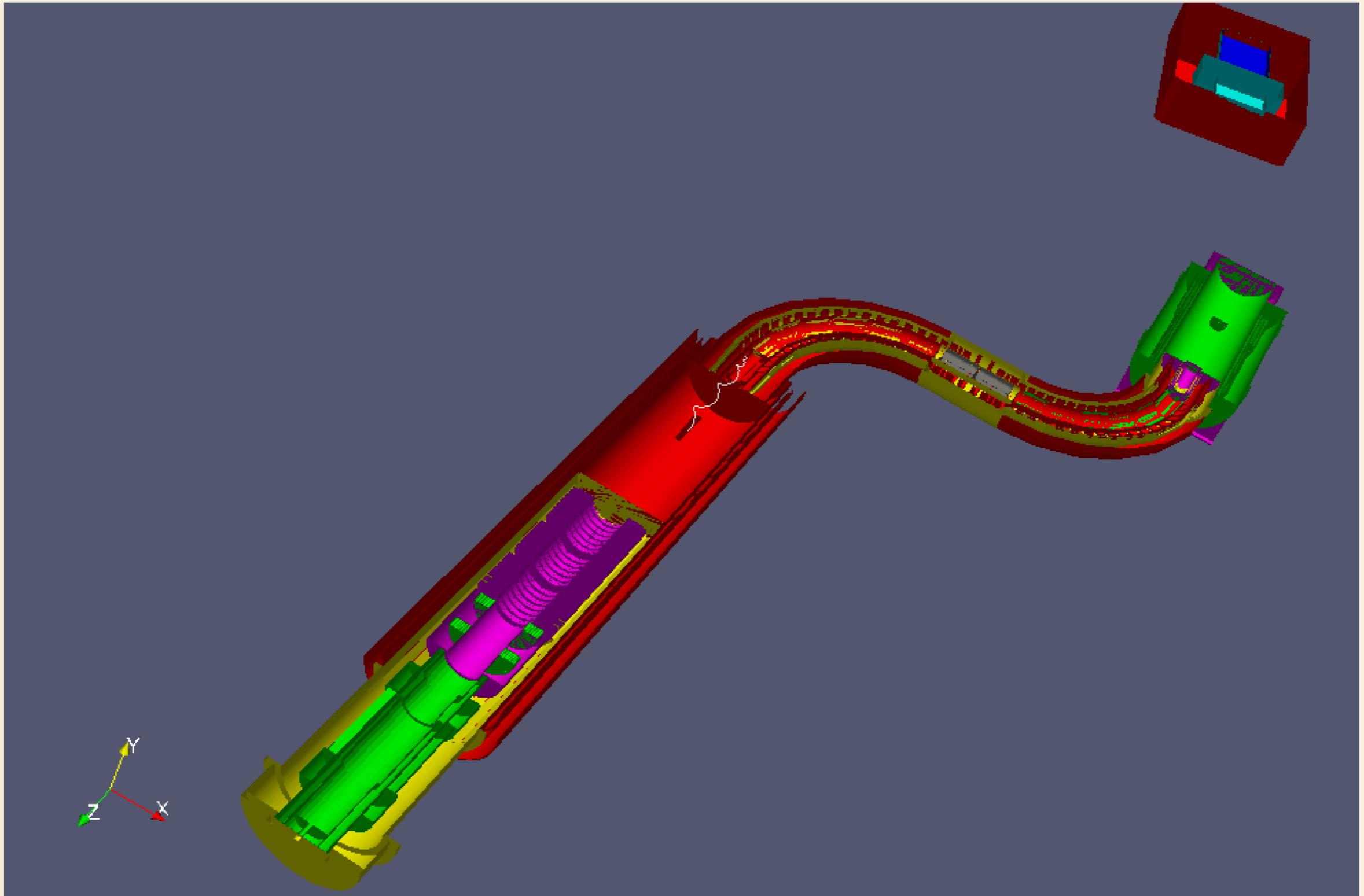
# LAr Reconstruction



# Mu2e



# Mu2e



# Getting data into ParaView

---

**Need to convert your data to VTK objects (so far I've done HepRepXML and text [csv] files with python scripts)**

- o **Python scripts outside of ParaView**

- o **“Python Programmable source” within ParaView**

**art/root object conversion is possible, but some thought to optimize would be necessary – perhaps do some conversion to VTK within art**

**Don't underestimate the utility of text files for debugging (e.g. stuck Geant use case)**

- o **Can connect ParaView to a running process supplying VTK objects (in-situ processing/Catalyst)**

- 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 had mixed results – don't underestimate the power of your laptop's GPU.**

# Next steps

---

**Fun stuff! It's addictive!**

**SCD Visualization Strategy may involve ParaView**

**Connect to art with Catalyst**

**I'll be giving a more extensive Computing Techniques Seminar with demo on *November 3rd***

**Document things so far**