mesh2gdml

Norman Graf (SLAC)

CHEP 2019, November 7, 2019

CAD to Geant4?

- Often requested from user community, despite recognized limitations of:
  - difficulty accessing proprietary formats
  - mismatch between level of detail, approaches to parent-child relationships, material definitions
  - performance issues
- Most existing solutions target recognized interchange formats such as STEP and IGES, but even these can have problems
  - complicated file format, usually not open source
  - possible loss of hierarchy or material association
  - little or no mapping to primitives

Geant4 "Primitives" vs Tesselations

- Geant4 Primitives
  - + Geant4 provides a very rich library of basic geometrical shapes plus the ability to define complex compound geometries via boolean operations.
  - + Analytic, or optimized geometric calculations of "inside" or distance to boundary, with reasonable CPU performance.
  - + Parameterizations also available, reducing memory footprint.
  - - Cumbersome to define irregular shapes
  - - Labor-intensive manual intervention to implement CAD designs
- Tesselations
  - + Complex (CAD) geometries with minimal human intervention.
  - + Support for irregular shapes, e.g. biological phantoms
  - - Complex geometrical calculations increase CPU
  - - Large number of vertices and facets increase memory

STL: Lowest Common Denominator

+ Simple format : list of three dimensional corner point coordinates (vertex) and flat triangles (facet).
+ Ubiquitous as an export format for CAD and other 3D software.
- No topological information about the mesh.
- No guarantee of correctness
  - single facets, holes, overlaps, etc.
- No material or other attributes
- Format is verbose, making file sizes large and subject to error from roundoff precision
  - e.g. shared vertices are listed explicitly n times.
- Overlap problems can arise when combining STL files from different software packages, or exported with different tolerances.

STL to GDML

+ Identifies topologically distinct elements in the file
+ Converts STL facets directly to G4TriangularFacet and creates G4TessellatedSolids, write as GDML.
+ Create world volume for standalone use, or leave as individual volume to aggregate or incorporate into a common world volume later.
+ Basic checking of geometry integrity.
+ Can assign material at creation time, e.g.

  > java StlToGdml model.stl model.gdml Aluminum

- Different materials must be manually assigned

Recent Developments

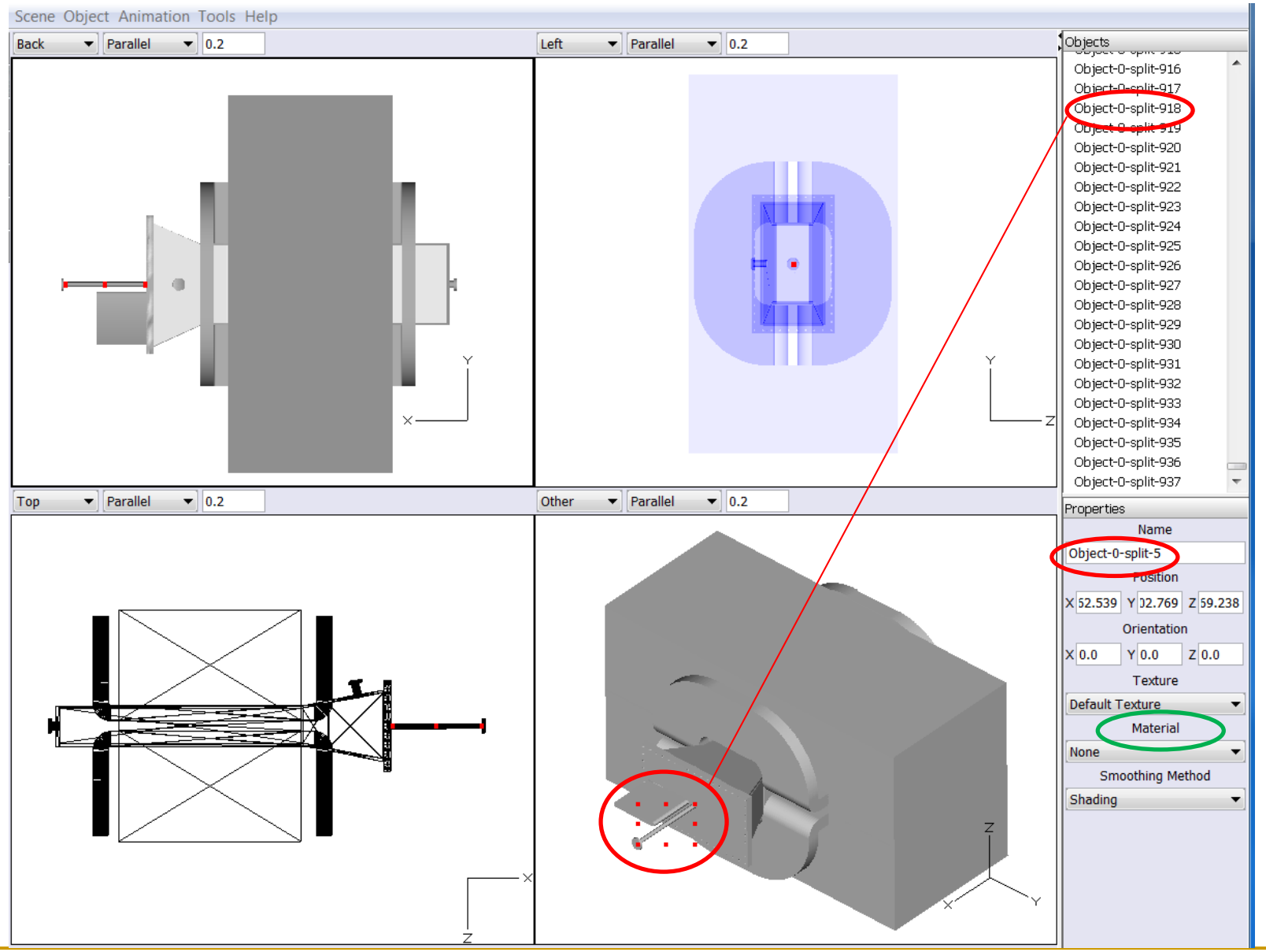
- GUI to aid translation process
  - Currently targeting Art of Illusion
    - Java-based, so easily deployable and “runs anywhere”
  - Easy-to-use plug-in architecture

- Allows user to select volumes and:
  - Create hierarchies and place volumes into it.
  - Delete unwanted volumes
  - Assign materials
    - by name (prone to mistake, e.g. Aluminum vs Aluminium)
    + from drop-down list (predefined, e.g. NIST or G4_*)
    - from material editor

Geometry Editor GUI (Art of Illusion)

Default Scene Views

Customizable Display Options



Supports many formats
STL, OFF, PLY, OBJ, …

mesh2gdml plugin splits mesh on import

Picking on list highlights selected object

Picking on object highlights name in list

Edit selected object's properties

NIST materials available from drop-down list

mesh2gdml plugin writes gdml file on export

Future Plans

Short Term

- Code review, refactoring, cleanup
- Documentation
- Release

Intermediate Term

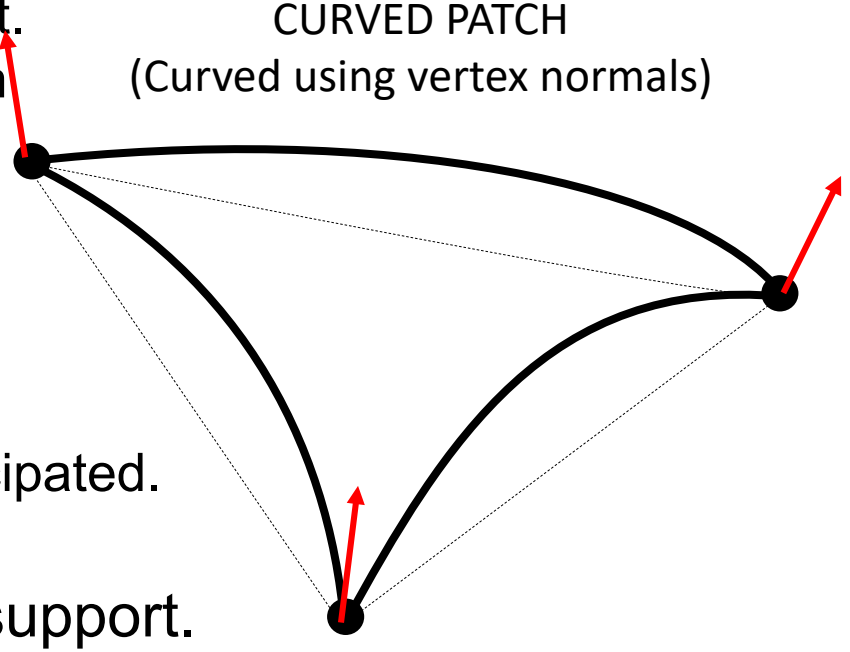
- Introduce support for tetrahedral meshes to improve performance
  - See Fast Tessellated Solid Navigation in GEANT4 by Poole et al.

Longer Term

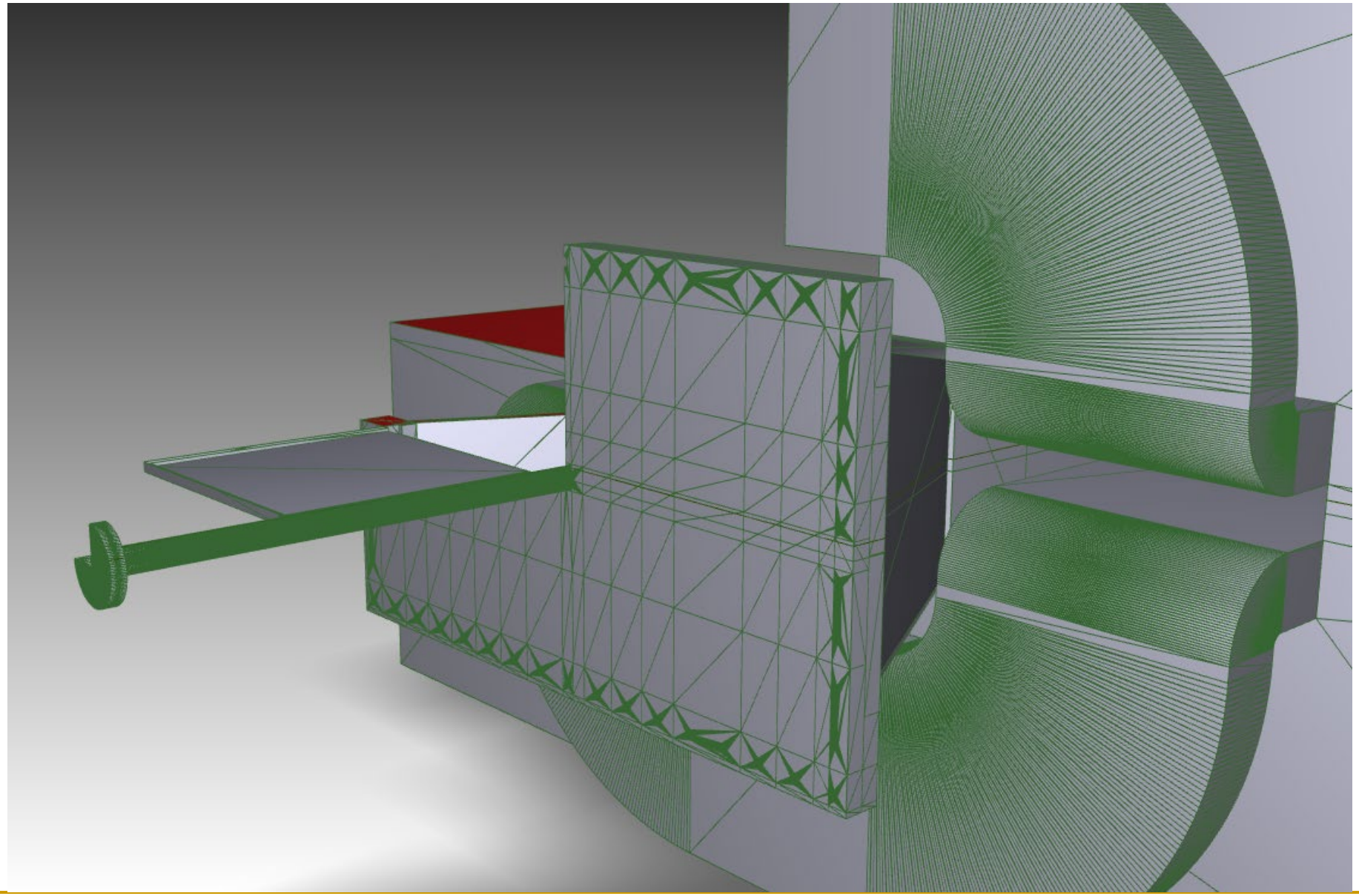
- Code to aid conversion to G4 primitives
  - User selects volume, then selects candidate simple primitive type (e.g. Box, Tube), code “fits” and extracts parameters, writes to GDML
- Support for newer 3D file formats?
  - AMF  3MF 

File Format Future: STL → AMF ?

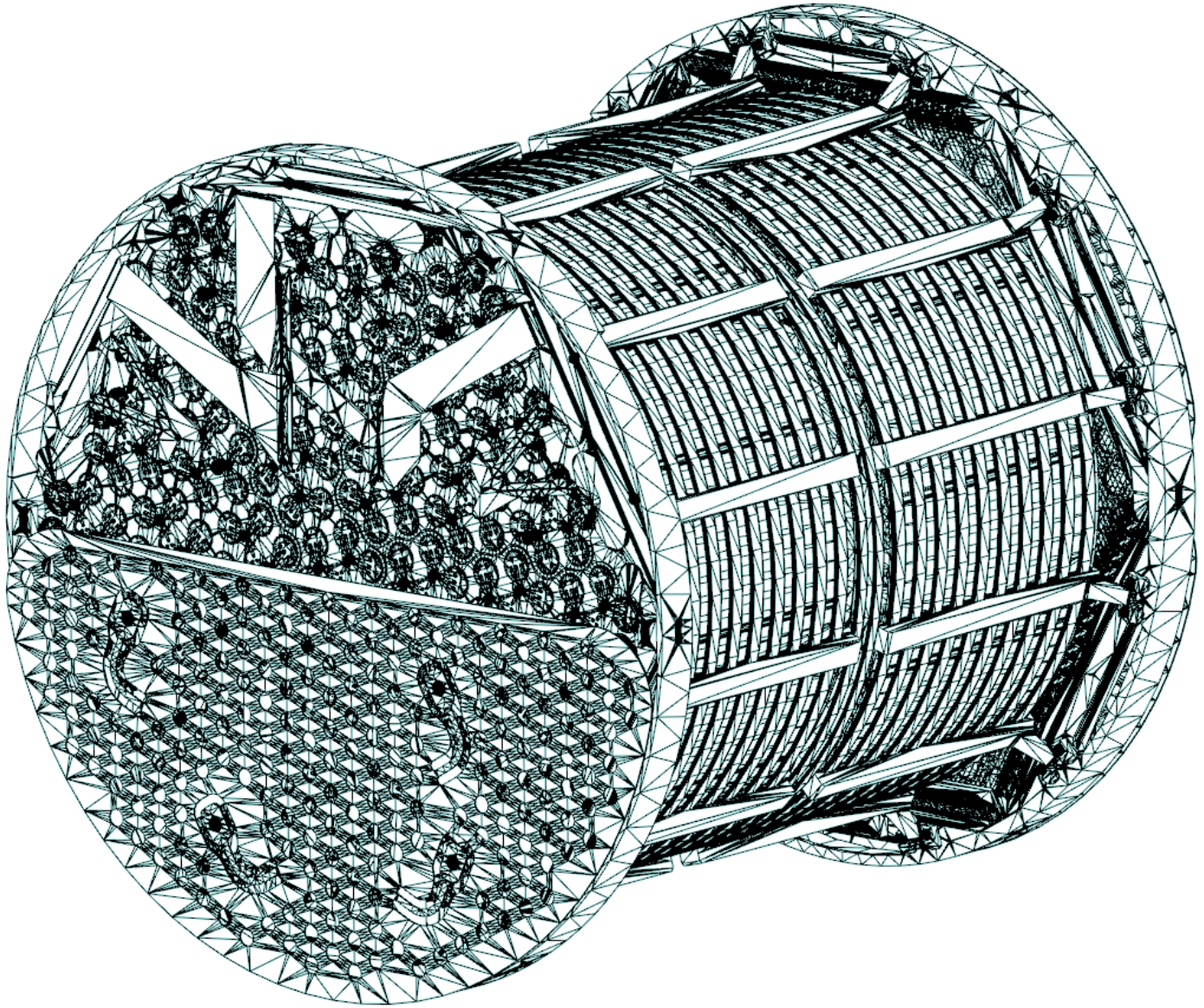
- ASTM F2915 defines a standard specification for Additive Manufacturing File Format (AMF)
- Solves most of the problems associated with STL.
- Takes STL format for vertices and facets and adds support for:
  - **<object>** Defines a volume associated with a material ID for printing.
  - **<material>** Optional element defines one or more materials for printing.
  - **<texture>** Optional element defines images or textures for color or texture mapping.
  - **<constellation>** Optional element provides hierarchy support.
  - **<metadata>** Optional element contains additional information
- Extensible:
  - **<mesh>** Facets can be curved with addition of **<normal>** tags
  - 3D bitmap <voxel> and functional representation <frep> anticipated.
- Efficient: higher precision with smaller size
- ASTM and ISO approved, expect CAD vendors to add support.



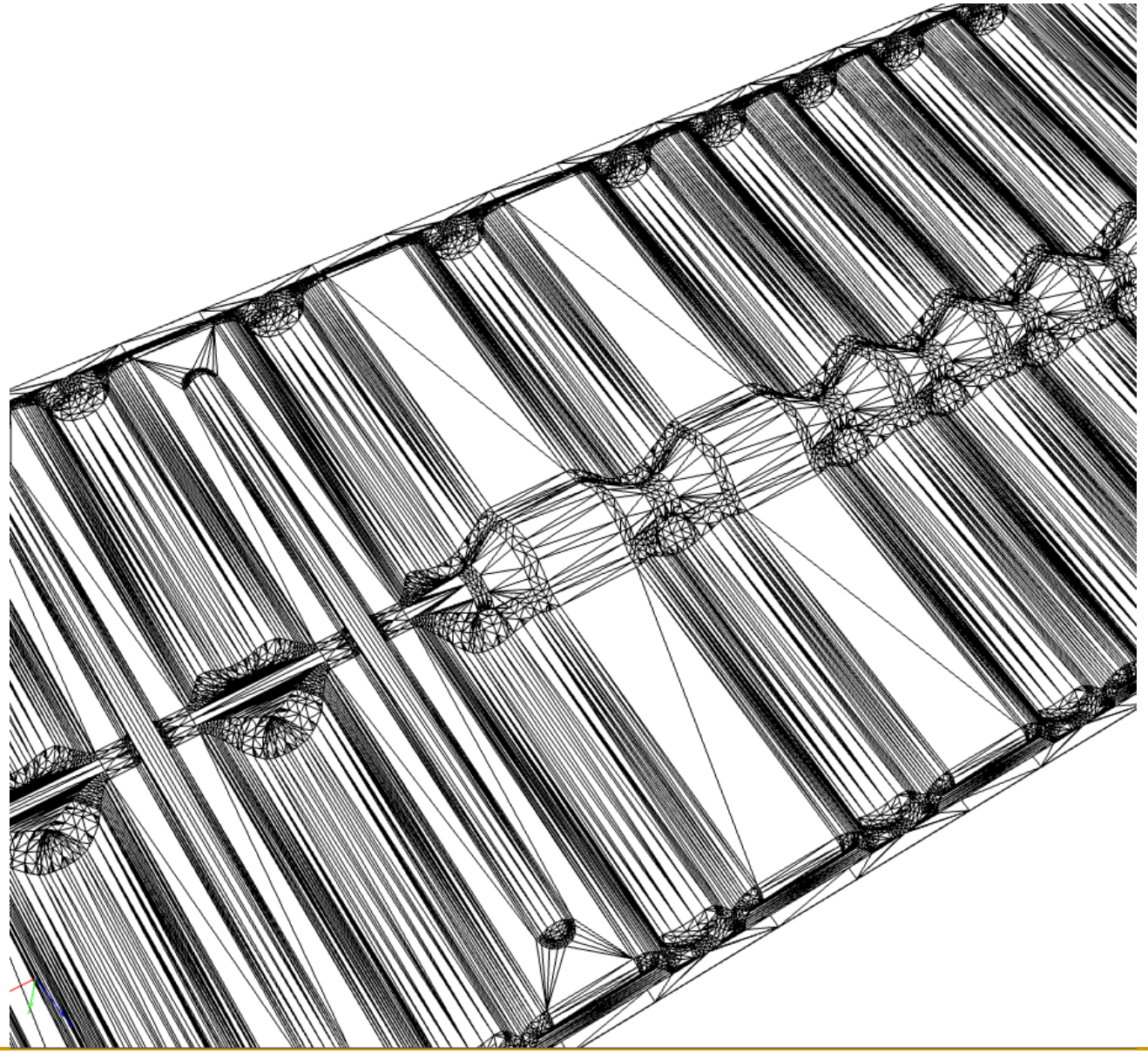
Heavy Photon Search (HPS)



EXO



LHCb RF Foil

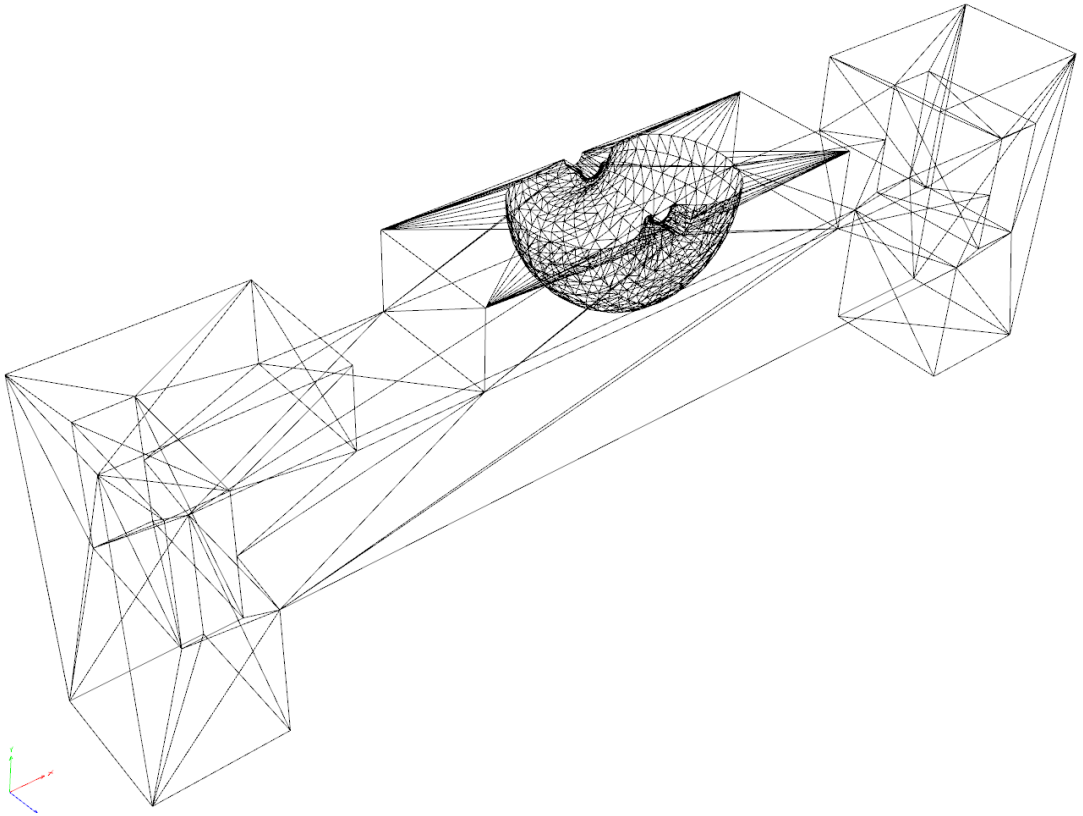


I have my geometry. Now what?

- Hard-code your G4-based application to read in and access your geometry directly
  - geant4/examples/extended/persistency/gdml
- Use slic, a general-purpose, Geant4-based simulation application
  - “Detector” completely defined at run-time via xml input file / G4 macros
  - lcdd fully encapsulates detector definition, including non-geometry aspects like sensitive detectors, magnetic fields (DOI: 10.1016/j.nima.2015.03.081)

slic Use Case I Accelerator Shielding

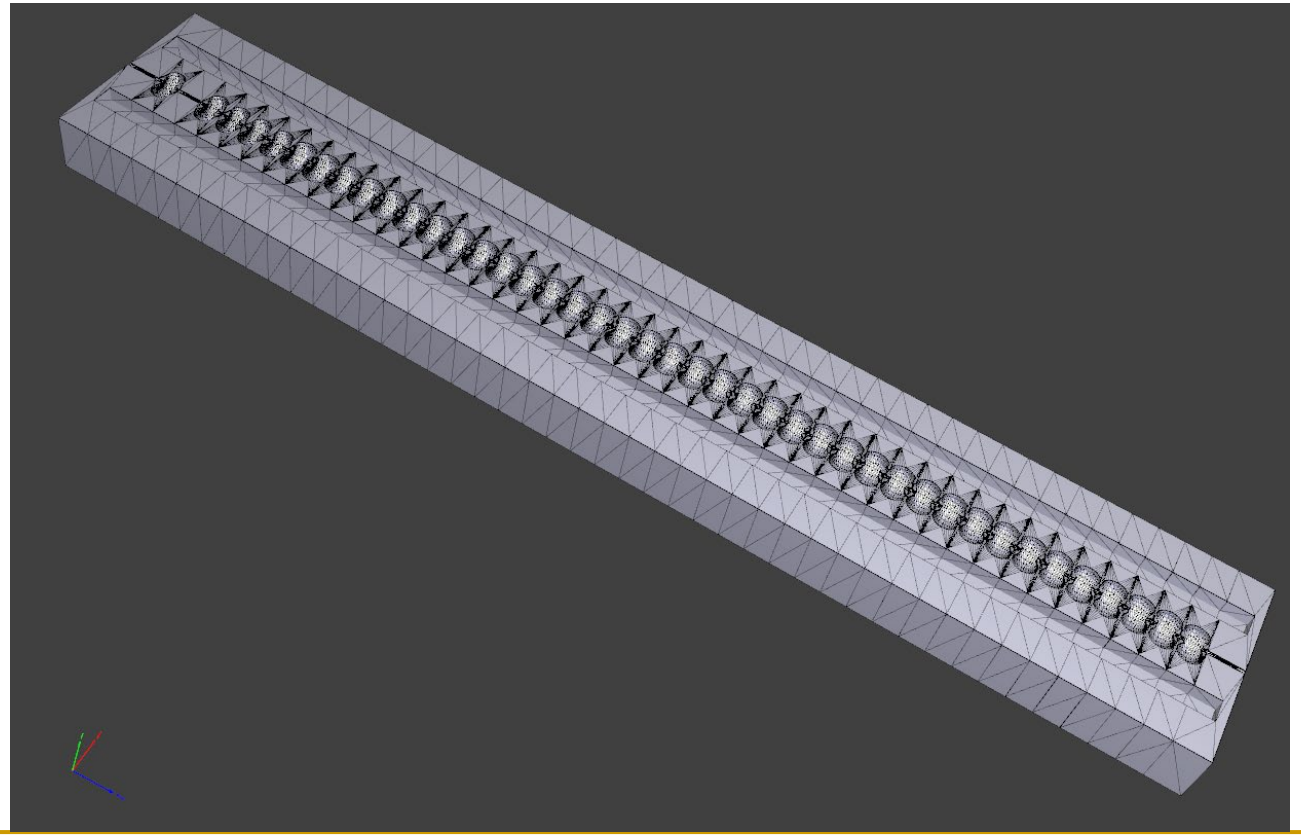
- Import geometry as plain gdml, use GPS as particle source and G4 scoring planes as detectors/analysis.
- Take CAD model, export as STL, convert to gdml



Basic unit cell of accelerating structure

slic Use Case I

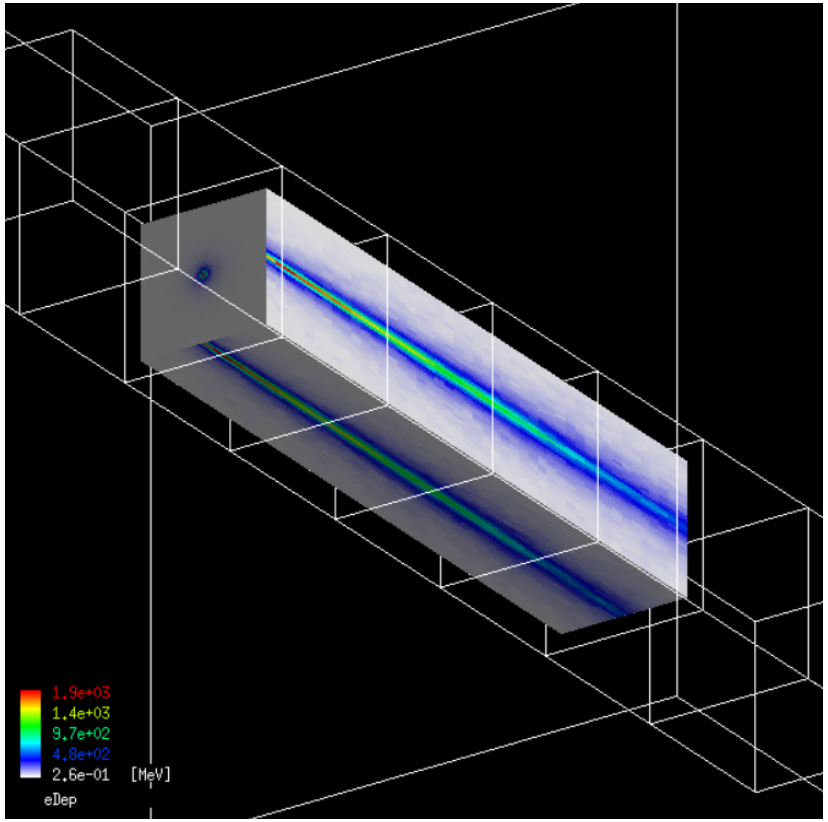
- Import geometry as plain gdml, use GPS as particle source and G4 scoring planes as detectors/analysis.
- slic –g accelerator.gdml –m scoring.mac

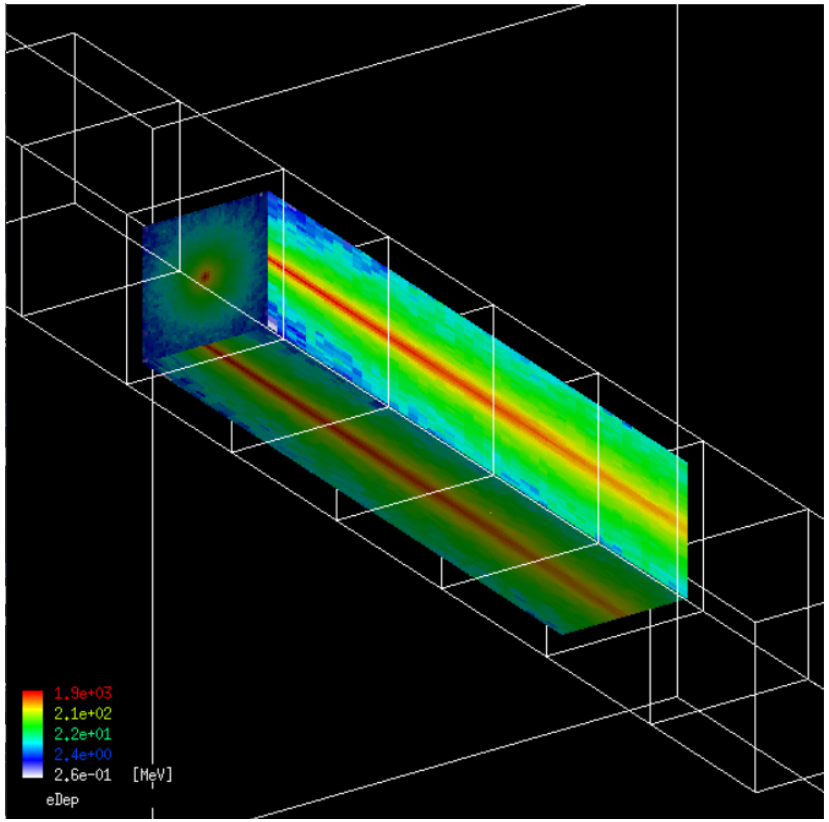


Full structure composed of unit cells plus end modules

slic Use Case I

- Scoring meshes can be drawn within Geant4 visualization or exported as CSV files for further analysis.

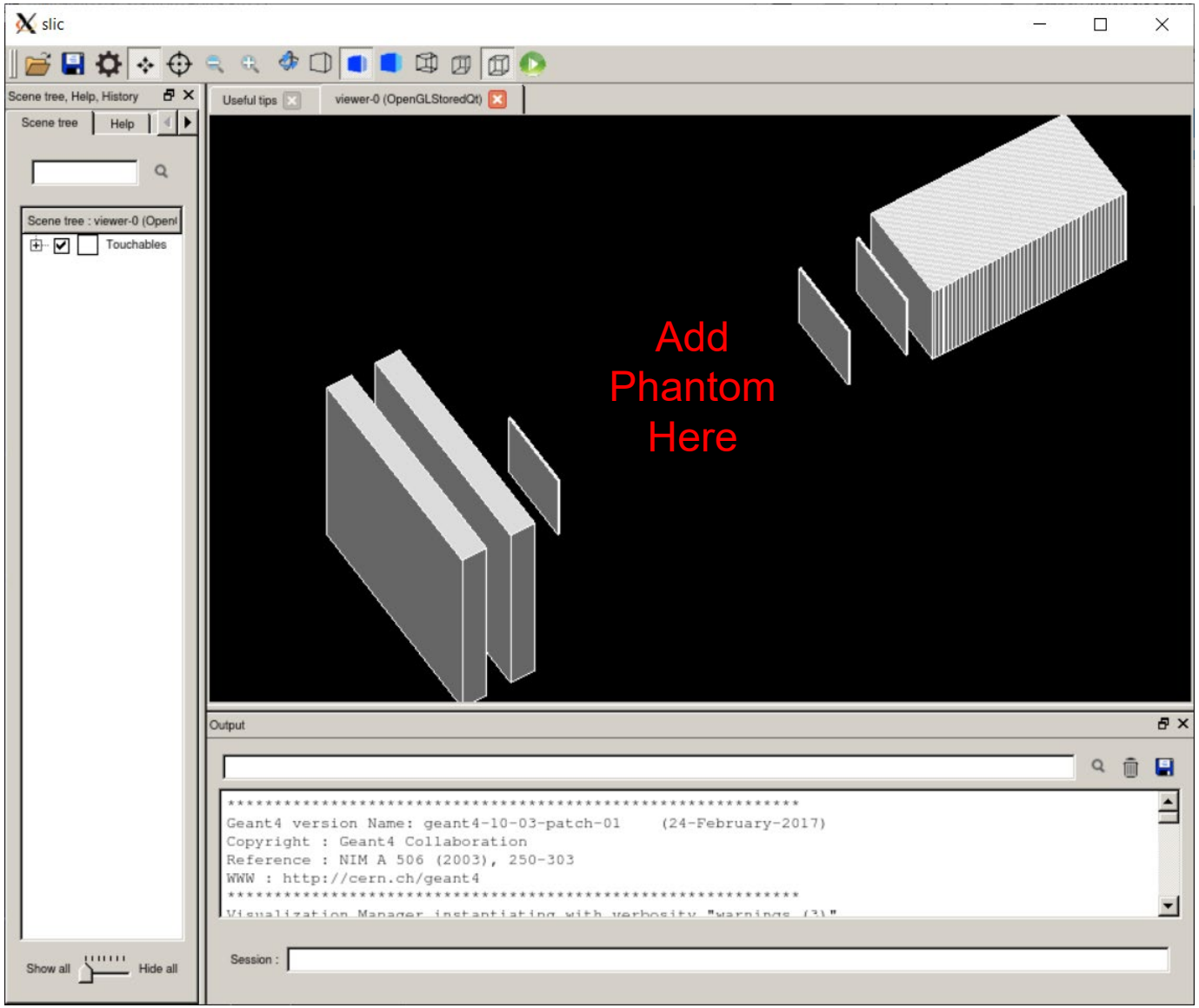




slic Use Case II

- Combine “physics detector” with gdml geometry of “target”
  - Fully define detector (e.g. tracker and/or calorimeter) as a sensitive detector using lcdd, use tessellated gdml as “target”
- Proton Computed Tomography
  - Define silicon strip/pixel or scintillating fiber tracker + calorimeter
  - Introduce phantom or medical model as target
    - DICOM $\rightarrow$ STL $\rightarrow$ gdml
- Muon Tomography
  - Define scintillating fiber tracker in lcdd or use scoring mesh for emulsion
  - Pyramid/volcano from archaeology/geology model

PCT



Detector model is defined as a static setup of moderator, tracking stations and calorimeter.

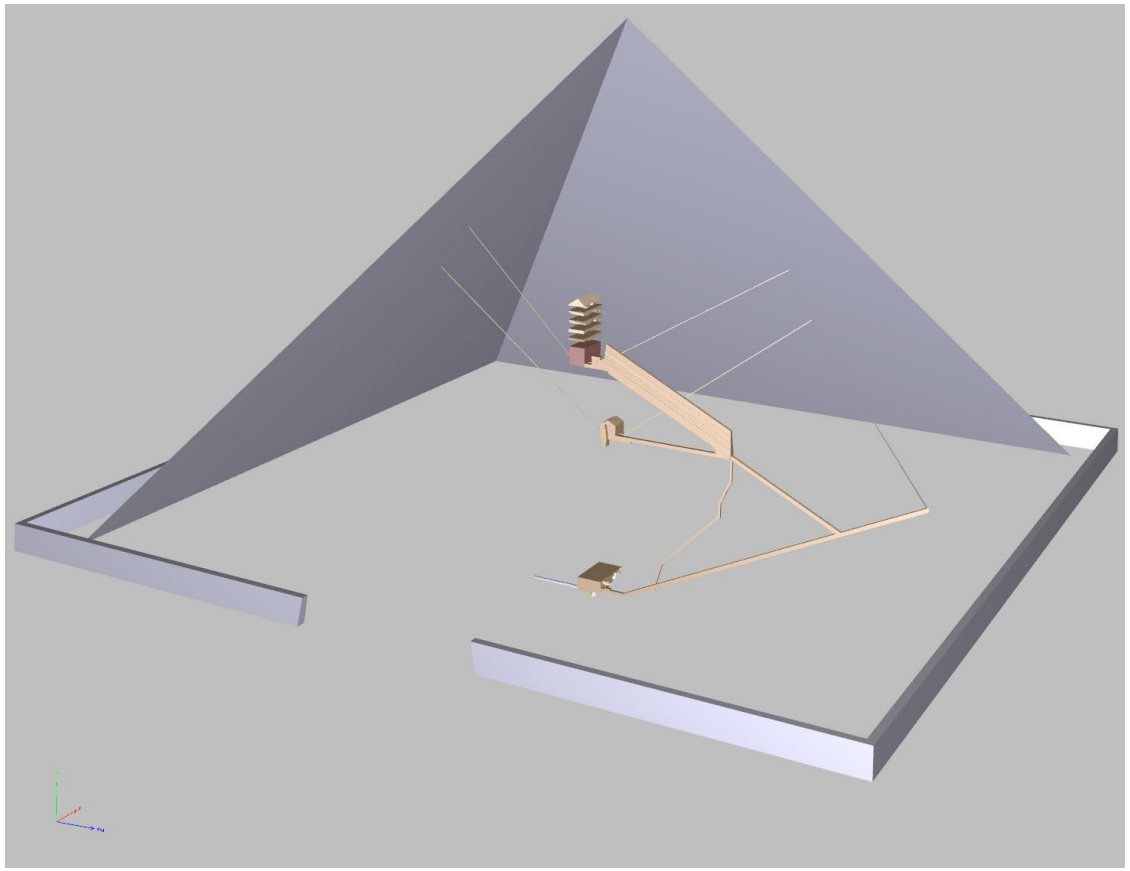
Phantom can be added as tessellated gdml model simply by editing text file.

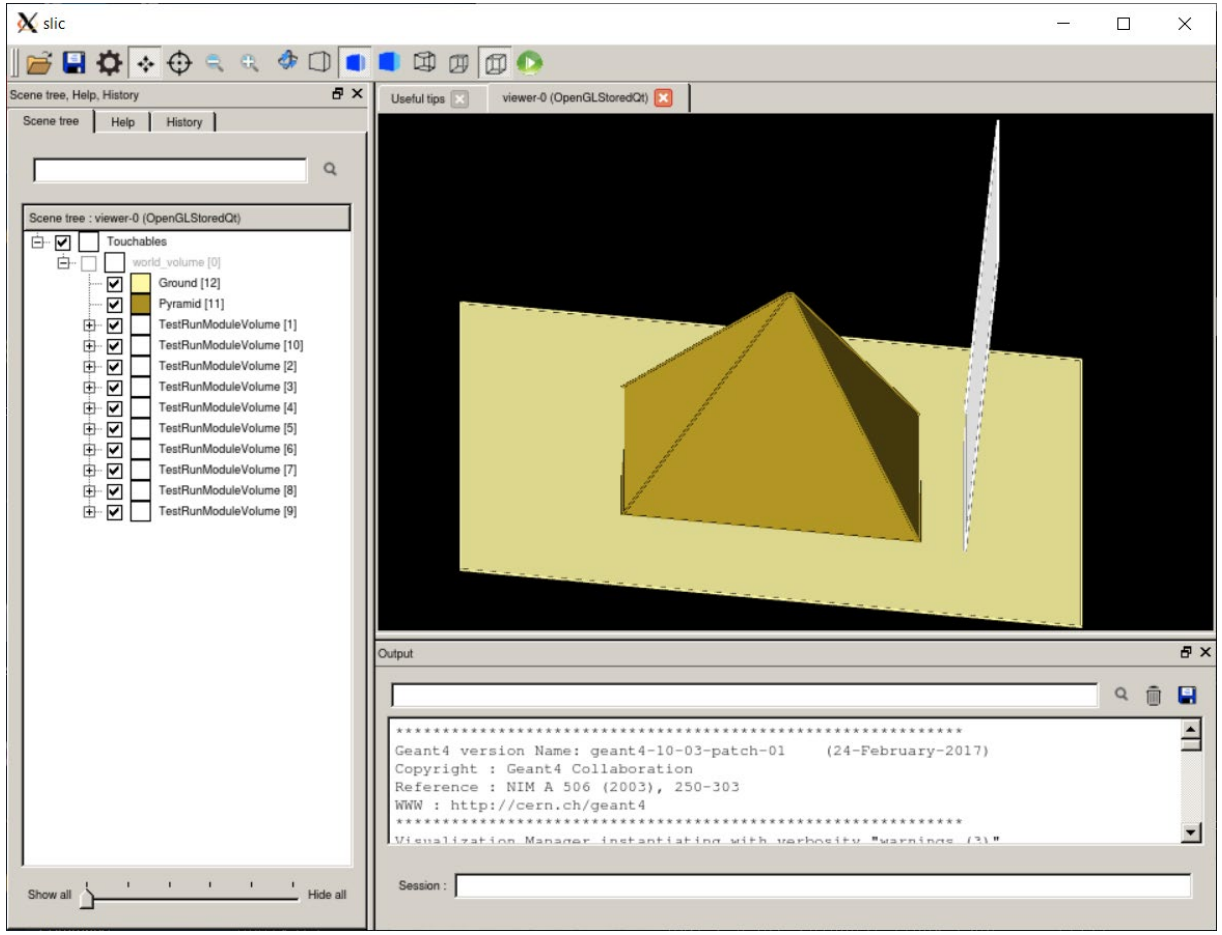
Beam can be provided by GPS or by generated beam particles in stdhep format.

Muon Tomography of Pyramid

Export VRML model of great pyramid of Khufu to STL, then convert to gdml

Detector is a series of scintillating fiber stations.
Cosmic rays generated using CRY, converted to stdhep.





Summary

- Extremely complex geometries can be modeled with Geant4 primitive or custom shapes. But sometimes it is necessary to simulate objects not easily modeled with them, such as
  - Biological phantoms
  - Complex CAD models
- STL provides a common denominator for most 3D models
- Mesh2gdml provides a pathway from meshed geometries to Geant4 via the G4TessellatedSolids in GDML.
- slic provides a standalone executable with which a number of different types of analyses can be conducted “out-of-the-box” without having to write any code.