SLAC-PUB-15298

mesh2gdml: from CAD to Geant4

Norman A. Graf

***Abstract*—As particle detectors and physics analyses become more complex, the need for more detailed simulations of the detector response becomes increasingly important. Traditionally, two-dimensional engineering drawings have been interpreted by physicists who then implement approximations to the individual parts in software simulation packages. This process requires a high level of experience in both the abstraction of the physical volumes as well as the implementation in code, and is prone to error. mesh2gdml is a program which allows three-dimensional tesselated geometrical volumes to be converted into a format which the Geant4 simulation toolkit can import directly. This provides a pathway for simulating the response of detector elements which have been designed in CAD programs without having to write any code.**

I. Introduction

THE ability to directly import CAD geometries into the detector response simulation package Geant4 [1] is a feature often requested by end users. There are many obstacles to a fully automatic workflow, including but not limited to: the difficulty in accessing proprietary formats, the mismatch between level of detail in producing a part and simulating it, the often disparate approaches to parent-child relationships and the difficulty in maintaining or assigning material definitions to parts. Despite these recognized limitations, the possibility of having one single geometric description of a detector, apparatus or experimental setup which can be used for both construction and simulation is quite appealing.

Geant4 provides a very rich library of basic geometrical shapes, often referred to as primitives, plus the ability to define compound geometries via boolean operations such as union, subtraction, and intersection. It is therefore capable of supporting extremely complex physical geometries composed of simple primitives. Most CAD systems also incorporate primitive volumes, but their definitions differ between programs and often do not map onto the Geant4 primitives, making the conversion difficult at best. However, one can also define a solid in Geant4 as a volume composed of surface facets. This G4TessellatedSolid can be composed of either triangular or quadrangular facets and therefore provides a mechanism for the programmatic importation of shapes and volumes defined in many CAD systems. In addition to CAD programs, there are very many 3D modeling programs which provide the user with convenient graphical user interfaces to create solid models. Usually aimed at gaming or rendering engines, these could be useful as a front end for a graphical geometry editor. Many output formats are supported, including tesselations. Furthermore, the use of tesselated objects to define a geometry provides a useful solution in cases where the objects are intrinsically irregular, such as biological phantoms.

The main impediment to the importation of CAD files into Geant4 has been their proprietary formats. Some existing solutions target recognized interchange formats such as STEP or IGES, but even these formats provide challenges, such as complicated file formats, possible loss of hierarchy or material association and little or no mapping to primitives. In this paper, we present mesh2gdml, a program which converts tesselated 3D objects in a number of open formats into a file which can be imported directly into Geant4. The Geometry Description Markup Language (GDML) [2] is a specialized XML-based language designed as an application-independent persistence format for describing the geometries of detectors associated with physics measurements. It provides support for all of the Geant4 primitives and can be used by Geant4 for the description of detector elements. By targeting this intermediate file format, we make it possible for end users to import geometries without having to write any additional code. This is not a ”silver bullet” which provides automatic translation of CAD models into Geant4 solids, but does provide a pathway for importing shapes which are otherwise too complicated to easily model with Geant4 primitives.

II. Tesselated Objects

Most CAD programs use analytic 3D surfaces and curves to define the topological boundaries of faces and edges of volumes. These boundary representations, also known as BREPs, can be very efficiently implemented, but vendors develop their own proprietary file formats. In addition to being undocumented, these formats also differ between programs, and often change between program versions. There are two major vendor-neutral data formats which enable exchange of models between CAD systems, STEP (Standard for the Exchange of Product model data) and IGES(Initial Graphics Exchange Specification). However, as their names imply, both standards encompass digital data well beyond the description of 3D volumes. Therefore, implementing a program which will be able to read and interpret generic STEP or IGES files is very difficult. Additionally, programmatically mapping arbitrary 3D volumes onto Geant4 primitives would be an almost impossible task. However, one can always approximate these volumes with a mesh of small polygons in a process normally referred to as tesselation. Geant4 provides support for such meshes in the form of the G4TessellatedSolid object. It is defined as a list of facets, either triangular or quadrangular, implemented as G4TriangularFacet and G4QuadrangularFacet, respectively. Thanks to the proliferation of rapid prototyping and additive manufacturing processes, the surface tesselation language STL (also known as STereo Lithography) format is

Manuscript received November 19, 2012. This work was supported in part by the U.S. Department of Energy.

N. A. Graf is with the SLAC National Accelerator Laboratory, Menlo Park, CA 94025 USA (telephone: 650-926-5317, e-mail: Norman.Graf@slac.stanford.edu).

Presented at IEEE 2012 Nuclear Science Symposium, Medical Imaging Conference
Anaheim, California, October 29 - November 3, 2012

Work supported by US Department of Energy contract DE-AC02-76SF00515.

Fig. 1. Analytic or "primitive" sphere. A point is inside the sphere if $r < r_{sphere}$. Other calculations such as distance to edge are simple as well.

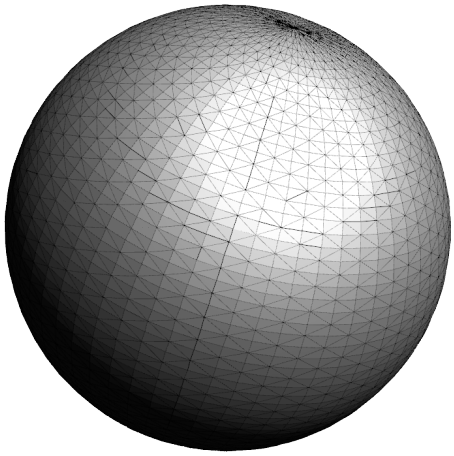

Fig. 2. A tesselated approximation to a sphere. Each facet must be checked to see whether a point is inside the volume.

the industrial standard for handling triangulated meshes and is ubiquitous as an export format for both CAD and other 3D modelling software. The format consists of a plain list of three dimensional corner point coordinates (vertex) and flat triangles (facet) with an associated normal vector, making it an ideal candidate for importation into Geant4. In what follows we restrict our comments to the use of tesselated solids described in STL format, but other mesh formats, such as OBJ, OFF, PLY, etc. can also be imported. By targeting STL as the lowest-common-denominator geometry format, we can rely on the CAD programs to convert their native representations into meshes. This ensures that the exported geometry is closed, complete, and has the correct level of detail and precision as determined by the design engineers.

III. From STL to GDML

There are both plain text and binary formats defined for STL files. The ASCII format is outline below.

solid name
  facet normal $n_1\ n_2\ n_3$
    outer loop
      vertex $v1_x\ v1_y\ v1_z$
      vertex $v2_x\ v2_y\ v2_z$
      vertex $v3_x\ v3_y\ v3_z$
    end loop
  } repeated nFacet times (1)
end solid name

Although verbose, the format is quite simple and easily parsed. The STL facets are translated directly into G4TriangularFacets which are used to create a G4TessellatedSolid. If the file describes a single solid, this is all which needs to be done. There is, however, no way to specify how many topologically distinct objects are described in any file, and most often many different volumes are included in a single file. Since there is no other structure in an STL file, one has to also solve the problem of creating "topology from a bucket of facets." This has been done in a manner which is efficient enough to handle the test cases encountered to-date (see following sections for examples). No further tests of the validity of the objects are currently performed; it is assumed that the facets fully enclose a space which is the solid, that solids do not overlap, and that the precision of the tesselation is sufficient to accurately model the geometry. In addition to defining the geometrical properties of the solid one needs to add physical characteristics in order to create a physical volume for Geant4. The most important characteristic is the material of the volume. The assignment of material to the newly created solid or solids requires manual intervention. If all of the volumes in a file are composed of the same material this assignment can be done by simply providing the material name as a command-line argument to the conversion program. The most efficient semi-automated workflow would involve exporting from the CAD program all volumes of the same material to separate STL files and converting each in turn, applying the correct material at conversion time. Work is ongoing to develop a graphical user interface which allows the user to graphically select individual geometrical objects and assign material to them. Figure 9 shows a screen capture of the current program. Having read in an STL file describing the HPS detector (see following section) the code has successfully identified the almost thousand topologically distinct detector elements and created individual tesselated volumes. The graphical interface is being used to select each of the distinct elements (bounding box vertices highlighted in red) and material selected from a drop-down list of defined materials will be assigned. Additionally, the ability to create geometrical hierarchies for the geometry is being implemented. Finally, one can either create a world volume from the bounding box of the volume(s) found in the STL file to use standalone within Geant4, or keep the individual volumes to aggregate or incorporate into a common world volume at a later stage. The output geometry is stored as GDML.

IV. Performance Issues

Much, if not most, of the time spent during a Geant4 simulation is used to calculate such geometrical quantities

as which volume the current stepping point is in and what the distance to the next boundary is. One clear advantage of Geant4 primitives or boundary representations is the ability to analytically calculate such quantities. Doing so for tesselated volumes requires looping over all of the facets defining the object, which can be very CPU intensive. For instance, as shown in Figures 1 and 2, calculating whether a point is inside a sphere involves a simple comparison of the radius of the point to the radius of the sphere in the local coordinates of the sphere. Doing so for a tesselated approximation of a sphere involves looping over each of the individual facets, calculating the orientation of the point with respect to the polygon and maintaining a running tally before finally deciding whether the point is inside the volume and, if so, the distance to the volume's surface. This clearly increases faster than linearly with the resolution of the tesselated approximation. In order to benchmark the performance of Geant4 using tesselated volumes, we have also written code which allows the exportation of Geant4 geometries in STL format. This geometry can then be reimported into Geant4 after being processed with mesh2gdml, allowing us to directly compare the CPU time difference between a geometry composed of primitives and tesselated solids. A number of systematic studies are currently underway. A side effect of this STL export is the ability to create a real 3D model of the Geant4 geometry on 3D printers. This would enable rapid prototyping of parts, direct comparison of the modeled geometry to CAD geometry, communication with colleagues and outreach to the public.

V. Example Implementations

We recently tested the implementation on a number of detectors for which we had access to the CAD models. The designs were exported in STL format directly from the modelling software and segregated by material. The files were converted to GDML and imported into Geant4 as a proof of concept.

A. EXO

The Enriched Xenon Observatory (EXO) [3] is an experiment designed to detect neutrino-less double beta decay using a liquid Xenon Time Projection Chamber (TPC). The need to reduce radioactive backgrounds led to a detector design with a minimum number of materials, allowing essentially the full detector to be exported to only four STL files. The roughly seven thousand topologically distinct detector elements were correctly identified from the over two million facets in the STL files. These volumes were assigned materials, converted to GDML and then imported into Geant4. Although the CPU time to process such a large number of facets makes this model prohibitive for large scale simulations it does serve as an end-to-end demonstration of the technical feasibility of this conversion approach.

B. HPS

The Heavy Photon Search (HPS) [4] is a fixed-target experiment at Jefferson Lab aimed at discovering a hidden-sector, heavy photon. Electron-positron pairs produced in the decay of such a particle are momentum-analyzed using silicon strip detectors inside a dipole magnetic field and their energy is measured in a crystal calorimeter array. The time to design and optimize the detector via simulations was extremely short, yet the need to precisely model the very high backgrounds very close to the beam was critical to the success of the recently completed test run. The support structures and vacuum enclosures were exported from the CAD design and used directly in the Geant4 simulations. Because these elements were only rarely hit by particles, one could afford to use the tesselated volumes in the production simulation of events. Figures 5 and 6 show the CAD model and the GDML model used in the Geant4 simulations.

C. LHCb

The LHCb experiment [5] is designed to study the subtle differences between matter and antimatter produced at CERN's Large Hadron Collider. The silicon sensors of the vertex detector are positioned as close as possible to the interaction region, in a vacuum vessel separated by a very thin sheet of aluminum from the primary vacuum of the beam pipe. The shape of this "rf foil" is very complicated and extremely difficult to model using Geant4 primitives. Figure 7 shows the CAD model of the "rf foil" and Figure 8 shows a detail view of the resulting GDML model. This would clearly not be used for large-scale simulations of millions of events, but can be critical in understanding subtle differences between the predicted and measured track parameter resolutions, especially for low momentum tracks.

VI. Future Directions

STL was originally defined for monochrome 3D printers and, as seen above, does not contain any information regarding color, texture or other common CAD model attributes such as volume hierarchy. Recently, a new standard has been proposed to serve as a replacement for STL. ASTM F2915 [6] defines a standard specification for an Additive Manufacturing File Format (AMF). It takes the STL format for vertices and facets and adds native support for color, materials, and hierarchies, known as constellations. The format is xml-based, with the following tags:

<**object**>
: Defines a volume associated with a material ID.

<**material**>
: Optional element defines one or more materials.

<**texture**>
: Optional element defines images or textures for color or texture mapping.

<**constellation**>
: Optional element provides hierarchy support.

<**metadata**>
: Optional element contains additional information.

The specification has just recently been approved but it is expected that CAD vendors will begin to add support for this format. AMF goes a long way towards resolving the deficiencies of STL and, in expectation of industry support

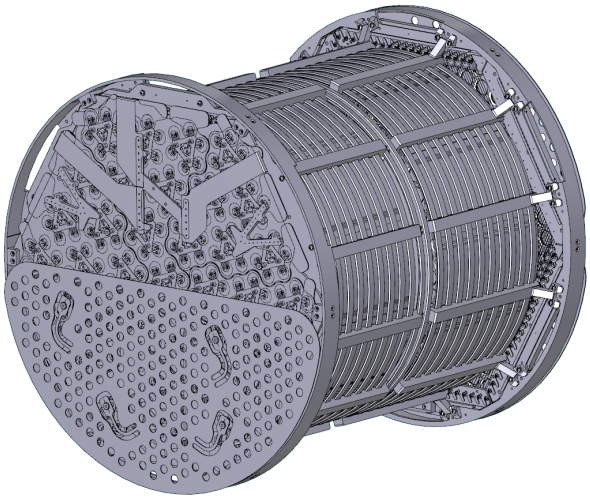

Fig. 3. The EXO TPC as modelled in SolidEdge with over 7000 discrete elements.

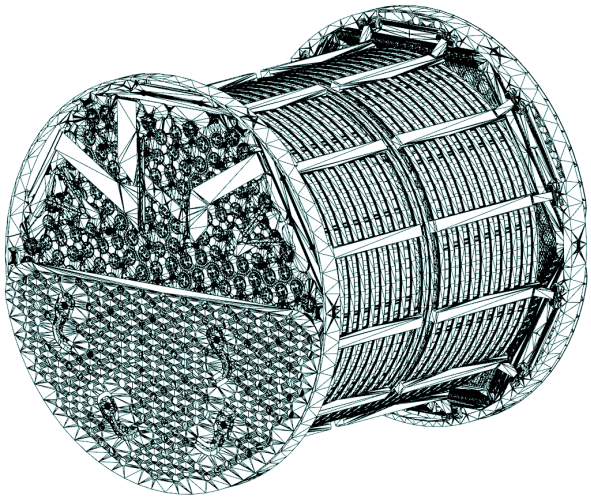

Fig. 4. A tesselated approximation to the EXO TPC with over 2 million facets.

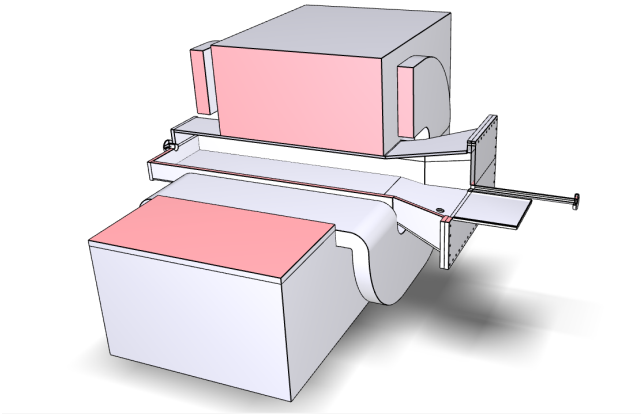

Fig. 5. The HPS dipole magnet and vacuum vessel CAD model.

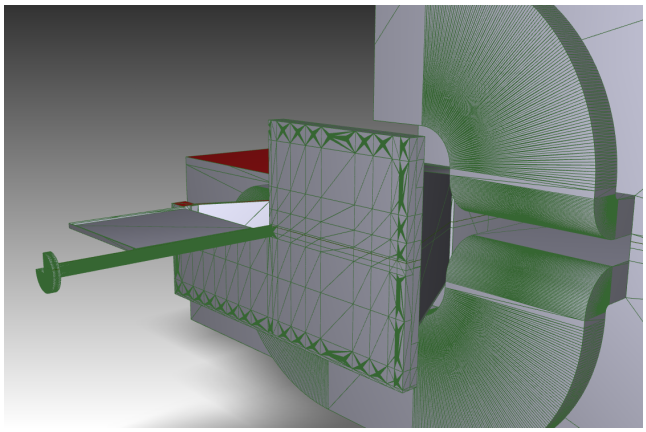

Fig. 6. A closeup view of the tesselated approximation to the HPS dipole magnet and vacuum vessel

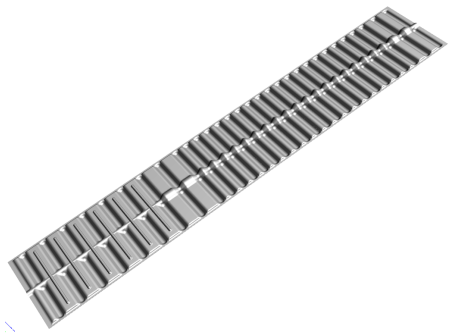

Fig. 7. The LHCb "rf foil".

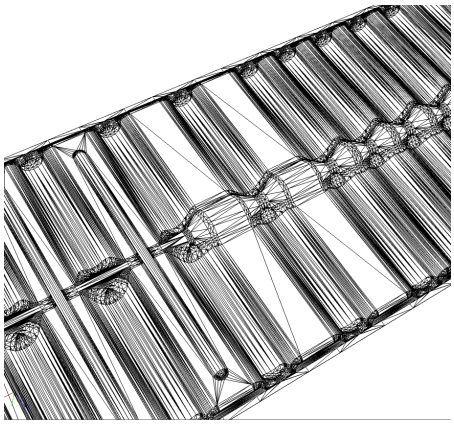

Fig. 8. A closeup view of the tesselated approximation to the LHCb "rf foil."

for this standard, we have begun implementing code which targets the additional information available in this format.

Being able to recognize volumes which correspond to Geant4 primitives and provide the user with the ability to replace the tesselated shape with its primitive counterpart would improve the performance of the resulting model. Automating this procedure would, however, be difficult for anything other than a few of the very simplest shapes.

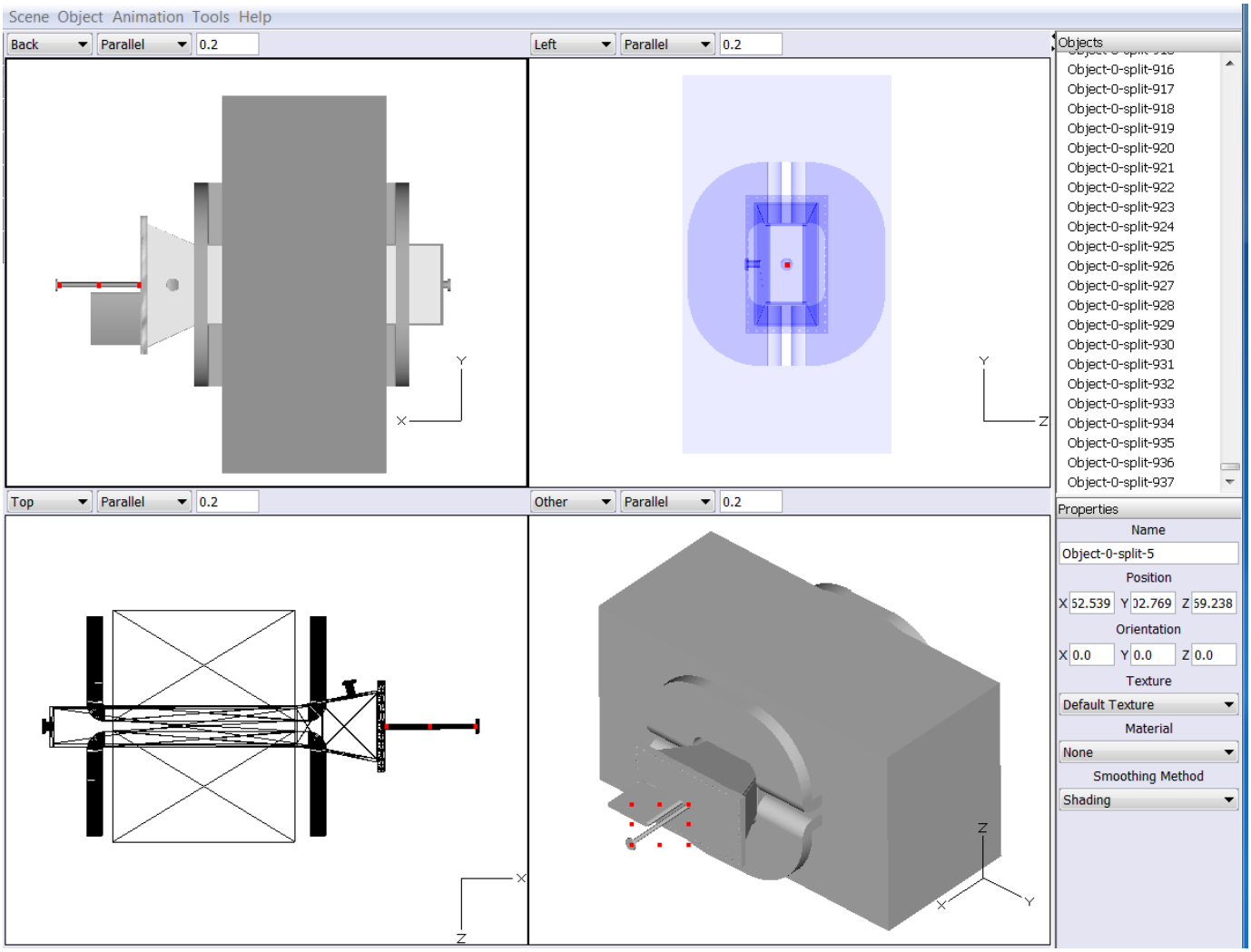

Fig. 9. Screen capture of the graphical user interface used to assign materials to the discrete volumes identified within an STL file before conversion to GDML.

VII. Conclusions

The program mesh2gdml provides a workflow whereby geometrical volumes defined by triangular or quadrangular meshes, such as STL, can be converted into volumes appropriate for simulation using Geant4. Writing the resulting volumes to GDML files makes them available for simulation within Geant4 without any further coding. Despite the inherent performance issues related to navigating through geometries composed of many individual facets and the requirement that material be assigned manually to volumes during the translation process, we believe the approach outlined in this talk provides access to a wider range of geometry inputs and will prove to be useful to a number of user communities interested in using Geant4 to simulate complex geometries without having to develop any code. The package is still under development and testing but we anticipate a release of the program by the second quarter of 2013.

References

[1] S. Agostinelli et al., Nucl. Instr. and Meth. A, 506 (2003) 250
J. Allison et al., IEEE Trans. On Nucl. Sci., 53 (2006) 270
http://geant4.cern.ch/

[2] R. Chytracek, J. McCormick, W. Pokorski, G. Santin, IEEE Trans. On Nucl. Sci., 53 (2006) 2892

[3] http://www-project.slac.stanford.edu/exo/

[4] https://confluence.slac.stanford.edu/display/hpsg/Heavy+Photon+Search+Experiment

[5] http://lhcb.web.cern.ch/lhcb/

[6] www.astm.org/Standards/F2915.htm