A CAD Interface for GEANT4

Christopher M Poole, Iwan Cornelius, Jamie V Trapp, Christian M Langton

***Abstract*—Typically used as a tool for Monte Carlo simulation of high energy physics experiments, GEANT4 is increasingly being employed for the simulation of complex radiotherapy treatments. Often the specification of components within a clinical linear accelerator treatment head is provided in a CAD file format. Direct import of these CAD files into GEANT4 may not be possible, and complex components such as individual leaves within a multi-leaf collimator may be difficult to define via other means. Solutions that allow for users to work around the limited support in the GEANT4 toolkit for loading predefined CAD geometries has been presented by others, however these solutions require intermediate file format conversion using commercial software. Here within we describe a technique that allows for CAD models to be directly loaded as geometry without the need for commercial software and intermediate file format conversion. Robustness of the interface was tested using a set of CAD models of various complexity; for the models used in testing, no import errors were reported and all geometry was found to be navigable by GEANT4.**

***Index Terms*—Monte Carlo, GEANT4, BEAMnrc, CAD, geometry, radiotherapy**

I. Introduction

GEOMETRY and Tracking (GEANT4) is a C++ toolkit specifically designed to track particles traversing a geometry whilst being subject to physical processes, it finds application in fields such as nuclear and particle physics and space engineering, with increasing use in medical physics [2], [6], [10], [11], [14]. Numerous physical processes can be modeled including photo-nuclear interactions, optical processes such as scintillation and Cherenkov radiation and other particle interactions over a wide energy range (250 eV up to TeV energies); the full gamut of processes available to the user is described by others [2]. Additionally, the toolkit provides functionality for the inclusion and exclusion of the desired processes as well as sufficient extensibility to included custom or user defined processes. Fast and effective geometry definition is also available to the user for relatively simple objects using constructs such as `G4Orb` for defining orbs or spheres, `G4Box` for defining rectangular prisms and the concept of boolean solids [2]. At the most fundamental level, the GEANT4 geometry hierarchy is divided into solids, logical volumes and physical volumes where solids described shape, logical volumes define material properties and mother daughter relations, and physical volumes define placement within the mother volume.

Geometry Description Mark-up Language (GDML) is a comprehensive geometry description format using Extensible Mark-up Language (XML) that allows for the persistence of many aspects of a geometry, including material properties and assemblies [7]. Functionality within the toolkit readily allows for geometries to be saved and reloaded as GDML, however there is little support for converting pre-existing user defined Computer Aided Design (CAD) models to GDML or indeed directly loading these same CAD models as geometry [8]. Of the two methods available to the user for accomplishing this task, both are reliant of intermediate file format conversion using commercial software [5], [8].

Many CAD packages export files compliant to the standard for the exchange of product model data (STEP - ISO 10303), a standard designed to supersede the still widely used initial graphics exchange specification (IGES) [4], [13]. As such, ISO standard STEP has been the target format for loading CAD geometry into GEANT4. Persistence example, with identifier 'G02', distributed with GEANT4 describes loading STEP Tools (STEP Tools Incorporated, New York) files directly, with the intermediate conversion of STEP to STEP Tools using the commercial ST-Viewer program or ST-Developer libraries (STEP Tools Incorporated, New York), refer to Fig.1(a). This process allows for assemblies of components to be loaded directly, however the STEP Tools programs and libraries may be prohibitively expensive for some users. Constantine et al described the process of converting STEP to GDML using the commercial software FastRad (Tests & Radiation - Toulouse) [5], refer to Fig.1(b). FastRad again may be considered prohibitively expensive for some users with the requirement of annual licensing, in addition to this, the trial version limits the conversion process to no more than 20 elements per assembly.

Principally, CAD file formats such as stereo-lithography format (STL) or Stanford polygon file format (PLY) describing a 3-dimensional volume use tessellated polygon meshes (usually triangular or quadrangular) to define its closed surface. A cloud of points in 3-dimensional space define the vertexes of the mesh and a collection of faces define the interconnects between the vertexes. The equivalent geometric construct in GEANT4 to this scheme is the tessellated solid (`G4TessellatedSolid`); except for the two previously mentioned techniques, mappings between the `G4TessellatedSolid` construct and CAD file formats are not readily available [8]. Programming toolkits however, already exist for the creation and manipulation of triangular and quadrangular meshes; the kind exportable by many modern CAD packages [1], [3]. Most notably the templated C++ library VCGLIB (Visual Computing Laboratory, Italy) offers advanced mesh manipulation routines and point cloud surface reconstruction algorithms. In addition to this, VCGLIB offers an import/export interface for many common CAD file formats [1].

Correspondence can be directed to `christopher.poole@qut.edu.au`

All authors are with the Discipline of Physics, Faculty of Science and Technology, and the Institute of Health and Biomedical Innovation at Queensland University of Technology, Brisbane, Australia.

This work is funded by the Queensland Cancer Physics Collaborative, and Cancer Australia (Department of Health and Ageing) Research Grant 614217.

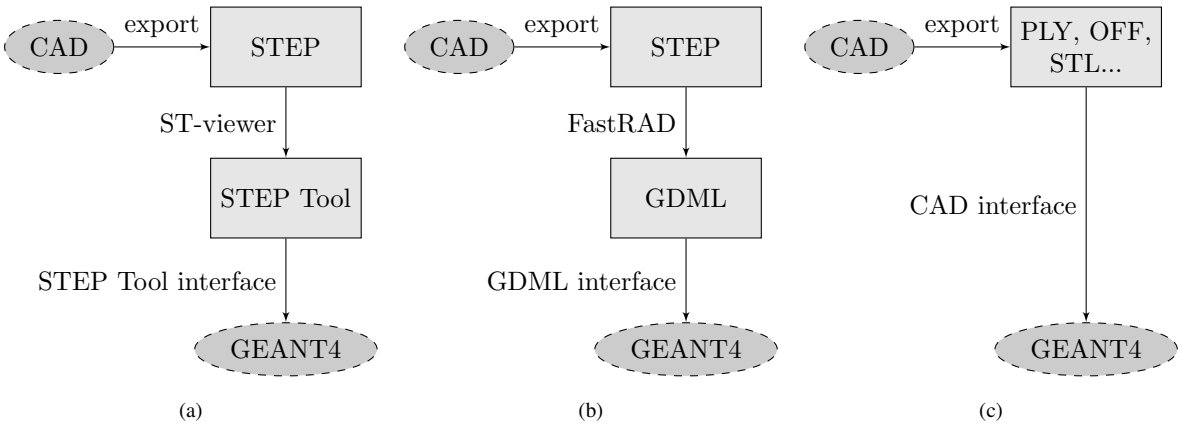


Fig. 1: A diagrammatic comparison between two currently available CAD import techniques where (a) shows the technique described in persistence example G02, (b) shows the STEP to GDML conversion technique [8], and (c) shows the new direct import technique proposed here within.

Here within we describe a very simple technique that allows for CAD models to be directly loaded as geometry in GEANT4 using VCGLIB. This technique is capable of handling many file formates exportable by CAD programs whilst not relying on intermediate file format conversions.

II. Methods

A. GEANT4 CAD Interface

Three C++ classes were created describing a vertex, face and a triangular mesh. The class `CADVertex` of base type `vcg::Vertex` was used to store a single vertex defined by three double precision values $(x,\ y,\ z)$. For faces, the class `CADFace` of base type `vcg::Face` was used to store a reference to all vertices that define the face $(a,\ b,\ c)$. Finally, the class `CADMesh` templated on two `std::vector` containers was created. The first vector container was templated on the custom type `CADVertex`, and the second on the custom type `CADFace` - in effect the `CADMesh` type contained two collections, one describing all the vertices in a mesh and another describing all of the interconnect between the vertexes (the faces). Through an instance of a `CADMesh`, the description of a triangular mesh with a cloud of vertices interconnected by a group of faces could be created.

Wrappers in VCGLIB extend the VCGLIB core to provide additional mesh manipulation functionality including importers and exporters. Using the `Open` method in the wrapper `vcg::tri::io::ImporterPLY` and parsing to it a PLY file and a reference to an instance of `CADMesh`, the instance of `CADMesh` was populated with a 3-dimensional triangular facet mesh described the PLY file. Other triangular facet mesh importers include OBJ, OFF, and STL file formats [1]. Random access to faces in the mesh was available using the `CADMesh::FaceIterator` iterator where each face provided a pointer to each vertex $(a,\ b,\ c)$ of the current face and each vertex provided pointers to its coordinates $(x,\ y,\ z)$.

For each `CADMesh` a `G4TessellatedSolid` was initialised in GEANT4 (version 9.4, patch 01, 25^{th} February 2011). Using the `CADMesh::FaceIterator` iterator, the `AddFacet` method of the `G4TessellatedSolid` was called once for each mesh face. The `AddFacet` method took three `G4ThreeVector` arguments describing the coordinates of each vertex that define the current face. Once the iterator looped over all faces, the `G4TessellatedSolid::SetSolidClosed` method was called, preventing further faces from being added to the volume. The `G4TessellatedSolid`, was now available for material assignment and placement within the GEANT4 user geometry using standard programmatic techniques in GEANT4.

At the user level, all functionality provided by the CAD interface could be incorporated into any pre-existing geometry with the inclusion of a single header file in the user detector construction. Initialisation of a `CADMesh` object along with a volume described in a CAD file, would provide a `G4TessellatedSolid` suitable for inclusion in the user geometry in the same manner as with any typical `G4Solid` object. Following the standard GEANT4 geometry hierarchy, material properties and volume meta-data unique to the volume could be assigned with association to a `G4LogicalVolume`, and placement of the `G4TessellatedSolid` within a mother volume with association to a `G4PhysicalVolume`, see code listing 1.

B. Test Cases

Six test volumes were used to verify the performance of the GEANT4 CAD interface. Three simple geometries, a truncated cone and a sphere generated using MeshLab (Visual Computing Laboratory, Italy) and an artificial hip (generated in-house) were saved in all formats capable of import by the CAD interface. Additionally, a flattening filter from a Varian clinical linear accelerator, a single leaf from a Varian multi-leaf collimator and a model of a pelvis from a CIRS Pelvic Phantom (Model 048) were loaded into GEANT4 using the CAD interface [12]. Each volume, when loaded was visually inspected for qualitative geometric integrity using the GEANT4 OpenGL viewer. Tessellated solid meta-data was dumped for

Listing 1: Basic usage of the CADMesh class in a user detector constructor

```
#include "CADMesh.hh"
...
CADMesh mesh;
G4VSolid * solid;
G4LogicalVolume * logical;
G4VPhysicalVolume * physical;
...
solid = mesh.LoadMesh("sphere.stl", "STL");
logical = new G4LogicalVolume(solid, water,
          "logical", 0, 0, 0);
physical = new G4PVPlacement(0,
           G4ThreeVector(), logical,
           "physical", world, false, 0);
```

Name	Verts	Faces	Inspection	Dump/Navigation
cone	50	96	Pass	Pass
sphere	642	1280	Pass	Pass
hip	502	1000	Pass	Pass
leaf	120	236	Pass	Pass
filter	1235	2346	Pass	Pass
pelvis	4986	10000	Pass	Pass

TABLE I: Properties of the six volume loaded into GEANT4 via the proposed CAD interface.

each solid using the `G4TessellatedSolid::DumpInfo` method providing a list of each face and vertex coordinate used to define the GEANT4 volume. This data was then compared directly to the original CAD file describing the same volume. A pass level of 100% matching was set, no rounding errors of the vertex coordinates were accepted and vertex and face counts had to agree exactly.

So as to ensure the loaded geometries were navigable by the GEANT4 kernel, each volume was assigned the material `G4_WATER` and positioned at the center of a `G4_AIR` filled world volume. A general particle source (GPS) was initialised and configured to produce a beam of geantinos (the standard GEANT4 debugging pseudo-particle) aimed at the test volume. Each test volume was bombarded with $100,000$ geantinos with the tracking verbosity level set to one and the step length set to $0.1\ mm$, ensuring any navigation errors associated with the imported CAD geometry were output; the angular distribution of the beam was set so as to target the entire volume ensuring all faces were inside the beam. At any time, if the navigator was unable to determine if it was inside or outside of the tessellated volume as a consequence of invalid volume definition, the test was failed.

III. Results

Fig.2(a), Fig.2(b) and Fig.2(c) show three simple geometries loaded into GEANT4 via the CAD interface described in section II-A; the geometries were visualised using the GEANT4 OpenGL viewer. Further to this, Fig.2(d), Fig.2(e) and Fig.2(f) show more complex geometries loaded into GEANT4. All test volumes imported using the CAD interface, independent of the CAD file format, passed qualitative visual inspection - no corruption of the geometry was visible. Table I displays the vertex and face counts for each volume; there was no difference between the vertex and facet counts reported by VCGLIB or GEANT4. A comparison between the original CAD file and the dumped face and vertex information from each tessellated solid generated no errors. When bombarded with a simulated source of geantinos, no navigation errors were reported. In the case that a face were to be purposefully excluded from the `G4TessellatedSolid`, the volume would be unnavigable as a consequence of an undefined 'inside' or 'outside'; a non-closed solid.

IV. Discussion & Conclusion

Using the templated C++ mesh manipulation library VCGLIB, we have demonstrated a technique whereby CAD models may be directly imported as geometry into GEANT4 without the express need for file format conversion using commercial software. Fig.2 shows a collection of CAD models successfully loaded into GEANT4 via the CAD interface. Reliability of the interface in terms of preserving model integrity during import was evaluated quantitatively by comparing the vertex coordinates as reported by GEANT4 to the actual vertex coordinates described in the source CAD file. Whilst the interface does not parse material properties or other GEANT4 specific meta-data, a CAD model imported using the interface may be saved as part of an assembly in GDML using tools that are already part of the toolkit. By removing the intermediate file format conversion step, unnecessary expenditure on commercial third party software can be avoided and any CAD model described by a triangular tessellated surface may be directly loaded as geometry within GEANT4. Further work will focus on increasing the number of CAD file formats available for direct import using the technique as well as enabling support for assemblies and the preservation of volume specific meta-data such as material composition. Currently this work is part of software tool using GEANT4 for the simulation of clinical linear accelerators [9], which is freely available (along with accompanying documentation) upon request made to the corresponding author. Source code for the CAD interface described here within may be acquired from: `https://code.google.com/p/cadmesh/`

References

[1] A portable C++ templated library for the manipulation, processing of triangle and tetrahedral meshes. Computer software. http://vcg.sourceforge.net/, 2011.

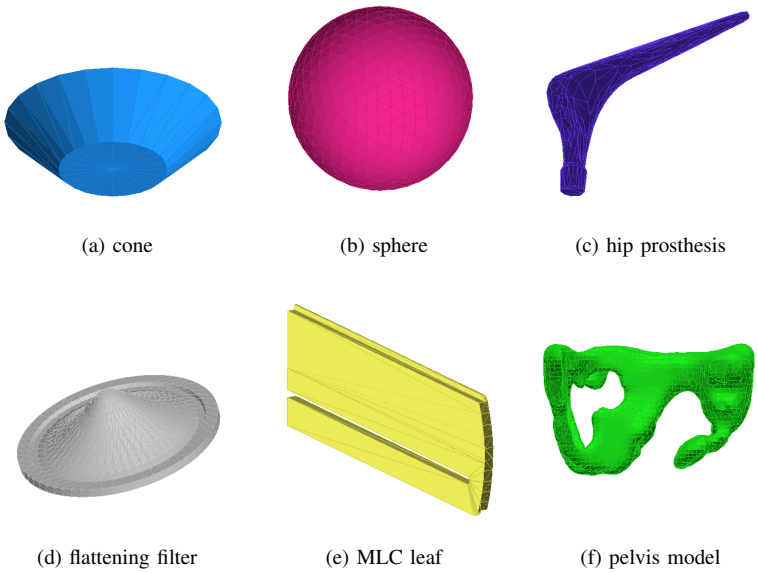
[2] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G. Barrand, et al. Geant4 simulation toolkit. *Nuclear Instruments and Methods in Physics Research-Section A Only*, 506(3):250–303, 2003.

[3] N. Amenta, S. Choi, and R.K. Kolluri. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266. ACM, 2001.

[4] US Product Data Association. Initial Graphics Exchange Specification 5, 2007.

[5] T. Beutier, E. Delage, M. Wouts, O. Serres, and P. F. Peyrard. FASTRAD new tool for radiation prediction. In *Radiation and Its Effects on Components and Systems, 2003. RADECS 2003. Proceedings of the 7th European Conference on*, pages 181–183, 2003.

[6] B. Caccia, C. Andenna, and G. A. P. Cirrone. MedLinac2: a GEANT4 based software package for radiotherapy. *Annali dell'Istituto superiore di sanità*, 46:173–177, 2010.



(a) cone (b) sphere (c) hip prosthesis

(d) flattening filter (e) MLC leaf (f) pelvis model

Fig. 2: Six test geometries loaded directly into GEANT4 using the proposed CAD interface.

[7] R. Chytracek, J. McCormick, W. Pokorski, and G. Santin. Geometry description markup language for physics simulation and analysis applications. *Nuclear Science, IEEE Transactions on*, 53(5):2892–2896, 2006.

[8] M. Constantin, D. E. Constantin, P. J. Keall, A. Narula, M. Svatos, and J. Perl. Linking computer-aided design (CAD) to Geant4-based Monte Carlo simulations for precise implementation of complex treatment head geometries. *Physics in Medicine and Biology*, 55:N211, 2010.

[9] I. Cornelius, B. Hill, N. Middlebrook, C. Poole, B. Oborn, and C. Langton. Commissioning of a Geant4 based treatment plan simulation tool: linac model and DICOM-RT interface. *Arxiv preprint arXiv:1104.5082*, 2011.

[10] L. Grevillot, T. Frisson, D. Maneval, N. Zahra, J. N. Badel, and D. Sarrut. Simulation of a 6 MV Elekta Precise Linac photon beam using GATE/GEANT4. *Physics in Medicine and Biology*, 56:903, 2011.

[11] S. Jan, D. Benoit, E. Becheva, T. Carlier, F. Cassol, P. Descourt, T. Frisson, L. Grevillot, L. Guigues, L. Maigne, et al. GATE V6: a major enhancement of the GATE simulation platform enabling modelling of CT and radiotherapy. *Physics in Medicine and Biology*, 56:881, 2011.

[12] C. M. Poole, I. Cornelius, J. V. Trapp, and C. M. Langton. Vectorised DICOM-RT regions of interest as G4TessellatedSolids. In *1st Geant4 Australian School and User Workshop*, 2011.

[13] M.J. Pratt. Introduction to ISO 10303the STEP standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1:102, 2001.

[14] E. Spezi and G. Lewis. An overview of Monte Carlo treatment planning for radiotherapy. *Radiation protection dosimetry*, 2008.