MOLLER Simulation Software: GEMC Implementation

Dustin McNulty
Idaho State University
mcnulty@jlab.org

Thanks to the SoLID group for paving the way

March 24, 2011

MOLLER Simulation Software: GEMC Implementation

Outline

- GEMC Intro
- Towards a MOLLER GEMC App
- Issues and Summary

GEMC (GEant4 Monte Carlo)

(Slide taken from Maurizio)

- Developed and maintained by Maurizio Ungaro. Used by CLAS12, GlueX, HPS, and others. GEMC homepage: https://gemc.jlab.org
- GEMC is a C++ program that uses GEANT4 to simulate the passage of particles through matter
- Detector information is stored in a jlab mysql server. Configuration changes immediately available to users without recompiling code
- Developers interact with database and do not need to know C++ or Geant4 to build detector and run the simulation
- Hit Process Factory: associate detectors with external digitization routines at runtime (Nice feature for seamless development of DAQ hardware and experiment analyzer)

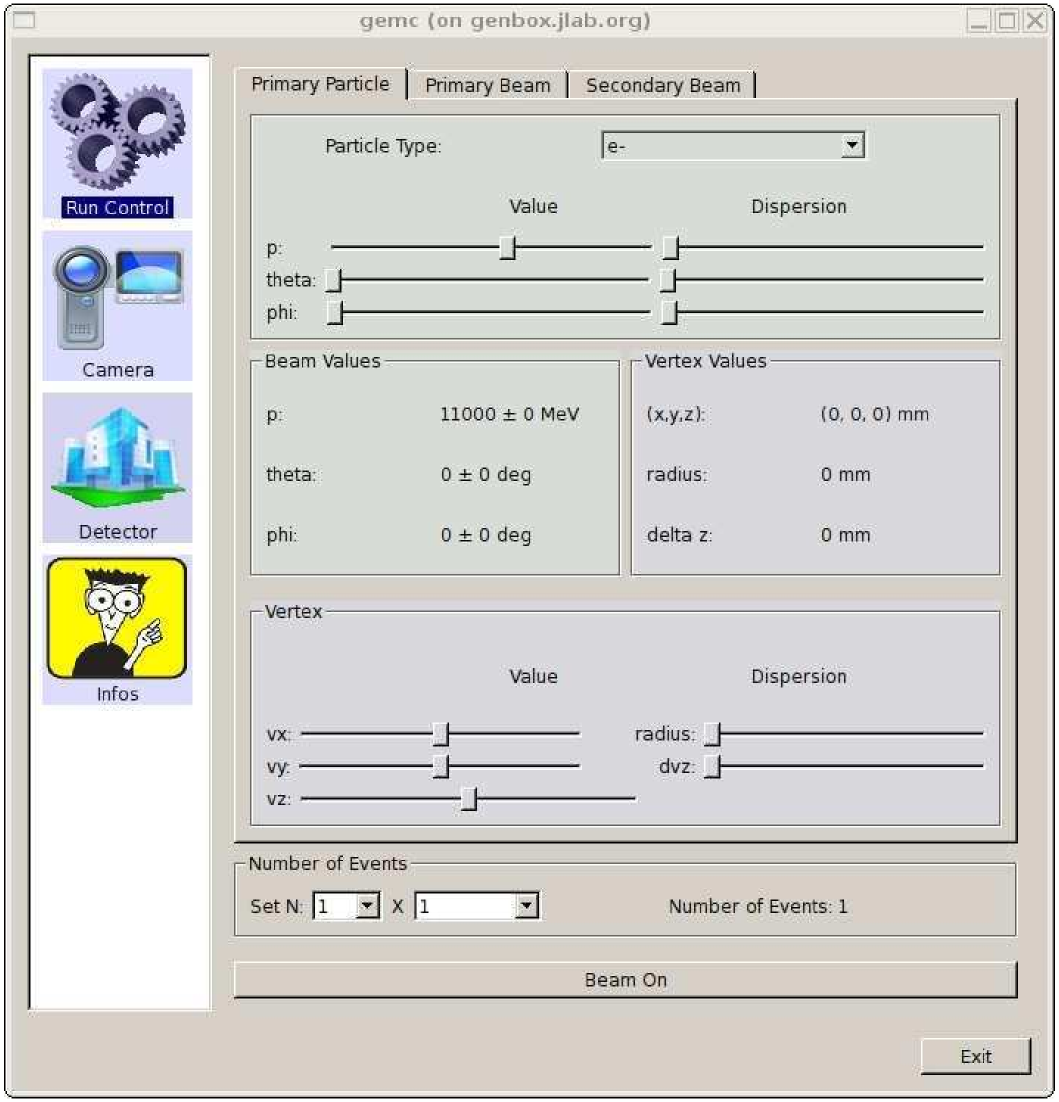
GEMC’s Detector Database

- GEMC uses its own well-defined DB table format for specifying a detector’s geometry, material, sensitivity, and output hit types
- The idea is to use same DB for design, simulation, and reconstruction
- Perl scripts are used to generate the detector information which is then loaded into DB; several examples/tools exist
- DB Advantages: Easy to use, extremely flexible, immediately available, no recompiling, detector sensitivity incorporated
- DB Disadvantages: DB can become quite fluid during initial code development which can cause confusion if not careful; also, the DB does not necessarily address our desire to go from CAD to GDML for seamless geometry import (maybe a pipe-dream anyway)

More GEMC Details

- GEMC has several predefined detector outputs or “banks” (these can be modified or new ones created)
- GEMC has support for reading in fieldmaps generated by TOSCA
- GEMC supports text and evio (CODA raw data) output formats where evio can be converted to root format given a custom written conversion utility–this is one of the drawbacks of GEMC
- GEMC supports external event generation using a Lund-type input file format. Standalone generator output files need to be converted from root to Lund; tools exist
- GEMC has a nice GUI interface and visualization display using Qt/OpenGL

GEMC GUI



Particle Type: e-

Beam Values

p:	11000 ± 0 MeV
theta:	0 ± 0 deg
phi:	0 ± 0 deg

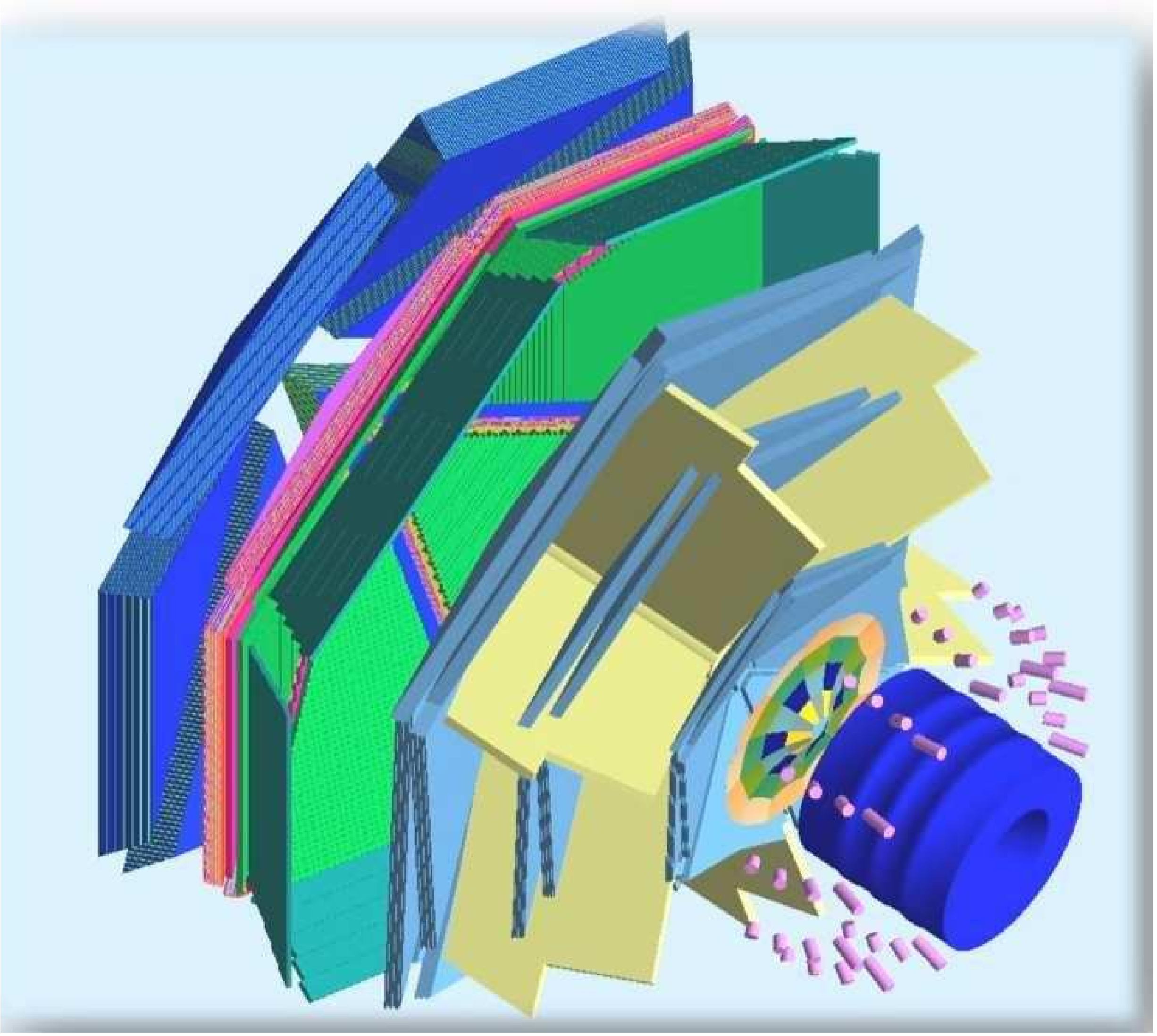
Vertex Values

(x,y,z):	(0, 0, 0) mm
radius:	0 mm
delta z:	0 mm

Number of Events

Set N: 1 X 1 Number of Events: 1

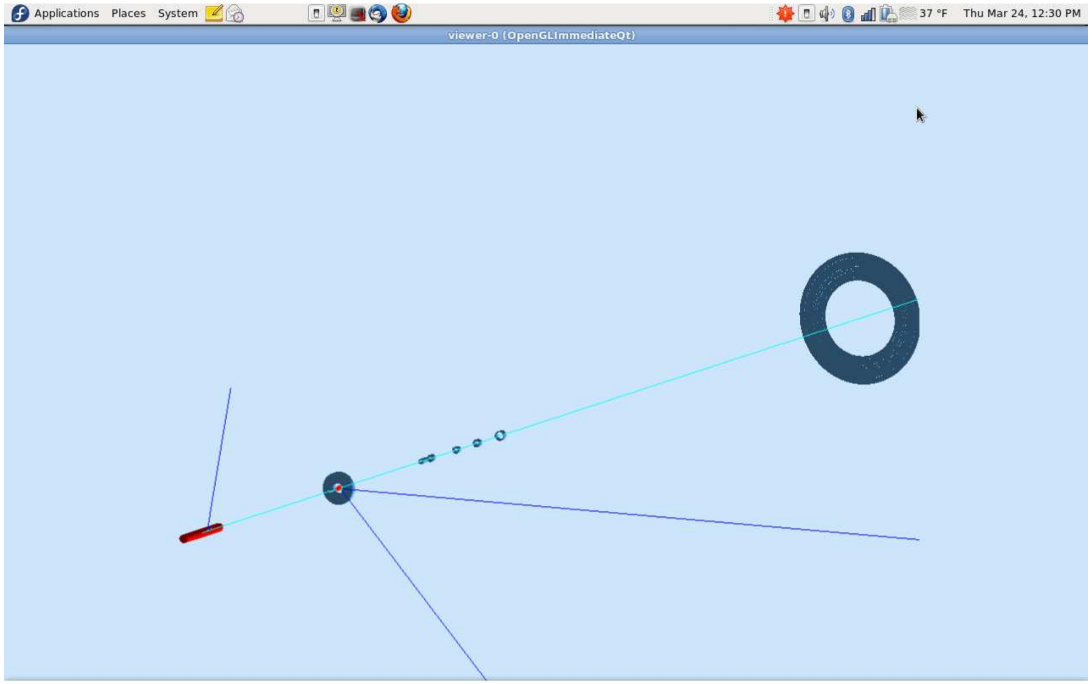
GEMC Geometry Display



Working Towards a MOLLER GEMC: molgemc

- For this development, our strategy is to follow as much as possible the development of the SoLID GEMC (solgemc)–which is quite advanced at this point
- First, install GEMC locally...not trivial, but finally achieved using Fedora 14 OS; also setup local mysql server for GEMC DB
- Define a baseline geometry using perl scripts – target, collimators, detector rings
- Incorporate fieldmaps: Not yet done. I have encountered several issues here and need assistance from Juliette/Mauri
- Incorporate event generators: In progress
- Just as SoLID does, we will use GEMC as a precompiled shared object library to build against–requires minimal modification to GEMC and enables us to add our own Classes as needed...

Working Towards a MOLLER GEMC: molgemc



- We will also setup a dedicated mysql server through the jlab computer center (with offsite access capability)
- Once a basic working version is established, we will add molgemc to our svn repository–getting close

Current issues with fieldmap import

- We have some file format issues: For phi-segmented cylindrically symmetric map, gemc wants datafile columns in this order – phi, transverse-coord, longitudinal-coord, Bx, By, Bz
- Also, GEMC expects the longitudinal coordinate to vary first, then the transverse, then azimuthal...our maps are opposite this
- GEMC assumes that map covers only half the segment (Currently we specify the entire segment)
- GEMC takes the one segment defined in the field map and generates the other 6 segments based on symmetry calculations...I’m not sure if an odd number of segments poses a problem here
- The dimension unit “m” is not implemented yet
- GEMC does not currently support the need for two fieldmaps...the potential fix for this is to combine the two maps into one file...