Visualization in MaGe/Geant4

Oleksandr Volynets

Max-Planck-Institute for Physics

MaGe Workshop 2010, Munich

January 18, 2010

Visualization in MaGe/Geant4, or Ways to see what you do in MaGe/Geant4

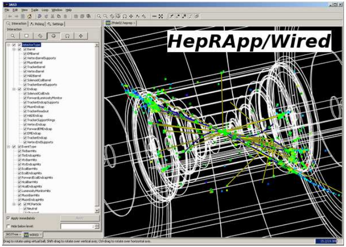
Oleksandr Volynets

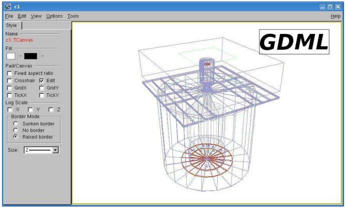
Max-Planck-Institute for Physics

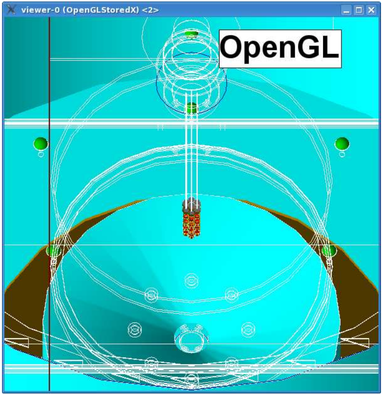
MaGe Workshop 2010, Munich

January 18, 2010

Examples


HepRApp/Wired


c1
File Edit View Options Tools Help
GDML


viewer-0 (OpenGLStoredX) <2>
OpenGL

Outline

- Motivation

Outline

- Motivation
- General visualization commands

Outline

- Motivation
- General visualization commands
- Examples of command sequences

Outline

- Motivation
- General visualization commands
- Examples of command sequences
- More detailed information on:
  - GDML
  - HepRep/HepRApp/Wired
  - OpenGL
  - Extra features of OpenGL

Motivation

This presentation gives a basic information on how to visualize a detector and/or an experiment using the standard Geant features.

Motivation

This presentation gives a basic information on how to visualize a detector and/or an experiment using the standard Geant features.

Why do we need this?

Motivation

This presentation gives a basic information on how to visualize a detector and/or an experiment using the standard Geant features.

Why do we need this?

- To see (before event simulation) the detector, it's geometry and possible overlaps;

Motivation

This presentation gives a basic information on how to visualize a detector and/or an experiment using the standard Geant features.

Why do we need this?

- To see (before event simulation) the detector, it's geometry and possible overlaps;
- To see (after the simulation is done) tracks, hits, energy deposits etc.

Motivation

This presentation gives a basic information on how to visualize a detector and/or an experiment using the standard Geant features.

Why do we need this?

- To see (before event simulation) the detector, it's geometry and possible overlaps;
- To see (after the simulation is done) tracks, hits, energy deposits etc.
- Even for more interesting things (not really useful but nice). Later on this...

Ways to visualize in Geant4

There are 8 ways to visualize (according to the Geant4 manual):

Ways to visualize in Geant4

There are 8 ways to visualize (according to the Geant4 manual):

1. OpenGL
2. Qt
3. OpenInventor
4. HepRep
5. DAWN
6. VRML
7. RayTracer
8. ASCIITree

Ways to visualize in Geant4

There are 8 ways to visualize (according to the Geant4 manual):

1. OpenGL
2. Qt
3. OpenInventor
4. HepRep
5. DAWN
6. VRML
7. RayTracer
8. ASCIITree

and one as a separate tool (included into Geant4.9.2 but implemented in MaGe)

1. GDML + ROOT

Ways to visualize in Geant4

There are 8 ways to visualize (according to the Geant4 manual):

1. OpenGL *
2. Qt
3. OpenInventor
4. HepRep *
5. DAWN
6. VRML
7. RayTracer
8. ASCIITree *

and one as a separate tool (included into Geant4.9.2 but implemented in MaGe)

1. GDML + ROOT *

* Covered in this presentation

General visualization commands

General visualization commands

Visualization is done in *macro/command line* of MaGe after the detector was defined and the */run/initialize* command executed so Geant4 *command line* activated. Further steps are:

General visualization commands

Visualization is done in *macro/command line* of MaGe after the detector was defined and the */run/initialize* command executed so Geant4 *command line* activated. Further steps are:

- Open a visualization driver, such as:
  - /vis/open HepRepFile [OGLIX, RayTracer]

General visualization commands

Visualization is done in *macro/command line* of MaGe after the detector was defined and the */run/initialize* command executed so Geant4 *command line* activated. Further steps are:

- Open a visualization driver, such as:
    - /vis/open HepRepFile [OGLIX, RayTracer]
- Add the detector geometry
    - /vis/drawVolume

General visualization commands

Visualization is done in *macro/command line* of MaGe after the detector was defined and the */run/initialize* command executed so Geant4 *command line* activated. Further steps are:

- Open a visualization driver, such as:
  - /vis/open HepRepFile [OGLIX, RayTracer]
- Add the detector geometry
  - /vis/drawVolume
- ⋆ If using an immediate viewer, such as OpenGL, set camera parameters and drawing style (wireframe/surface):
  - /vis/viewer/set/style wireframe
  - /vis/viewer/set/viewpointThetaPhi 70 20

General visualization commands

Visualization is done in *macro/command line* of MaGe after the detector was defined and the */run/initialize* command executed so Geant4 *command line* activated. Further steps are:

- Open a visualization driver, such as:
  - /vis/open HepRepFile [OGLIX, RayTracer]
- Add the detector geometry
  - /vis/drawVolume
- ⋆ If using an immediate viewer, such as OpenGL, set camera parameters and drawing style (wireframe/surface):
  - /vis/viewer/set/style wireframe
  - /vis/viewer/set/viewpointThetaPhi 70 20
- Declare what data should be added to the scene (default is to just add full set of detector volumes)
  - /vis/scene/add/trajectories
  - /vis/scene/add/hits

General visualization commands

- Run simulation with appropriate options to store trajectory information:
    - /run/beamOn 1

General visualization commands

- Run simulation with appropriate options to store trajectory information:
    - /run/beamOn 1
- Execute the visualization (done automatically with each /run/beamOn, but needed by some drivers if you want to output geometry without running an event):
    - /vis/viewer/flush

General visualization commands

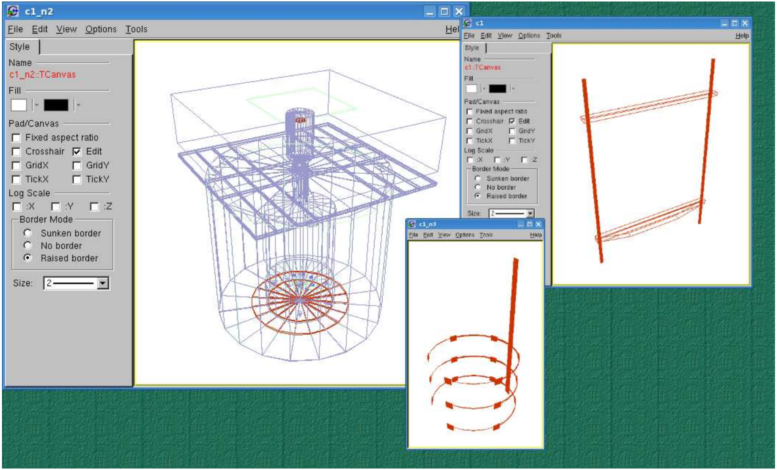
- Run simulation with appropriate options to store trajectory information:
  - /run/beamOn 1
- Execute the visualization (done automatically with each /run/beamOn, but needed by some drivers if you want to output geometry without running an event):
  - /vis/viewer/flush

⋆ If using an external viewer, such as for HepRepFile:
  - import the .heprep file into HepRApp/Wired, set camera parameters, drawing style, etc., view the visualization

Example Visualization Command Sequences

- Visualize a detector in OpenGL (Linux or Mac)
  - /vis/open OGLIX
  - /vis/drawVolume
- Visualize trajectories and hits for 10 events using HepRep/HepRApp
  - /vis/open HepRepFile
  - /vis/drawVolume
  - /vis/scene/add/trajectories
  - /vis/scene/add/hits
  - /run/beamOn 10

GDML (Geometry Description Markup Language)



GDML3.0 + ASCIITree + ROOT example sequence

- Open ASCIITree driver and create geometry tree
  - /vis/open ATree
  - /vis/ASCIITree/verbose 1
  - /vis/drawTree
- Set up GDML format: filename, output format and levels to modularize (see GDML manual):
  - /MG/geometry/GDML/outputName GerdaArray
  - /MG/geometry/GDML/outputFormat false
  - /MG/geometry/GDML/modularizeLevels 0 1 2
- Dump geometry to files
  - /MG/geometry/GDML/write
- Import geometry into ROOT
  - `gSystem->Load("libGeom");`
  - `gSystem->Load("libGdml");`
  - `TGeoManager::Import("GerdaArray.gdml");`
  - `gGeoManager->GetTopVolume()->Draw();`

GDML (Geometry Description Markup Language)

Advantages:

- Possibility to view each part of the detector separately e.g. in ROOT (using ROOT macro)
- You can write AND read the geometry to/from file (based on the XML language)

GDML (Geometry Description Markup Language)

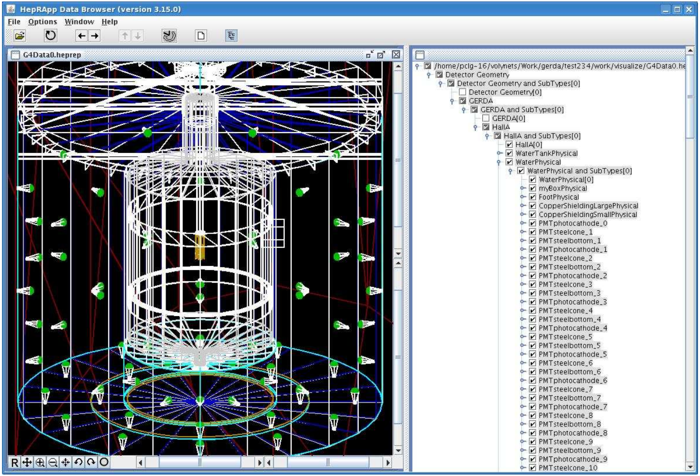
Advantages:

- Possibility to view each part of the detector separately e.g. in ROOT (using ROOT macro)
- You can write AND read the geometry to/from file (based on the XML language)

Disadvantages:

- GDML is included in Geant4.9.2 and later releases so there appeared number of incompatibilities in MaGe (way of reading/writing files was changed) - to be discussed during the Computing Session
- Creating geometry for older releases of GDML (version $<= 2.10$) does not mean that it will be read by Geant4-GDML

HepRApp/Wired (HepRep)



HepRApp/Wired (HepRep)

It is fast and simple way to see the detector.

HepRApp/Wired (HepRep)

It is fast and simple way to see the detector.

Advantages:

- Flexible view of the detector, easy control: zoom/rotate using mouse;
- Many visual options;
- Easy manual creation of .eps files

HepRApp/Wired (HepRep)

It is fast and simple way to see the detector.

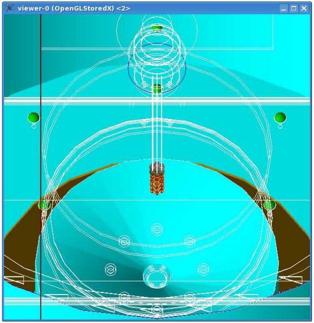
Advantages:

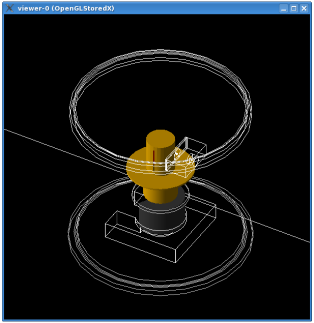
- Flexible view of the detector, easy control: zoom/rotate using mouse;
- Many visual options;
- Easy manual creation of .eps files

Disadvantages:

- Rather slow when rotating a complicated geometry;
- Not a better way of making high-quality pictures (i.e. for Posters or Technical Design Report)

OpenGL





OpenGL

Advantages:

- Can be viewed directly in Geant4;
- Uses a video graphic card so it is much faster way than the others.

OpenGL

Advantages:

- Can be viewed directly in Geant4;
- Uses a video graphic card so it is much faster way than the others.

Disadvantages(read as: "features"):

- You need a good graphic card and correctly installed driver for this!
- All movements of the detector/camera should be assigned in *macro/command line*, no mouse control available

OpenGL

Advantages:

- Can be viewed directly in Geant4;
- Uses a video graphic card so it is much faster way than the others.
- Gives extra possibilities. See next slide...

Disadvantages(read as: "features"):

- You need a good graphic card and correctly installed driver for this!
- All movements of the detector/camera should be assigned in *macro/command line*, no mouse control available

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

- Rotating detector;

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

- Rotating detector;
- Zooming detector;

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

- Rotating detector;
- Zooming detector;
- Particle tracks;

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

- Rotating detector;
- Zooming detector;
- Particle tracks;
- Particle showers

Creating a movie with OpenGL

Not really useful, but very fascinating. What can we see in motion:

- Rotating detector;
- Zooming detector;
- Particle tracks;
- Particle showers

See the movie about GERDA...

How to create a movie using OpenGL (OGLX)

Here is my way described. As reported in geant4 presentations (see references) there is easier way to do this.

How to create a movie using OpenGL (OGLX)

Here is my way described. As reported in geant4 presentations (see references) there is easier way to do this.

- Put some loop to MaGe/Geant4 macro at the very end: *RunGerda.mac:*
  - /control/loop movie.loop phi 0 100 5
    loop over $\phi = 0..100$ *with step* 5 *degrees*

  movie.loop:
  - /vis/viewer/set/viewpointThetaPhi 30 {phi}
  - /vis/ogl/printEPS

How to create a movie using OpenGL (OGLX)

Here is my way described. As reported in geant4 presentations (see references) there is easier way to do this.

- Put some loop to MaGe/Geant4 macro at the very end: *RunGerda.mac:*
  - /control/loop movie.loop phi 0 100 5
    loop over $\phi = 0..100$ *with step* 5 *degrees*

  movie.loop:
  - /vis/viewer/set/viewpointThetaPhi 30 {phi}
  - /vis/ogl/printEPS
- Convert all *.eps* to *.yuv* files, create conversion macro for *mpeg2encode* and encode pictures to video file. See the reference for details.

How to create a movie using OpenGL (OGLX)

Have you seen “Time Warp” show on “Discovery Channel”?

How to create a movie using OpenGL (OGLX)

Have you seen “Time Warp” show on “Discovery Channel”?

See slowed down motion of the particles (for free!)...

Conclusions

Conclusions

- There are plenty of ways to visualize the detector, use the way you like

Conclusions

- There are plenty of ways to visualize the detector, use the way you like
- Each method has Advantages and Disadvantages

Conclusions

- There are plenty of ways to visualize the detector, use the way you like
- Each method has Advantages and Disadvantages
- There are other also good ways not covered in this presentation (DAWN, RayTracerX etc.). Try them

Conclusions

- There are plenty of ways to visualize the detector, use the way you like
- Each method has Advantages and Disadvantages
- There are other also good ways not covered in this presentation (DAWN, RayTracerX etc.). Try them
- If you are a geek* you have much space to fulfil your needs. Just use your imagination

* http://en.wikipedia.org/wiki/Geek

Acknowledgements

Thanks to Joseph Perl from SLAC
(I took his presentations' materials while creating my talk)

References

HepRApp Tutorial
- http://geant4.slac.stanford.edu/Presentations/vis/G4HepRAppTutorial/G4HepRAppTutorial.html

OpenGL Tutorial
- http://geant4.slac.stanford.edu/Presentations/vis/G4OpenGLTutorial/G4OpenGLTutorial.html

GDML manual
- http://gdml.web.cern.ch/GDML/doc/GDMLmanual.pdf

Geant4 Visualization Commands
- http://geant4.slac.stanford.edu/Presentations/vis/G4VisCommands.ppt (and .pdf)

Geant4 Advanced Visualization
- http://geant4.slac.stanford.edu/Presentations/vis/G4VisAdvanced.ppt (and .pdf)

How to Make a Movie
- http://geant4.slac.stanford.edu/Presentations/vis/HowToMakeAMovie.ppt (and .pdf)

Visualization Chapter of the Geant4 User's Guide for Application Developers
- http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/

Movies and other useful manuals and tutorials
- http://geant4.slac.stanford.edu/Presentations/vis/