Short Introduction to Geant4

- principle: step by step and particle by particle
- geometries: rebuilding a setup in the computer
- particles: from Geantinos to heavy ions
- processes: and the winner is ... random
- visualization: blue ions and green $\gamma$-rays
Step by Step – For Each Particle

- produce particle with a particle gun
  - fix particle type, $E_{\text{kin}}$, momentum direction, ...

- go through the world in steps
  - step size determined by cross sections
  - steps end at volume borders
  - apply processes
    - may create, change, destroy particles

- continue until no particles are left
  - or all outside the world volume
Geometries

- Geant4 handles geometrical, logical, and “physical” volumes separately
  - geometrical volumes specify shapes
  - logical volumes specify materials
  - physical volumes specify placements in the setup
- organization in a tree
  - world volume → mother volume → ... → volume
“solids” in Geant4 parlance

- construction from simple shapes
  - many simple shapes available
    - see Geant4 Documentation
  - shapes may be combined: intersection, union, ...
- construction from CAD data is also possible
  (I never tried this)
Logical Volumes

- association of shape and material
- grouping of child volumes
  - must be completely inside the mother volume
- allows for repeated placement
  - e.g. 8 Si detectors, but only one logical volume
- assignment of visualization attributes
  - e.g. color, transparency
- “sensitive detector” for data readout (later)
Physical Volumes

- placement of logical volumes
- each “replica” has its number for identification in data read-out
G4VSolid* s = new 
G4Tubs("s_nai_crystal", 0, r, 
1/2, 0, 360*deg);

G4LogicalVolume* l = new 
G4LogicalVolume(s,GetMat("NaI"), 
"l_nai_crystal");

new G4PVPlacement(TranslateZ3D(l/2), 
l, "nai_crystal", mother, 
false, 0 );
Sensitive Detectors

- need to get information from interaction of particles with detector material
- done using a “sensitive detector”
- for each particle passing, some information may be collected and stored
  - e.g. energy loss of a single particle
  - e.g. A, Z of an ion (i.e. information not accessible in an experiment)
- data storage not directly included in Geant4
  (I use ROOT)
many predefined particles:

- ions by A, Z with mass, ...
- photons
- ...
- “Geantino” for testing purposes
Physics = Processes

- each particle has a list of processes that apply
- list has to be made for each particle in the simulation program
  - best to look at examples provided with Geant4
- different types of processes
  - AlongStep, PostStep, AtRest
- many processes already implemented
AlongStep Processes

- apply for all steps of the particle
- continuous processes
  - also those where a microscopic model is not feasible or too slow
- example
  - energy loss of ions in matter
AtRest Processes

- apply if the particle is at rest
- selection by remaining lifetime

example

- radioactive decay

( seems to be implemented only for particles at rest in present Geant4 versions )
PostStep Processes

- apply for moving particles
- selection by interaction length
- examples:
  - elastic scattering
    - already implemented in Geant4
  - knock-out, in-flight $\gamma$-ray emission, ...
    - processes may be added by the user
Process Selection (PostStep)

- choose n.o. interaction lengths (IL)
  - randomly, $e^{-x}$ distribution
- current IL depends on material of volume
  - here is the cross section
  - angular distributions etc. defined by the process
- process with shortest remaining IL wins
  - this length is the step length
  - new step, subtract from n.o. ILs left
- reaching the volume border is also a “process”
Visualization

- several types of viewers
  - DAWN – PostScript, non-interactive
  - VRML, HepRep – 3D, interactive
- most viewers not included in Geant4
- display of geometries, but also of events
Some (Partial) Examples

- program structure
- script for viewing the geometry
- script for running a simulation
need to define user classes:

- MyConstruction – create the detector geometry
- MyPhysicsList – assign processes to particles
- MyPrimaryGeneratorAction – create particles
- MyEventAction – read out data for each event
Visualization script

- switch off debug messages
  
  /run/verbose 0
  /event/verbose 0
  /tracking/verbose 0

- initialize run manager
  
  /run/initialize

- test for overlapping volumes
  
  /geometry/test/grid_test
  \textarrow{true}

- draw to VRML2
  
  /vis/scene/create
  /vis/scene/add/volume
  \textarrow{world}
  /vis/scene/add/axes
  /vis/open VRML2FILE
  /vis/viewer/flush

- viewer: e.g. freewrl
Visualization Example

screenshot of
- CACTUS
- SiRi
- target
- support structures

with freewrl viewer
Simulation script

- set detector parameters
  - /GamSim/cactus/enable true
  - /GamSim/beamtube/enable true
  - /GamSim/beamtube/diameter 12.7 cm
  - /GamSim/target/enable true
  - /GamSim/target/material Cobalt
  - /GamSim/target/size 14 mm
  - /GamSim/target/thickness 0.5 mm

- initialize run manager
  - /run/initialize

- select output file
  - /E450/rootTree/filename cactus_60Co.root

- select primaries
  - 60Co, 2 γ rays

- run (i.e. wait 😊)
  - /E450/gun/ion 0 0
  - /E450/gun/eKin 0 MeV
  - /E450/gun/product 0 0
  - /E450/gun/beamDiameter 1 cm
  - /E450/gun/beamUniform true
  - /E450/gun/zRange 0.5 mm
  - /E450/gun/addEnergy 1173 1
  - /E450/gun/addEnergy 1332 1
  - /run/beamOn 20000000
Simulation Example 1

- $^{60}$Co source
- 200,000 $\gamma$ rays
  - 50% 1173 keV
  - 50% 1332 keV
- gaussian resolution
  - preliminary
  - artificial (while storing)
Simulation Example 2

- $^4$He beam, 30 MeV
- 100 ions
- drawing trajectories
  - ions
  - $\gamma$ rays
  - electrons
Other Topics

- biasing
  - often it is necessary to enhance the desired process
    - e.g. reaction in the target, most particles would just go through
- examples
  - Geant4 includes a lot of examples, also for “physics lists” (i.e. process assignments to particles)
- problems
  - Geant4 does not like some geometries (intersections) – trial and error