Tutorial Live Notes

0.1 Overview of Simulation

This is a GEANT4 based simulation of the calorimeter test-stand to be built at Stockholm University. This is a single slice of the calorimeter that we will use in a test beam to validate the design and TDAQ system. The first goals of this simulation (the next year) are

- **Verify range measurement followed by light collection in lead-glass will work**
    - Range + Absorption concept presented here on **Slide 3**
- **Verify segmentation needed in lead-glass to have**
- **Characterize gamma backgrounds effect on pileup,**
- **Test different thicknesses of plastic scintillators effect on energy reconstruction**

The next (parallel?) steps

- **Add subdetector systems**
    - **silicon tracking in vacuum tube**
    - **TPC**
- **Add readout/services**
    - **Wavelength shifting fibers**
    - **Electronics**
    - **Support structure/mechanical material**

The much further steps (1 year+ away)

- **Create full detector system**
    - **Submodules of different subdetectors**
        - **Silicon tracking**
        - **TPC**
        - **Calorimeter**
        - **Support structures**
        - **Services**
    - Full materials list from Anders started here

0.2 Required Software

- **Install Geant4:** http://geant4-userdoc.web.cern.ch/geant4-userdoc/UsersGuides/InstallationGuide/html/installguide.html
- **Install Root:** https://root.cern.ch/root/html534/guides/users-guide/InstallandBuild.html
- **Clone nnbar-sim Repository:** https://github.com/kdunne/nnbar-calo-sim
    - Follow README for installation/build instructions

1. Geant4 Documentation

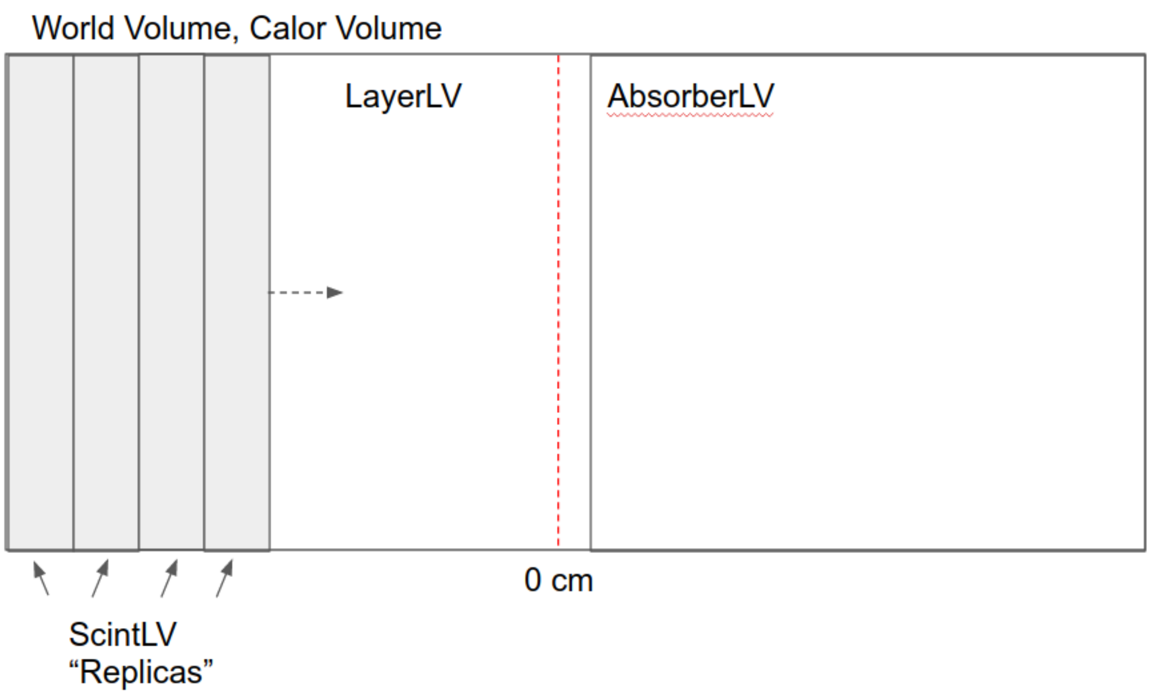
There are many tutorials online. This one will take common themes from them and high light the parts relevant to our calorimeter simulation specifically. Here are helpful documentation/tutorials I have used often.

- **Doxygen: http://www.apc.univ-paris7.fr/~franco/g4doxy/html/index.html**
    - Searchable reference for class/object descriptions
- **Helpful Tutorials:**
    - https://www.ge.infn.it/geant4/training/portland/basicStructure.pdf
    - https://indico.cern.ch/event/294651/sessions/55918/attachments/552022/760640/UserActions.pdf
- **GEANT4 built-in examples:**

1. Geometry

Defined in DetectorConstruction.cc

- Detector Materials
  - Optical Photon Processes
    - https://arxiv.org/pdf/1612.05162.pdf
- Volumes



- Replica volumes use rotation coordinate (sensitive integer?) and translation coordinates (vector)
  - Rotations? Need to know more down the line

2. Sensitive Detectors

- Declare geometric elements "sensitive" to passage of particles
  - Record physical quantities -> energy, time, PID, et.
- How to:
  - Write your own SD class
    - ScintillatorSD.cc

        - AbsorberSD.cc
        - TubeSD.cc
    - Attach it to a Logical Volume
        - DetectorConstruction.cc

```
//....

void DetectorConstruction::ConstructSDandField()
{
  G4SDManager::GetSDMpointer()->SetVerboseLevel(1);


  // declare vacuum as TubeSD
  G4String tubeDetectorName = "TubeLV" ;
  TubeSD* tubeDetector = new TubeSD(tubeDetectorName);
  G4SDManager::GetSDMpointer()->AddNewDetector(tubeDetector);
  SetSensitiveDetector("TubeLV", tubeDetector);

  // declare Scintillator as SinctillatorSD
  G4String scintDetectorName = "ScintLV" ;
  ScintillatorSD* scintDetector = new ScintillatorSD(scintDetectorName);
  G4SDManager::GetSDMpointer()->AddNewDetector(scintDetector);
  SetSensitiveDetector("ScintLV", scintDetector);

  // declare absorber as AbsorberSD
  G4String absorberDetectorName = "AbsoLV" ;
  AbsorberSD* absorberDetector = new AbsorberSD(absorberDetectorName);
  G4SDManager::GetSDMpointer()->AddNewDetector(absorberDetector);
  SetSensitiveDetector("AbsoLV", absorberDetector);


}
```

3. Hits

- **A hit can be made for each step (interaction), or it can accumulate quantities**
- **Hits are stored in HitCollections**
- **Write your own class that inherits from G4VHit to define data structure for gathering hit information**
    - NNbarHit.cc
- **Call ProcessHits() in SD class definition to manipulate hit data**

```
//.....
void ScintillatorSD::Initialize(G4HCofThisEvent*)
{

    HitsCollection = new NNbarHitsCollection(sensitiveDetectorName,
                                                             collectionName[0]);
}

//.....
G4bool ScintillatorSD::ProcessHits(G4Step* aStep, G4TouchableHistory* )
{

    //if (aStep -> GetPreStepPoint() -> GetPhysicalVolume() -> GetName() != "Scint") return false;
    if (aStep -> GetPreStepPoint() -> GetPhysicalVolume() -> GetName() != "Layer") return false;


    // Get Direction
    G4Track * theTrack = aStep  ->  GetTrack();
    G4ThreeVector stepDelta = aStep->GetDeltaPosition();
    G4double direction = stepDelta.getZ();

    //Get particle name
    G4ParticleDefinition *particleDef = theTrack -> GetDefinition();
    G4String particleName =  particleDef -> GetParticleName();

    // Get particle PDG code
    G4int pdg = particleDef ->GetPDGEncoding();
```

4. Histograms

- **Histograms are created in RunAction.cc**
  - i.e. happens once at beginning of each run
- **Fill Histograms from HitCollections of each SD**
  - Histograms are filled in EventAction.cc
    - i.e. happens once at end of each event
- **Creating/Filling handled by analysisManager instance**

Histogram List (any not listed here and found in code are under development)

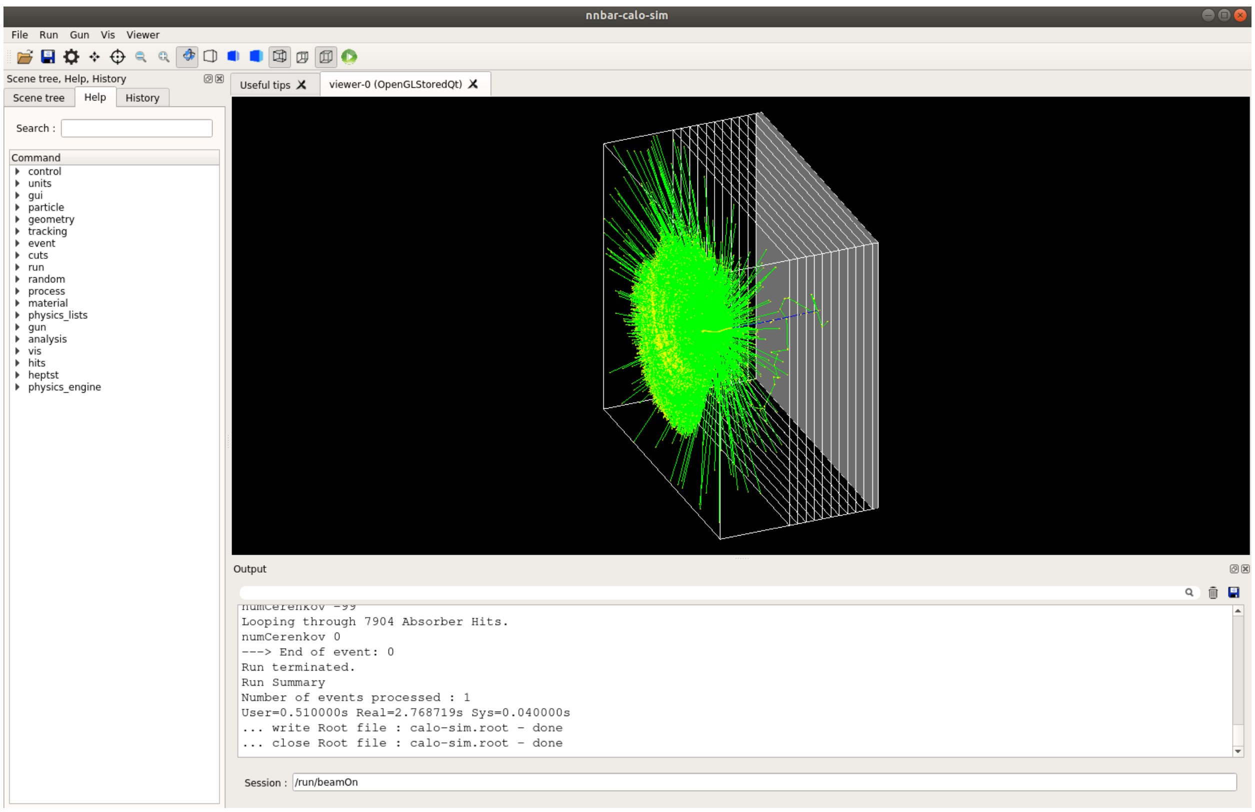
Histogram Name	**xaxis**	**yaxis**
NumCerenkov	Number Cerenkov photons created in lead-glass	Number Events
PhotonTime	Time each unique photon created since beginning of run	Number Events
DecayTime	Time the primary particle's kinetic energy == 0	Number Events
Range	Position where primary particle stops	Number Events
EdepScint	Energy deposited in all scintillators	Number Events
EdepAbs	Energy deposited in lead glass	Number Events
EdepTube	Energy deposited in Vacuum tube	Number Events
eDepvRange	Total energy deposited	Position where primary particle stops
eDepvCerenkov	Energy deposited in lead-glass	Number of Cerenkov photons created in lead-glass
RangevCerenkov	Number of Cerenkov photons created in lead-glass	Total range

- Need to build TTree support for ntuple support

- Don’t want to rerun simulation/analysis constantly!

5. Running the Simulation

- **Interactive Mode--look at in instructions in git repo**
  - ./nnbar-calo-sim
  - Opens GUI, useful for sanity check on 1 event



- **Macro Mode**
  - ./nnbar-calo-sim -m macroname
  - nnbar-calo-sim.in is provided in repository

```
# Macro file for test

/run/initialize
/gun/particle mu+

# mu+ 50 MeV
#/gun/momentum 50 MeV
#
# mu+ 75 MeV
#/gun/momentum 0 0 75 MeV
#
# mu+ 100 MeV
#/gun/momentum 0 0 100 MeV
#
# mu+ 200 MeV
#/gun/momentum 0 0 200 MeV
#
## mu+ 400 MeV
/gun/momentum 0 0 400 MeV

/run/beamOn 10
```

6. Future Work

- **Histogram management as separate code** -> e.g. HistogramManager.cc to handle creating and filling histograms
- **Plotting tool as integrated function** currently using my own plotting script on the output of the simulation. Working on a general NNbar groot style we can apply to all plots, and a function to automatically create useful plots when the simulation is run
- **Segmentation of lead-glass** segment lead-glass into a grid using replica method scintillators have now