Table of Contents

CERN

We are using the repository for HGCal Geant4 standalone simulations located at https://github.com/pfs/PFCal/tree/master/PFCalEE/analysis, this can be obtained by doing:

```
$ git init

$ git remote add origin https://github.com/pfs/PFCal/tree/master/PFCalEE

$ git clone -b BASELINE-TP https://github.com/pfs/PFCal

$ cd PFCal/PFCalEE/
```

To set up the environment and compile (SL6 machines), go to the PFCalEE directory and run:

```
$ cd PFCal/PFCalEE

$ source g4env.sh

$ cd userlib

$ mkdir lib

$ mkdir obj

$ mkdir bin

$ make -j 5

$ cd ../

$ make -j 5
```

FNAL

In the cmslpc (SL6) is possible that the setting up of the environment doesn't work, it is required to connect to a SL6 note

```
$ ssh cmslpc41.fnal.gov
```

After this, copy the repository to your working area

```
$ cd workdir
$ git init
$ git remote add origin https://github.com/pfs/PFCal/tree/master/PFCalEE
$ git clone https://github.com/pfs/PFCal
$ cd PFCal/PFCalEE/
```

Updated to the desired git tag version

```
$ git tag
$ git fetch --all
$ git checkout (-f) Version
```

Setup the environment (bash)

```
$ cp /uscms/home/yumiceva/work/HGCal/PFCal/PFCalEE/g4env_fnal.sh .
bash
$ source g4env_fnal.sh
```

Setup the environment (csh)

```
$ cp /uscms/home/yumiceva/work/HGCal/PFCal/PFCalEE/g4env_fnal.csh .
$ rehash
$ source g4env_fnal.csh
```

Compile the files

```
$ mkdir -p userlib/{lib,obj,bin} && cd userlib && make dictionary && make
-j 5 && cd - && make -j 5
```

Generation of events

The code submitProd.py is created to submit jobs in parallel to the batch system (using LSF), to run do:

```
$ python submitProd.py -options
```

In case of options conflicts it can be used also:

```
$ ./submitProd.py -options
```

The options for the code are:

- -n : Number of events to generate (default 1000)
- -g : Use particle gun
- -d : Data or type of particle to shoot (default e-)
- -s : Short batch queue (default 1nd)
- -l : Long batch queue (default 2nw)
- -t : Git tag use it
- -r : Run stats
- -m : Detector model
- -v : Detector version
- -e : eos Path to save root file
- -o : Output directory
- -a : Incidence angle in radians (default 0)
- -b : Magnetic field in Tesla (default 0)
- -f : Path to the HepMC input file
- -S : Not submit batch job

Particle Gun

To run this example modify the submitProd.py file to set energy loops and run:

```
$ for i in seq 0 5; do python submitProd.py -s 1nd -q 2nd -t V00-00-00 -g -r ${i} -v 3 -m 0 -e /s
```

This example is running 2500 events of a particle gun shooting electrons to the detector model.

The energy of the initial particles can be changed in the script SubmitProd.py

At then moment of running the program other particles can be chosen for the particle gun, like muon (mu-) or pions (pi-)

For a complete code of the particles in Geant4 see:

http://geant4.cern.ch/G4UsersDocuments/UsersGuides/ForApplicationDeveloper/html/

Analysis

outputs

A sample of the results are presented in the root files attached

The file DigiPFcal contains the histograms for the noise check and the reconstructed hits in the detector

The file HGcal contains 4 branches:

The file HGcal contains 4 branches:

Sampling Section: Contains the information of the volume; the total, absorbed y measured energy; the fraction of particles, etc.

SimHit : Contains histograms such as the number of particles, energy, time, layer, etc

GenParticle : Contains the information of the generated particle

Analysis code SimHits

An analysis script that plot histograms of the variables contained in the root files previously obtained is obtained using the MakeClass command in root.

The codes are named HGCal (see attached files), and are located in /afs/cern.ch/user/h/hhernand/work/codes
To run the codes successfully in root is required to load one library for the main directory of G4, for this run:

```
$ ln -s ~/work/PFCal/PFCalEE/userlib/include .
```

At the moment of start root, some libraries need to be loaded(everytime start root):

```
$ gSystem->Load("~/work/PFCal/PFCalEE/userlib/lib/libPFCalEEuserlib.so");
```

Then we need load some files:

```
$ .L HGcal.C;

$ HGcal t;

$ t.Loop();
```

The output of this script will be a root file that contains all the desired histograms, plus some .txt files with information about the events

As a sample of the results the Z-X profile of the detector under the action of the different particle guns can be found in the attached files

Analysis code

For the Digitization the code Digitizer.cpp located in the directory .../PFCal/PFCalEE/userlib/test/

To execute this code run the command "make", the resulting executable will be located in the directory PFCal/PFCalEE/userlib/bin/

To run the executable:

$ ./HGCal-SimHit (number of events) (complete path to input file) (complete path to output file) (granularity layeri-layerj:value, laye...) (noise layeri-layerj:value...) (threshold layeri-layerj:value...) (random seed value, default = 0) (debug, default = 0) (save sim hits, default = 0) (save digi hits, default = 0)

For the ECAL CALICE-like HG:

number of events : choosing 0 or negative numbers will lead to take the full number of events

granularity : 0-19:4,20-29:6

noise: 0-29:0.12

hreshold : 0-29:2

The result will be a root file located in the output directory, this file will contain the Reconstructed information.

As a sample of the results the Z-X profile of the detector under the action of the different particle guns can be found in the attached files.

Geometry Visualization

To visualize the geometry that is used for the standalone simulation the files DetectorConstruction.cc (~/.../PFCalEE/src) and DetectorCosntruction.hh (~/.../PFCalEE/include) contains the information of all the available geometries, e.g:

```
enum DetectorVersion {
    v_CALICE=0,
    v_HGCALEE_Si80=1,
    v_HGCALEE_Si120=2,
    v_HGCALEE_Si200=3,
    v_HGCALEE_Si500=4,
    v_HGCALEE_gap1=5,
    v_HGCALEE_CALICE=6,
    v_HGCALEE_inverted=7,
    v_HGCALEE_concept=8,
    v_HGCALEE_W=9,
    v_HGCALEE_gap4=10,
    v_HGCALEE_prePCB=11,
    v_HGCAL=20,
    v_HGCALHE=21,
    v_HGCALHEScint=22,
    v_HGCALHE_CALICE=23
```

```
  };

  enum DetectorModel {
    m_SIMPLE_20=0,
    m_SIMPLE_50=1,
    m_FULLSECTION=2,
    m_SIMPLE_100=3
  };
```

In the PFCalEE directory the file PFCalEE.cc can be modified to change to use the desired detector and model, the line to change is:

```
int version=0; //DetectorConstruction::v_HGCAL;
  //int model=DetectorConstruction::m_FULLSECTION;
  int model=DetectorConstruction::m_SIMPLE_20;
```

After the file is modified to compile run:

```
$ make -j 5
```

After this, run the following command (Changing the path to one in your directory):

```
$ export G4DAWNFILE_DEST_DIR=/afs/cern.ch/user/h/hhernand/work/PFCal/PFCalEE/
```

Copy the file vis.mac to the directory where the PFCalEE executable is (geant4_workdir/bin/Linux-g++).

To run the visualization is necessary to go to the directory where the PFCalEE executable is and run:

```
$ PFCalEE vis.mac
```

A sample and a complete geometry construction for the CALICE detector can be found in the attached files.

A sample and a complete geometry construction for the HGCalHe-CALICE (like) can be found in the attached files.

HGcal Particle Flow Reconstruction

use fireworks to display under CMSSW

Do it in slc6:

Set the environment:

For tcsh:

```
$ source /uscmst1/prod/sw/cms/cshrc prod
```

For bash:

```
$ source /uscmst1/prod/sw/cms/cshrc prod

$ cmsrel CMSSW_6_2_0_SLHC16

$ cd CMSSW_6_2_0_SLHC16/src && cmsenv

$ git cms-merge-topic lgray:HGC-Linking

$ git cms-merge-topic vandreev11:hgcal--recogeometry-fix--jumps

$ git cms-merge-topic 5063

$ git cms-addpkg RecoParticleFlow/PFProducer

$ git cms-addpkg DataFormats/ParticleFlowReco

$ git cms-addpkg DataFormats/ParticleFlowCandidate

$ scram b -j 9
```

To run the file:

```
$ cmsShow -c
${CMSSW_BASE}/src/RecoParticleFlow/PFClusterProducer/test/hgcal_rechits.fwc
-g /afs/cern.ch/user/l/lgray/work/public/xHGCAL/cmsRecoGeom1-HGCAL.root
--sim-geom-file
/afs/cern.ch/user/l/lgray/work/public/xHGCAL/cmsSimGeom-14-HGCAL.root
/afs/cern.ch/user/l/lgray/work/public/CMSSW_6_2_X_SLHC_2014-07-17-0200/src/matri
```

sample simulation

sample file name:/TTGamma_TuneZ2star_8TeV-madgraph-tauola/Summer12_DR53X-PU_RD1_START53_V7N-

You can use DAS system to display the file names associated to the sample and also download one file. Alternative, you can copy a file from EOS at Fermilab to your area.

From DAS:

go to DAS web:https://cmsweb.cern.ch/das/

Then copy the file name in and search then we can download the file with the command(example):

```
$ xrdcp
root://xrootd.unl.edu//store/mc/Summer12_DR53X/TTGamma_TuneZ2star_8TeV-madgraph-
/some/local/path
```

From EOS:

From EOS: start valid proxy: voms-proxy-init --voms cms

Go to DAS web and fine the file, the entire name is the LFN, so we use the command to copy the file from EOS:

```
$ xrdcp root://cmsxrootd.fnal.gov//store/path/to/file /some/local/path
```

If the space is not enough to download the entire file, then we can change our working area:

To show the area:

```
$ quota
```

To check area we are working at now:

```
$ pwd
```

To change to other area:

```
$ cd /uscms_data/d1/xu0724/
```

This topic: Main > Shiyuan-HGCal
Topic revision: r18 - 2014-11-06 - ShiyuanXu