Photon Detection Analysis Software

Introduction

This document describes the software used to analyze a six-layer system model with each layer consisting of a 10 mm thick region filled with a 70/30 mixture of Ar/CO2, representing a GEM detector, sandwiched between an 8-micron thick tungsten foil and a 300-micron thick Kapton sheet. The layout of the first 2 layers of the system analyzed is shown in figure 1.

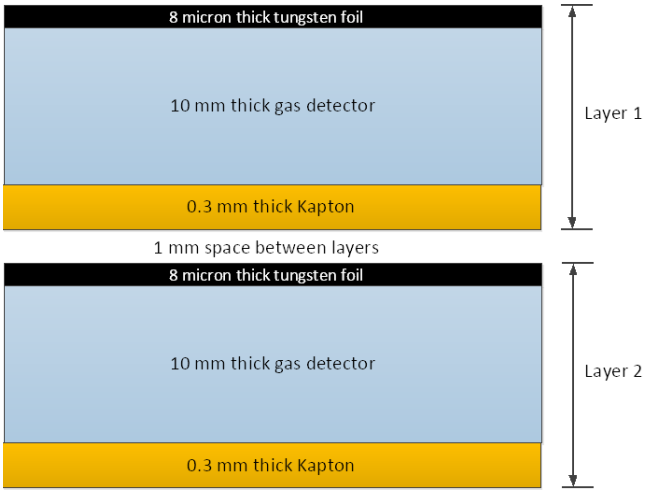


Figure 1.This figure shows 2 layers of a 6-layer stack including the dimensions of each of its parts

The software consists of 2 independent parts. The first part is a Geant4 simulation of photons with various energies and angles of incidence shot into the model. In this simulation photons that ionize the tungsten foil produce photoelectrons that pass into the following gas region. These electrons can ionize gas molecules and a number of statistics related to these ionizations, such as location and electron energy, are saved in a results ROOT file.

The second part of the software consists of a number of ROOT scripts that read the simulation results file and compute various statistics on the simulation results. These scripts generate a number of histograms of the statistics which are displayed locally in an X-window and can be saved in `.png` or `.pdf` format. In addition, one ROOT script also saves some simulation results in `.csv` format that can be transferred to a computer with MATLAB installed to enable viewing electron tracks.

The Geant4 Simulation

A version of the simulation corresponding to the current six-layer model can be run as a pre-compiled executable. However, if changes to the simulation are desired, the source code of the simulation that can be edited as desired is also resident on the cluster. Following the description on how to run the pre-compiled simulation is a description of the source code and instructions on how to compile the modified source

Running the Pre-Compiled Simulation

To run the simulation, first log in to the cluster as geantuser. If not at the cluster console, either log in to the cluster from a command prompt using the command `ssh -X geantuser@163.118.42.1,` or use PuTTY to log into `geantuser@163.118.42.1`. Logging in to the cluster may require connecting to the FIT network using the FortiClient VPN. After login, change to the directory `geantUsers/mLuntz/newSixLayerModel_build`. Note that there are some other directories whose names include `sixLayer`. The most up-to-date software is in the one mentioned above.

Set Simulation Parameters

In this `build` directory is a macro file named `run4.mac` that is used to specify the simulation parameters. The contents of that file are given below. The only lines that need to be possibly adjusted before running a simulation are in bold faced font. The line beginning `/gun/energy` is used to set the photon energy. The angle of incidence of the photon is set in one of the lines beginning `/gun/direction`. If an angle of incidence other than one of the ones provided is desired, a line can be added with the 3 values following the word `direction` being -sin(θ), 0, and cos(θ) where θ is the angle of incidence. Only one of these `direction` lines should be left uncommented.

The `/process/inactivate` line controls which physics processes are used in the simulation. Both the photoelectric effect and Compton scattering are used in the simulation when this line is commented out. If the comment is removed and no other change is made, only photoelectric effect is simulated and if the line is changed to `/process/inactivate phot`, then only Compton scattering is simulated. The final simulation configuration line, `/run/beamOn`, sets the number of photons generated during the simulation run.

```
# Macro file for example B1
#
# Can be run in batch, without graphic
# or interactively: Idle> /control/execute run1.mac
#
# Change the default number of workers (in multi-threading mode)
#/run/numberOfThreads 4
#
# Initialize kernel
/run/initialize
#
/control/verbose 0
/run/verbose 0
/event/verbose 0
/tracking/verbose 0
#
# gamma 6 MeV to the direction (0.,0.,1.)
#
/gun/particle gamma
```

```
/gun/energy 200 keV
/gun/direction 0 0 1 # 0 deg aoi
# /gun/direction -.258819 0 .965926 # 15 deg aoi
# /gun/direction -.5 0 .866025 # 30 deg aoi
# /gun/direction -1 0 1 # 45 deg aoi
# /gun/direction -.866025 0 .5 # 60 deg aoi
#/process/inactivate compt
#
# cause everything except electrons to be filtered out
#/vis/filtering/trajectories/create/particleFilter
#/vis/filtering/trajectories/particleFilter-0/add e-
#/vis/filtering/trajectories/particleFilter-0/add gamma
#
#/vis/drawOnlyToBeKeptEvents
/run/beamOn 1000000
#
#/vis/viewer/reset
#/vis/viewer/set/viewpointThetaPhi 90 0
#/vis/viewer/zoom 4
#/vis/viewer/pan .04 .04
#/vis/viewer/zoom 4
```

Running the Simulation

Before running a simulation, you must issue the command `geant410Source` from the command line. This command sets some environment variables needed by Geant4. After these variables are set, running the simulation is simply a matter of issuing the command `./exampleB1 run4.mac` from the `build` directory. The simulation should then run to completion and when complete will return to a command prompt. Upon completion the simulation results will be written to the `build` directory with the filename `simulationResults.root`. If desired for archival purposes, this file can be copied to another file whose name reflects the simulation parameters used.

Modifying the Simulation

The simulation was originally developed as a modification of the Geant4 basic example B1 so many of the files retain legacy B1 names. The source code is located in `geantUsers/mLuntz/newSixLayerModel`. In this directory is the main executable file, `exampleB1.cc`, a Makefile, some macros used for execution, and two subdirectories, `src` and `include`. The `include` subdirectory holds the header files used by the primary simulation files in the `src` subdirectory.

Structure of the Simulation

Of the source files in the `src` directory only 4 of them are likely to need modification for minor changes to the simulation. The file `B1DetectorConstruction.cc` is the main file describing the six-layer model. Each layer, consisting of a tungsten foil, a gas filled region, and a Kapton

sheet, of the six-layer stack is defined in this file. The file also specifies the dimensions, locations, and materials of each part of the stack.

The fundamental approach to the simulation is to shoot a photon particle into the stack and collect data at each step in the simulation occurring in any of the gas-filled detectors. These collected data are then saved in a file for off-line analysis.

The file `B1SteppingAction.cc` is where this data collection occurs. At each step in the simulation this file checks to see if the step involves an electron and occurs in one of the 6 gas-filled detectors. If so, it collects data about the step, such as its location (x, y, z), the electron momentum, the electron KE, etc.

The file `B1RunAction.cc` defines the results data to be collected and specifies the filename of the results file. At the end of the simulation run, this file also writes the data collected in `B1SteppingAction.cc` to the `simulationResults.root` file.

The file `B1PrimaryGeneratorAction.cc` defines the location of the source of particles and the number of particles generated per event. Earlier versions of this file also defined the initial direction of the particle but this required re-compiling the executable to change the direction. This direction is now defined in the simulation parameters macro so these lines are now commented out.

In addition to these 4 files there are 7 other files in the `src` directory. These files, `B1ActionInitialization.cc`, `B1EventAction.cc`, `PhysicsList.cc`, `PhysicsListMessenger.cc`, `PhysicsListEmStandard.cc`, `StepMax.cc`, and `StepMaxMessenger.cc` support the 4 main files. There should be no need to modify any of these files unless larger changes to the simulation such as adding new physics is anticipated. There should also be no need to modify the main simulation file `exampleB1.cc`.

In addition, there should be no need to modify the header files in the `include` directory when making minor changes to the simulation, with one possible exception. As stated above, the simulation results are written to a ROOT file. These results can instead be written to a .csv file by changing the header file `B1Analysis.hh`. To change to a `.csv` results file, just comment the line `#include "g4root.hh"` and uncomment the line `//#include "g4csv.hh"`. When writing to a .csv file the default filename will be `simulationResults_nt_Results.csv` rather than `simulationResults.root`.

Compiling a Simulation with Minor Changes

This section assumes that any changes requiring a re-compile were made on files located in the `newSixLayerModel` directory. If the source code has been copied to another directory, or if the modification involves new or renamed source files, then some preliminary steps described later will be required.

If only minor changes have been made to any of the source or header files, the compile process is very simple. The simulation was developed using Geant4 version 10.07. To use the correct version of Geant4, one must first give the command `geant410Source` from the command line. Next, change to the `newSixLayerModel_build` directory. Then to re-compile one need only type the command `make -j 2 exampleB1`.

Compiling a Simulation with More Significant Changes

If there have been more significant changes to the code, such as adding new source files, or if the entire `newSixLayerModel` directory is copied to a new location, then some additional steps are required before the executable can be generated. As an example, assume that the entire `newSixLayerModel` directory has been copied to the directory `somewhere/newLocation`. If the source main code filename in this directory has been changed from `exampleB1.cc` to something else, then the file CMakeLists.txt will need to be edited to reflect the new source filename and executable.

Under the directory `somewhere`, make a new directory `newLocation_build` and change to that directory. From that directory type the command

```
cmake -DGeant4_DIR=/usr/local/share/geant4-10.7.4-install/lib64/Geant4-10.7.4 ../newLocation
```

This should generate the Makefile that can be used from this directory to compile the executable as described earlier.

Processing the Simulation Results

This section assumes that the simulation was run from the directory `newSixLayerModel_build`. After the simulation runs and returns to the command prompt, change the directory with the command `cd ../rootProcessing`. In that directory are a number of processing scripts that read the simulation results and generate numeric and graphics results. Prior to processing the results, you need to issue the command `root624Source` from the command line. This sets up some paths needed to run ROOT.

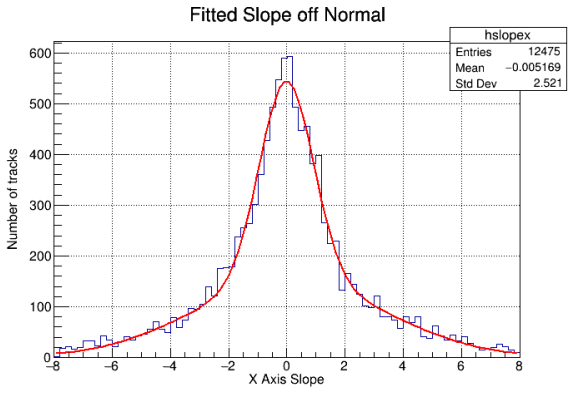


Figure 2. This is an example of a histogram of electron track slopes projected on the x axis along with a double Gaussian fit to the histogram. This histogram was generated from a

The first script is `axisSlopeWithFit.C`. This script computes a best fit line to each electron track and generates histograms of the slope between the z axis and that fit line projected on the x-z and y-z planes. Because it was found that the electron scatter was the same at each level, the electron tracks for all levels are used to develop these histograms. The script also performs a double Gaussian fit to each of these histograms. If X-windows has

been installed on the local machine, these histograms, of which an example is shown in figure 2, will be visible locally. If the local machine does not have X-windows, the root script on the cluster can be edited to uncomment the lines that save the graphics files `xaxisHist.png` and `yaxisHist.png`. When uncommented these files will be saved in the same directory as the script and, if desired, transferred from the cluster for viewing.

The fit parameters of these histograms, the mean and standard deviation of the core fit and the mean and standard deviation of the tail fit, are printed at the end of the script run. An average electron direction is computed by averaging all best fit lines as 3-d vectors and a pair of angles are reported as an overall direction. The angle theta is the angle that the average vector makes with the normal to the plane of the foil. The angle phi is the angle that it makes with the x axis.

Also printed at the end of the run is the number of photoelectrons generated in each tungsten foil layer that pass into that layer's detector along with the total for all layers.

In addition to these values, this script generates a file that can be used to visualize the electron scatter. This file, `trackRslts.csv`, contains the x, y, and z coordinates of some of a desired layer's electron tracks along with the coordinates of their line fit. A simple MATLAB script such as the one below can be used to visualize the tracks.

```
dat=importdata("trackRslts.csv");
events=unique(dat(:,1));
figure
for k=1:min(length(events),100)
    z=-dat(dat(:,1)==events(k),4);
    x=dat(dat(:,1)==events(k),2);
    y=dat(dat(:,1)==events(k),3);
    xfit= dat(dat(:,1)==events(k),5);
    yfit = dat(dat(:,1)==events(k),6);


    plot3(x,y,z,'-or','MarkerSize',3,'MarkerFaceColor', ...
        [0 0 0],'MarkerEdgeColor','none')
    hold on
    plot3(xfit,yfit,z,'-ob','MarkerSize',3,'MarkerFaceColor', ...
        [0 0 0],'MarkerEdgeColor','none')
end
grid
```

An example of the tracks plotted is shown in figure 3. In this figure the black dots connected by red lines indicate a point on the track where the electron ionized the gas. The red lines are the electron tracks, and the blue lines are the best linear fit to a track. Where there were only 2 points in the electron track, the fit line coincided with the track and obscured the track.

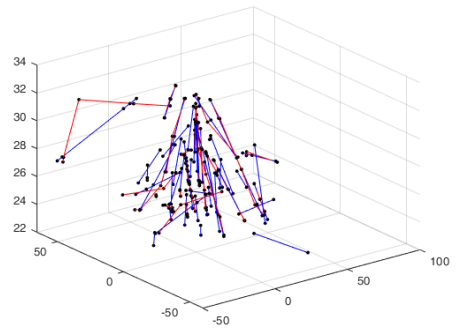
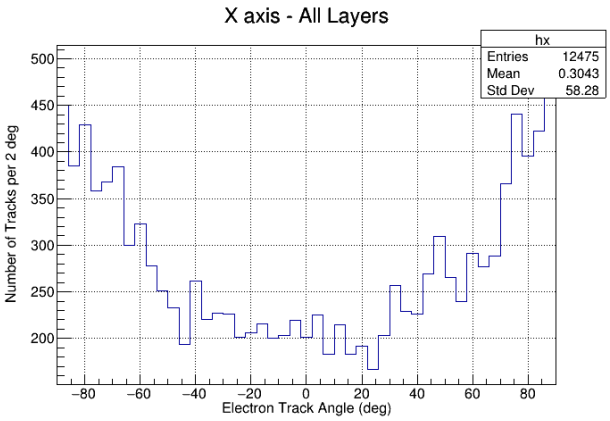


Figure 3. This is an example of the electron tracks and the associated linear fit to these tracks. The red lines are the tracks and the blue lines are the linear fits. The tracks were the results of a simulation of 200 keV photons hitting the first layer of the stack.

To run this ROOT script, all that is necessary is to start ROOT from the command line and at the root prompt type `.x axisSlopeWithFit.C(n)`, where `n` is a number from 1 through 6 specifying the layer for which the track results file is desired. Entering the number 7 will generate a `trackRslts.csv` file that contains tracks for all 6 layers. This file can also be viewed using the same MATLAB script.

The second script, `slopeAngleDensity.C`, generates histograms equivalent to `xaxisHist.png` and `yaxisHist.png` for the total of all 6 layers but with the track slope expressed as the equivalent angle. If the statements saving these histograms are uncommented, these equivalent histograms are saved in the files `histTrackAnglX.png` and `histTrackAnglY.png`. The file is run from the ROOT command prompt by entering `.x slopeAngleDensity.C`. An example of this histogram is shown in figure 4.



In comparing with figure 2, the reason that the histogram continues to increase with increasing angle while the slope falls off with increasing angle is because of the tangent function relationship between the slope and the angle. When the angle is large, a small change in angle has a much larger change in slope than the equivalent angle change when the angle is small.

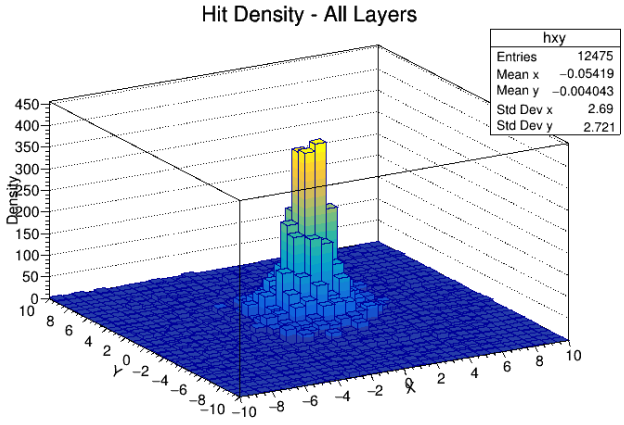


Figure 5. This histogram provides a 3-d view of the scatter of electrons based on a linear fit to the electron track. The histogram shows the density of electrons 1 mm below the tungsten foil

The third processing script that can be run is `slopeDensity.C`. Running this script is simply a matter of entering `.x slopeDensity.C` from the ROOT command prompt. This script generates 3 different histograms of the simulation results. If the print statements are uncommented, the script stores them in graphics files that can be viewed on another computer. The first histogram, an example of which is shown in figure 5, is `histSlope3d.png`. This histogram

illustrates the density of electrons in a plane 1 mm from the top of the detectors. This is an approximation to the actual electron density since it assumes that each electron is travelling along its fit line rather than its actual path.

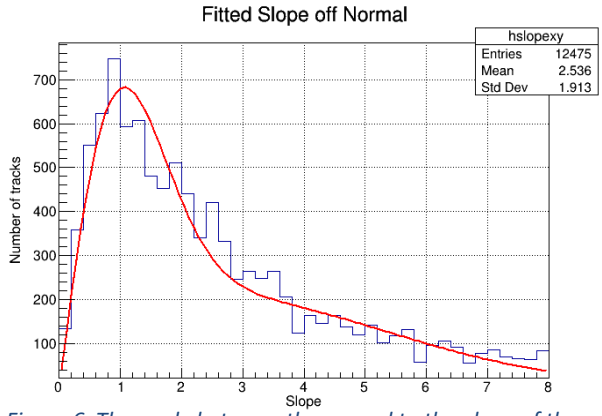


Figure 6. The angle between the normal to the plane of the foil and the linear fit to the electron track is shown in this histogram. This figure also includes a fit to the measured histogram using a function that assumes both the x and y axis histograms are double Gaussian functions

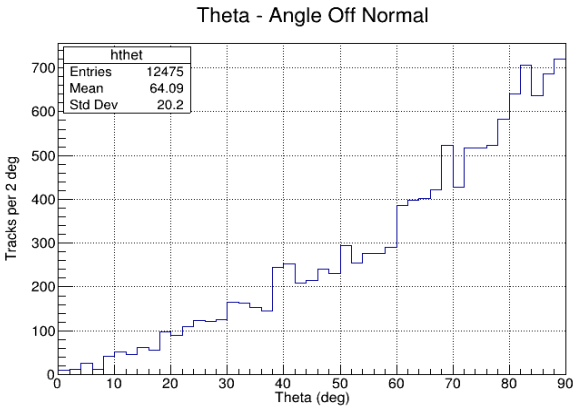


Figure 7. This is the equivalent to figure 6 but with the slope expressed as an angle. The reason that the histogram increases with increasing angle is explained in the text.

The second histogram, an example of which is shown in figure 6, is saved in `histThetSlope.png`. This is a histogram of the slope that the fitted electron tracks make as measured in the plane defined by the z axis and the fit line. This is the slope equivalent of the angle theta. This histogram includes a fit using a function that assumes the x and y axis slopes are double Gaussian functions with equal mean values.

The last histogram, saved in `histThetAngle.png`, is just a histogram equivalent to `histThetSlope.png`, but plotted with the slope converted to its equivalent angle. An example of this histogram is shown in figure 7. The reason that this histogram continues to increase with increasing angle although the slope falls off with increasing slope is due to the nonlinearity in the tangent function relating slope to angle. The difference in slope for a given change in angle increases as the slope/angle increases.