CaloSim

A Geant Simulation of the Tungsten/Scintillating Fiber calorimeter for the new g-2 experiment.

Outline

1. Geant Code
    a. Calosim.cc
    b. CaloSimPrimaryGeneratorAction.cc
    c. CaloSimPrimaryGeneratorMessenger.cc
    d. CaloSimDetectorConstruction.cc
    e. CaloSimSD.cc
    f. CaloSimRunAction.cc
    g. CaloSimDetectorHit.cc
    h. CaloSimMaterialsConstruction.cc
    i. CaloSimPhysicsList.cc
    j. CaloSimSteppingVerbose.cc
2. Compiling
3. Running
    a. Interactive Mode
    b. Messenger Commands
    c. Macros
    d. Batch mode
4. Visualizations
5. Root Tree Info
    a. Blocks
    b. Hodoscope
    c. Positions
        i. Init
        ii. Calo

Geant Code

CaloSim.cc

This a sort of "main" file for the calosim simulation. I've not edited it much since I've been using it. The only things I've set have been the default output file name ("out.root"), and also there is a section here where you set the colors on visualizations. I had to create a transparent color to get rid of gammas in showers. Also here you set the default macro to use, usually set to "vis.mac".

CaloSimPrimaryGeneratorAction.cc

This is the file and program called to create particles. I have set up 2 modes, one where you read electrons in from a ".dat" file, and another where you use a random number generator to produce particles. Setting the parameters for either reading from a file or randomly generating particles can be done either in this file or using the messenger functions, which will be discussed later

CaloSimPrimaryGeneratorMessenger.cc

This file allows changing of some parameters of particle generation without re-compilation. This is one of the few files where looking at the .hh file really helps with understanding how it works. To add messenger functions I've not added, follow the functions already made (that's how I did it). More to come when I discuss messenger functions

CaloSimDetectorConstruction.cc

Where the geometry of the detector is set. The pattern is as follows: first you make a solid (create the dimensions of the object) then you define a logical volume (fill it with a material) and then a physical placement (the final position). One nifty thing about this is that one doesn't have to define each little fiber – using "mother volumes" one only needs to define a few and then they get copied. The way the calo is defined is:

1. define a "box" to contain the calo
2. Define blocks, and position them with a for loop
3. Define a "layer" of optical glue. Make these 10 layers inside one mother calo block, and the layers get copied to all calo blocks
4. Define a "fiber", make 10 of the fibers, make them in a mother layer, they will be copied to all layers in the block, as well as every block

In this file the block, layers, fibers are defined. Also, a hodoscope has been made.

CaloSimSD.cc

The SD stands for sensitive detector. This is where we make the calorimeter sensitive to measuring data. The NTuple is defined here, as well as steps for defining how data is taken, such as energy or positions. At the end, the ntuple is filled, with units determined.

CaloSimRunAction.cc

Not a whole lot worth noting here, since at the end of each run we don't do much. If you're upset about the "/vis/viewer/update" command error message you keep getting you can comment it out here, but be careful to remember to undo if you want to use visualizations.

CaloSimDetectorHit.cc

I don't fully understand this file, but also have not needed to edit it as of yet.

CaloSimMaterialsConstruction.cc

A file to define materials to be used in detectorconstruction. Haven't edited it yet.

CaloSimPhysicsList.cc

This file sets the physical processes involved in the simulation. Also haven't edited this yet either.

CaloSimSteppingVerbose.cc

Once again, don't know if I even need this file.

Compiling

Compiling is pretty simple – I've just been using gmake whenever I edit a file. I've not had problems when I don't do a gmake clean, so to save time when you edit a file, just using gmake should work.

Running Geant

There are two ways to run geant, first interactively, and second in "batch mode"

Interactive Mode

To run the program, at the prompt just enter the executable found at

CaloSim/bin/Linux-g++/CaloSim

The program automatically executes the vis.mac macro. In it I've simply set the verbosity (how much stuff it prints to the terminal) to a functional minimum. It then gives you a prompt inside geant looking like:
Idle>

Here commands like "cd" and "ls" work. Unfortunately, tab completion and hitting the up key for a history does not work. Here you can access options, mainly thru the run/ directory as well as Gun/. To simply run a few particles, the

run/beamOn 10

command will run 10 particles, and write the result to a root file, here out.root, because that was the default and we did not specify a certain file to use.

Messenger Commands

I've created some useful messenger commands that will help with particle generation without recompilation. To find these, run the interactive mode and do

Idle> cd Gun/
Idle> ls

You'll see a list of commands:

1. Usefile
    a. This is a command that sets a flag for whether or not you'd like to use an input file. If you set it to 0, then you do not use an input file. If you set it to 1, then you do
    b. Use like: Idle> UseFile 1
2. inputFile
    a. This sets the name of the input file you'd like to use
    b. Use Lke: Idle> inputFile /data/npluser2/nschroed/g2electrons/wiggle1.dat
    c. Note: this will only be used if you do UseFile 1 before hand

3. Energy
    a. Sets energy of particles (GeV)
    b. Use like : Idle> Energy 2
4. Angle
    a. Sets starting angle of random beam, in degrees, measured as angle away from perpendicular to normal to calo, turning inward (to the right as you look down beam)
    b. Use like: Idle> Angle 5
5. AngleRand
    a. Sets range of random angles to give beam. If you have an angle of 5 and you set angle rand to 1, you will have a uniform dist of angles from 5 to 6
    b. Use Like Idle> AngleRand 1
6. XInit
    a. Sets Initial X of random beam, in cm, measured with inward to the right, positive X
    b. Use Like Idle> XInit -3
7. XinitRand
    a. Sets random offset for XInit. If you have XInit -3 and XInitRand 10, you'll have a uniform distribution of initial X from -3 to 7
    b. Use Like Idle> XInitRand 10
8. YInit
    a. Same as Y Init, negative is down
    b. Use Like Idle> YInit
9. YinitRand
    a. Same as XInitRand
    b. Use Like Idle> YInitRand

Macros

The macros that geant uses are essentially just lines of code you'd have to type yourself in an interactive mode. The default vis.mac file is bare bones, the verbose commands at the beginning simply determine how much out put gets printed to terminal. I recommend leaving it as low as you see.

You can make your own macros, and a lot of Ron's old macros are in the CaloSim/mac/ folder. Macros are useful because you wont have to retype things like filenames over and over again when debugging. I have noticed that sometimes a segmentation fault will arise when using something in a macro, but you wont get that fault when using the interactive mode. This usually means you're trying to use a messenger function to change something that doesn't have a variable assign to the info you want to change.

The "#" sign comments out lines in a macro.

Having a separate visualization macro is also a good idea. There are lots of commands that have to be used for a visualization to occur. More on that later

Batch Mode

To run in “batch mode” you put the executable, the macro you wish to run, and then specify the root output.

```
$> ./bin/Linux-g++/CaloSim   mac/vis.mac    ./myrootfile.root
```

That sequence would execute the CaloSim program with the macro mac/vis.mac and output the results to ./myrootfile.root

When run this way, the program will exit upon completion. Here at UIUC, we used a shell script to run the program and then submitted that shell script to the pion machines. This will have to be tailored to whatever computing services you plan on using. For reference, the shell script I use is rungeant.sh.

Visualizations

I used the HepRepXML program to create images of the calo block. To do so, the HepRep software must be installed in your directory as far as I can tell. In the macro you use, make sure the commands

/vis/open HepRepXML
/vis/viewer/set/style wirefram
/vis/drawVolume
/vis/scene/add/trajectories

are included. If you would like a pileup of images, include

/vis/scene/endOfEventAction accumulate

To view the files, you must have the java applications suite installed. I used the command

jas3-0.8.3/jas3 scene-0.heprep.zip

to open a visualization file. I do not know how to change the output name for visualizations, they all come out as scene-0.heprep.zip

Once you've opened a file, you open the viewer using file –new –wired 4 window. Once inside, you can zoom in and out, and on the left there are icons with other viewing tools. You can also turn off objects with the buttons on the left menu. You can turn things off in the detectorconstruction file by changing vis attributes to "offAtt".

Finally, most of what I know about this is by trial and error. For more info, look at http://geant4.slac.stanford.edu/Presentations/vis/G4HepRAppTutorial/G4HepRAppTutorial.html

Root Tree Info

The root tree contains several pieces of info:

1. block energies
    a. b01 corresponds to the block on the bottom left as you look from the beam (dave, look at the map from the run).
    b. The leaves contain info for energy deposited in a block
2. Hodoscope
    a. Same thing as block energies, only its for the vertical and horizontal hodoscopes
    b. Leaves contain energy deposited by electrons in hodoscope staves
3. Positions
    a. InitX is the initial position of the electron
    b. CaloX SHOULD be the position of impact on the calorimeter. I'm 90% certain this is done correctly.
4. Initial Momentums
    a. initPX is the initial X momentum of the electron
    b. same for initPY and PZ
5. WEdep
    a. Energy deposited in the tungsten
6. totalEdep
    a. total energy deposited in calo
7. FiberEdep
    a. Energy deposited in all fibers
8. glueEdep
    a. energy deposited in glue
9.