MPAGS Particle Techniques
Project Documentation
(v1.01)

Kostas Nikolopoulos

January 2022

1 Intro

The code in: https://github.com/knikolop-bham/MPAGS-EPPT-B5 is practically Geant4 exampleB5 with minor amendments[1]. Remember to keep up-to-date with changes in the code with:

```
git pull
```

2 Setting-up

Example for Birmingham Particle Physics system: log in to eprexa, eprexb or another node of your choice

```
cc7 #(to change to centos7)
mkdir EPPT; cd EPPT
source /cvmfs/sft.cern.ch/lcg/views/LCG_94/x86_64-centos7-gcc8-opt/setup.sh
git clone https://github.com/knikolop-bham/MPAGS-EPPT-B5.git
mkdir build; cd build
cmake ../MPAGS-EPPT-B5/
make -j 10
```

once the compilation completes try to see if everything works well:

```
./exampleB5 run1.mac
```

For those with no access to cvmfs, download Geant4 from https://geant4.web.cern.ch/support/download (there is the option to download pre-compiled libraries for specific operating systems, otherwise one would have to compile the source code) and then download the example code from git, and compile it.

3 Executing the programme and understanding the output

Executing the programme:

```
./exampleB5
```

brings up a graphical user interface with a command line and event viewer.
The set-up of the experiment is a two arm spectrometer. Each arm consists of a hodoscope and the 5 layers of drift chambers with distance of 0.5 m. The axis of the first arm is along the z axis, and the particles are initially directed towards the positive z-axis. The positive x-axis is on the “horizontal” and the positive is towards the “left”. The angle between the two arms can be set, but start with them in a straight line (i.e. angle equals 0). Between the two arms a cubic magnet, with a magnetic field along the y-axis, that is “vertical” in an actual experiment). Behind the second spectrometer arm, there is a homogeneous electromagnetic calorimeter (CsI)

[1] Many thanks to Jack Matthews for typing up the first version of these notes in January 2021.

and then a hadronic calorimeter (Pb/Scintillator). Changes to geometry can be verified by looking at the GUI event viewer.
The programme can be executed with macro file (in this case mymacro.mac) as an argument.

```
./exampleB5 mymacro.mac >mymacro.out
```

Example macro file. Please note that within a line whatever follows a # is a comment and is ignored. Note that Geant4 is now multi-threaded. This makes the code run faster, but the log output is a bit more difficult to decode. Remember lines starting with “G4WTx” is output of thread “x”.

```
# Change the default number of workers
# (in multi-threading mode which is the default)
#/run/numberOfWorkers 4

# Initialize kernel
/run/initialize

# do not randomize type of primary particle
/B5/generator/randomizePrimary FALSE

# Set the ange between the two arms
/B5/detector/armAngle 0. deg

# Set the magnetic field strength
/B5/field/value 1. tesla

# Set the particle that will be generated
# (choice: mu+, e+, pi+, kaon+, proton)
/gun/particle mu+

# Set the particle's initial momentum
/B5/generator/momentum 50. GeV

# Set standard deviation of initial momentum.
# Zero means all initial particles have exactly the same momentum.
/B5/generator/sigmaMomentum 0. GeV

# Set standard deviation of initial angle.
# Zero means all initial particles have exactly the same direction.
/B5/generator/sigmaAngle 0. deg

# Run five events
/run/beamOn 5
```

The output of the code can be found in mymacro.out. A few pointers in understanding the log file.
The output below provides a list and information about the defined materials. In this case: Air, Argon, Plastic (Vinyltoluene), CsI, Lead. The materials defined are :

```
***** Table : Nb of materials = 5 *****

 Material:     G4_AIR     density:  1.205 mg/cm3  RadL: 303.921 m   Nucl.Int.Length: 7
                              Imean:  85.700 eV   temperature: 293.15 K  pressure:  1.00 atm

   ---> Element: C (C)    Z =  6.0    N =    12    A = 12.011 g/mole
            ---> Isotope:    C12    Z =  6    N =  12    A =  12.00 g/mole    abundance:
            ---> Isotope:    C13    Z =  6    N =  13    A =  13.00 g/mole    abundance:
1.070 %
```

```
         ElmMassFraction:   0.01 %  ElmAbundance   0.02 %

  --->  Element: N (N)   Z =  7.0   N =    14   A = 14.007 g/mole
        --->  Isotope:   N14   Z =  7   N =  14   A =  14.00 g/mole   abundance:
        --->  Isotope:   N15   Z =  7   N =  15   A =  15.00 g/mole   abundance:
0.368 %
         ElmMassFraction:  75.53 %  ElmAbundance  78.44 %

  --->  Element: O (O)   Z =  8.0   N =    16   A = 15.999 g/mole
        --->  Isotope:   O16   Z =  8   N =  16   A =  15.99 g/mole   abundance:
        --->  Isotope:   O17   Z =  8   N =  17   A =  17.00 g/mole   abundance:
0.038 %
        --->  Isotope:   O18   Z =  8   N =  18   A =  18.00 g/mole   abundance:
0.205 %
         ElmMassFraction:  23.18 %  ElmAbundance  21.07 %

  --->  Element: Ar (Ar)   Z = 18.0   N =    40   A = 39.948 g/mole
        --->  Isotope:  Ar36   Z = 18   N =  36   A =  35.97 g/mole   abundance:
0.337 %
        --->  Isotope:  Ar38   Z = 18   N =  38   A =  37.96 g/mole   abundance:
0.063 %
        --->  Isotope:  Ar40   Z = 18   N =  40   A =  39.96 g/mole   abundance:
         ElmMassFraction:   1.28 %  ElmAbundance   0.47 %
 ...
```

Subsequenrly a geometry check follows, ensuring that no volumes are overlapping.

```
Checking overlaps for volume magneticPhysical ... OK!
Checking overlaps for volume firstArmPhysical ... OK!
Checking overlaps for volume fSecondArmPhys ... OK!
...
Checking overlaps for volume EMcalorimeterPhysical ... OK!
Checking overlaps for volume HadCalorimeterPhysical ... OK!
Checking overlaps for volume HadCalScintiPhysical ... OK!
```

Then the Geant4 physics lists are provided. The physics lists are the physics models that will be used to simulate the interactions of particles with matter. For the same process, different models may apply for different energy ranges and for different particle species. In this case we have the default physics lists which are a compromise between precision and speed of execution. For specific applications one may need to use specialised physics lists. Also through the physics lists one may decide to "turn on/off" certain processes to study the effect.
After the configuration of the programme is provided, the simulation of events starts. The output is given below, and it gives you information about the various hits in the drift chambers, information about the energy deposited in the calorimeter, etc

```
G4WT0 > ### Run 0 starts on worker thread 0.
G4WT0 > Going to create slave ntuples from main
G4WT0 > --> Event 1 starts with initial seeds (99594690,49590222).
G4WT0 >
G4WT0 > >>> Event 1 >>> Simulation truth : proton (0,0,50000)
G4WT0 > Hodoscope 1 has 1 hits.
G4WT0 >   Hodoscope[7] 4.987661834418 (nsec)
G4WT0 > Hodoscope 2 has 1 hits.
G4WT0 >   Hodoscope[11] 43.355402605649 (nsec)
G4WT0 > Drift Chamber 1 has 5 hits.
G4WT0 >   Layer[0] : time 6.6721237031776 (nsec) --- local (x,y) -0.00823144773101
G4WT0 >   Layer[1] : time 8.3402380527246 (nsec) --- local (x,y) -0.00420319400435
```

```
G4WT0 >   Layer[2] : time 10.008352404277 (nsec) --- local (x,y) 0.00724155042907
G4WT0 >   Layer[3] : time 11.676466757471 (nsec) --- local (x,y) 0.021133638863061
G4WT0 >   Layer[4] : time 13.344581112151 (nsec) --- local (x,y) 0.037879768327896
G4WT0 > Drift Chamber 2 has 5 hits.
G4WT0 >   Layer[0] : time 35.030584784339 (nsec) --- local (x,y) -29.783993947147,
G4WT0 >   Layer[1] : time 36.69881820035 (nsec) --- local (x,y) -35.757554806523,
G4WT0 >   Layer[2] : time 38.367051581667 (nsec) --- local (x,y) -41.730219971081,
G4WT0 >   Layer[3] : time 40.03528486714 (nsec) --- local (x,y) -47.700455502051,
G4WT0 >   Layer[4] : time 41.703518135083 (nsec) --- local (x,y) -53.670213960774,
G4WT0 > EM Calorimeter has 57 hits. Total Edep is 1479.3131886983 (MeV)
G4WT0 > Hadron Calorimeter has 15 hits. Total Edep is 1447.6490617717 (MeV)
G4WT0 > --> Event 2 starts with initial seeds (11291749,28987050).
```

Information about the run are stored in the output file, named “B5.root”. Opening this with root:

```
>  root B5.root
   ------------------------------------------------------------
  | Welcome to ROOT 6.14/04                 http://root.cern.ch |
  |                                (c) 1995-2018, The ROOT Team |
  | Built for linuxx8664gcc                                     |
  | From tags/v6-14-04@v6-14-04, Aug 23 2018, 17:00:44          |
  | Try '.help', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------


root [0]
Attaching file B5.root as _file0...
(TFile *) 0x1a66d00
root [1] _file0->ls()
TFile**         B5.root
 TFile*         B5.root
  KEY: TTree    B5;1     Hits
  KEY: TH1D     Chamber1;1      Drift Chamber 1 # Hits
  KEY: TH1D     Chamber2;1      Drift Chamber 2 # Hits
  KEY: TH2D     Chamber1 XY;1   Drift Chamber 1 X vs Y
  KEY: TH2D     Chamber2 XY;1   Drift Chamber 2 X vs Y
```

The file contains some histograms (ignore) and a TTree. The contents of the tree are given below.

```
root [2] B5->Print()
******************************************************************************
*Tree    :B5        : Hits
*
*Entries :        5 : Total =           16529 bytes  File  Size =       8373 *
*        :          : Tree compression factor =   1.23
*
******************************************************************************
*Br    0 :Dc1Hits   : Int_t B5
*
*Entries :        5 : Total  Size=         698 bytes  File Size  =        200 *
*Baskets :        2 : Basket Size=       32000 bytes  Compression=   1.00
*
 *...........................................................................*
*Br    1 :Dc2Hits   : Int_t B5
*
*Entries :        5 : Total  Size=         698 bytes  File Size  =        200 *
*Baskets :        2 : Basket Size=       32000 bytes  Compression=   1.00
```

```
*
 *............................................................................*
*Br    2 :ECEnergy  : Double_t B5
*
*Entries :        5 : Total  Size=        731 bytes  File Size  =        222 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br    3 :HCEnergy  : Double_t B5
*
*Entries :        5 : Total  Size=        731 bytes  File Size  =        222 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br    4 :Time1     : Double_t B5
*
*Entries :        5 : Total  Size=        716 bytes  File Size  =        216 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br    5 :Time2     : Double_t B5
*
*Entries :        5 : Total  Size=        716 bytes  File Size  =        216 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br    6 :ECEnergyVector : vector<double> B5
*
*Entries :        5 : Total  Size=       3969 bytes  File Size  =       2120 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.62
*
 *............................................................................*
*Br    7 :HCEnergyVector : vector<double> B5
*
*Entries :        5 : Total  Size=       1569 bytes  File Size  =        793 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.32
*
 *............................................................................*
*Br    8 :Dc1HitsVector_x : vector<double> B5
*
*Entries :        5 : Total  Size=        974 bytes  File Size  =        446 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br    9 :Dc1HitsVector_y : vector<double> B5
*
*Entries :        5 : Total  Size=        974 bytes  File Size  =        446 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   1.00
*
 *............................................................................*
*Br   10 :Dc1HitsVector_z : vector<double> B5
*
*Entries :        5 : Total  Size=        974 bytes  File Size  =        446 *
```

```
*Baskets :          2 : Basket Size=      32000 bytes  Compression=   1.00
*
*............................................................................*
*Br   11 :Dc2HitsVector_x : vector<double> B5
*
*Entries :          5 : Total  Size=        990 bytes  File Size  =        462 *
*Baskets :          2 : Basket Size=      32000 bytes  Compression=   1.00
*
*............................................................................*
*Br   12 :Dc2HitsVector_y : vector<double> B5
*
*Entries :          5 : Total  Size=        990 bytes  File Size  =        462 *
*Baskets :          2 : Basket Size=      32000 bytes  Compression=   1.00
*
*............................................................................*
*Br   13 :Dc2HitsVector_z : vector<double> B5
*
*Entries :          5 : Total  Size=        990 bytes  File Size  =        462 *
*Baskets :          2 : Basket Size=      32000 bytes  Compression=   1.00
*
*............................................................................*
```

ECEnergyVector and HCEnergyVector are std::vector<double> giving the information about the energy deposited in each calorimeter cell.
The Drift chamber vectors (DC1, DC2) give the exact X and Y positions of hits in layers, no resolution effects included, and the Z layer that the hit occurred in. Note that ”Row” shows which event the data corresponds to. The information of the drift chamber before/after the particle passes through the magnetic field can be used to calculate momentum of particle.

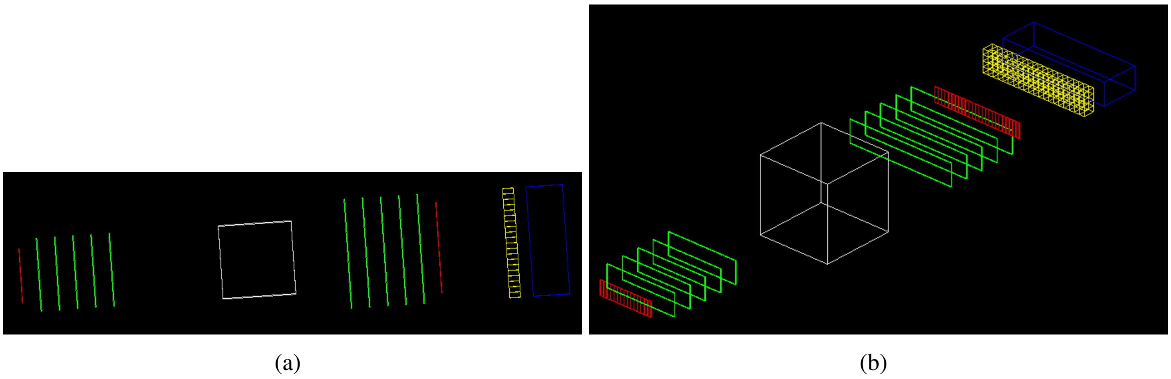
```
root [3] B5->Scan(”Dc1HitsVector_x:Dc1HitsVector_y:Dc1HitsVector_z”)
***********************************************************
*    Row   * Instance * Dc1HitsVe * Dc1HitsVe * Dc1HitsVe *
***********************************************************
*        0 *        0 * 0.0129406 * 0.0089895 *         0 *
*        0 *        1 * 0.0313354 * 0.0004807 *         1 *
*        0 *        2 * 0.0504493 * -0.016349 *         2 *
*        0 *        3 * 0.0697113 * -0.037151 *         3 *
*        0 *        4 * 0.0887113 * -0.053507 *         4 *
*        1 *        0 * 0.0163292 * -0.038325 *         0 *
*        1 *        1 * 0.0354586 * -0.056851 *         1 *
*        1 *        2 * 0.0610413 * -0.076174 *         2 *
*        1 *        3 * 0.0916985 * -0.098445 *         3 *
*        1 *        4 * 0.1298779 * -0.123623 *         4 *
...
```

4 Geant4 code

exampleB5.cc is the main file, creates all the different objects needed by Geant4. In general, you won’t need to modify this.
Geant4 has runs, with certain action being undertaken at the beging and the end of the run. Runs are made up of events, and certain actions can be undertaken at the start or end of an event. Events are made up of tracks, and tracks are made up of steps.
Geometry is defined in B5DetectorConstruction. Important part is the Construct() function. Initially the materials are constructed, and points are prepared to the various materials needed for the geometry. Then a ”World”

volume is created for detector geometry to go in. Volumes are made up of a solid volume which defines the shape, a logical volume, which defines the material, and a physical volume which defines the placement.
Magnetic field is set up by creating a tube volume in the size and shape of the magnetic field. It is then placed within the world with a 90 degree rotation. Step limits are set to ensure interactions occur.
Regions of air for the arms are defined and placed, with the second having options for the angle to change. The physical volume of the arm is placed within the world volume, and then the hodoscopes are placed into the arm volume. Drift chambers are placed using a for loop to get the set of layers. Similar done for second arm.
Calorimeter volume is created, then individual cells are produced. Hadronic calorimeter creates alternating layers of different materials.
VisAttributes sets colours for the GUI view.
ConstructSDandField() sets a volume as a sensitive detector, which means that information will be tracked in that volume and desired information can be read out - applied to each detector. Magnetic field also constructed here.
Sensitive Detectors: Hits are created at steps where energy is deposited. Hits are given to the sensitive detector to be processed and stored - ProcessHits() function does this. In B5DriftChamberSD.cc, ProcessHits() begins by obtaining the track the step belongs to and checks the charge, so that the processing can be stopped if the particle is neutral. Gets the information about the particle at the start point of a step, and then saves it to the hit which is stored. Hit class contains instructions on what to store.
Run Actions: Creates ntuples, which store the information. Begin of run action - at the start of the run create the output file. End of run action - write to and save the output file.
Event Actions: Begin of event action - sets things up. End of event action - loops over the different drift chambers, gets the hit collection for each, and loops over the hits to get and store the information in them.
Summary of run/event action: new variables are created in run action, and filled in in event action.
Primary Generator Action: GeneratePrimaries creates the initial particles in the event, defining particle type, momentum, etc.

5 Detector Geometry and Sample Events


(a)

(b)

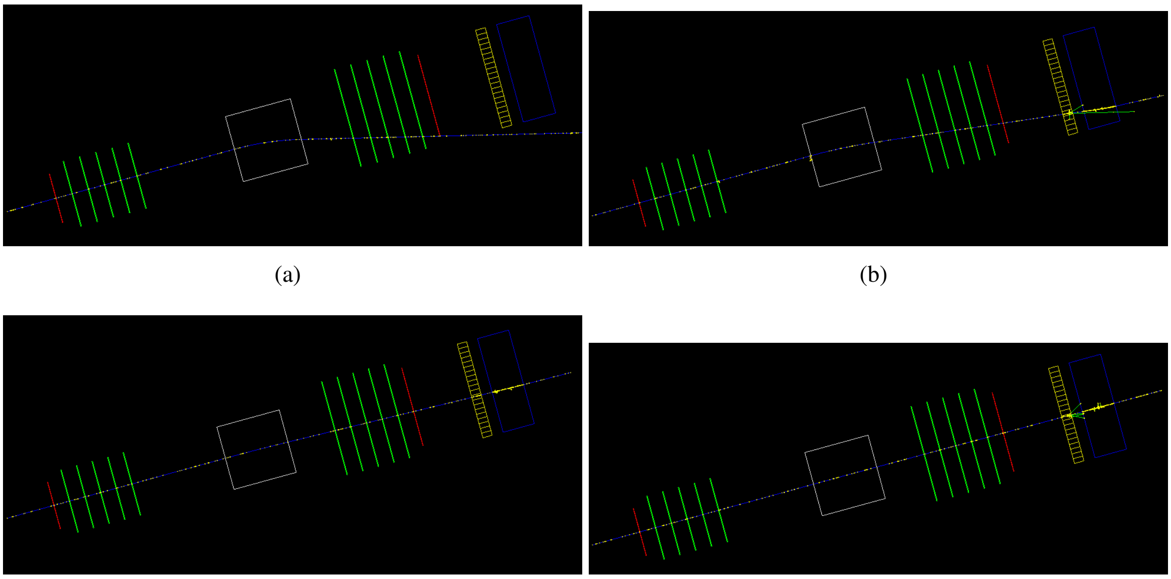
Figure 1: (a) View of the experimental set-up from abovr. (b) ”side” view of the experimental set-up.

6 Homework

6.1 Tracking

Having discussed the basic structure of the code, and how to run/modify it, the first exercise is the following:

- For a magnetic field of 0.5 T, an angle of the spectrometer of 0 degrees (i.e. the axes of the tow arms are on the same line), send 1000 antimuons (mu+) along z+ with a momentum of 100 GeV. The precision coordinate is ”x”, and the second coordinate is ”y”. Assume a precision along x of 100μm, and along y of 1cm. Reconstruct the track of the muons along the x-z plane and estimate their momentum. Draw the distribution of reconstructed momentum, and find the resolution.



(a)

(b)

(c)

(d)

Figure 2: Example μ^+ events for different momentum values: (a) 2.5 GeV, (b) 5 GeV, (c) 20 GeV (d) 100 GeV

- Estimate the momentum resolution if the magnetic field was 0.25 T or 1T.
- For a magnetic field of 0.5 T, estimate the momentum resolution for 50 GeV and 200 GeV muons
- What is the momentum resolution for 100 GeV muons if you assume perfect knowledge of the position along x? why?
- Add 5cm of Pb immediately before the magnetic and estimate the momentum resolution. Repeat by place the Pb immediately after the magnet. What is the effect?
- (Optional) Perform a complete fit to the track that takes as input the measurements in the x-z plane for both DC1 and DC2.

6.2 Calorimetry

- For a magnetic field of 0.5 T, an angle of the spectrometer of 0 degrees (i.e. the axes of the tow arms are on the same line), send 1000 antimuons (mu+) along z+ with a momentum of 100 GeV. Study the energy depositions in the Electromagnetic and Hadronic calorimeters.
- Repeat with positrons, but also try to reconstruct the positron energy using the calorimeter information.
- Repeat with protons, but also try to reconstruct the energy of the protons using the calorimeter information.