Lund University Department of Physics

Geant 4 Homework Report

Skorda Eleni

LUND
UNIVERSITY

October 23, 2018

Project description

In this project we want to scan the container of a truck with thin aluminium walls, to reveal its contents without opening it. The most common way is to use X-ray photons to scan it. Here we try γ photons and neutrons. Both of them are absorbed by the material we want to scan. By knowing they way the materials absorb radiation we can even know, what type of materials are inside the truck. The other very interesting thing we can do is use cosmic muon to scan the truck. Since the scope of the project is to apply these principles using Geant4 simulation toolkit, we will not discuss the physics behind each case.

Detector Geometry

The first step is to construct the truck, place objects inside it and then create the detectors around it. We use NIST manager for all the materials used but CsI. This was constructed with the method presented in the lectures. Inside the container there are :

- orb made of concrete
- tubes from stainless steel (created with parametrized volume)
- small tube from uranium

The DetectorConstuction method is included in the end of the report as an extra material.Picture 1 is showing how the dector looks like and also 10 photons that hit the container from the opposite side from the photon detector.

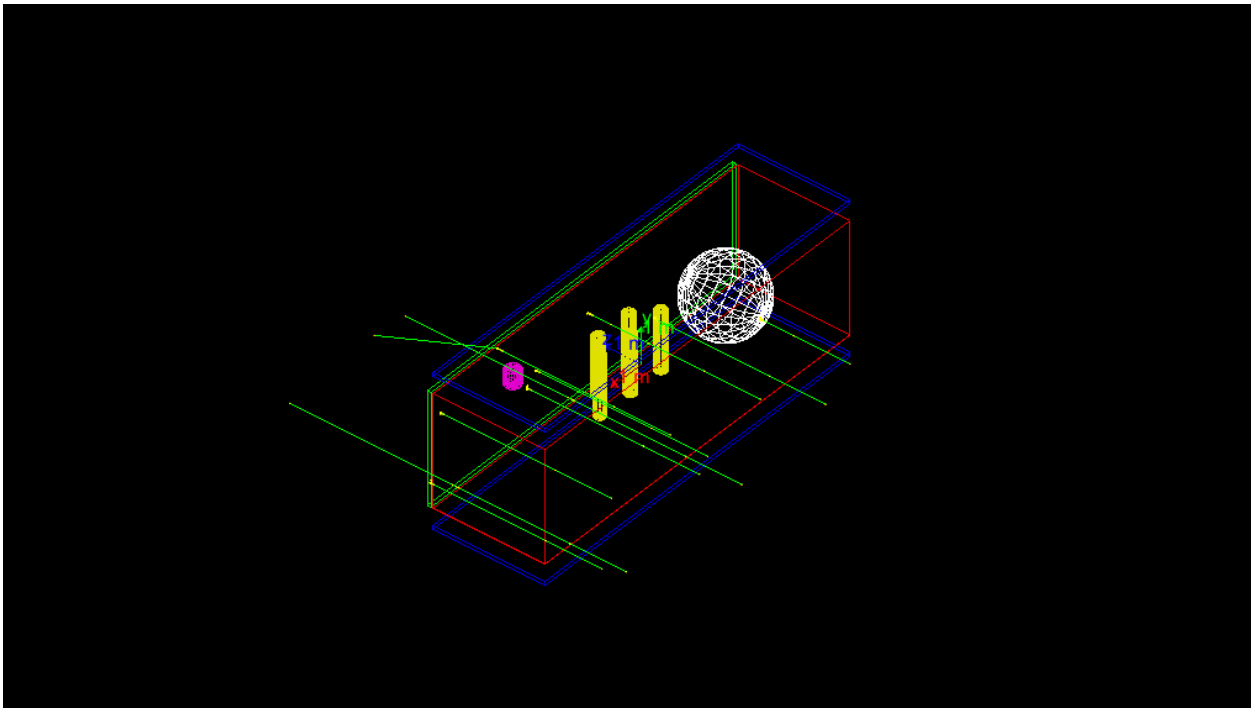

Figure 1: Picture for 5MeV photons. We generated 10 events

Simulation with γ

The first Part of the exercise is to create photons that will scan the container on the (x,y) plane. For this purpose we generate many events and in each of them there is only one photon of 5 MeV energy hitting the detector with direction parallel to z axis and at a random point in the (x,y) plane.

For this part a command-base scorer is used and the macro for this is included in the end of this report. Figure 2 shows the picture from the detector when we simulate 1million events of 1 photon per event

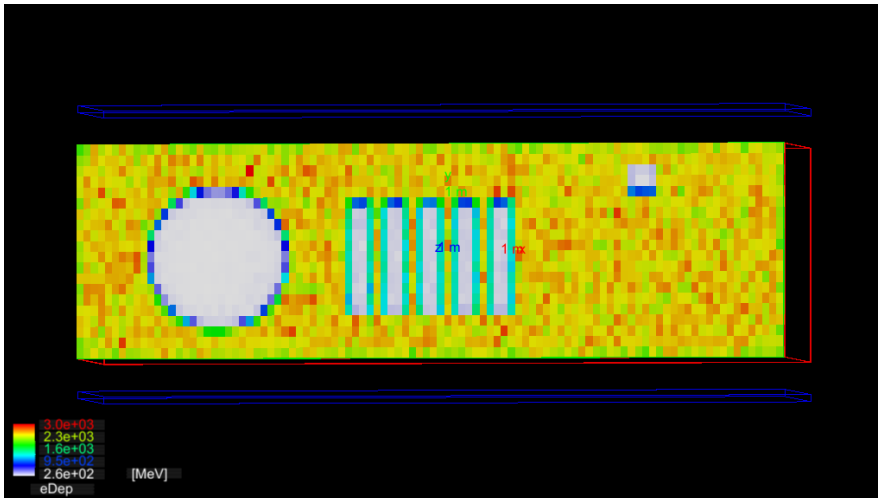

Figure 2: Picture for 5MeV photons. We generated 1 million events

Simulation with n

The first Part of the exercise is to create photons that will scan the container on the (x,y) plane. For this purpose we generate many events and in each of them there is only one neutron of 50 MeV energy hitting the detector with direction parallel to z axis and at a random point in the (x,y) plane.

For this part a command-base scorer is the same as the previous case. Of course we change the particle in the particle gun and in the DetectorConstruction the detector material to scilinator. Figure 3 shows the picture we get when we simulate 1million events of 1 neutron per event

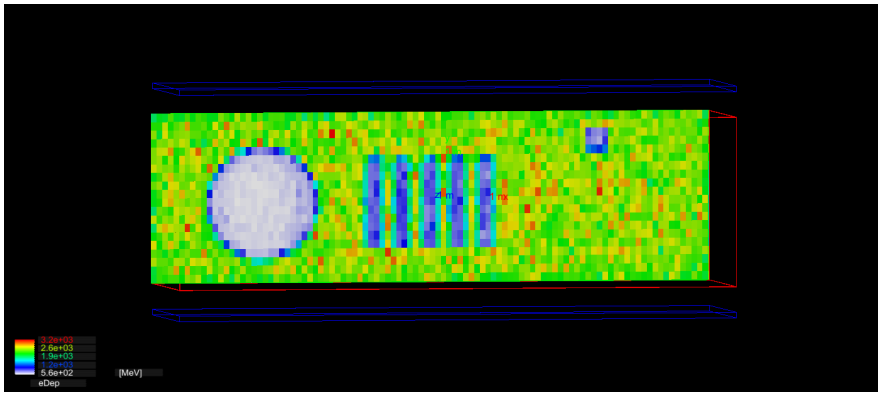

Figure 3: Picture for 50MeV neutrons. We generated 1 million events

Simulation with μ

In this case we want to use muons to inspect the interior. We know that muons scatter differently for each material. We are going to use their scattering angle to create the picture of the detector.

In the simulation, in each of the two detectors (one placed on top of the truck and the other on the bottom) we take two points : The prestep point and the post step point. The prestep point belongs to the current volume while the post step to the next. In the sensitive detector class, in the method we process the hits we can calculate the vector that is the difference between the post step and prestep point.This vector is stored as extra information about the hit. We should also mention that the prestep point is required to be on the boundary of the volume to make sure we get the very first hit.

By doing that for both of the detectors (top and bottom) we end up with two vectors. The next step is to use the definition of the inner product of two vectors to calculate the angle between them :

$$\vec{a}\ \vec{b} = |\vec{a}|\ |\vec{b}|\ cos\theta \Rightarrow \theta = cos^{-1}\frac{\vec{a}\ \vec{b}}{|\vec{a}|\ |\vec{b}|}$$

Figure 4 has the points in xz plane and the color. In the Z axis of this histogram is the number of muons. Figure 5 shows the points of the incoming muons and in the Z axis of the histogram we put the scattering angle. We see how the muons are scattered by the different materials in the container

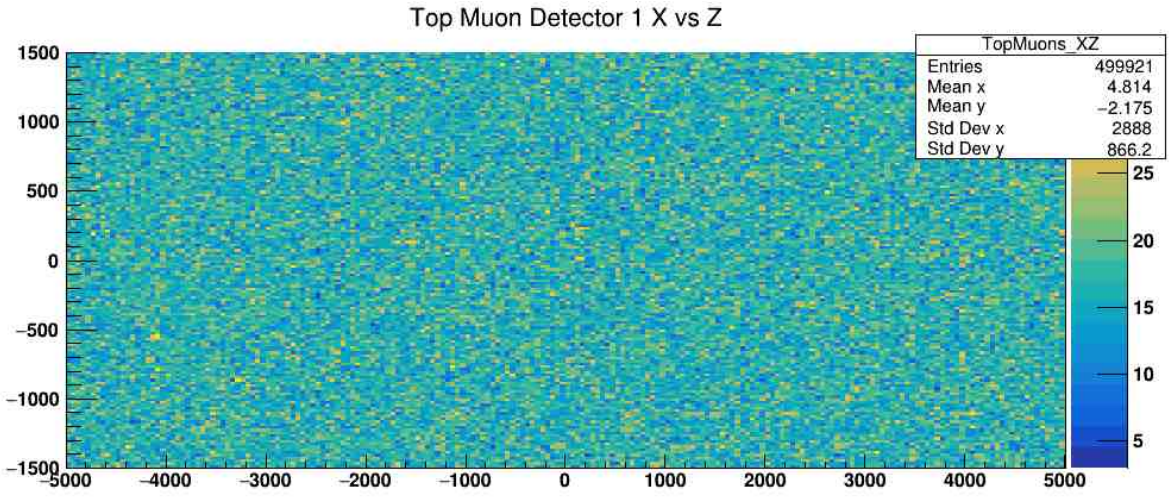

Figure 4: Picture for 4GeV muons. We generated 500000 events. Top Muon detector

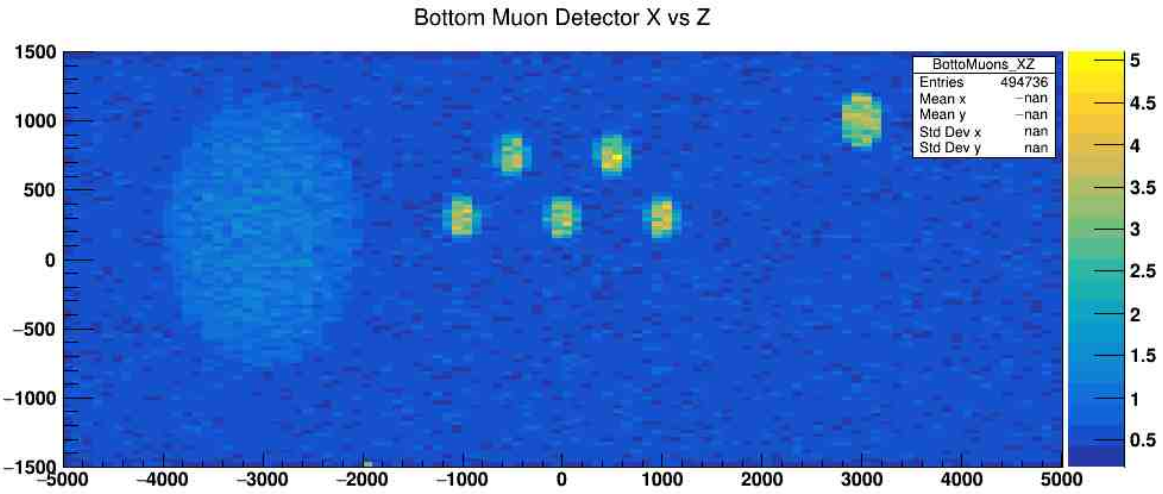

Figure 5: Picture for 4GeV muons. We generated 500000 events. Bottom Muon detector

Something in the calculation of the scattering angle was wrong and the second plot doesn't show any of the detectors interior.I noticed that I didn't impose the condition

if(step-¿GetPostStepPoint()-¿GetStepStatus() == fGeomBoundary)

So this might be the reason for this image.

main() function

The main function used for this project is the following. The commented parts were used for the first step. When muons are simulated the graphic environment consumes a lot of time. Also any printed messages were commented after making sure everything works to run with more events.

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
```

```
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
// $Id: example.cc 70284 2013-05-28 17:26:43Z perl $
//
/// \file example.cc
/// \brief Main program of the analysis/ example

#include "DetectorConstruction.hh"
#include "ActionInitialization.hh"

#ifdef G4MULTITHREADED
#include "G4MTRunManager.hh"
#else
#include "G4RunManager.hh"
#endif

#include "G4UImanager.hh"
#include "FTFP_BERT.hh"
#include "G4StepLimiterPhysics.hh"

#include "G4VisExecutive.hh"
#include "G4UIExecutive.hh"

#include "G4ScoringManager.hh"

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

int main(int argc,char** argv)
{
  //Detect interactive mode (if no argument) and define UI session
  // G4UIExecutive* ui = 0;
  // if ( argc == 1 ) { //No commands line argument
  //   //Let G4UIExecutive guess what is the best available UI
  //   ui = new G4UIExecutive(argc,argv);
  // }

  // Construct the default run manager
  // Note that if we have built G4 with support for Multi-threading we set
       it here
// #ifdef G4MULTITHREADED
//   G4MTRunManager* runManager = new G4MTRunManager;
```

```
//   //Set the default number of threads to be the number of available
    cores of the machine
//   //If not specified use 2 threads
//   runManager->SetNumberOfThreads( G4Threading::G4GetNumberOfCores() );
// #else
//   G4RunManager* runManager = new G4RunManager;
// #endif
  G4RunManager* runManager = new G4RunManager;

  // Activate UI-command base scorer
  // G4ScoringManager * scManager = G4ScoringManager::GetScoringManager();
  // scManager->SetVerboseLevel(1);

  // Mandatory user initialization classes

  //====================
  //The Geometry
  runManager->SetUserInitialization(new DetectorConstruction);

  //====================
  //The Physics
  G4VModularPhysicsList* physicsList = new FTFP_BERT;
  physicsList->RegisterPhysics(new G4StepLimiterPhysics());
  runManager->SetUserInitialization(physicsList);

  //====================
  // User action initialization
  runManager->SetUserInitialization(new ActionInitialization());


  runManager->Initialize();

  // start a run
  int numberOfEvent = 100000;
  runManager->BeamOn(numberOfEvent);

  // // Visualization manager construction
  // G4VisManager* visManager = new G4VisExecutive;
  // // G4VisExecutive can take a verbosity argument - see /vis/verbose
      guidance.
  // // G4VisManager* visManager = new G4VisExecutive("Quiet");
  // visManager->Initialize();

  // // Get the pointer to the User Interface manager
  // G4UImanager* UImanager = G4UImanager::GetUIpointer();

  // if (argc>1) {
  //   // execute an argument macro file if exist
  //   G4String command = "/control/execute ";
  //   G4String fileName = argv[1];
  //   UImanager->ApplyCommand(command+fileName);
```

```
  // }
  // else {
  //   //We have visualization, initialize defaults: look in init_vis.mac
     macro
  //   UImanager->ApplyCommand("/control/execute init_vis.mac");
  //   if (ui->IsGUI() ) {
  //     UImanager->ApplyCommand("/control/execute gui.mac");
  //     }
  //   ui->SessionStart();
  //   delete ui;
  // }
  // Job termination
  // Free the store: user actions, physics_list and detector_description
     are
  // owned and deleted by the run manager, so they should not be deleted
  // in the main() program !

  //delete visManager;
  delete runManager;

  return 0;


}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......
```

User defined Classes

The classes presented here are edited from the hands on code we worked on during the tutorial days. For this reason some of the names might be miss-leading.

Parametrisation Class

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
```

```
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
// $Id: CellParameterisation.cc 76474 2013-11-11 10:36:34Z gcosmo $
//
/// \file CellParameterisation.cc
/// \brief Implementation of the CellParameterisation class

#include "CellParameterisation.hh"

#include "G4VPhysicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4SystemOfUnits.hh"


//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

CellParameterisation::CellParameterisation()
: G4VPVParameterisation()
{
}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

CellParameterisation::~CellParameterisation()
{}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

void CellParameterisation::ComputeTransformation
(const G4int copyNo,G4VPhysicalVolume *physVol) const
{
  //G4int px = -3*m;

  G4double px =(copyNo-2.)/2;
  G4double pz = 0;
  (copyNo%2==0) ? pz =-0.2 : pz=0.25;
```

```
  physVol->SetTranslation(G4ThreeVector(px*m,0*m,pz*m));

  //rotate the barrels
  G4RotationMatrix* xRot = new G4RotationMatrix;
  xRot->rotateX(M_PI/2*rad);


  physVol->SetRotation(xRot);

}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......


// G4Material* CellParameterisation::ComputeMaterial(const G4int copyNo,
    G4VPhysicalVolume* physVol,const G4VTouchable* parentTouch)
// {
//   G4Material* water = G4Material::GetMaterial("G4_WATER");


//   G4Material* mat= G4Material::GetMaterial("G4_AIR");
//   return mat;
// }


   // material, sensitivity, visAtt
```

Detector Construction Class

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
```

```
// * use in resulting scientific publications, and indicate your *
// * acceptance of all terms of the Geant4 Software license.      *
// ****************************************************************
//
// $Id: DetectorConstruction.cc 77656 2013-11-27 08:52:57Z gcosmo $
//
/// \file DetectorConstruction.cc
/// \brief Implementation of the DetectorConstruction class

#include "DetectorConstruction.hh"
//#include "MagneticField.hh"
#include "CellParameterisation.hh"
//#include "HodoscopeSD.hh"
#include "DriftChamberSD.hh"
//#include "EmCalorimeterSD.hh"
//#include "HadCalorimeterSD.hh"

#include "G4FieldManager.hh"
#include "G4TransportationManager.hh"
#include "G4Mag_UsualEqRhs.hh"
#include "G4AutoDelete.hh"

#include "G4Material.hh"
#include "G4Element.hh"
#include "G4MaterialTable.hh"
#include "G4NistManager.hh"

#include "G4VSolid.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4Orb.hh"
#include "G4LogicalVolume.hh"
#include "G4VPhysicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVParameterised.hh"
#include "G4PVReplica.hh"
#include "G4UserLimits.hh"

#include "G4SDManager.hh"
#include "G4VSensitiveDetector.hh"
#include "G4RunManager.hh"
#include "G4GenericMessenger.hh"

#include "G4VisAttributes.hh"
#include "G4Colour.hh"

#include "G4ios.hh"
#include "G4SystemOfUnits.hh"

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......
```

```
DetectorConstruction::DetectorConstruction()
: G4VUserDetectorConstruction(),
  fMessenger(0),muonUPdetLogical(0),muonDOWNdetLogical(0),
  // fHodoscope1Logical(0),
  //fWirePlane1Logical(0),
  fVisAttributes()//,
  //fArmAngle(0.*deg), fArmRotation(0), fSecondArmPhys(0)

{
    fArmRotation = new G4RotationMatrix();
    fArmRotation->rotateY(fArmAngle);
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......

DetectorConstruction::~DetectorConstruction()
{
    delete fArmRotation;
    delete fMessenger;

    for (G4int i=0; i<G4int(fVisAttributes.size()); ++i)
    {
      delete fVisAttributes[i];
    }
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......

G4VPhysicalVolume* DetectorConstruction::Construct()
{
    // Construct materials
    ConstructMaterials();
    G4Material* air = G4Material::GetMaterial("G4_AIR");
    //G4Material* argonGas = G4Material::GetMaterial("G4_Ar");
    G4Material* aluminum = G4Material::GetMaterial("G4_Al");
    G4Material* concrete = G4Material::GetMaterial("G4_CONCRETE");
    G4Material* steel = G4Material::GetMaterial("G4_STAINLESS-STEEL");
    G4Material* scintillator = G4Material::GetMaterial("
       G4_PLASTIC_SC_VINYLTOLUENE");
    G4Material* my_CsI = G4Material::GetMaterial("CsI");
    G4Material* warhead = G4Material::GetMaterial("G4_U");
    G4bool checkOverlaps = true;


    // geometries
       ------------------------------------------------------------
```

```
// the truck volume  (world volume)
// make solid of alluminium and dig a hole of air


//create first a significantly bigger world volume
G4VSolid* worldSolid
  = new G4Box("worldBox",20.*m,6.*m,6.*m);
G4LogicalVolume* worldLogical
  = new G4LogicalVolume(worldSolid,air,"worldLogical");
G4VPhysicalVolume* worldPhysical
  = new G4PVPlacement(0,G4ThreeVector(),worldLogical,"worldPhysical"
     ,0,
                      false,0,checkOverlaps);

// Define a box that is our container, create physical volume and
   place it--> fill it with aluminium
// we will make a whole inside later --> color it BLUE
G4VSolid* containerWallsSolid
  = new G4Box("containerWallsBox",5*m,1.5*m,1.5*m);
G4LogicalVolume* containerWallsLogical
  = new G4LogicalVolume(containerWallsSolid,aluminum,"
     containerWallsLogical");
new G4PVPlacement(0,G4ThreeVector(0.0*m,0.0*m,0.0*m),
   containerWallsLogical,
                  "containerWallsPhysical",worldLogical,false,0,
                     checkOverlaps);

G4VSolid* containerSolid
  = new G4Box("containerBox",4.995*m,1.495*m,1.495*m);
G4LogicalVolume* containerLogical
  = new G4LogicalVolume(containerSolid,air,"containerLogical");
new G4PVPlacement(0,G4ThreeVector(0.0*m,0.0*m,0.0*m),containerLogical,
                  "containerPhysical",containerWallsLogical,false,0,
                     checkOverlaps);

G4VSolid* containersBox
  = new G4Box("barrelBox",1.3*m,1.*m,0.5*m);
G4LogicalVolume* barrelsbLogical
  = new G4LogicalVolume(containersBox,air,"barrelsbLogical");
new G4PVPlacement(0,G4ThreeVector(0.0*m,-0.1*m,0.5*m),barrelsbLogical,
                  "barrelbPhysical",containerLogical,false,0,
                     checkOverlaps);


G4VSolid* concSphere
  = new G4Orb("concreteSphere",1*m);
G4LogicalVolume* concSphereLogical
  = new G4LogicalVolume(concSphere,concrete,"concSphereLogical");
new G4PVPlacement(0,G4ThreeVector(-3.*m,-0.1*m,0.2*m),
   concSphereLogical,
                  "concSpherePhysical",containerLogical,false,0,
```

```
                        checkOverlaps);


//now place stuff inside the container
G4RotationMatrix* xRot = new G4RotationMatrix;
xRot->rotateX(M_PI/2*rad);



G4VSolid* warh
  = new G4Tubs("uranium tube",0.*m,0.2*m,0.2*m,0.0*deg,360.0*deg);
G4LogicalVolume* warheadLogical
  = new G4LogicalVolume(warh,warhead,"warheadLogical");
 new G4PVPlacement(xRot,G4ThreeVector(3.*m,1.*m,1.*m),warheadLogical,
                   "warPhysical",containerLogical,false,0,checkOverlaps
                      );


 G4VSolid* mybarrel
   = new G4Tubs("waterbarrels",0.*m,0.15*m,0.8*m,0.0*deg,360.0*deg);
 G4LogicalVolume* barrelLogical
   = new G4LogicalVolume(mybarrel,steel,"barrelLogical");


G4VPVParameterisation* objParam = new CellParameterisation();
new G4PVParameterised("cellPhysical",barrelLogical,barrelsbLogical,
   kXAxis,5,objParam);


// new G4PVPlacement(0,G4ThreeVector(0.5*m,-1.1*m,0.5*m),barrelLogical
   ,
//                   "barrelPhysical",containerLogical,false,0,
   checkOverlaps);



// create photon/neutron detector
// photon: CsI
// neutron : change material to scilinator

G4VSolid* gammadetSolid
  = new G4Box("gammadetBox",5*m,1.5*m,0.05*m);
G4LogicalVolume* gammadetLogical
  = new G4LogicalVolume(gammadetSolid,my_CsI,"gammadetLogical");
new G4PVPlacement(0,G4ThreeVector(0.0*m,0.0*m,1.6*m),gammadetLogical,
                   "containerPhysical",worldLogical,false,0,
                      checkOverlaps);

//create muon detectors
```

```
 G4VSolid* muonUPdetSolid
   = new G4Box("muonUPdetBox",5*m,0.05*m,1.5*m);
muonUPdetLogical
  = new G4LogicalVolume(muonUPdetSolid,my_CsI,"muonUPdetLogical");
new G4PVPlacement(0,G4ThreeVector(0*m,2.0*m,0.0*m),muonUPdetLogical,
                  "muonUPdetPhysical",worldLogical,false,0,
                     checkOverlaps);

G4VSolid* muonDOWNdetSolid
  = new G4Box("muonDOWNdetBox",5*m,0.05*m,1.5*m);
 muonDOWNdetLogical
   = new G4LogicalVolume(muonDOWNdetSolid,my_CsI,"muonDOWNdetLogical")
      ;
 new G4PVPlacement(0,G4ThreeVector(0*m,-2.0*m,0.0*m),
    muonDOWNdetLogical,
                   "muonDOWNdetPhysical",worldLogical,false,0,
                      checkOverlaps);


// visualization attributes
   ------------------------------------------------

G4VisAttributes* visAttributes = new G4VisAttributes(G4Colour
   (1.0,1.0,1.0));
visAttributes->SetVisibility(false);
worldLogical->SetVisAttributes(visAttributes);
fVisAttributes.push_back(visAttributes);

visAttributes = new G4VisAttributes(G4Colour(0.8888,0.0,0.0));
//visAttributes->SetVisibility(false);
containerLogical->SetVisAttributes(visAttributes);
fVisAttributes.push_back(visAttributes);

visAttributes = new G4VisAttributes(G4Colour(0.8888,0.0,0.0));
//    visAttributes->SetVisibility(false);
containerWallsLogical->SetVisAttributes(visAttributes);
fVisAttributes.push_back(visAttributes);

visAttributes = new G4VisAttributes(G4Colour(0.8888,0.0,0.8));
//    visAttributes->SetVisibility(false);
warheadLogical->SetVisAttributes(visAttributes);
fVisAttributes.push_back(visAttributes);

visAttributes = new G4VisAttributes(G4Colour(0.8888,0.0,0.8));
visAttributes->SetVisibility(false);
barrelsbLogical->SetVisAttributes(visAttributes);
fVisAttributes.push_back(visAttributes);


 visAttributes = new G4VisAttributes(G4Colour(0.88,0.88,0.0));
//    visAttributes->SetVisibility(false);
```

```
    barrelLogical->SetVisAttributes(visAttributes);
    fVisAttributes.push_back(visAttributes);

    visAttributes = new G4VisAttributes(G4Colour(0.0,0.9,0.0));
    gammadetLogical->SetVisAttributes(visAttributes);
    fVisAttributes.push_back(visAttributes);

    visAttributes = new G4VisAttributes(G4Colour(0.0,0.0,0.9));
    muonDOWNdetLogical->SetVisAttributes(visAttributes);
    muonUPdetLogical->SetVisAttributes(visAttributes);
    fVisAttributes.push_back(visAttributes);

    // return the world physical volume
       ----------------------------------------

    return worldPhysical;
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......

void DetectorConstruction::ConstructSDandField()
{
  // sensitive detectors
     ------------------------------------------------------
  G4SDManager* SDman = G4SDManager::GetSDMpointer();
  G4String SDname;

  G4VSensitiveDetector* UpMus = new DriftChamberSD(SDname="/UpMus");
  SDman->AddNewDetector(UpMus);
  muonUPdetLogical->SetSensitiveDetector(UpMus);

  G4VSensitiveDetector* DownMus = new DriftChamberSD(SDname="/DownMus");
  SDman->AddNewDetector(DownMus);
  muonDOWNdetLogical ->SetSensitiveDetector(DownMus);

}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......

void DetectorConstruction::ConstructMaterials()
{
    G4NistManager* nistManager = G4NistManager::Instance();

    // Air
    nistManager->FindOrBuildMaterial("G4_AIR");

    // Argon gas
    nistManager->FindOrBuildMaterial("G4_Ar");
```

```
    nistManager->FindOrBuildMaterial("G4_WATER");
   // Scintillator
    //(PolyVinylToluene, C_9H_10)
    nistManager->FindOrBuildMaterial("G4_PLASTIC_SC_VINYLTOLUENE");

    nistManager->FindOrBuildMaterial("G4_CONCRETE");
    nistManager->FindOrBuildMaterial("G4_U");
    // aluminium for the walls.

    nistManager->FindOrBuildMaterial("G4_Al");

    nistManager->FindOrBuildMaterial("G4_STAINLESS-STEEL");


    // CsI for gamma/muon detectors

    G4Element* el_i = new G4Element("Iodine","I", 53,126.9*g/mole);
    G4Element* el_cs = new G4Element("Cesium","Cs",55,132.9*g/mole);
    G4Material* mat_csi = new G4Material("CsI",4.51*g/cm3,2);
    mat_csi->AddElement(el_i,1);
    mat_csi->AddElement(el_cs,1);

    // Important: Never use a real gas with 0 density as materials.
    //            Physics processes requires density>0 and may give wrong
    //            results if they encounter 0 density material.
    //            Instead use one of the following methods if you need
    //            "vacuum" (e.g. a beam pipe internal)
    // Vacuum "Galactic"
    // nistManager->FindOrBuildMaterial("G4_Galactic");


    G4cout << G4endl << "The materials defined are : " << G4endl << G4endl
       ;
    G4cout << *(G4Material::GetMaterialTable()) << G4endl;
}


//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......
```

Sensitive Detector Class

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
```

```
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ******************************************************************
//
// $Id: DriftChamberSD.cc 76474 2013-11-11 10:36:34Z gcosmo $
//
/// \file DriftChamberSD.cc
/// \brief Implementation of the DriftChamber class

#include "DriftChamberSD.hh"
#include "DriftChamberHit.hh"

#include "G4HCofThisEvent.hh"
#include "G4TouchableHistory.hh"
#include "G4Track.hh"
#include "G4Step.hh"
#include "G4SDManager.hh"
#include "G4MuonMinus.hh"
#include "G4ios.hh"
#include "math.h"
//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

DriftChamberSD::DriftChamberSD(G4String name)
: G4VSensitiveDetector(name), fHitsCollection(0), fHCID(-1)
{
    collectionName.insert("driftChamberColl");
}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

DriftChamberSD::~DriftChamberSD()
{}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......

void DriftChamberSD::Initialize(G4HCofThisEvent* hce)
```

```
{
    fHitsCollection
      = new DriftChamberHitsCollection(SensitiveDetectorName,
          collectionName[0]);
    if (fHCID<0)
    { fHCID = G4SDManager::GetSDMpointer()->GetCollectionID(
       fHitsCollection); }
    hce->AddHitsCollection(fHCID,fHitsCollection);

}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo
    ......

G4bool DriftChamberSD::ProcessHits(G4Step* step, G4TouchableHistory*)
{

  G4double charge = step->GetTrack()->GetDefinition()->GetPDGCharge();

  if (charge==0.) return true;

  G4StepPoint* preStepPoint = step->GetPreStepPoint();

  //first step in the detecto

  G4TouchableHistory* touchable
    = (G4TouchableHistory*)(step->GetPreStepPoint()->GetTouchable());


  if ( step->GetTrack()->GetDefinition()==G4MuonMinus::MuonMinusDefinition
     () )  //check if muons
    if(preStepPoint->GetStepStatus() == fGeomBoundary) //check if the
       prestep is in the boundary

      {
        //  G4VPhysicalVolume* motherPhysical = touchable->GetVolume(); //
            mother
        //G4int copyNo = motherPhysical->GetCopyNo();
        // G4cout << "Particle ID ::  " << step->GetTrack()->GetTrackID()
        //        << " has a parent :: "<<step->GetTrack()->GetParentID()
          //      <<G4endl;
        G4ThreeVector worldPos = preStepPoint->GetPosition();
        G4ThreeVector localPos
          = touchable->GetHistory()->GetTopTransform().TransformPoint(
              worldPos);

        DriftChamberHit* hit = new DriftChamberHit();
        hit->SetWorldPos(worldPos);
        hit->SetLocalPos(localPos);
        hit->SetTime(preStepPoint->GetGlobalTime());
```

```
        G4ThreeVector worldPos2 = step->GetPostStepPoint()->GetPosition();
        G4ThreeVector localPos2
          = touchable->GetHistory()->GetTopTransform().TransformPoint(
              worldPos2);

        G4ThreeVector diff = localPos2-localPos ;

        hit->SetDirection(diff);
        // G4double angle = acos(numerator/denominator);
        fHitsCollection->insert(hit);
      }

  return true;
}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo
    ......
```

Scoring Macro

```
/run/initialize

#########################################
#
# define scoring mesh
#
/score/create/boxMesh gamma_mesh
#
#Create a mesh large as the box
/score/mesh/boxSize 5. 1.5 0.05 m
#Position it over the box
/score/mesh/translate/xyz 0. 0. 1.6 m
#mesh voxel size of 5 cm
/score/mesh/nBin 100 20 20
#
/score/quantity/energyDeposit eDep

/score/close
#
/score/list

/vis/disable
/run/verbose 1
/gun/particle gamma
/run/beamOn 1000000
/vis/enable

#########################################
#
# Dump scores to a file
```

```
#
/score/dumpQuantityToFile gamma_mesh eDep eDepGamma.txt

#/control/execute mydraw.mac
#/vis/ogl/printEPS
#
```