Exercise GEANT4 course, Lund 2018

Caterina Marcon, Particle Physics, caterina.marcon@hep.lu.se

October 8, 2018

1 Geometry

In Fig. 1 an overview of the geometry is shown: top and bottom muon detectors (orange), gamma/neutron detector (light blue) and container (purple). All the detectors are $10\times3\times0.1$ m^3, separated by a 10 cm gap from the $10\times3\times3$ m^3 container. Container walls are 0.5 cm thick, as requested.

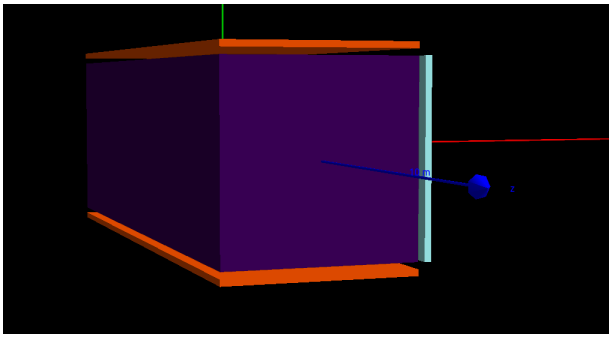

Figure 1: Container and detectors overview.

Inside the container, four types of objects are placed. In Fig. 2, the solid for a gold ingot is presented. It is defined as a G4Trd solid:

```
G4Trd("IngotSolid", 12.5*cm, 9.5*cm,3.5*cm,2.5*cm,2.*cm);
```

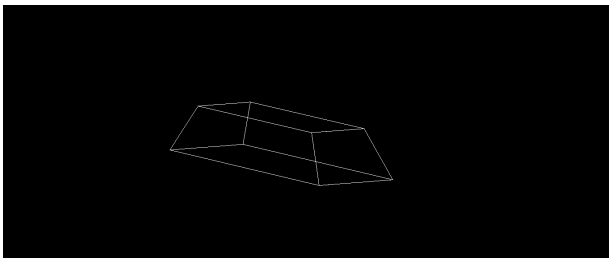

Figure 2: Single ingot.

The placement of 256 ingots is achieved by parameterised volumes. Placement is implemented as follows:

```
for (G4int copyNo=0;copyNo<256;copyNo++) {

    G4int column = copyNo / 16;
    G4int row = copyNo % 16;
    fYCell[copyNo] = row*4.1*cm;
    fZCell[copyNo] = 2.*m+column*7.5*cm;

}

fRot = new G4RotationMatrix();
fRot->rotateX(90.*deg);
physVol->SetRotation(fRot);
physVol->SetTranslation(G4ThreeVector(0., fYCell[copyNo], fZCell[copyNo]));
```

The resulting arrangement is sown in Fig. 3

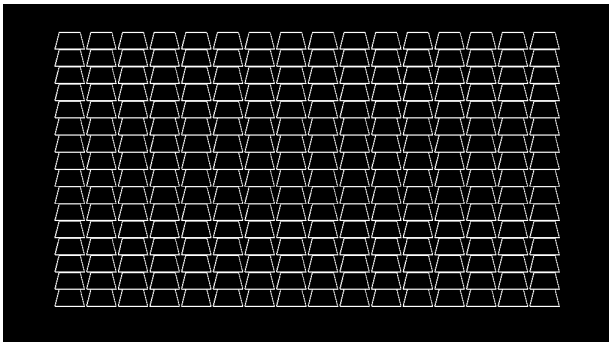

Figure 3: Wall of bricks.

A tungsten cube is positioned at the origin (0,0,0). On the left of Fig. 4, a G4Polycone, defined below, is used to approximate the shape of a human body.

```
G4double z[6] = {0.*cm,89.*cm,90.*cm,149.*cm,150.*cm,180.*cm};
G4double rin[6] = {0.*cm,0.*cm,0.*cm,0.*cm,0.*cm,0.*cm};
G4double rout[6] = {20.*cm,20.*cm,35.*cm,35.*cm,12.5*cm,12.5*cm};
fHuman = new G4Polycone("HumanSolid", 0., 360.*deg,6,z,rin,rout);
```

For this particular solid, an average material composition is selected:

```
// MATERIAL (From src/DetectorConstruction.cc)
G4Material *Body_01 = nistManager->FindOrBuildMaterial("G4_ADIPOSE_TISSUE_ICRP");
G4Material *Body_02 = nistManager->FindOrBuildMaterial("G4_BLOOD_ICRP");
G4Material *Body_03 = nistManager->FindOrBuildMaterial("G4_BONE_COMPACT_ICRU");
G4Material *Body_04 = nistManager->FindOrBuildMaterial("G4_BONE_CORTICAL_ICRP");
G4Material *Body_05 = nistManager->FindOrBuildMaterial("G4_BRAIN_ICRP");
G4Material *Body_06 = nistManager->FindOrBuildMaterial("G4_MUSCLE_SKELETAL_ICRP");
G4Material *Body_07 = nistManager->FindOrBuildMaterial("G4_MUSCLE_STRIATED_ICRU");
G4Material *Body_08 = nistManager->FindOrBuildMaterial("G4_MUSCLE_WITH_SUCROSE");
G4Material *Body_09 = nistManager->FindOrBuildMaterial("G4_MUSCLE_WITHOUT_SUCROSE");
G4Material *Body_10 = nistManager->FindOrBuildMaterial("G4_SKIN_ICRP");

// Approximate weight fractions are used to compose HBody
G4Material *HomogeneousBody = new G4Material("HBody",985*kg/m3,10);
HomogeneousBody->AddMaterial(Body_01,20.*perCent);
HomogeneousBody->AddMaterial(Body_02,7.*perCent);
HomogeneousBody->AddMaterial(Body_03,7.5*perCent);
HomogeneousBody->AddMaterial(Body_04,7.5*perCent);
```

```
HomogeneousBody->AddMaterial(Body_05,2.*perCent);
HomogeneousBody->AddMaterial(Body_06,10.*perCent);
HomogeneousBody->AddMaterial(Body_07,10.*perCent);
HomogeneousBody->AddMaterial(Body_08,10.*perCent);
HomogeneousBody->AddMaterial(Body_09,10.*perCent);
HomogeneousBody->AddMaterial(Body_10,15.*perCent);
```

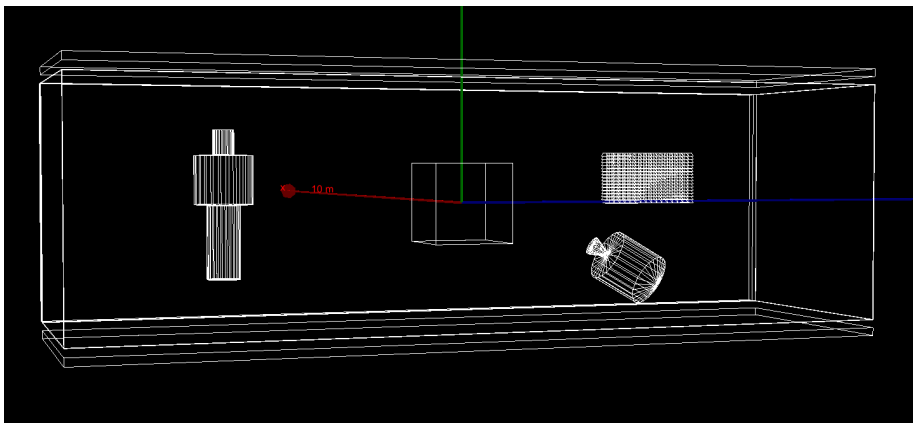

Figure 4: Overview of the four objects in the container.

Lastly, on the right side of Fig. 4, a model of a nuclear bomb is placed. The solid is entirely made of uranium and defined as a polycone in a similar way as for the human body:

```
G4double zLB[5] = {0., 25.*cm, 25.1*cm, 95.*cm, 110.*cm};
G4double rinLB[5] = {0.*cm,0.*cm,0.*cm,0.*cm,0.*cm};
G4double routLB[5] = {12.5*cm, 2.5*cm, 35.*cm, 35.*cm, 0.};

fLittleBoy = new G4Polycone("LittleBoySolid",0., 360.*deg, 6, zLB, rinLB, routLB);
```

2 Source code of main() with comments

```
#include "DetectorConstruction.hh" // Defining the geometry of the container,
                                   // the side gamma/neutron detector and the
                                   // top and bottom muon detectors.
                                   // The method ConstructSDandField(), which is
                                   // called by each thread, defines the sensitive
                                   // detectors (MuonSD class) for top and bottom
                                   // muon plates, registers them to the SD manager
                                   // and assigns them to the corresponding logical
                                   // volumecorresponding logical
                                   // volumes

#include "ActionInitialization.hh" // Instance of RunAction is created here to allow
                                   // the definition of the output histograms for muons
                                   // Interface to ROOT is selected in the Analysis.hh
                                   // header file and accessed via the AnalysisManager

// Allowing multithread, if available
```

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

#include "G4UImanager.hh"

// Including the header files for the physics list
#include "FTFP_BERT_HP.hh"
#include "G4StepLimiterPhysics.hh"

// Headers taking care of visualization and user interface management
#include "G4VisExecutive.hh"
#include "G4UIExecutive.hh"

// Enabling the use of command-based scorers (used for gammas and neutrons)
#include "G4ScoringManager.hh"

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 (i.e. only
                           // executable name is passed to main)
                           // 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

        G4ScoringManager * scManager = G4ScoringManager::GetScoringManager();
        scManager->SetVerboseLevel(1);

        //====================
        // The Geometry
        // Instance of the DetectorConstruction class is created
        // and assigned to the RunManager, which will take care of
        // calling the Construct() and ConstructSDandField() methods
        runManager->SetUserInitialization(new DetectorConstruction);


        //====================
        // The Physics
        // FTFP_BERT_HP is used, as requested, and assigned to the
        // RunManager as well
        G4VModularPhysicsList* physicsList = new FTFP_BERT_HP;
```

```
        //physicsList->RegisterPhysics(new G4StepLimiterPhysics());
        runManager->SetUserInitialization(physicsList);

        //====================
        // User action initialization
        // Instance of this class results in the initialization of the
        // RunAction, used in this code to create all the histograms
        // to store results (for muon)
        runManager->SetUserInitialization(new ActionInitialization());

        // 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 {

                // Initialize visualization defaults from init_vis.mac
                UImanager->ApplyCommand("/control/execute init_vis.mac");

                if (ui->IsGUI() ) {

                        UImanager->ApplyCommand("/control/execute gui.mac");
                }

                ui->SessionStart();
                delete ui;

        }

        // Job termination. RunManager will take care of deleting its pointers,
        // no need to care about them.

        delete visManager;
        delete runManager;

        return 0;

}
```

3 Simulation with Gammas

First run is performed with gammas as the source particle. Some sample interactions are shown in Fig. 5. In Fig. 6 the resulting imaging after 5×10^5 generated gammas is shown. Due to its small density, the human body material is not efficiently stopping the incoming radiation. This results in a blurred image.

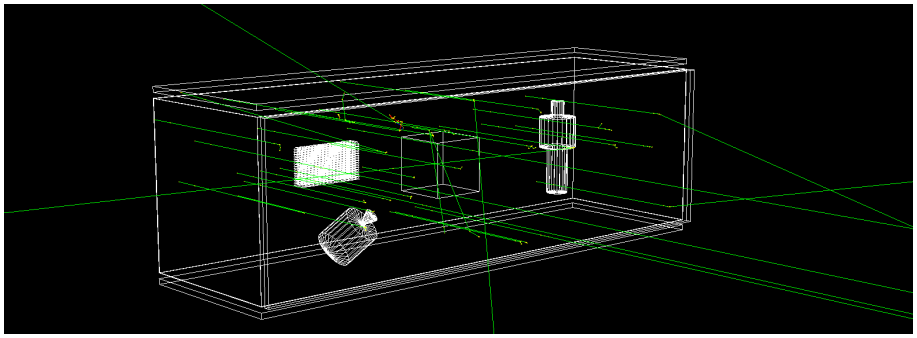

Figure 5: 30 interactions with gammas as the source particle.

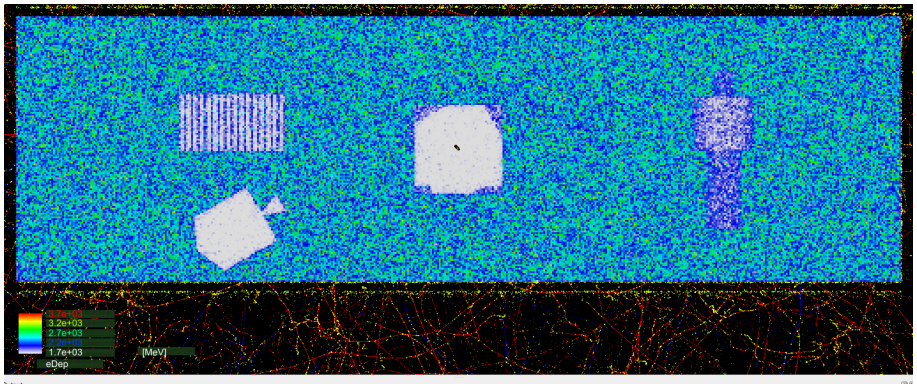

Figure 6: $5{\times}10^5$ interactions with gammas as the source particle.

4 Simulation with neutrons

For the simulation with neutrons, the material of the detector is set to a plastic scitillator (G4_PLASTIC_SC_VINYLTOLUENE) instead of CsI. In this case, the material of the human model, being a low density material, is a good neutron absorber; so the image is sharper than the previous case. In Fig. 7, 30 interactions of neutrons are shown and in Fig. 8 the resulting image obtained with 5×10^5 source neutrons is presented.

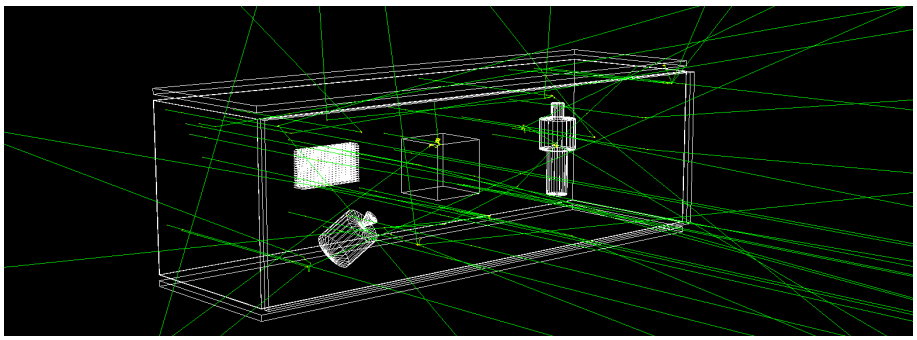

Figure 7: 30 interactions with neutrons as the source particle.

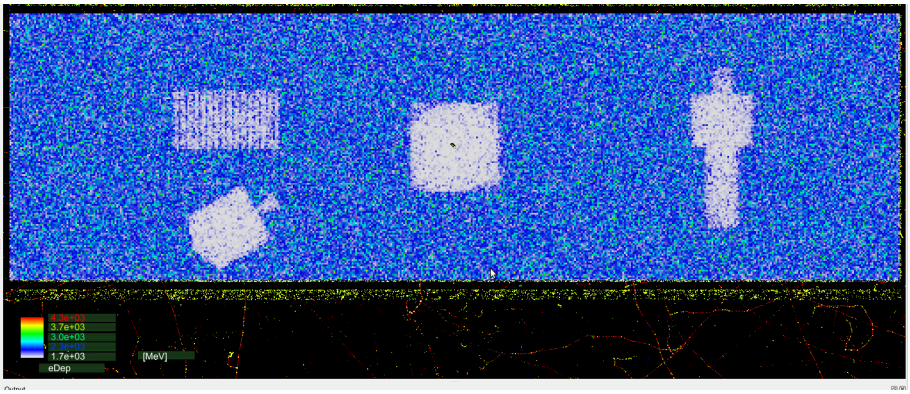

Figure 8: 5×10^5 interactions with neutrons as the source particle.

5 Simulation with μ-

For the muon simulation, source particles are originated from the top of the container (Fig. 9, red tracks) and their path is mostly straight through the container volume. High-Z materials are expected to be good scatterers, unlike the low density material of the human model. This results (Fig. 10) in the human model being completely transparent to muons.

The reconstructed image presented in Fig. 10 is obtained by recording in a ROOT histogram the incident position of the muons in the top Sensitive Detector together with the scattering angle associated to the track. The angle

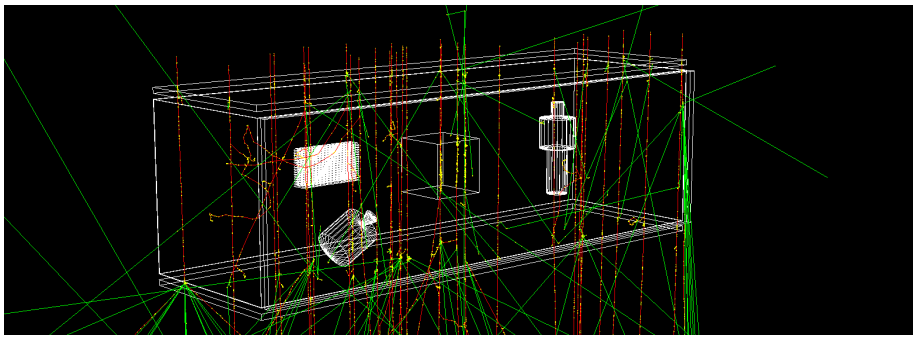

Figure 9: 30 interactions with muons as the source particle.

is calculated taking into account the first interaction with the top detector and the first interaction with the bottom detector.

The first interaction is identified by checking whether this condition is verified or not:

```
step->GetPreStepPoint()->GetStepStatus() == fGeomBoundary
```

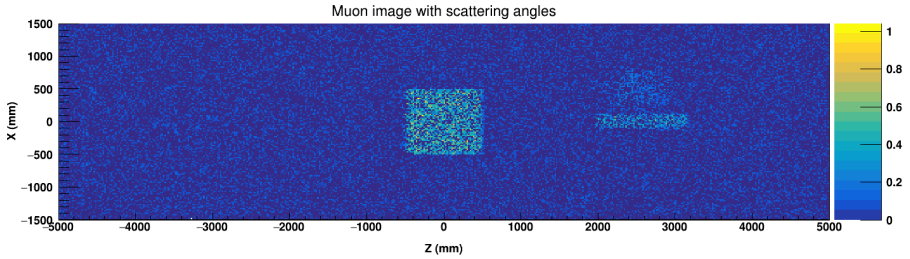

Figure 10: $5{\times}10^5$ interactions with neutrons as the source particle.