Geant4 Simulation

Xiandong Zhao

Department of Physics

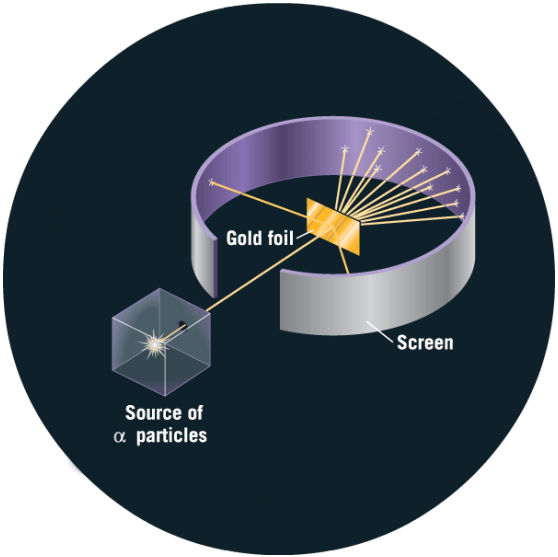
Southern Methodist University

Jan. 25, 2018

Introduction

- Motivation
- Detector Construction
- Primary Generator Action
- Main Function
- Visualization Setting File
- Output & Next Step

Motivation: Rutherford Scattering Experiment

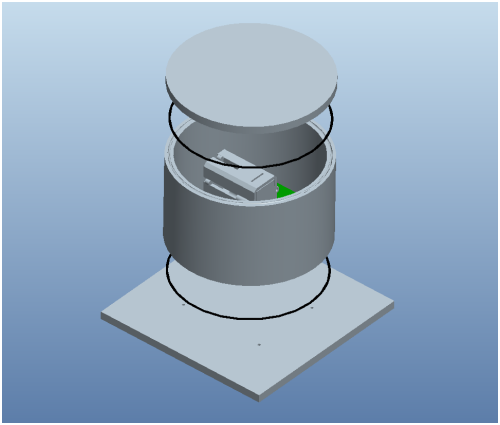


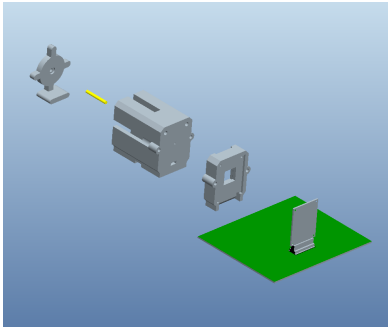
Schematic of the rutherford scattering experiment.

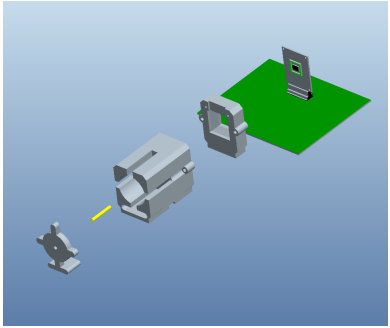
Details of our rutherford experiment:

- Source: ∼1 Ci, Am-241
- Alignment System: 1 millimeter diameter
- Gold foil: Gold leaf with 0.1∼0.125 micrometers
- Sensor: 4×7 millimeters, 5.3 micrometer/pixel

Motivation: Rutherford Scattering Experiment





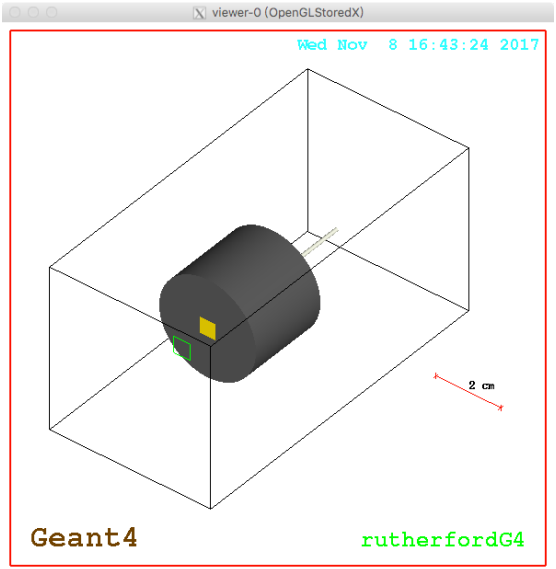


- Design a chamber to prove vacuum environment
- Design the alignment system

Detector Construction: Alignment System

Simulate alignment system, in my "DetectorConstruction" class:

- Collimator: stainless steel pipe
- Collimator frame: plastic
- Target: gold foil with 1 micrometer thickness
- Sensor: 4×7 millimeters area



Schematic of the alignment system in Geant4 simulation.

Detector Construction: World Volume

Material:

- G4Material* Vacuo = new
  G4Material("Galactic",z,a,density,kStateGas,temperature,pressure);

Volume:

- G4Box* solidWorld = new
  G4Box("World",world_sizeXY,world_sizeXY,world_sizeZ);
- G4LogicalVolume* logicWorld = new
  G4LogicalVolume(solidWorld,Vacuo,"World");
- G4VPhysicalVolume* physWorld = new
  G4PVPlacement(G4ThreeVector(),logicalWorld,"World");

Detector Construction: Collimator, the Stainless Steel Pipe

Material:

- G4Material* Iron = new G4Material("Iron",z,a,density);

Volume:

- G4Tubs* solidTube_01 = new G4Tubs("Tube_01",tube01_Rmin,tube01_Rmax,tube01_Dz/2.0,0.0,twopi);
- G4LogicalVolume* logicTube_01 = new G4LogicalVolume(solidTube_01,Iron,"Tube_01");
- new G4PVPlacement(G4ThreeVector(0.*mm,0.*mm,-17.5*mm),logicalTube_01,"Tube_01",logicWorld);

Detector Construction: Collimator Frame, Plastic

Material:

- G4Element* elH = new G4Element(“Hydrogen”,“H”,z,a);
- G4Element* elC = new G4Element(“Carbon”,“C”,z,a);
- G4Element* elO = new G4Element(“Oxygen”,“O”,z,a);
- G4int ncomponents = 3;
- G4int natoms;

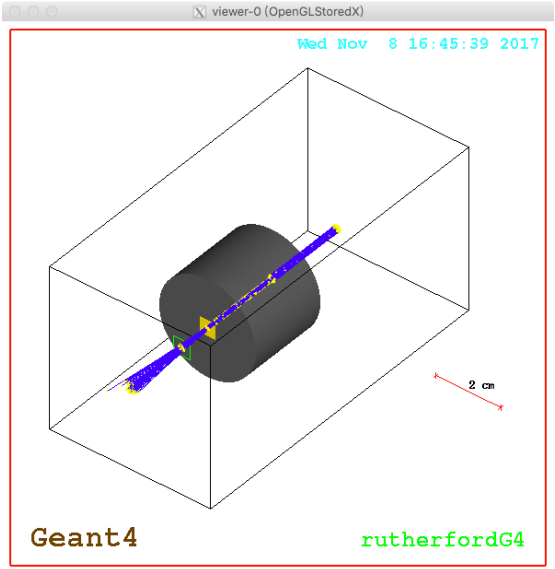
- G4Material* PLA = new G4Material(“Polylatic”,density,ncomponents);
- PLA->AddElement(elC, natoms=3);
- PLA->AddElement(elH, natoms=4);
- PLA->AddElement(elO, natoms=2);

Volume: Set as a tube, bundling the collimator pipe.

Primary Generator Action: α Source Simulation

Add the simulation for α Source, in my "PrimaryGeneratorAction" class:

- 1K α particles with rough selection
- 52 α particles hit the gold foil
- 51 α particles collected by sensor



Import α source in Geant4.

Primary Generator Action: Set Particle Gun

Set α particle:

- G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
- G4String particleName = "alpha";
- G4ParticleDefinition* particle = particleTable->FindParticle(particleName);

Set particle gun:

- G4int n_particle = 1;
- G4ParticleGun* fParticleGun = new G4ParticleGun(n_particle);
- fParticleGun->SetParticleDefinition(particle);

- fParticleGun->SetParticleEnergy(5.488*MeV);
- fParticleGun->SetParticlePosition(G4ThreeVector(posX*mm,posY*mm,-30.*mm));
- fParticleGun->SetParticleMomentumDirection(G4ThreeVector(vx,vy,vz));

Main Function

Main function is the *.cc file in the root directory of your codes. Set random engine:

- G4Random::setTheEngine(new CLHEP::RanecuEngine);
- Geant4 has several random algorithm can be chosen. So, you can choose different random "engines" and different random "distributions" to do your simulation.

Set physics list:

- G4VModularPhysicsList* physicsList = new QBBC;
- There are several available physics processes in Geant4, users can decide how much detail in the physics modeling is needed. You can construct your own physics list, or use the offered physics list.

Visualization Setting File

Visualization setting file (macro file) is the *.mac file in the directory you want to run the Geant4 simulation, vis.mac is the default macro file when you run the simulation directly. Every sentence in macro file should be a built-in command (except the comment). Usage of this file:

- Opens an OpenGL viewer to show your detector construction and trajectories of particles;
- Set a specify view angle and specify style for your detector construction.

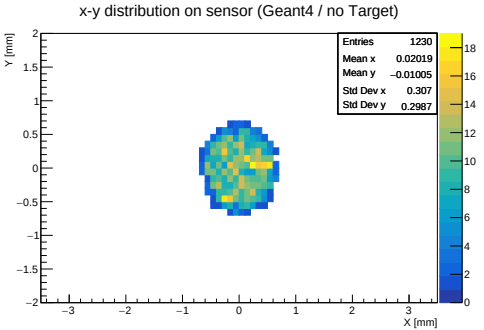
Built-in commands:

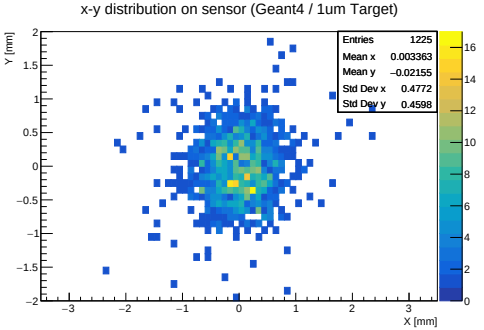
- Interactively via a Graphical User Interface - GUI
- In a macro file via /control/execute <command>
- You can also use the command in C++ code with the ApplyCommand function of GUImanager

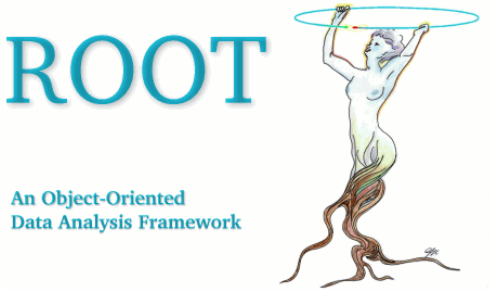
Output & Next Step

Generate more events in this simulation:

- 20K α particles with rough selection
- 1230 α particles hit the gold foil
- 1225 α particles collected by sensor







ROOT Logo (old version)

ROOT Logo (new version)

ROOT is a very useful tool in particle physics analysis:

- An object-oriented program and library developed by CERN
- Almost all the plots in the published article of ATLAS experiment are generated by ROOT.
- Please find more details about ROOT in its website: https://root.cern.ch/