Primary Particle

山下智弘
JST CREST/神戸大学
Borrowing especially from presentations of M. Asai(SLAC)
Geant4 Tutorial @ Japan 2007
17-19 Oct, 2007 @ RCNS, based on Geant4 9.0.p01

Outline

I. G4VPrimaryGeneratorAction class
II. Particle gun
III. General particle source
IV. Using Particle gun

I. G4VPrimaryGeneratorAction class

- This class is one of mandatory user classes
  - control the generation of primaries.
  - This class itself should NOT generate primaries
    - invoke GeneratePrimaryVertex() method of primary generator(s) to make primaries.
- Constructor of G4VPrimaryGeneratorAction
  - Instantiate primary generator(s))
  - Set default values to it(them)
- GeneratePrimaries() method
  - Randomize particle-by-particle value(s)
  - Set these values to primary generator(s)
    - Never use hard-coded UI commands
  - Invoke GeneratePrimaryVertex() method of primary generator(s)

I. G4VPrimaryGeneratorAction class
1. Primary vertices and particles

- Primary vertices and primary particles are stored in G4Event in advance to processing an event.
  - G4PrimaryVertex and G4PrimaryParticle classes
    - These classes don't have any dependency to G4ParticleDefinition nor G4Track.
  - Capability of bookkeeping decay chains
    - Primary particles may not necessarily be particles which can be tracked by Geant4.
- Geant4 provides some concrete implementations of G4VPrimaryGenerrator.
  - G4ParticleGun
  - G4HEPEvtIInterface, G4HEPMCInterface
  - G4GeneralParticleSource

II. Particle gun

- Concrete implementations of G4VPrimaryGenerator
  - A good example for experiment-specific primary generator implementation
- It shoots one primary particle
  - a certain energy from a certain point at a certain time to a certain direction.
  - Various set methods are available
  - Intercoms commands are also available for setting initial values
- One of most frequently asked questions is :
  - I want “particle shotgun”, “particle machinegun”, etc.
- Instead of implementing such a fancy weapon, in your implementation of UserPrimaryGeneratorAction, you can
  - Shoot random numbers in arbitrary distribution
  - Use set methods of G4PartticleGun
  - Use G4PartticleGun as many times as you want
  - Use any other primary generators as many times as you want to make overlapping events

III. GenerralParticleSource

- A concrete implementation of G4VPrimaryGenerrator
- Suitable especially to space applications
- Specifically, it allows the specifications of the spectral, spatial and angular distribution of the primary source particles.
- Implementation

```
MyPrimaryGeneratorAction::MyPrimaryGeneratorAction()
{ generator = new G4GeneralParticleSource; }
void MyPrimaryGeneratorAction::
                GeneratePrimaries(G4Event* anEvent)
{ generator->GeneratePrimaryVertex(anEvent); }
```

- Detailed description
  - http://reat.space.qinetiq.com/gps/

IV. Using Particle gun
1. PrimaryGeneratorActtion

```
class G4ParticleGun;
class G4Event;

class MyPrimaryGeneratorAction : public G4VUserPrimaryGeneratorAction
{
public:
    MyPrimaryGeneratorAction();
   ~MyPrimaryGeneratorAction();

public:
    void GeneratePrimaries(G4Event*);

private:
    G4ParticleGun* particleGun;
};
```

Inherit G4VUserPrimaryGeneratorAction

User have to define this method

Use G4ParticleGun as primary generator

IV. Using Particle gun
1. PrimaryGeneratorActtion *contd*.

```
MyPrimaryGeneratorAction::MyPrimaryGeneratorAction()
{
    G4int n_particle = 1;
    particleGun = new G4ParticleGun(n_particle);

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

    particleGun->SetParticleDefinition(particle);
    particleGun->SetParticleMomentumDirection(G4ThreeVector(0., 0., -1.));
    particleGun->SetParticleEnergy(150.0*MeV);
    G4double position = 0.0*m;
    particleGun->SetParticlePosition(
            G4ThreeVector(0.*cm, 0.*cm, position) );
}
```

Get proton definition

Set default particle, its energy, direction, and vertex

IV. Using Particle gun
1. PrimaryGeneratorActtion *contd*.

```
MyPrimaryGeneratorAction::~MyPrimaryGeneratorAction()
{
   delete particleGun;
}

void MyPrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
   particleGun->GeneratePrimaryVertex(anEvent);
}
```

IV. Using Particle gun
2. Particle gun commands(1)

- In G4UIterminal,

  Idle> ls /gun

  shows list of commands

Commands :
List * List available particles.
particle * Set particle to be generated.
direction * Set momentum direction.
energy * Set kinetic energy.
position * Set starting position of the particle.
time * Set initial time of the particle.
polarization * Set polarization.
number * Set number of particles to be generated.
ion * Set properties of ion to be generated.

IV. Using Particle gun
2. Particle gun commands(2)

```
Idle> help energy

Command /gun/energy
Guidance :
Set kinetic energy.

Parameter : Energy
 Parameter type  : d
 Omittable       : True
 Default value   : taken from the current value

Parameter : Unit
 Parameter type  : s
 Omittable       : True
 Default value   : GeV
 Candidates      : eV keV MeV GeV TeV PeV J electronvolt kiloelectronvolt
     megaelectronvolt gigaelectronvolt teraelectronvolt petaelectronvolt joule

Idle>
```

IV. Using Particle gun
2. Particle gun commands(3)

/gun/particle proton
/gun/energy 150 MeV
/gun/position 0 0 0 m
/gun/direction 0 0 -1

- Save a macrofile with 5 liens above, as *gunProton150.mac*
- In Geant4 UIterminal (or in another macro file)

  Idle> /control/execute macros/gunProton150.mac

  will set the particle gun to shoot proton with the energy from the position to the direction in the macro file

IV. Using Particle gun
2. Randomize particle energy

```
…
#include "Randomize.hh“
…
void MyPrimaryGeneratorAction::GeneratePrimaries(
                    G4Event* anEvent) {
   G4double kinE = 150*MeV;
   G4double sigma = 10*MeV;
   kinE = G4RandGauss::shoot(kinE, sigma);
   particleGun->SetParticleEnergy(kinE);
   particleGun->GeneratePrimaryVertex(anEvent);
}
```

Randomize the energy with gaussian

- Other quantities are also able to be randomized through Set method.