Quick Intro to

Particle source in Geant4

Karim Laihem

Geant4 Training event – Calorimetry in HEP

DESY Zeuthen 10 -13 May 2011

Refrence talk: Makoto Asai (SLAC)

Contents

- Lets have a look at G4VUserPrimaryGeneratorAction
- Built-in primary particle generators
  - Particle gun
  - General particle source
- Particle Gun or General Particle Source?

```
PrimaryGeneratorAction::PrimaryGeneratorAction(DetectorConstruction* myDC)
:myDetector(myDC)
{
  G4int n_particle = 1;
  G4ParticleGun*  particleGun = new G4ParticleGun(n_particle);

// default particle
  G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
  G4ParticleDefinition* particle = particleTable->FindParticle("e-");
// gun setings
  particleGun->SetParticleDefinition(particle);
  particleGun->SetParticleMomentumDirection(G4ThreeVector(0.,0.,1.));
  particleGun->SetParticlePosition(G4ThreeVector(0.*cm,0.*cm,-100.*cm));
  particleGun->SetParticleEnergy(10.0*GeV);
}
```

const

Init

//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....

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

~dest

//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....

```
void PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{


  particleGun->GeneratePrimaryVertex(anEvent);
}
```

Beam On

G4ParticleGun Built-in primary particle generators

- **Concrete implementations of G4VPrimaryGenerator**

  It shoots one primary particle of a certain energy from a certain point at a certain time to a certain direction.

  (a complete set of function is available)

- **UI commands are also available for setting initial values**

/gun/List	List available particles
/gun/particle	Set particle to be generated
/gun/direction	Set momentum direction
/gun/energy	Set kinetic energy
/gun/momentum	Set momentum
/gun/momentumAmp	Set absolute value of momentum
/gun/position	Set starting position of the particle
/gun/time	Set initial time of the particle
/gun/polarization	Set polarization
/gun/number	Set number of particles to be generated (per event)
/gun/ion	Set properties of ion to be generated [usage] /gun/ion Z A Q

```
PrimaryGeneratorAction::PrimaryGeneratorAction(DetectorConstruction* myDC)
:myDetector(myDC)
{
  G4int n_particle = 1;
  G4ParticleGun*  particleGun = new G4ParticleGun(n_particle);

// default particle
  G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
  G4ParticleDefinition* particle = particleTable->FindParticle("e-");
// gun setings
  particleGun->SetParticleDefinition(particle);
  particleGun->SetParticleMomentumDirection(G4ThreeVector(0.,0.,1.));
  particleGun->SetParticlePosition(G4ThreeVector(0.*cm,0.*cm,-100.*cm));
  particleGun->SetParticleEnergy(10.0*GeV);
}
```

const — Init

```
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....
```

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

~dest

```
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....
```

```
void PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{

  particleGun->SetParticleDefinition(particle=„gamma“);
  particleGun->SetParticleEnergy(10.0*GeV);

  particleGun->GeneratePrimaryVertex(anEvent);
}
```

Beam On

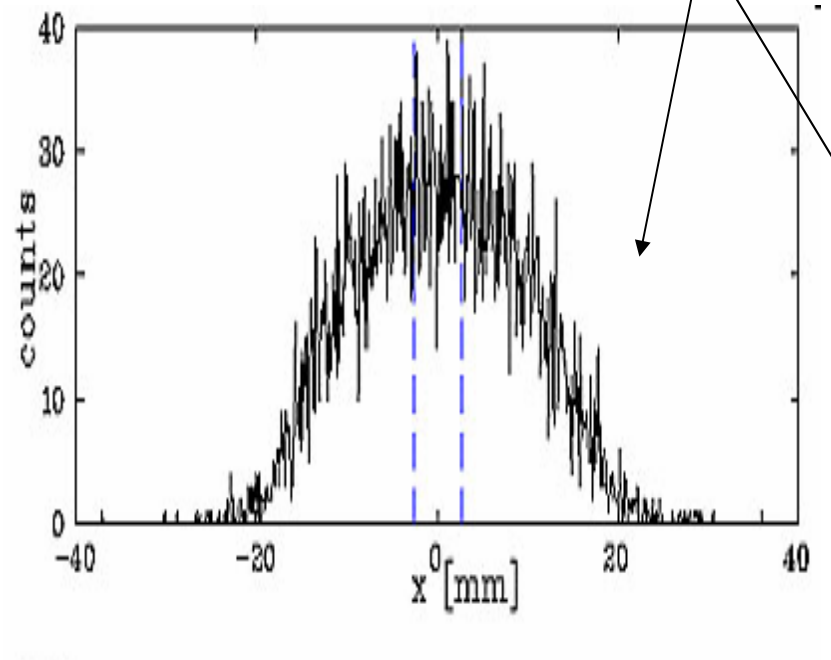
G4ParticleGun

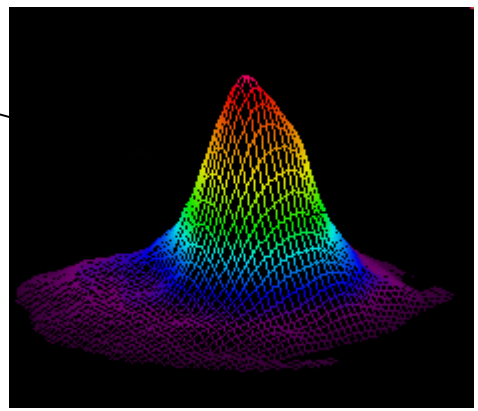
- It shoots one primary particle of a certain energy from a certain point at a certain time to a certain direction.
  - Various set methods are available
  - Intercoms commands are also available for setting initial values
- One of most frequently asked questions is :

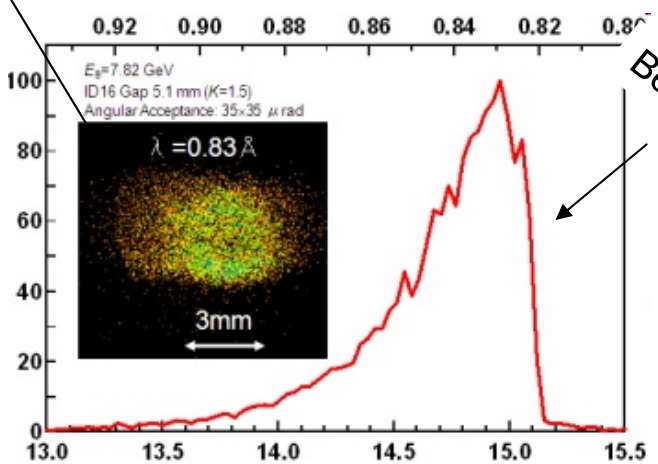
  I want “particle shotgun”, “particle machinegun”, etc.
- Instead of implementing such a fancy weapon, in your implementation of UserPrimaryGeneratorAction, you can
  - Shoot random numbers in arbitrary distribution
  - Use set methods of G4ParticleGun
  - Use G4ParticleGun as many times as you want

Some real situation

Beam Profile







Beam Energy

G4VUserPrimaryGeneratorAction (example of settings Randomization)

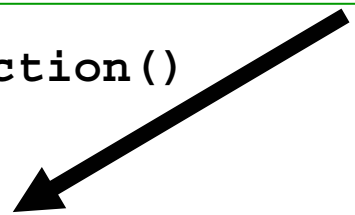
```
void T01PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{ G4ParticleDefinition* particle;
  G4int i = (int)(5.*G4UniformRand());

 switch(i)
  { case 0: particle = positron; break; ... }
  particleGun->SetParticleDefinition(particle);

  G4double pp = momentum+(G4UniformRand()-0.5)*sigmaMomentum;
  G4double mass = particle->GetPDGMass();
  G4double Ekin = sqrt(pp*pp+mass*mass)-mass;

  particleGun->SetParticleEnergy(Ekin);

  G4double angle = (G4UniformRand()-0.5)*sigmaAngle;
  particleGun->SetParticleMomentumDirection
           (G4ThreeVector(sin(angle),0.,cos(angle)));

  particleGun->GeneratePrimaryVertex(anEvent);
}
```

- You can repeat this for generating more than one primary particles.

G4GeneralParticleSource

- A concrete implementation of G4VPrimaryGenerator

G4ParticleGun* particleGun = new G4ParticleGun(n_particle);



```
PrimaryGeneratorAction::PrimaryGeneratorAction()
{
generator = new G4GeneralParticleSource;
-------------
---------------
}
```

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

- Detailed description

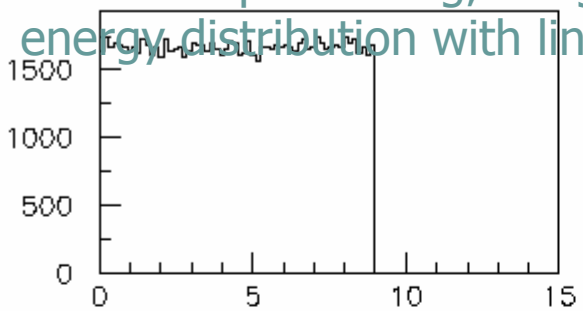
  http://reat.space.qinetiq.com/gps/

Square

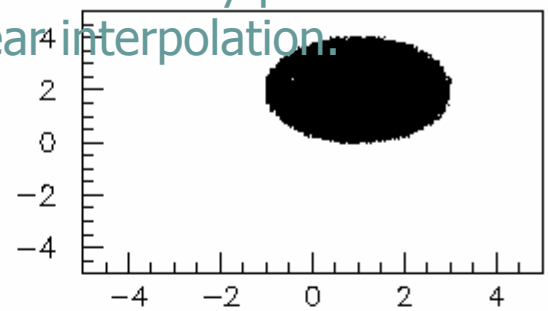
Spheri

Cylindri
energy

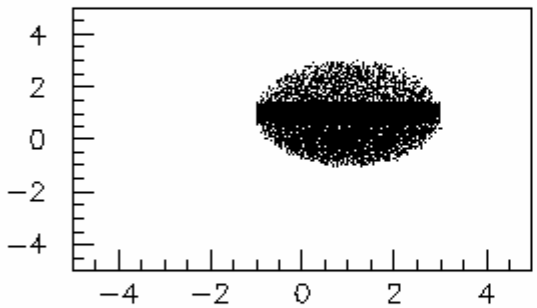
Spherical volume with z biasing, isotropic radiation with theta and phi biasing, integral arbitrary point-wise energy distribution with linear interpolation.



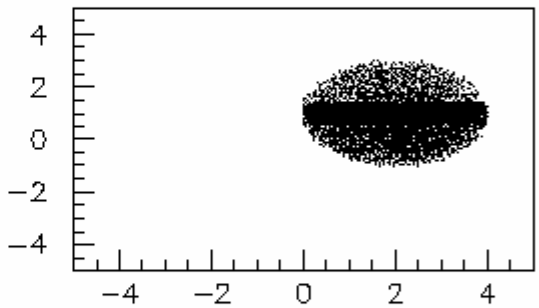
Source Energy Spectrum



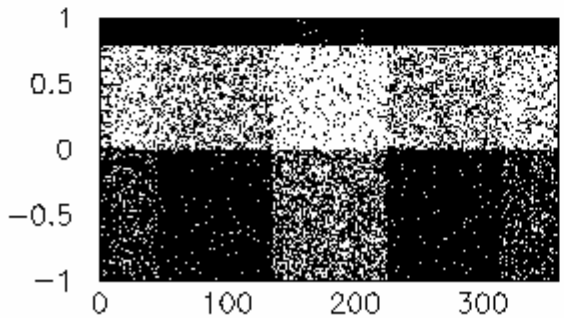
Source X–Y distribution



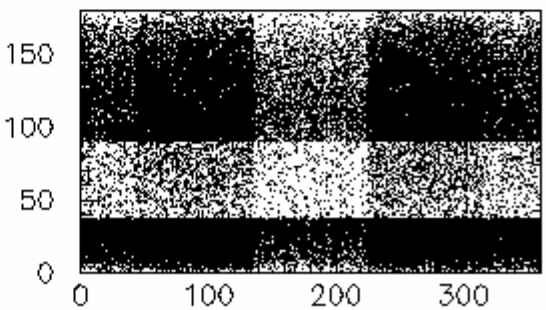
Source X–Z distribution



Source Y–Z distribution



Source cos(theta)–phi distribution



Source theta/phi distribution

Karim Laihem

G4GeneralParticleSource (GPS)

Many examples are available here :

http://reat.space.qinetiq.com/gps/examples/examples.htm

- Example 1

```
/gps/particle proton

/gps/ene/type Mono
/gps/ene/mono 500 MeV

/gps/pos/type Plane
/gps/pos/shape Rectangle
/gps/pos/rot1 0 0 1
/gps/pos/rot2 1 0 0
/gps/pos/halfx 46.2 cm
/gps/pos/halfy 57.2 cm
/gps/pos/centre 0. 57.2 0. cm

/gps/direction 0 -1 0

/run/beamOn ...
```

mono energetic beam
500 Mev

planar emission from a z×x plane
along -y axis

What to do and where to do

- In the constructor of your **UserPrimaryGeneratorAction**
  - Instantiate G4ParticleGun
  - Set default values by set methods of G4ParticleGun
    - Particle type, kinetic energy, position and direction
- In your macro file or from your interactive terminal session
  - Set values for a run
    - **Particle type, kinetic energy, position and direction**
- In the **GeneratePrimaries()** method of your **UserPrimaryGeneratorAction**
  - Shoot random number(s) and prepare track-by-track or event-by-event values
    - **Kinetic energy**, **position** and **direction**
  - Use set methods of **G4ParticleGun** to set such values
  - Then invoke **GeneratePrimaryVertex()** method of **G4ParticleGun**
  - If you need more than one primary tracks per event, loop over randomization and **GeneratePrimaryVertex().**
- examples/extended/analysis/A01/src/A01PrimaryGeneratorAction.cc is a good example to start with.
  - Also, new example to demonstrate how to use ParticleGun to alternate most of the GPS functionalities is to be included in Geant4

Particle Gun vs. General Particle Source

- **Particle Gun**
  - Simple and intuitive
  - Shoot one track at a time
    - You can shoot more than one tracks within an event.
  - Easy to handle.
    - Use set methods to alternate track-by-track or event-by-event values.

- **General Particle Source**
  - Powerful
  - Controlled by UI commands.
    - Almost impossible to control through set methods
  - Capability of shooting particles from a surface of a volume.
  - Capability of randomizing kinetic energy, position and/or direction following a user-specified distribution (histogram).

- If you need to shoot primary particles from a surface of a complicated volume, either outward or inward, GPS is the choice.
- If you need a complicated distribution, GPS is the choice.
- Otherwise, use Particle Gun.

Thanks