Primaries generation

Vienna workshop on simulations 2024
23rd April

Carlo Mancini Terracciano
carlo.mancini-terracciano@uniroma1.it

Outline

- How Geant4 handles primary generation
- The G4VUserPrimaryGeneratorAction class
- G4ParticleGun
- G4GeneralParticleSource
- Some examples

User classes

- You **have** to write the **main**()
- Initialisation classes:
  - **G4VUserDetectorConstruction**
  - **G4VUserPhysicsList**
  - **G4VUserActionInitialization**
- Action classes
  - **G4VUserPrimaryGeneratorAction**
  - **G4UserRunAction**
  - **G4UserEventAction**
  - **G4UserStackingAction**
  - **G4UserTrackinAction**
  - **G4UserSteppingAction**

classes written in red are mandatory!

G4VUserPrimaryGeneratorAction

- It is one of the mandatory user classes
- it controls the generation of primary particles
- does not directly generate primaries but invokes G4VPrimaryGenerator::GeneratePrimaryVertex()
- registers the primary particle(s) to a **G4Event** instance
- Via the **GeneratePrimaries(G4Event*)** pure virtual method

G4VUserPrimaryGeneratorAction.hh

```
class G4VUserPrimaryGeneratorAction
{
  public:
    G4VUserPrimaryGeneratorAction();
    virtual ~G4VUserPrimaryGeneratorAction();

  public:
    virtual void GeneratePrimaries(G4Event* anEvent) = 0;
};
```

G4VPrimaryGenerator

- base class for particle generators
- called by G4VUserPrimaryGeneratorAction::GeneratePrimaries(G4Event*) to produce an initial state
- A subclass of **G4VPrimaryGenerator** must be implemented
- Any **G4VPrimaryGenerator** subclass must override the pure virtual method **GeneratePrimaryVertex(G4Event*)**
- In Geant4 there are 3 classes derived from **G4VPrimaryGenerator**:
  - G4ParticleGun
  - G4GeneralParticleSource (GPS for friends!)
  - G4HEPEvtInterface (not described here)

G4ParticleGun

- Can be used for experiment-specific primary generator implementation

- shoots one primary particle of a given energy from a given point at a given time to a given direction

- You can use these methods to set primary properties:

```
void SetParticleEnergy(G4double aKineticEnergy);
void SetParticleMomentum(G4double aMomentum);
void SetParticlePosition(G4ThreeVector aPosition);
void SetNumberOfParticles(G4int aHistoryNumber);
```

G4ParticleGun an example

```
myPrimaryGenerator::myPrimaryGenerator ()
: G4VUserPrimaryGeneratorAction(), fParticleGun(nullptr)
{
  fParticleGun  = new G4ParticleGun();

  // set defaults
  fParticleGun->SetParticleDefinition(G4Gamma::Definition());
  fParticleGun->SetParticleMomentumDirection(G4ThreeVector(0.,0.,1.));
  fParticleGun->SetParticleEnergy(6.*MeV);
}


myPrimaryGenerator::~myPrimaryGenerator ()
{
  delete fParticleGun;
}
```

G4ParticleGun an example

```
myPrimaryGenerator::GeneratePrimaries(G4Event* evt)
{
  // Randomization event by event
  G4double cosT = -1.0 + G4UniformRand()*2.0;
  G4double phi = G4UniformRand()*twopi;
  G4double sinT = sqrt(1 - cosT*cosT);
  G4ThreeVector direction(sinT*sin(phi), sinT*cos(phi), cosT);

  G4double ene = G4UniformRand() * 6*MeV;

  fParticleGun->SetParticleDirection(direction);
  fParticleGun->SetParticleEnergy(ene);

  fParticleGun->GeneratePrimaryVertex(evt);
}
```

G4ParticleGun UI commands

- You could also use User Interface commands
- No randomisation
- You could change parameters between to runs
- UI settings are overwritten by code in **GeneratePrimaries()**

```
/gun/energy 10 MeV
/gun/particle mu+
/gun/direction 0 0 -1
/run/beamOn 100
/gun/particle ion
/gun/ion 55 137
/gun/position 10 10 -10 cm
/run/beamOn 100
```

G4GeneralParticleSource

- Designed to allow specification of multiple particle sources each with independent definition of particle type, position, direction and energy distribution

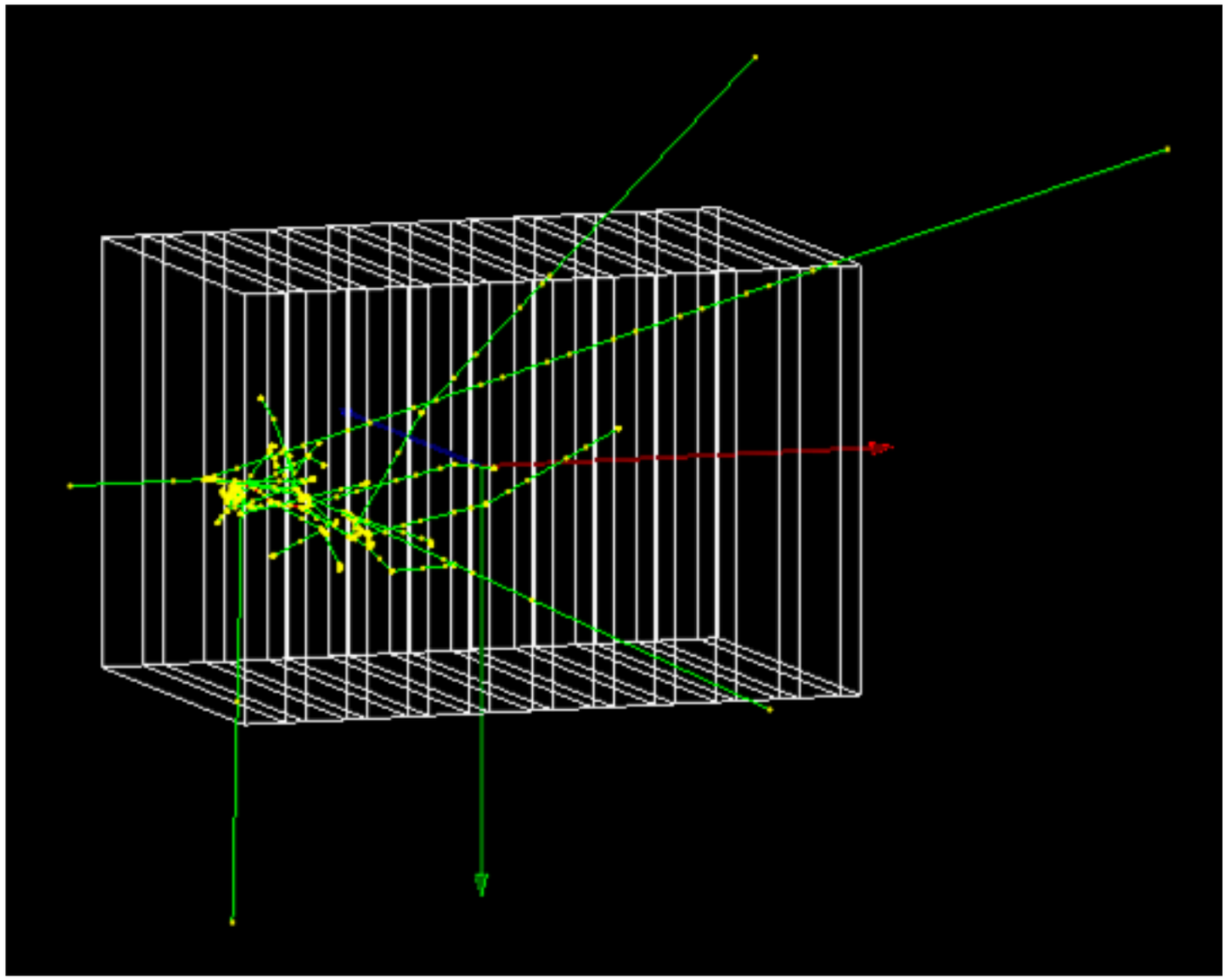
- Primary vertex can be randomly chosen on the surface of a certain volume, or within a volume

- Momentum direction and kinetic energy of the primary particle can also be randomized

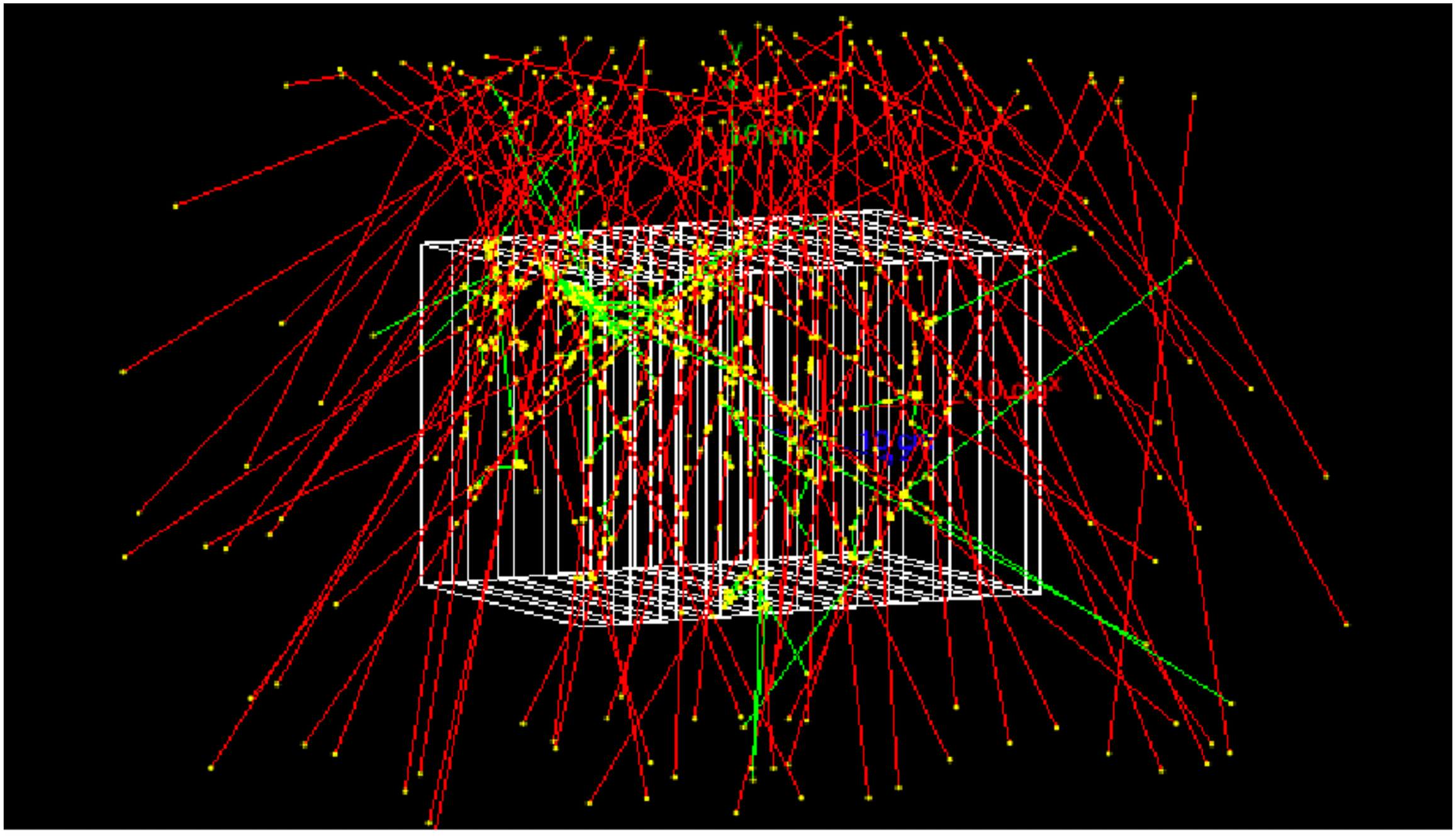
- All via UI commands

G4GeneralParticleSource an example

/gps/energy 100 MeV
/gps/particle gamma
/gps/direction 1 0 0
/run/beamOn 1



G4GeneralParticleSource an example



G4GeneralParticleSource an example

```
/gps/particle mu-
/gps/position 0 10 0 cm

/gps/pos/type Plane
/gps/pos/shape Rectangle
/gps/pos/centre 0 10 0 cm
/gps/pos/halfx 10 cm
/gps/pos/halfy 10 cm

#angular distribution
/gps/ang/type user
/gps/hist/type theta
/gps/hist/point 0.010862          0
/gps/hist/point 0.045685  0.0024035
/gps/hist/point …
/gps/hist/point   1.5083 0.00050073
```

```
#energy distribution
/gps/ene/type Arb
#/gps/ene/diffspec 0
/gps/hist/type arb
/gps/hist/point 2143.10399    239.57
/gps/hist/point 4426.79262    117.08
/gps/hist/point …
/gps/hist/point 98058.0265    0.22311
/gps/hist/inter Lin

#switch axes to make mu coming from Y
/gps/pos/rot1 1. 0. 0.
/gps/pos/rot2 0. 0. 1.

#finally...
/run/printProgress 1000
/run/beamOn 100
```

Custom G4VPrimaryGenerator implementation

- If you need:
  - An interface to a non-HEPEvt external generator
  - neutrino interaction, Higgs decay, non-standard interactions
  - many particles from one vertex, or many vertices (e.g.: double beta decay)
  - Time difference between primary tracks

- You have to implement a class inheriting from **G4VPrimaryGenerator**
  - You must implement the pure virtual method **GeneratePrimaryVertex(G4Event* evt)**
  - Instantiate one (or more) **G4PrimaryVertex** and attach to it **G4PrimaryParticle**
  - Add vertice(s) to the event **evt->AddPrimaryVertex()**

Hands on

- Primaries generation via Particle Gun and GPS
- https://geant4.lns.infn.it/vienna2024/task2