Particles and Physics Processes

Geant4 Collaboration

KEK/CRC

Contents

Particles

Process Interface

Production cuts

User limits

PARTICLES

Structure from G4Track to process

G4Track

"*snapshot*" of a particle state
- position, time, etc

propagated by the tracking

handled by G4 kernel

G4DynamicParticle

"*Dynamic*" physical properties of a particle,
- such as momentum, energy, spin, etc.
- representing an individual particle.

G4ParticleDefinition

"*Static*" properties of a particle
- THE particle type: *G4Electron*, *G4Gamma*, …
- charge, mass, life time, decay channels, etc.

G4ProcessManager

manage process list

each concrete physics process

Process 1

configured by users
a.k.a. ***user physics list***

PROCESS INTERFACE

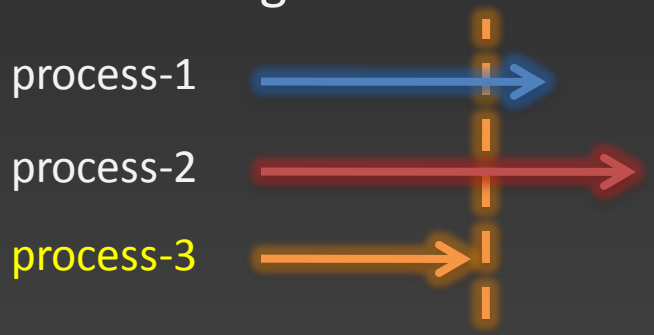
Process Interface

Abstract base class (*G4VProcess*) defining the common interface of all processes in Geant4:

- derived by *all physics processes*
- *Particle transportation is a process* as well, by which a particle interacts with geometrical volume boundaries and field of any kind.
- Users can define their own process class deriving from G4VProcess.

Each particle has its own list of applicable processes (*G4ProcessManager*).

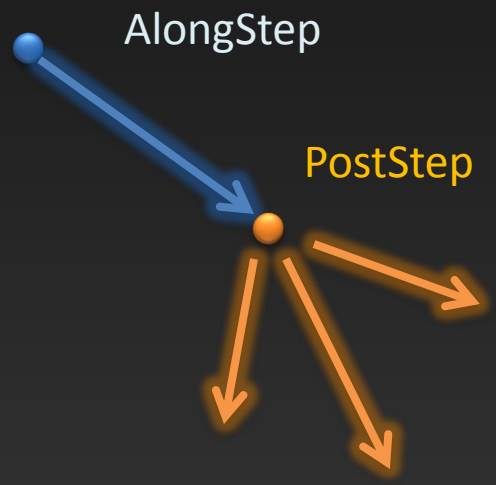
- At each step, all processes listed are invoked to get proposed physical interaction lengths.
- The process which requires the shortest interaction length limits the step.



Process types in Geant4

Three kinds of processes:

- AtRest process:
  - ✓ e^+ annihilation …
- AlongStep process:
  - ✓ To describe continuous processes,
  - ✓ occuring along the path of the particle,
  - ✓ ionisation, multiple scattering, ...
- PostStep process
  - ✓ To describe point-like reactions
  - ✓ decay in flight, compton scattering, ...



A process can implement any combination of these types:

- e.g. decay = AtRest + PostStep

G4VProcess: action methods

Each action defines two mandatory (virtual) methods:

- *GetPhysicalInteractionLength()*
  - ✓ used to limit a step:
  - ✓ the process triggers an interaction,
  - ✓ or any other reasons, like fraction of energy loss, geometry boundary, user's limits …
- *DoIt()*
  - ✓ implements the actual action to be applied on the track
  - ✓ the related production of secondaries

Other useful methods

- *G4bool IsApplicable(const G4ParticleDefinition &)*
  - ✓ used to check if a process can handle a given particle type

Sequence of processes for a particle travelling

At the beginning of the step, determine the step length

- define the step length as the smallest of the lengths among :
  - ✓ All AlongStep/GetPhysicalInteractionLenght()
  - ✓ All PostStep/GetPhysicalInteractionLength()

Apply all *AlongStepDoIt()* actions, at once:

- changes are computed from particle state at the beginning of the step,
- accumulated in the *G4Step*,
- then applied to the *G4Track*, from the *G4Step*.

Apply *PostStepDoIt()* actions sequentially:

- apply *PostStepDoIt()* of the process which limit the step

Physics List

G4ProcessManager maintains three *vectors* of processes (for AtRest/AlongStep/PostStep)

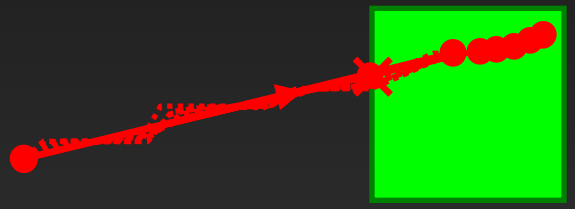
These vectors are provided by users a.k.a. "Physics List".

- a list of processes to be taken into account
- process ordering

A word about process ordering

Ordering of following processes is critical:

- Assuming n processes, the ordering of the AlongGetPhysicalInteractionLength() of the last processes should be:
  - ✓ [n-2] multiple scattering
  - ✓ [n-1] ionization
  - ✓ [n] transportation



Why?

- Processes return a true path length.
- Multiple scattering *virtually folds up* this true path length into a shorter *geometrical* path length.
- Based on this new length, the transportation can geometrically limits the step.

Other processes ordering usually *does not matter*.

PRODUCTION CUTS

Threshold for Secondary Production (1)

Every simulation developer must answer the question:

- *At what energy do I stop tracking particles?*

This is a balancing act

- need to go low enough to get the physics interested in
- can't go too low because some processes have *infrared divergence* causing much CPU-time consumption.

The traditional Monte Carlo solution is to impose an absolute cut-off in energy:

- particles are stopped when this energy is reached
- remaining energy is dumped at that point

Threshold for Secondary Production (2)

But, such a cut may cause *imprecise stopping location and deposition of energy*

There is also a particle dependence

- range of 10 keV γ in Si is *a few cm*
- range of 10 keV e^- in Si is *a few microns*

And a material dependence

- suppose you have a detector made of alternating sheets of Pb and plastic scintillator
- if the cut-off is OK for Pb, it will likely be wrong for the scintillator which does the actual energy deposition measurement

Threshold for Secondary Production (3)

Geant4 solution: impose a production threshold

- this threshold is *a distance, not an energy*
- default = *1 mm*
- a charged particle loses energy by producing secondary electrons or gammas
- if the particle no longer has enough energy to produce secondaries which travel at least 1mm, two things happen:
  - ✓ discrete energy loss ceases (*no more secondaries produced*)
  - ✓ the particle is *tracked down to zero energy* using continuous energy loss

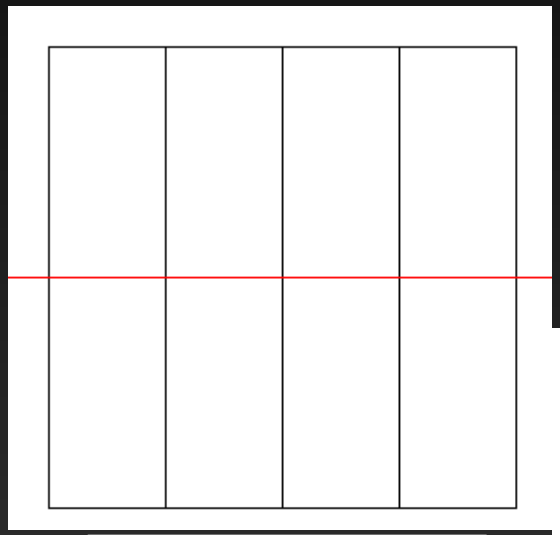
Stopping location is therefore correct.

Only one value of production threshold distance is needed for all materials

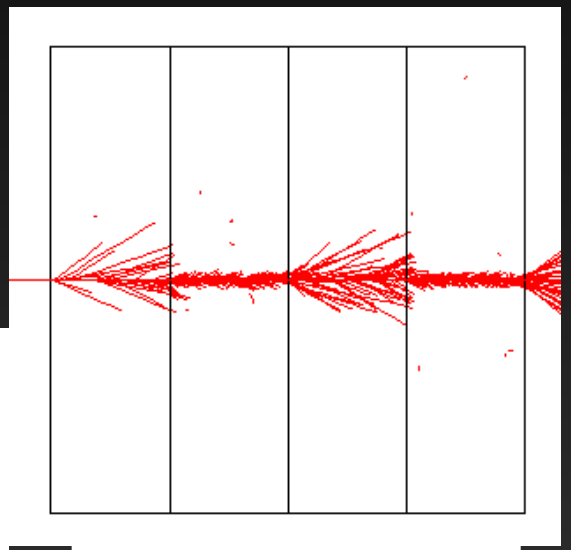
- correspond to different energies depending on material.

Production Threshold vs. Energy Cut

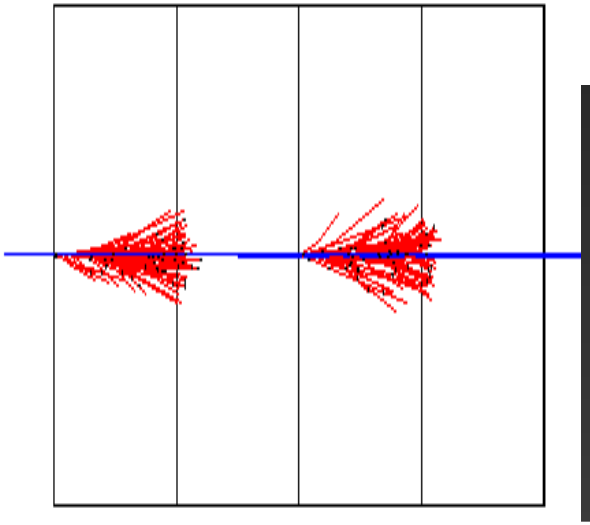
500 MeV p in LAr-Pb sampling calorimeter



Cut = 2 MeV



Cut = 450 keV



Production cut =1.5 mm

USER LIMITS

G4UserLimits

User limits are artificial limits affecting to the tracking.

```
G4UserLimits(   G4double ustepMax = DBL_MAX,
                G4double utrakMax = DBL_MAX,
                G4double utimeMax = DBL_MAX,
                G4double uekinMin = 0.,
                G4double urangMin = 0. );
```

- fMaxStep; // max allowed Step size in this volume
- fMaxTrack; // max total track length
- fMaxTime; // max global time
- fMinEkine; // min kinetic energy remaining (only for charged particles)
- fMinRange; // min remaining range (only for charged particles)

- Blue : affecting to step / : affecting to track

You can set user limits to logical volume and/or to a region.

- User limits assigned to logical volume do not propagate to daughter volumes.
- User limits assigned to region propagate to daughter volumes.
- If both logical volume and associated region have user limits, those of logical volume win.

Processes co-working with G4UserLimits

In addition to instantiating G4UserLimits and setting it to logical volume/region, you have to assign the following processes to particle types you want to affect.

Limit to step:

- *G4StepLimiter* process must be defined to affected particle types.

Limits to track:

- *G4UserSpecialCuts* process must be defined to affected particle types.