GetVelocity

Issues
A proposed solution

Hisaya Kurashige
Kobe Univ.

GetVelocity method

- *G4Track::GetVelocity* gives

  Velocity of the track

  - For particles with mass
    - velocity = c_light*std::sqrt(T*(T+2.*mass))/(T+mass);
      - T : KineticEnergy
  - For mass-less particles (ex. Gamma)
    - velocity = c_light
  - For optical photon
    - velocity = group velocity of the photon
      - MaterialPropertiesTable has group velocity

$$v_g = \frac{c}{n(E)} + \frac{dn(E)}{d(\log E)}$$

Performance

- *G4Track::GetVelocity* costs CPU

  Why ? Because it is called several times in each step to update velocity information in step points

  - At the beginning of the tracking, *G4Step* uses it to initialize step points information
  - At the end of AlongStepDoIt, *G4Step* uses it to fill post-step point information
  - At every PostStepDoIt, *G4Step* uses it to fill post-step point information

Transportation

- *G4Transportation* uses *GetVelocity* to
  - Set *G4FieldTrack* (in magnetic fields)
  - Compute time of flight in the step

```
// The time was not integrated .. make the best estimate possible
G4double finalVelocity   = track.GetVelocity() ;
G4double initialVelocity = stepData.GetPreStepPoint()->GetVelocity() ;
```

NOTE: No difference between initial/finalVelocity except for optical photon

```
if (fpDynamicParticle->GetDefinition()== fOpticalPhoton)  {
      //  A photon is in the medium of the final point
      //  during the step, so it has the final velocity.
      deltaTime = stepLength/finalVelocity ;
} else if (finalVelocity > 0.0) {
```

GetVelocity: latest updates

- Use *G4PhysicsVector* in calculation of velocity to improve performance
  - track-V09-03-01
    - Use PhysicsVector for *G4Track::GetVelocity*
  - track-V09-03-04
    - Fix a bug in *G4Track::PrepareVelocityTable*
      - *GetVelocity* gives wrong value around E_{kin}/mass ~ 100

Charged particles: *PhysicsVector approach*

- Suppose a muon travels
  in the magnetic field
  - *G4Track::GetVelocity* is called 6+α times per step
    - G4Transportation: 2
    - At AlongStepDoIt :1
    - At PostStepDoIt :3 + α
  - But, energy of the muon changes once in most steps

  Caching mechanism improves performance
- *PhysicsVector* is used in calculation of velocity from kinetic energy

  → ~30% improvement in CPU time for muons in big volume

Mass-less particles

- Number of calls per step is half
  - *G4Track::GetVelocity* is called 3+α times per step
    - G4Transportation: 1
    - At AlongStepDoIt :1
    - At PostStepDoIt :1+α
- No problem for Gamma/neutrinos
  - No calculation : always return c_light
- Optical Photon is another issue
  - Need some calculation to get group velocity in *G4MaterialPropertyVector::GetProperty*()
  - Caching mechanism is implemented in GetVelocity
    - Use previous value if material and energy are same compared to the previous step

We can improve performance

But..

We have bad design concerning velocity of track

It cause difficulties/problems in

code maintenance

extension of particle types

Problems in Design

- *GetVelocity* takes care of all particle type
  - 'phonon' is planed to be added. It also require complex calculation for GetVelocity
- It is waste of time to call *GetVelocity* several times in each step

➢ Problems come from

- Other properties of *G4Track* are set by process via ParticleChange
- Only velocity is the exception

A proposal of design change

- Velocity is set by **processes**
  via particle change
    in same manner as other properties
  - Calculation of velocity occurs only if necessary
    because a process knows what happens in the step
    - After energy change, for ordinary particles
    - After entering a new volume or changing energy, for optical photon

Details: Track

- Introduce a new member and method of
  - *G4double G4Track::velocity*
  - *void G4Track::SetVelocity(G4double)*
- *GetVelocity* simply gives *G4Track::velocity*
- Current *GetVelocity* method is moved to
  - *G4double G4Track::CalculateVelocity()*
  - *G4double G4Track::GetGroupVelocityForOpticalPhoton()*

Details: process

- Following processes are responsible for updating velocity
  - Transportation

    I'd propose to have separate transportation
    - *G4Transportation*

      →no need to update velocity unless velocity change along the step is taken into account
    - *G4OpticalTransportation*
    - *G4PhononTransportation*
  - EnergyLoss processes
  - DiscreteProcesses which can change energy
    - No need to change process codes →see next

Details : ParticleChange

- Two methods are added to ParticleChange
  - G4double GetVelocity() const;
  - void ProposeVelocity(G4double);
  - These methods will be used by processes
  - Or by other methods of ParticleChange
    - For example, ProposeVelocity may be called in ProposeEnergy for particles with non-zero mass

- Velocity of the track will be updated in *G4Step::UpdateTrack*

Plans

- Make prototype to check

  The new design works successfully

  NO deterioration in performance

  Expendability

  → by end of 2010

- Implement new design after 9.4 release
  - Track
    - ParticleChanges
    - Track
    - Step
  - Transportation