June 2007

Geant4 Event Biasing

Marc Verderi, LLR

(Heavily copied from

Jane Tinslay, SLAC)

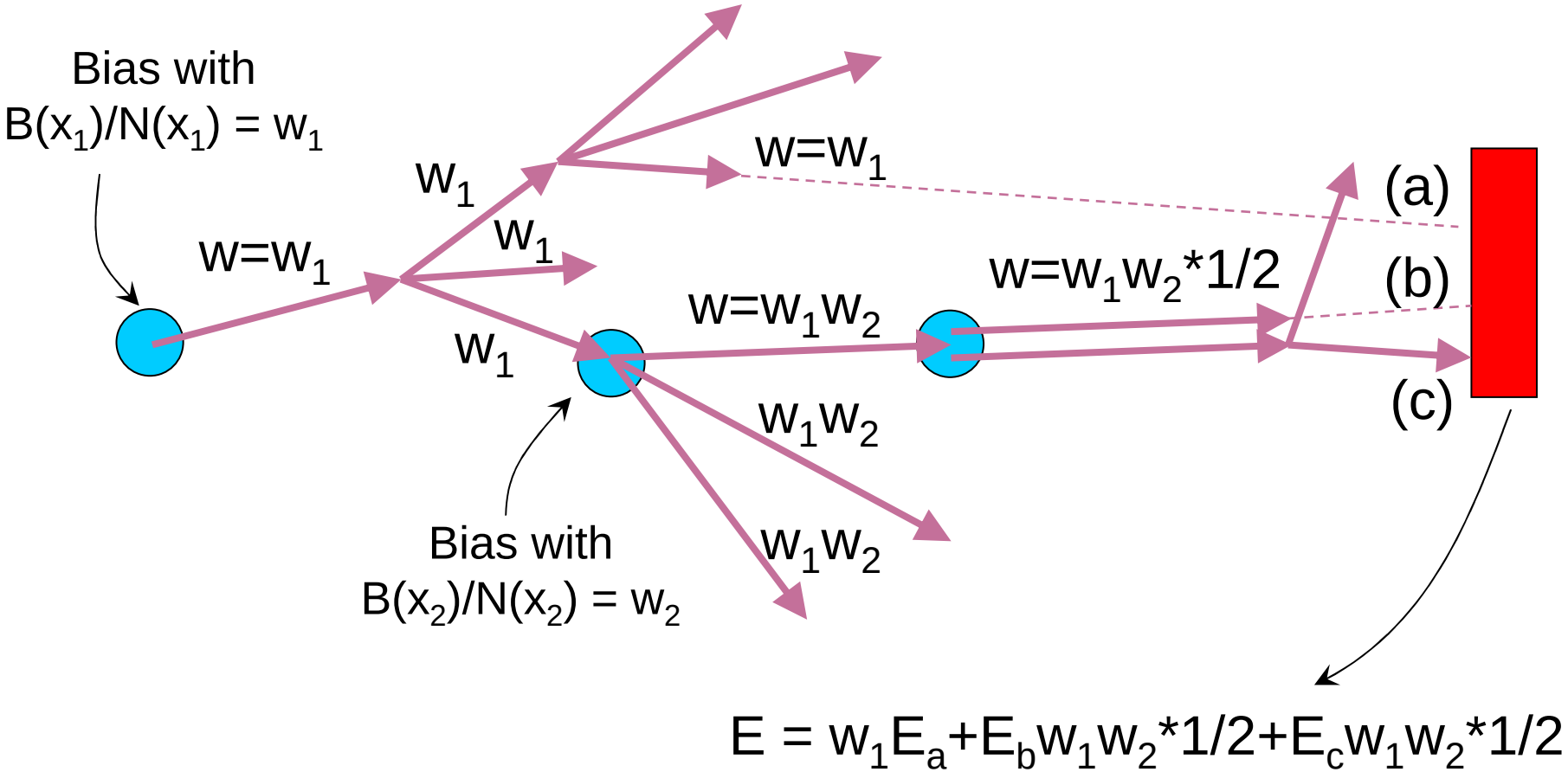
Outline

- Introduction
- Built in biasing options
  - Primary particle biasing
  - Radioactive decay biasing
  - General hadronic leading particle biasing
  - Hadronic cross section biasing
  - Geometrical biasing
    - Importance sampling
    - Weight windows & weight cutoff
- User defined biasing
  - G4WrapperProcess
  - Uniform bremsstrahlung splitting example
- Recent developments
- Summary

Introduction

- What is analogue simulation ?
  - Sample using natural probability distribution, N(x)
  - Predicts mean with correct fluctuations
  - Can be inefficient for certain applications

- What is non-analogue/event biased simulation ?
  - Cheat - apply artificial biasing probability distribution, B(x) in place of natural one, N(x)
    - B(x) enhances production of whatever it is that is interesting
    - To get meaningful results, must apply a weight correction
  - Predicts same analogue mean with smaller variance
    - Increases efficiency of the Monte Carlo
  - Doesn't predict correct fluctuations
  - Should be used with care


Bias with
B(x1)/N(x1) = w1
w=w1
w1
w1
w=w1
w1
w=w1w2
w=w1w2*1/2
(a)
(b)
(c)
w1w2
w1w2
Bias with
B(x2)/N(x2) = w2
E = w1Ea+Ebw1w2*1/2+Ecw1w2*1/2

- Geant4 simulation:
  - Analogue == regular processing
  - Non-analogue/event biased simulation == manipulated processes and/or process list
    - I.e, manipulate processing to effectively apply B(x) in place of N(x)
- Geant4 provides
  - Several built-in general use biasing techniques
  - Utility class, G4WrapperProcess to support user defined biasing
- Expect biasing to be used by experienced users
  - Should understand what a particular biasing technique does, it's constraints and side effects
  - Understand how processing works in Geant4

Built in Biasing Options

Biasing Technique	First Release Version
Primary particle biasing	3.0
Radioactive decay biasing	3.0
Mars hadronic leading particle biasing (obsolete)	4.0
General hadronic lead particle biasing	4.3
Hadronic cross section biasing	4.3
Geometrical Importance sampling	5.0
Geometrical weight window and weight cutoff	5.2

Primary Particle Biasing

- Use case:
  - Increase number of high energy particles in cosmic ray spectrum
- Increase number of primary particles generated in a particular phase space region of interest
  - Weight of primary particle modified as appropriate
- General implementation provided by G4GeneralParticleSource class
  - Bias position, angular and energy distributions

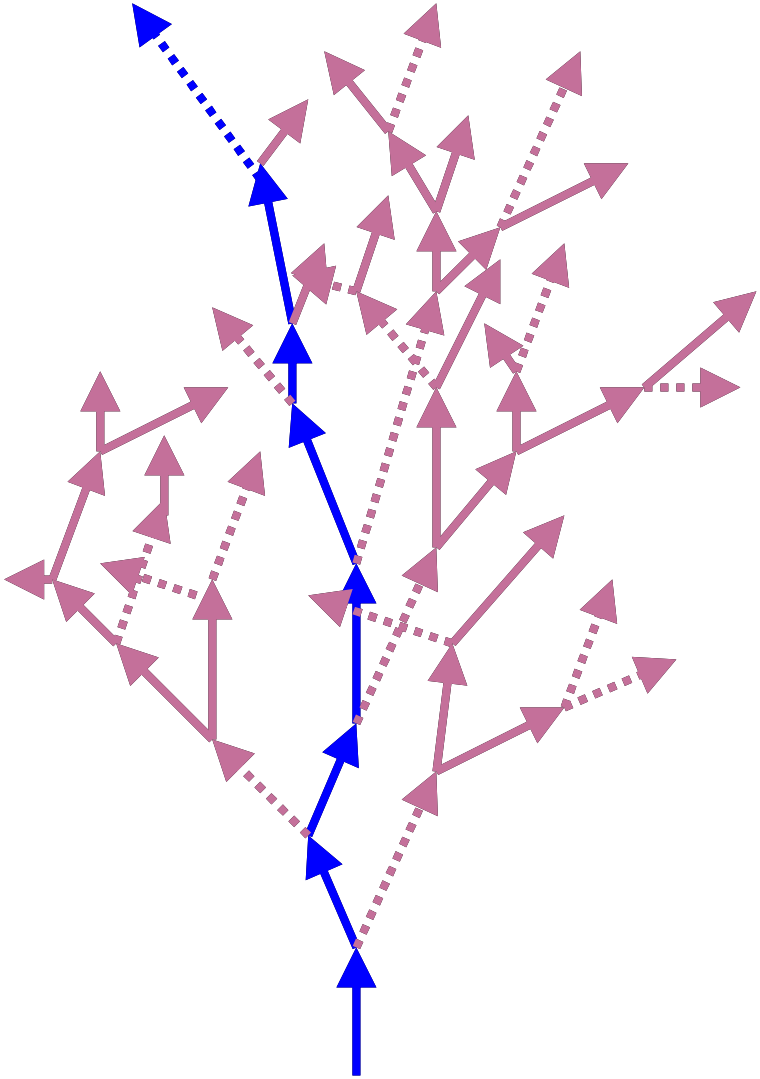
Radioactive Decay Biasing

- G4RadioactiveDecay simulates decay of radioactive nuclei

- Implements the following biasing methods
  - Increase sampling rate of radionuclides within observation times
    - User defined probability distribution function
  - Nuclear splitting
    - Parent nuclide is split into user defined number of nuclides
  - Branching ratio biasing
    - For a particular decay mode, sample branching ratios with equal probability

General Hadronic Leading Particle Biasing

- Built in utility for hadronic processes
  - Implemented in G4HadLeadBias class
- Keep only the most important part of the event, and representative tracks of given particle types
  - Keep track with highest energy
    - I.e, the leading particle
  - Of the remaining tracks, select one from each of the following types if they exist:
    - Baryon's, π^0's, mesons, leptons
  - Apply appropriate weight
- To activate, set SwitchLeadBiasOn environment variable



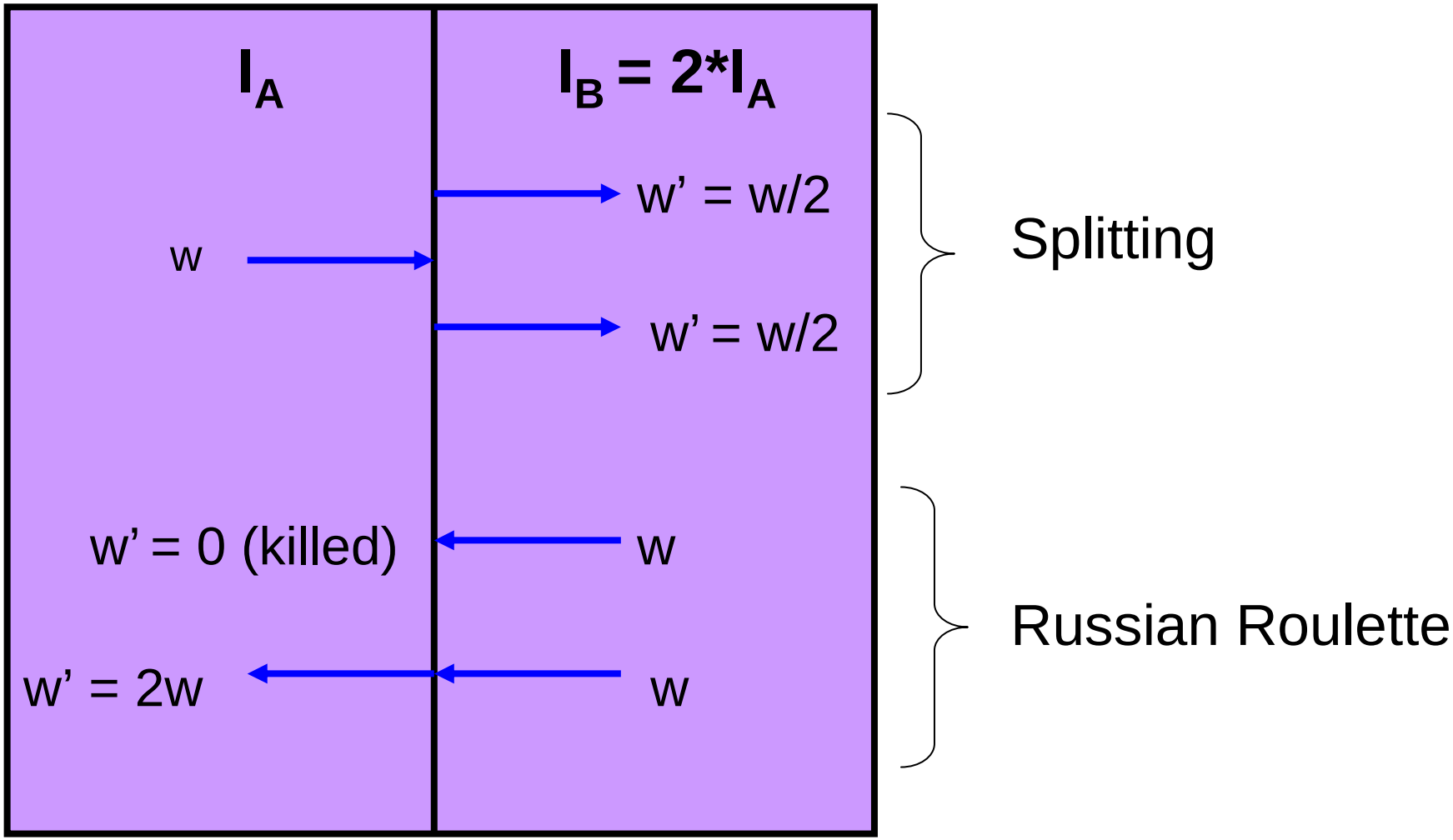
Hadronic Cross Section Biasing

- Built in cross section biasing in hadronics for PhotoInelastic, ElectronNuclear and PositronNuclear processes
- Artificially enhance/reduce cross section of a process
- Useful for studying
  - Thin layer interactions
  - Thick layer shielding

Geometrical Biasing

- Geometry based biasing implemented within common framework in Geant4
  - Importance sampling
  - Weight windows
  - Weight cutoff
- Process based approach
  - Process list is modified behind the scenes to apply biasing
- Applicable in mass or parallel geometries
  - Define physical volumes named cells
  - Ability to use parallel geometries to define importance sampling
    - Currently in Beta stage, should be there in release 9.0

Importance Cell Example



Weight Window

- Weight based enhancement to importance sampling
- Particles either split or Russian Roulette played based on space-energy cells
- User defines a weight window for each space cell, and optionally for different energies
- Can help control weight fluctuations introduced by other variance reduction techniques

Biasing Example B01

- examples/extended/biasing/B01
- Study punch through of 10 MeV neutrons incident upon thick concrete cylinder
- Demonstrates importance sampling & weight windows technique in mass geometry
- Geometry consists of an 80 cm high concrete cylinder divided into 18 slabs
- Importance value for slab n = 2^n

Geometrical Biasing Documentation

- Detailed examples can be found at
  - examples/advanced/Tiara
  - examples/extended/biasing
- Documentation on all geometrical biasing techniques at
  - http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/ch03s07.html

User Defined Biasing: G4WrapperProcess

- Implement user defined biasing through G4WrapperProcess
  - A process itself, I.e, inherits from G4VProcess
  - Wraps an existing process
    - By default, function calls are forwarded to existing process
  - Non-invasive way to manipulate the behavior of a process
- To use:
  - Subclass G4WrapperProcess and override appropriate methods, e.g, PostStepDoIt
  - Register subclass with process manager in place of existing process
  - Register existing process with G4WrapperProcess

G4WrapperProcess Structure

```
class G4WrapperProcess : public G4VProcess {

protected:
  G4VProcess* pRegProcess;
...

inline void
G4WrapperProcess::RegisterProcess(G4VProcess* process)
{
  pRegProcess = process;
...

}

inline G4VParticleChange*
G4WrapperProcess::PostStepDoIt(const G4Track& track,
                               const G4Step&  stepData)
{
  return pRegProcess->PostStepDoIt(track, stepData);
}
```

Example: Uniform Bremsstrahlung Splitting

- In this example, only interesting in scoring bremsstrahlung photons
- Want to increase Monte Carlo efficiency by reducing computing time spent tracking electrons
- Example of biasing through enhancing production of secondaries

Implementation

1) Create user class inheriting from G4WrapperProcess
2) Override PostStepDoIt method of G4WrapperProcess

```
class BremSplittingProcess : public G4WrapperProcess {

public:

  BremSplittingProcess();
  virtual ~BremSplittingProcess();

  // Override PostStepDoIt  method
  G4VParticleChange* PostStepDoIt(const G4Track& track, const G4Step& step);
...
```

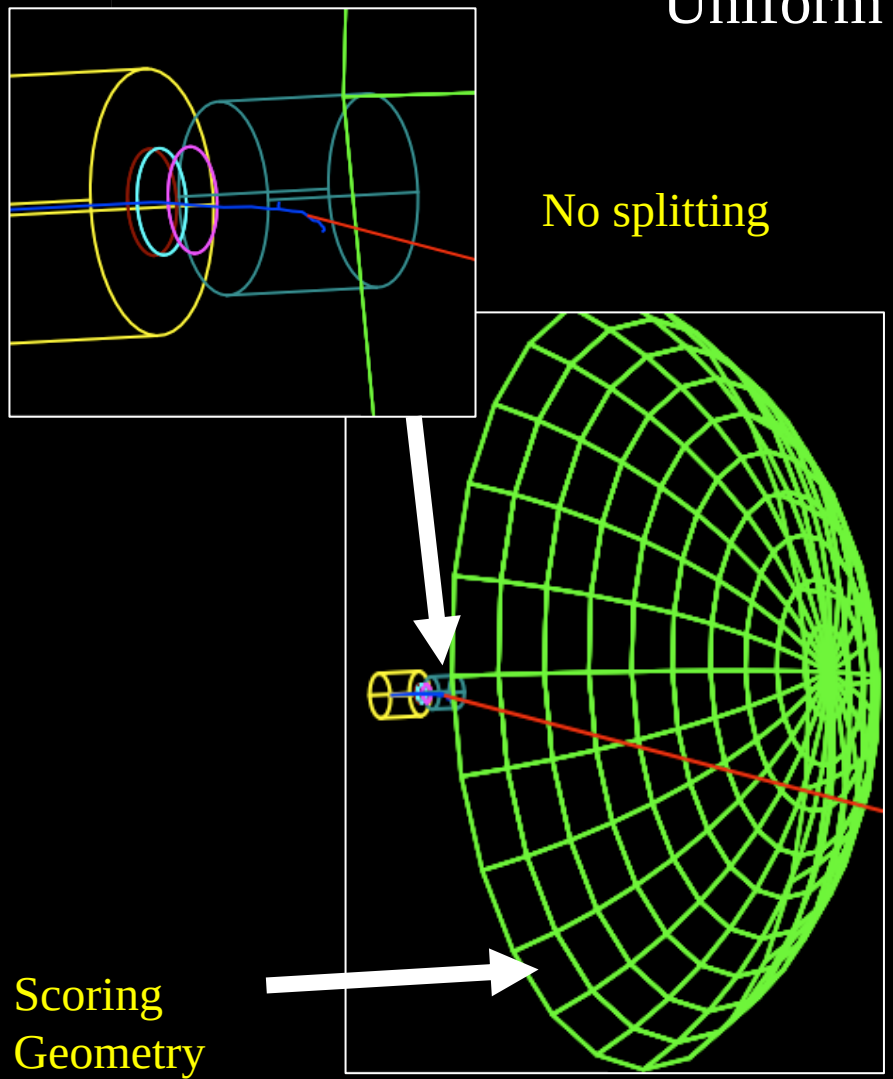
3) Implement overridden PostStepDoIt method

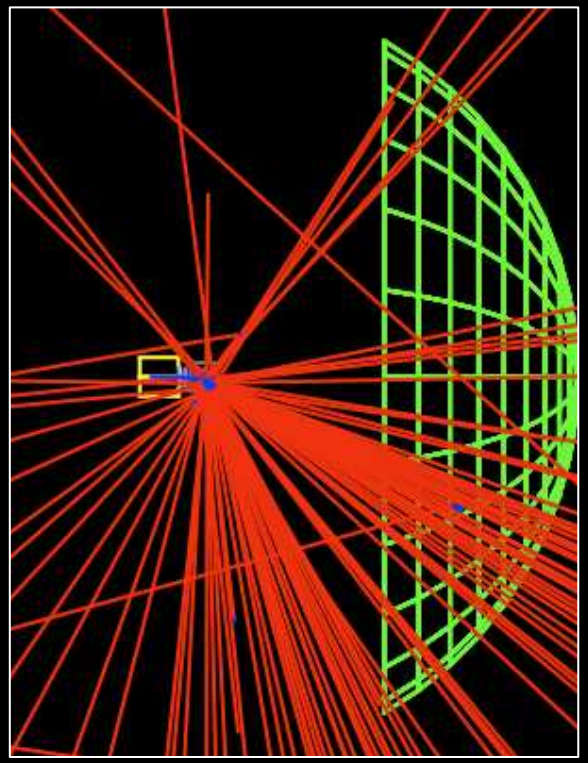
```
G4VParticleChange*
BremSplittingProcess::PostStepDoIt(const G4Track& track, const G4Step& step)
{
...
  G4double weight = track.GetWeight()/fNSplit;

  // Secondary store
  std::vector<G4Track*> secondaries;
  secondaries.reserve(fNSplit);

  // Loop over PostStepDoIt method to generate multiple secondaries.
  for (G4int i=0; i<fNSplit; i++) {
    particleChange = pRegProcess->PostStepDoIt(track, step);

    assert (0 != particleChange);
    particleChange->SetVerboseLevel(0);

    for (G4int j=0; j<particleChange->GetNumberOfSecondaries(); j++) {
      secondaries.push_back(new G4Track(*(particleChange->GetSecondary(j))));
    }
  }
  ...
```

4) Register wrapped process with process manager

```
G4LowEnergyBremsstrahlung* bremProcess = new G4LowEnergyBremsstrahlung();
bremSplitting = new BremSplittingProcess();
bremSplitting->RegisterProcess(bremProcess);
pmanager->AddProcess(bremSplitting,-1,-1, 3);
```

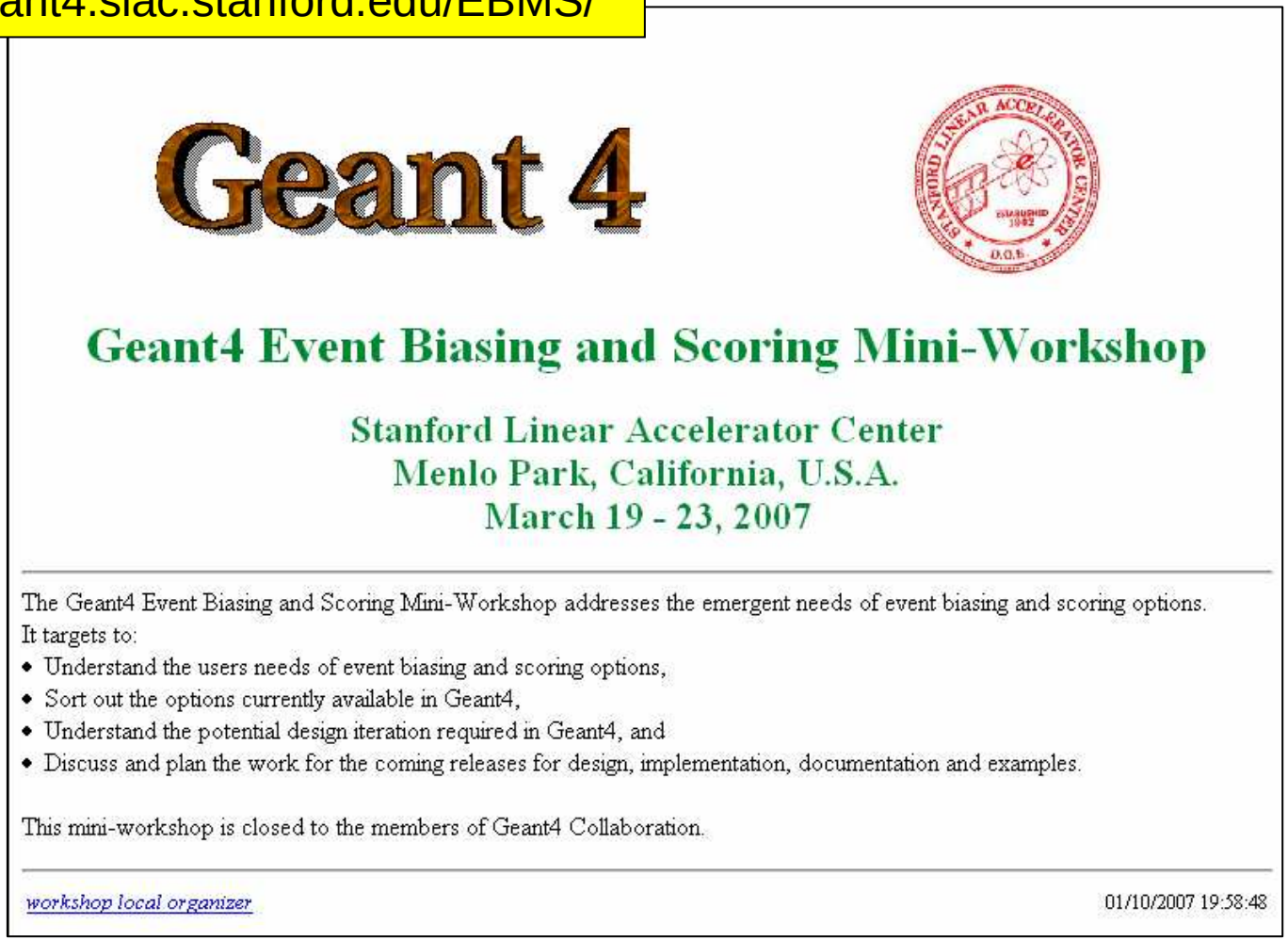
Uniform Bremsstrahlung Splitting





Recent Developments

http://geant4.slac.stanford.edu/EBMS/



Geant 4

Geant4 Event Biasing and Scoring Mini-Workshop

Stanford Linear Accelerator Center
Menlo Park, California, U.S.A.
March 19 - 23, 2007

The Geant4 Event Biasing and Scoring Mini-Workshop addresses the emergent needs of event biasing and scoring options.
It targets to:

- Understand the users needs of event biasing and scoring options,
- Sort out the options currently available in Geant4,
- Understand the potential design iteration required in Geant4, and
- Discuss and plan the work for the coming releases for design, implementation, documentation and examples.

This mini-workshop is closed to the members of Geant4 Collaboration.

workshop local organizer 01/10/2007 19:58:48

- Physics biasing
  - Existing physics based biasing fragmented
  - Identify missing biasing methods & variations between methods in other Monte Carlo codes
    - Implicit capture
    - General cross section biasing
    - Interaction forcing
    - Path length biasing
    - Advanced bremsstrahlung splitting
    - Leading particle biasing
  - Look at developing dedicated framework to provide general physics biasing in analogy with geometrical biasing
    - Manipulating physics processes/lists
  - Idea box for tomorrow:
    - Any request/input about biasing functionalities ?

Summary

- Number of popular event biasing techniques built into Geant4
- User defined biasing supported through G4WrapperProcess
- Ongoing developments aim to improve exiting Geant4 biasing, and provide new event biasing and scoring methods
- Documentation at
  - http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/ch03s07.html