



Monte Carlo Simulation of Photon Propagation and Detection by the IceCube Neutrino Detector



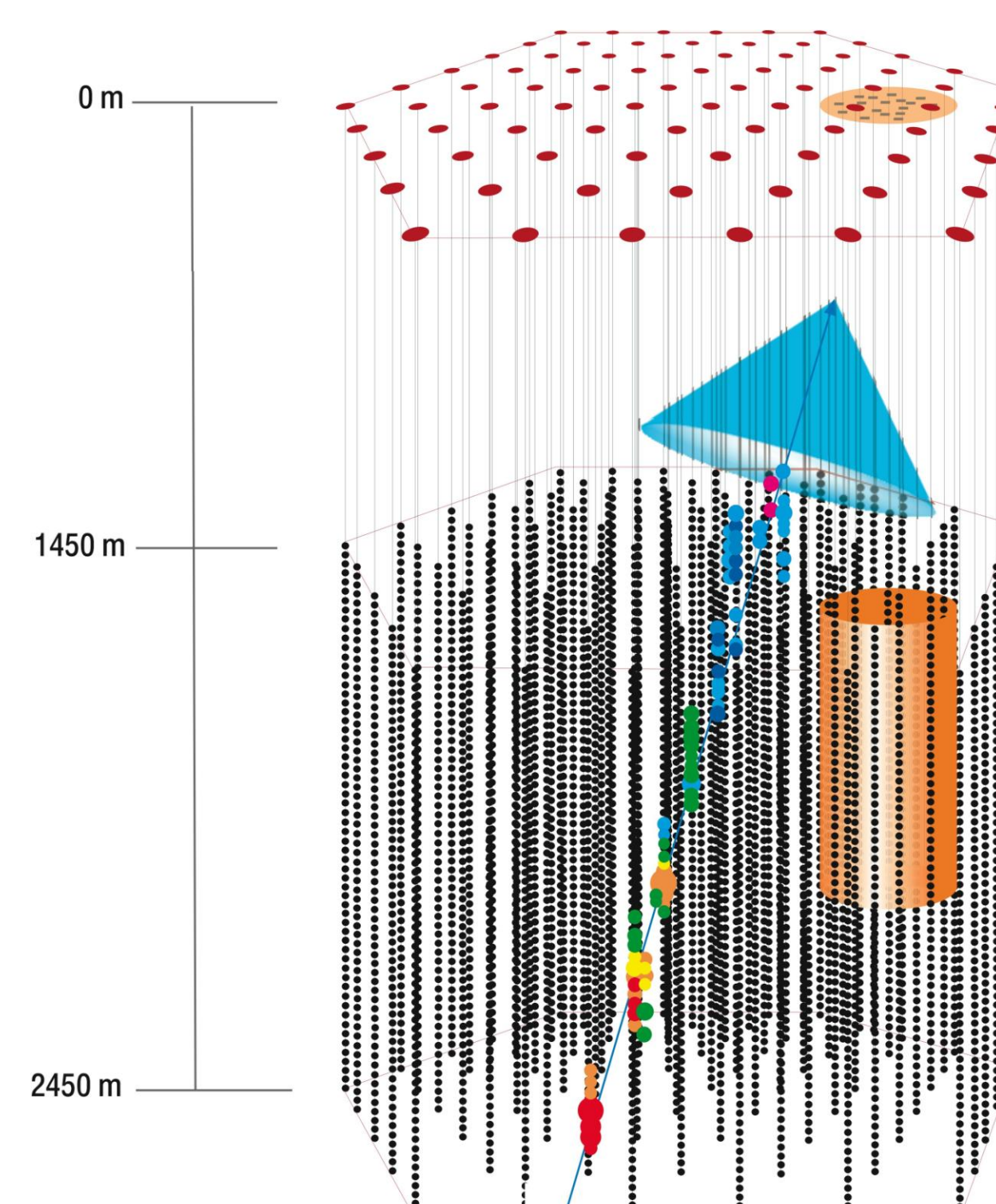
Tareq AbuZayyad (tareq@icecube.wisc.edu)
University of Wisconsin, River Falls WI 54022

Abstract:

The goal of the IceCube¹ experiment is to detect extraterrestrial neutrinos through the observation of the Cherenkov light signals generated by high-energy muons (by-products of neutrino interactions with matter) as they pass through the detector volume. Cherenkov photons suffer multiple scattering events on their way from the source to the optical modules comprising the detector. Most will be absorbed by the medium and never reach a detector element. The detector event rate is on the order of 1kHz (mainly atmospheric muons), with a typical event generating some 10^7 photons. Considering the event rate, a direct simulation of the detector response, a crucial step in understanding and interpreting the detector data, is computationally prohibitive.

To speed up the computation, "photon tables" are generated once and then are used in the simulations of individual events. A CPU implementation of the direct calculation of photon multiple scattering and detection was implemented by a fellow researcher working on the IceCube experiment. That version (after repeated optimizations) was found to be ~10x slower than using photon tables. Using CUDA, a version of the simulation program run on a GeForce 9800GT achieved ~100x speed up compared to the CPU version (run on a single core of an Intel Q6600). A few modifications were required to map the problem from the CPU (serial) to run efficiently on a GPU. The output of the two simulation programs however was found to be in excellent agreement.

In this poster we discuss the problem being solved and show a comparison of the results obtained using the GPU implementation and the CPU version.



IceCube is a neutrino observatory located at the geographic South Pole. IceCube strings are deployed 1,450-2,450 meters in the ice. A total of 80+ strings will be deployed by 2011. When completed the detector volume will reach 1 km^3 . Currently the detector is operating with 59 strings. Each string is comprised of 60 optical modules. The plot illustrates Cherenkov light emitted by a charged particle moving through ice which will trigger IceCube optical sensors resulting in a track as illustrated by multiple colored optical modules.

The Problem:

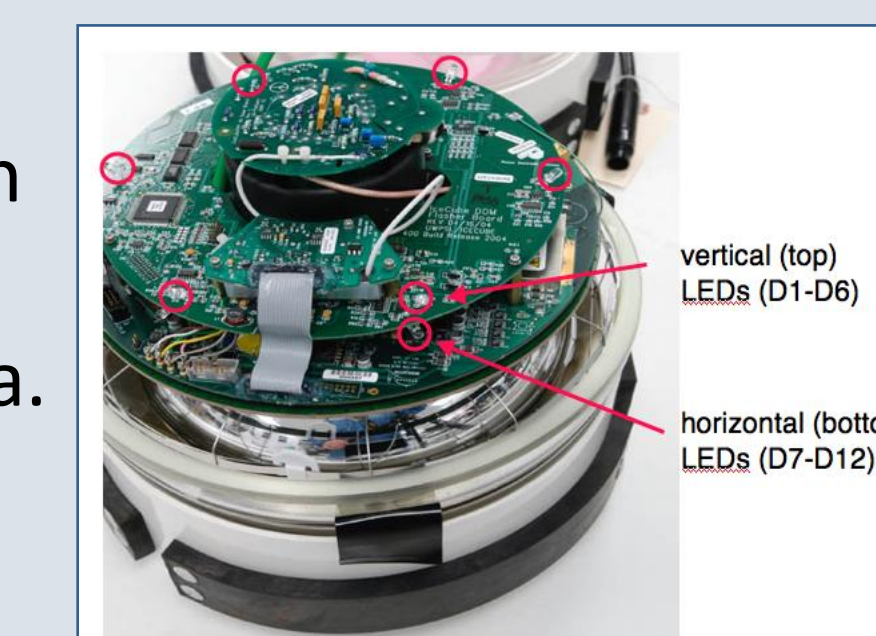
Follow photons (point particles) as they multiply scatter in the ice until they are either absorbed in the ice, or are detected by an optical module (a sphere with a radius of 18 cm). In the simulation, 3540 modules (59 strings x 60 modules) are used. The strings are located on a hexagonal grid about 125 m apart. The modules vertical spacing is ~17 meters.

(case A) Light produced by muons:

- 1) photons can be produced at any point along the track, non-uniform distribution of injection point governed by stochastic interactions superimposed over continuous energy losses. Photon initial direction at Cherenkov angle from muon direction (random phi)
- 2) Muons with different energies produce different numbers of photons. Muon energy generated randomly from an E^{-2} or E^{-1} spectrum.
- 3) Photon wavelength spectrum determined by Cherenkov light emission spectrum convoluted with wavelength-dependent optical module efficiency curve. Wavelength range 300-600 nm in thirty 10 nm bins.
- 4) **PURPOSE:** Assuming a given Ice Model test distributions of triggers against real data.

(case B) Light produced by LED flashers.

- 1) Each Optical Module contains a flasher board with twelve 405 nm LEDs. Flashers are mono-chromatic point sources with known position and known light output ($10^7 - 10^9$ photons).
- 2) **PURPOSE:** Vary Ice Model parameters to best fit calibration data. The goal here is to make sure that our modeling of the ice is done properly.



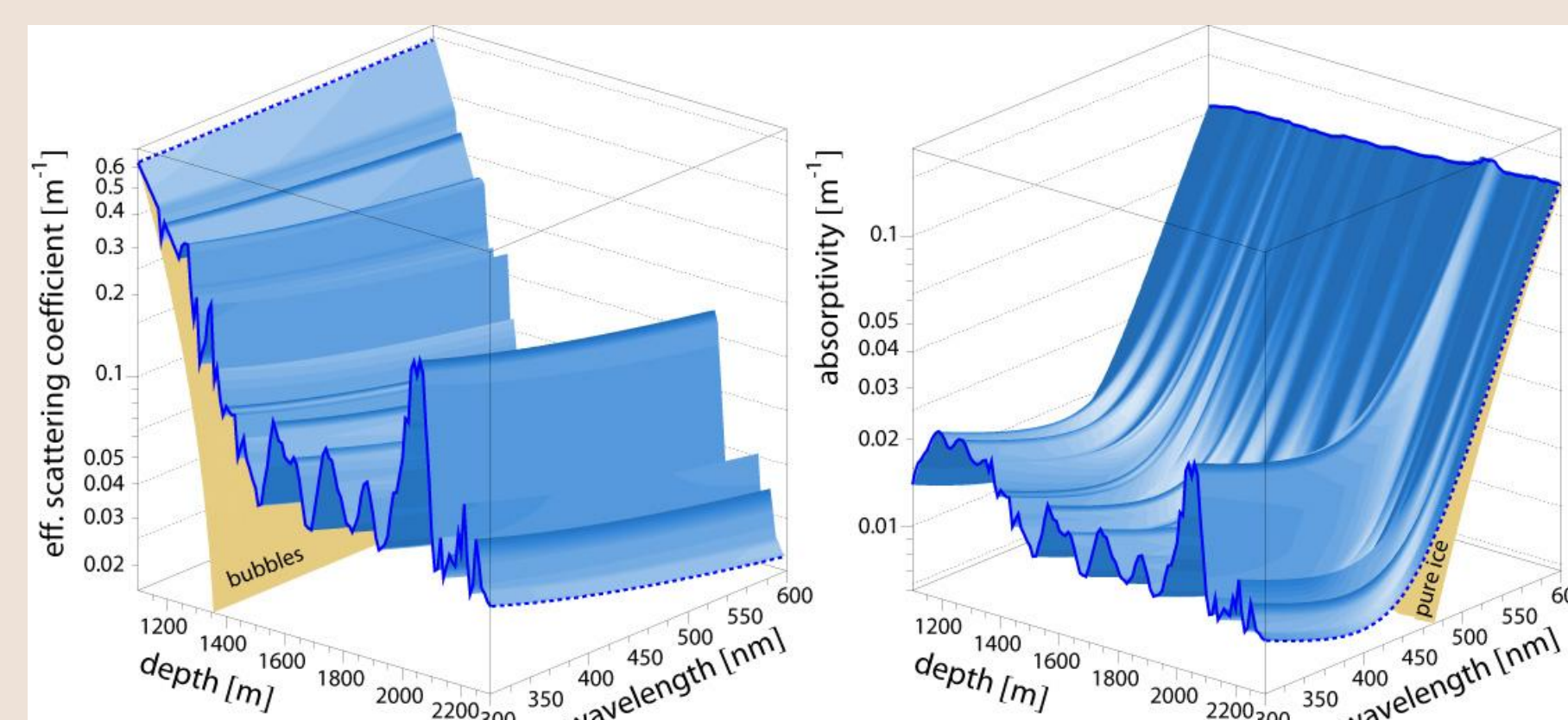
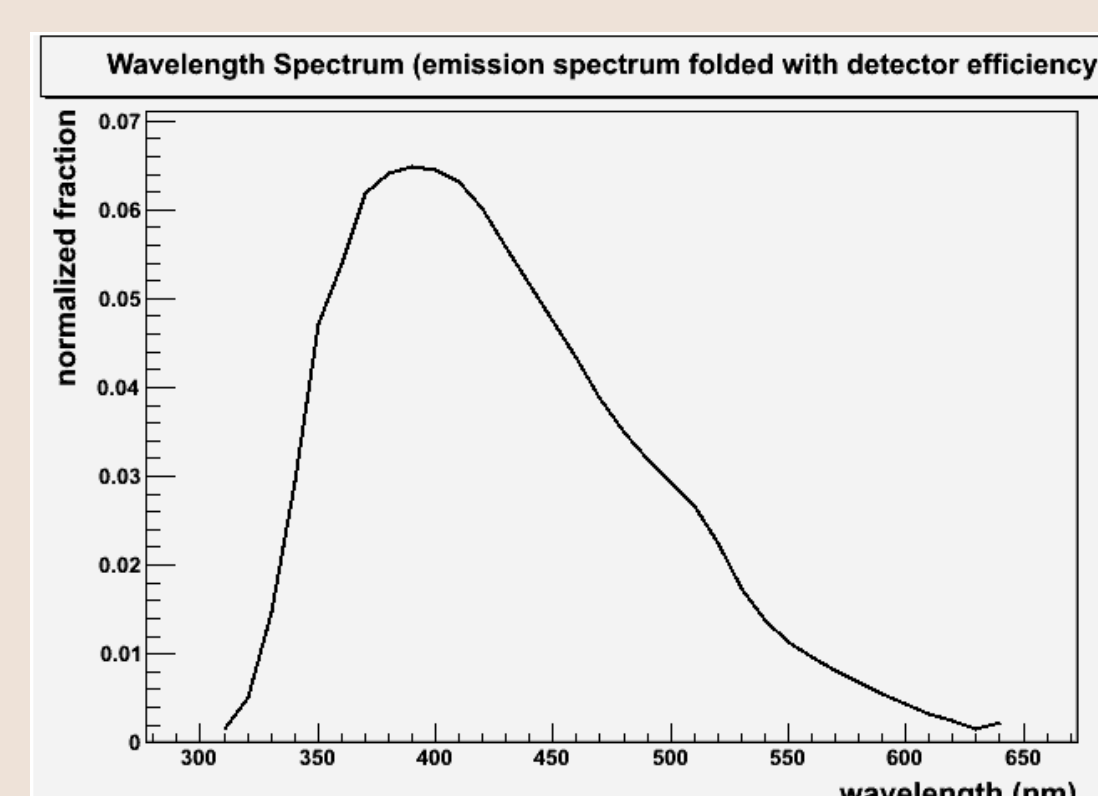
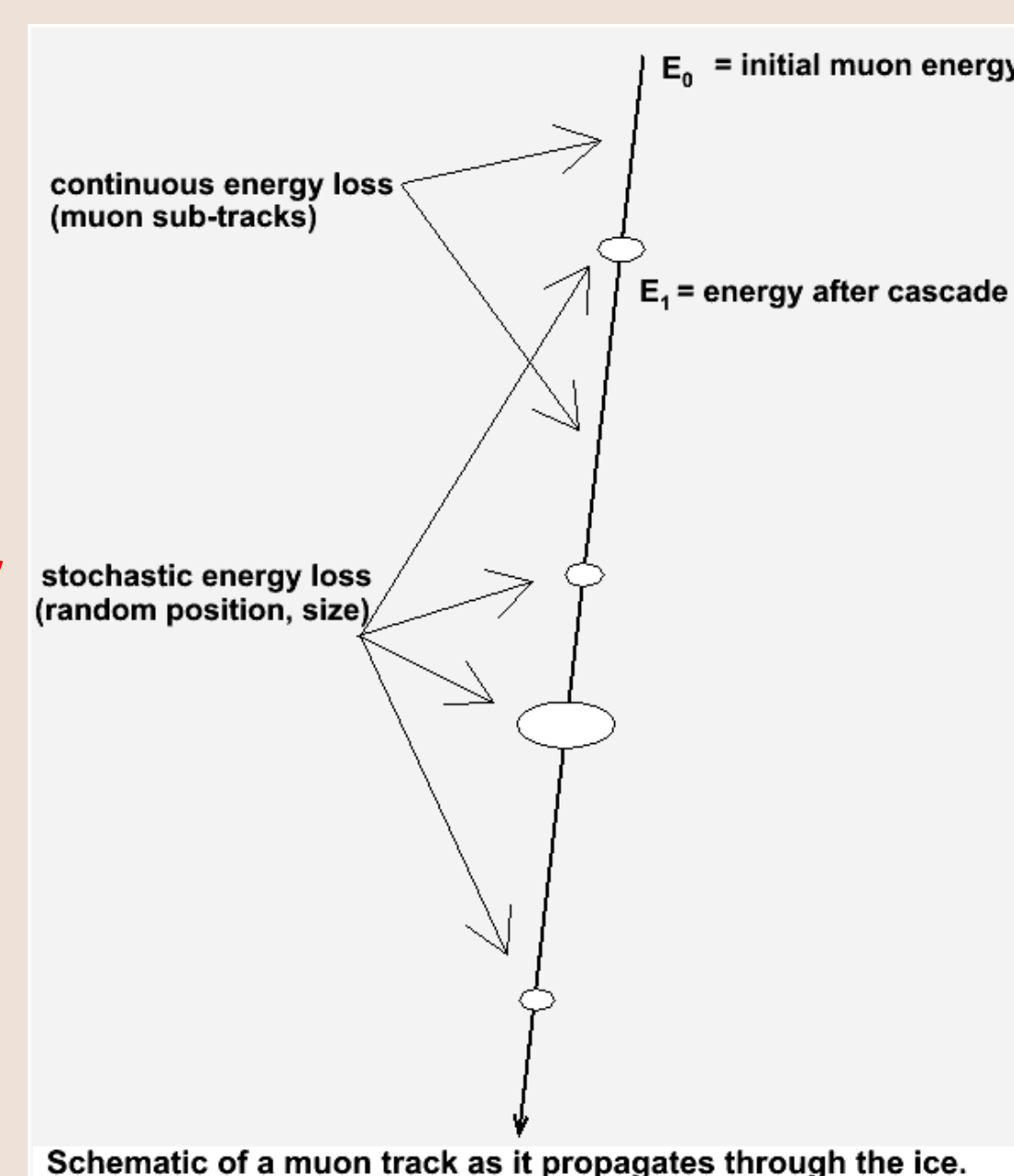
Implementation:

A) muons

- 1) To utilize the parallel processing capability of a GPU, independent events (muons passing through detector volume) are processed at the same time. **Each muon track is handled by one thread-block.**
- 2) Within a block, **one thread is used for each photon produced by the parent muon.**
- 3) The emission point of the photon on the muon track is selected randomly. The distribution is uniform over a sub-track, but not for the entire track.
- 4) Wavelength dependence is accounted for by performing 30 separate simulations; one for each wavelength bin of 10 nm.

B) Ice

- 1) The measured ice properties are used in the simulation: 174 layers each 10 m deep.
- 2) Each layer has a different absorption and scattering lengths. There are clear layers as well as highly scattering/absorbing layers due to the presence of air bubbles, dust, volcanic ash, etc.
- 3) The Henyey-Greenstein phase function with ($g=0.8$) is used for scattering calculations.



Implementation (cont'd)

C) The detector.

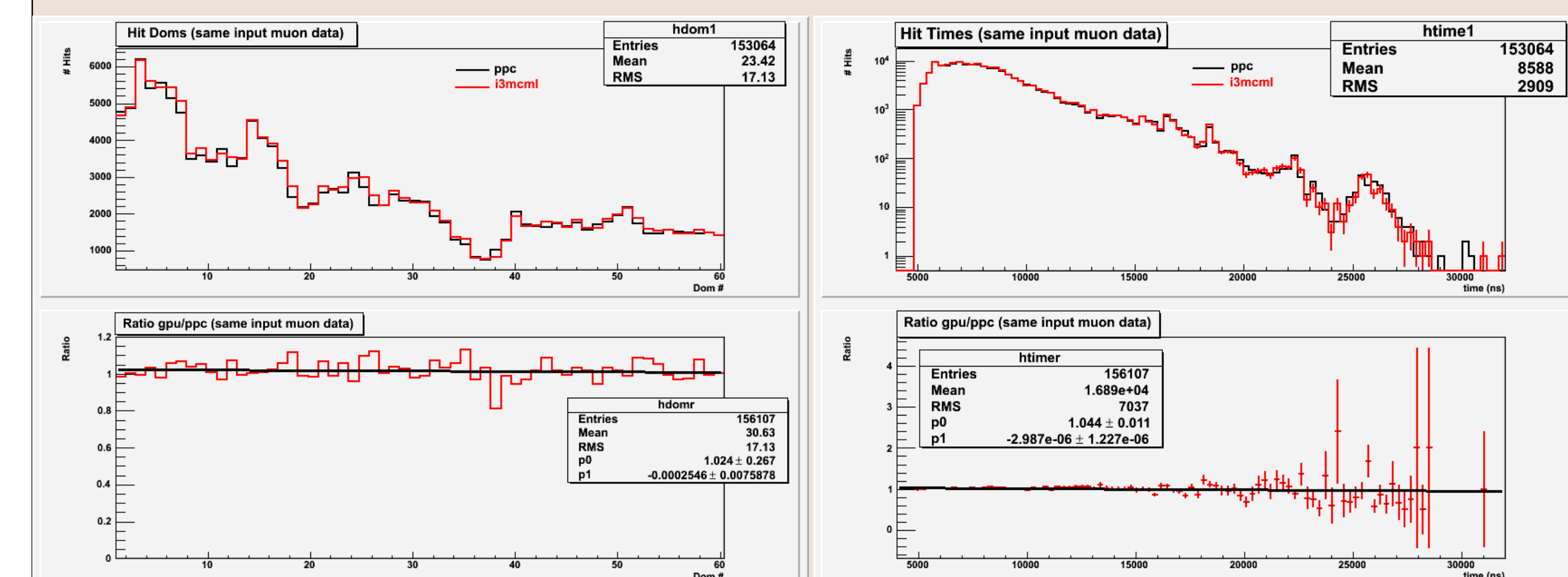
- 1) A photon is detected if it hits a detector module. The test for detection involves testing for the intersection of a line segment with a sphere.
- 2) A 3-d grid is used to keep track of where a photon is and which optical module (if any) it should be tested against.
- 3) The grid was constructed such that each cell contains at most one module. A module however can cross a cell boundary, and therefore can be in more than one cell.
- 4) Most cells do not contain modules: The inter-string separation is ~125 m, and the typical scattering length is on the order of a few meters.
- 5) Scattering is allowed only in-layer and layer number is used as the z-component of the detector grid.

Performance considerations:

- 1) Ice data stored in constant memory (cached).
*this is in part the reason why multiple simulations are used at different wavelengths. The overhead of running multiple simulations is relatively small.
- 2) Detector Grid (x-y) stored in constant memory (cached). The actual position of a module is retrieved from global memory only if an intersection-test is to be performed.
- 3) Muon data copied to shared memory. The use of shared memory resulted in a small (~10%) performance gain compared to global memory
- 4) Fast math (less accurate) functions were used. We did not compare the effect on speed but we can see from comparisons with the CPU program (double precision) that there is no effect.

Results:

Simulations were run using both the GPU and the CPU programs using the same input data. Since these are Monte Carlo programs, agreement can only be tested in a statistical manner. The histograms show distribution of hit optical modules and photon arrival time.



Acknowledgments:

The IceCube simulation described here is, to a large extent, based on the program CUDAMCML².

References:

- 1) <http://www.icecube.wisc.edu>
- 2) E. Alerstam, T. Svensson and S. Andersson-Engels, "Parallel computing with graphics processing units for high-speed Monte Carlo simulations of photon migration", Journal of Biomedical Optics Letters, 13(6) 060504 (2008)