Photon Transport Simulation

First Report

Sept, 2003

Nathaniel Tagg – University of Oxford

A PhotonTransport plugable object contains one or more of the following components. Each is used with a single function call:

Photon Computer

Blue Photon Tracker

WLS Fibre Model

Green Photon Tracker

```
Class PhotonTransportModule:: public Tobject
{
 PhotonTransportModule() { fRandom = gRandom; };
 virtual ~PhotonTransportModule() {};

 void SetRandom( TRandom* r ) { fRandom = r; };

 // These functions  need overriding:
 virtual void Reset(const VldContext& cx) { fContext = cx; };
 virtual void Configure( const Registry& ) {};

 // At least one of:

   virtual void ComputePhotons( const DigiScintHit* inHit,
                                PhotonCount& outCount );

 virtual Bool_t ScintPhotonToFibreHit( const DigiScintHit*    inHit,
                                       const UgliStripHandle& inStrip,
                                       DigiPhoton*            &outPhoton )

 virtual Bool_t FibreHitToGreenPhoton( const UgliStripHandle& inStrip,
                                       StripEnd::StripEnd_t   inDirection,
                                       DigiPhoton*            inBluePhoton,
                                       DigiPhoton*            &outGreenPhoton )

 virtual Bool_t GreenPhotonToPe( const UgliStripHandle& inStrip,
                                 DigiPhoton*            inGreenPhoton,
                                 DigiPE*                &outPe)
```

Photon Computer

Determines the number of photons that will be obtained from the DigitScintHit.

Works in one of several ways, by demanding certain number of blue photons, green photons, or photoelectrons.

Current Implimentation: Blue Photons ("Dead Reckoning")

- Take dE/dx from Hit
- Compute number of blue photons created using Birk's law.
- Tell the caller to simulate the correct number of blue photons

Ref: PDG and GMINOS code.

Example 2: Photoelectrons By Mapper Data

- Find the energy deposited in the strip by the hit at the position of the hit
- Use the Mapper or Muon calibration to determine the strip efficiency
- Use the Mapper or Muon calibration to determine the light output from this position along the strip
- Compute the expected number of photoelectrons.
- Tell the caller to run photons until N photoelectrons have successfully made it to the phototube.

Blue Photon Tracker

Makes a green photon from the DigiScintHit, and tracks it until it gets absorbed or gets into the fibre.
Returns the location and time of the hit, along with a DigiPhoton representing the end of the blue photon track.

Current Implimentation: Simple simulation
- Create a blue photon along the track of the DigitScintHit with random direction.
- Use the bulk properties of a generic blue photon: attenuation length, TiO2 refelctivity, etc,
- Track the photon until abosorbed or interacts in fibre
- Output the location and time of the hit.

Base of code by Brian Rebel
Ref: Lightyield.f from Keith Ruddick, via P Miyagawa via p Achenbach via...

Example 2: PDFs (much faster, requires work)
Use generic distributions of the time and displacement of the blue photon.

Example 3: Full simulation
Create the blue photon from the known wavelength spectrum.
Simulate using the full suite of wavelength-dependent measurements.

WLS Fibre Model

Creates a green photon from the blue photon.

Adds decay time of flouror.
Calculates green photon direction.
Throws out green photon if it's not trapped in the fibre.

Current Implimentation:
Contains no wavelength dependencies. This should be a good approximation.

Creates the green photon randomly in the volume of the fibre core at the longitudinal position of the blue fibre. (This is strictly wrong, and should include a simulation of the attenutation length of the blue photon in the fibre.)

Traps the green photon if it survives the first reflection.

Ref: CP Achenbach and JH Cobb, J. Opt A: Pure Appl. Opt 5 (2003) 239-249

Tracks the green photon and creates a DigiPE if it survives.

Current Implimentation:
Tracks time and attenuation of photon through green fibre (including pigtails) and green fibre.

Contains no wavelength dependencies; uses double-exponential instead.

Tracks the transport time using the longitudinal cosine of the green photon direction.

Makes no losses at connectors or due to fibre bends.

Ref: Uses the bulk attenutaion double-exponential formulae (green and clear) taken from GMINOS

Future work:
- Could use a complete wavelength spectrum, and fold in the wavelength dependence of the phototube.

- Could take attenutation curve from Mapper data (but would need correcting for the different photon-paths)

Green Photon Tracker

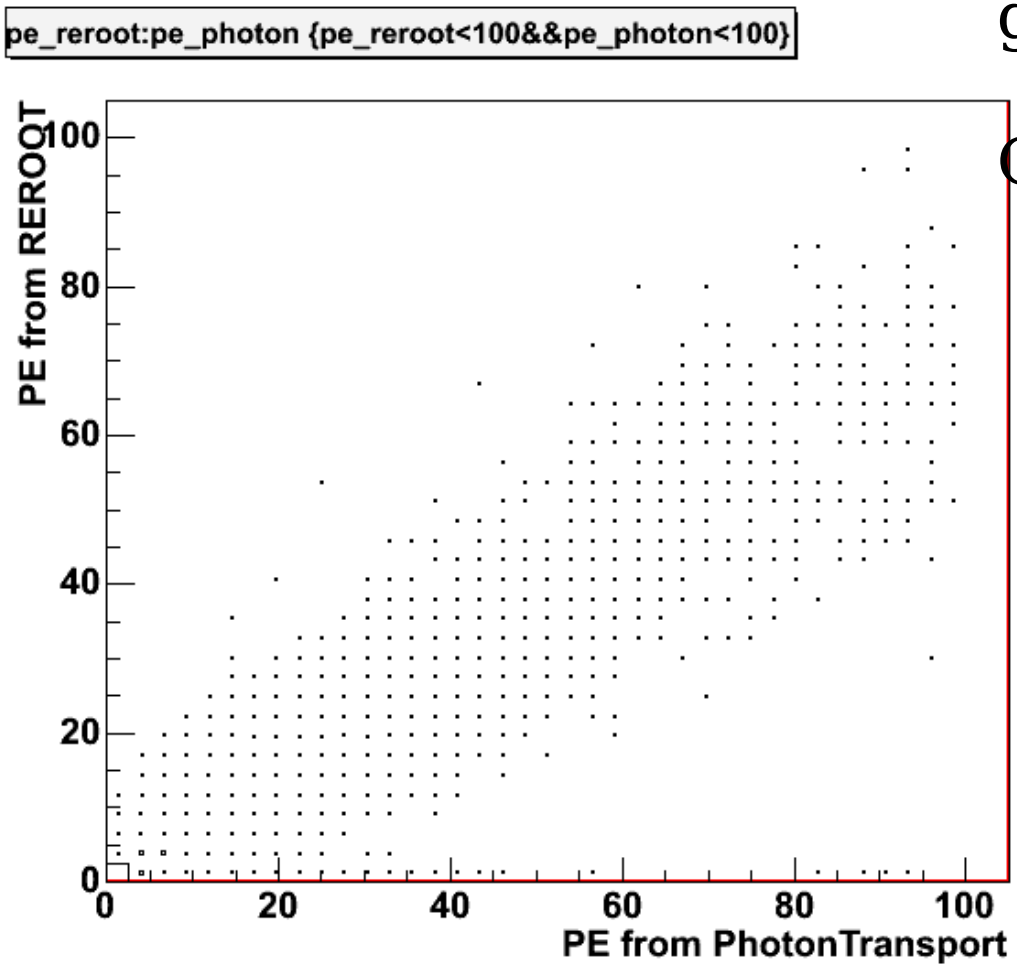
Photon Computer:

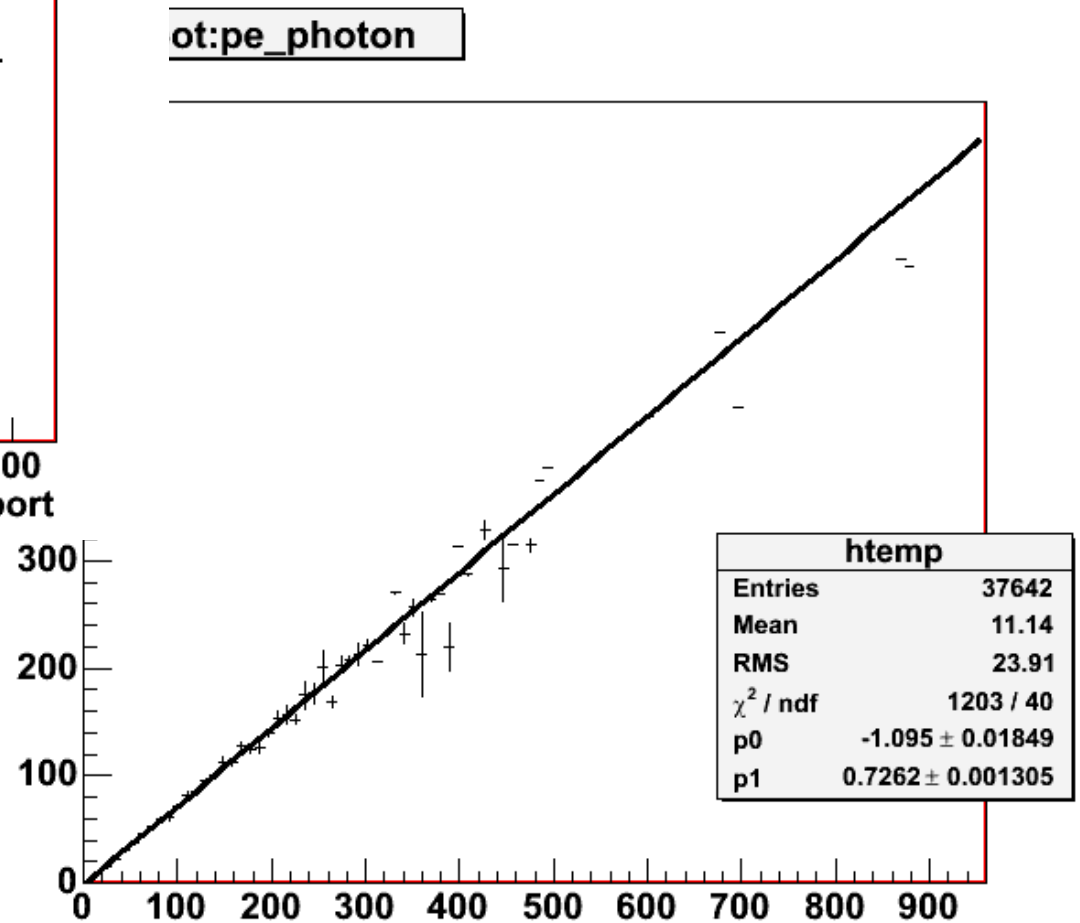
```
//get the number of photons
 //from the PDG 2002 Section 27 on Particle Detectors you get
 //about 1 photon/100eV of energy deposited.
 // .. so, 10photons/keV = 10^7 photons/GeV

 // From rough comparison with REROOT results, the total
 // constant for this algorithm should be ~27500

 // Here's what I can account for:
 const double kLightYield = ( 1/100.  // photons/eV
                              * 1e9 // GeV/ev
                              * 0.13 // QE of phototubes
                              * 0.20 // Trapping probability in fibre (for all angles)
                              );

 // This fudge factor comes from a comparison of using GMINOS FLS digits
 // and this method.  It has no physical justification.
 const double kFudgeFactor = 1.0;

 // Apply normalization and birks' constant.
 double meanBlue = kLightYield * kFudgeFactor
   * fOverallLightOutput
   * dE/(1. + fBirksConstant*dEdx);
```

Photon Computer:

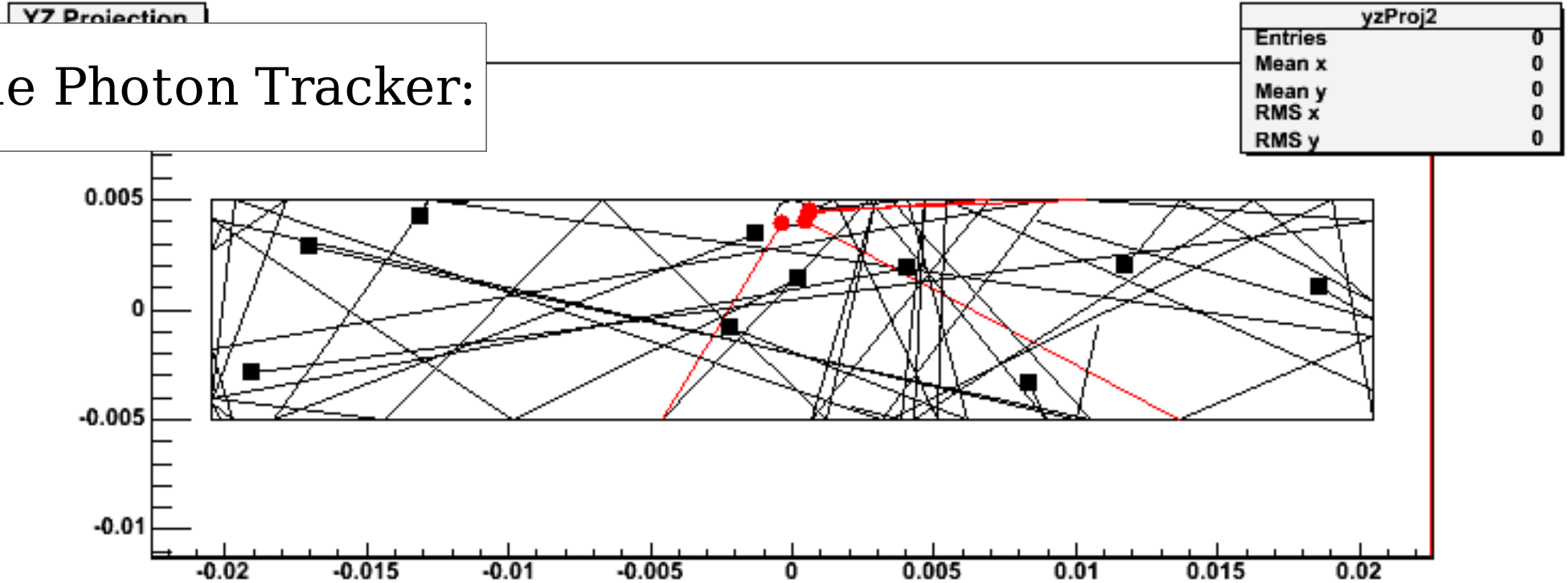
Hits from first 100 events of gm_far_le_noosc_1_hjun02.root

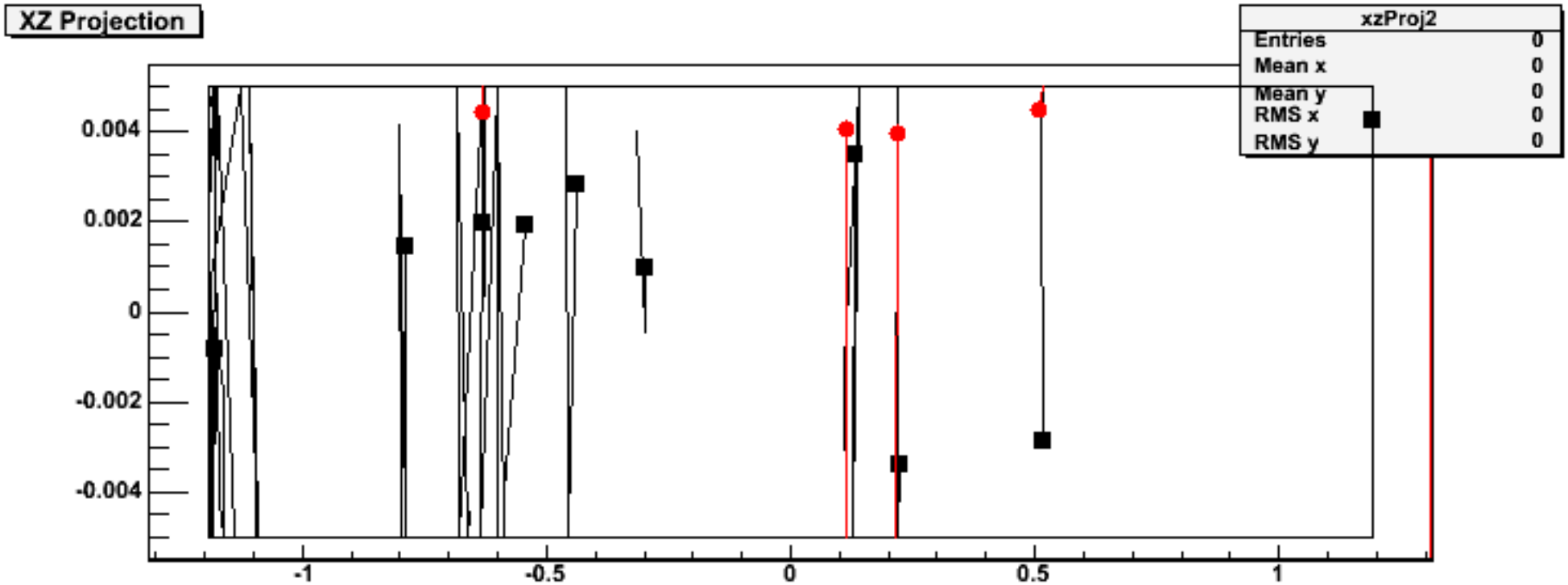
Compare stripend-by-stripend



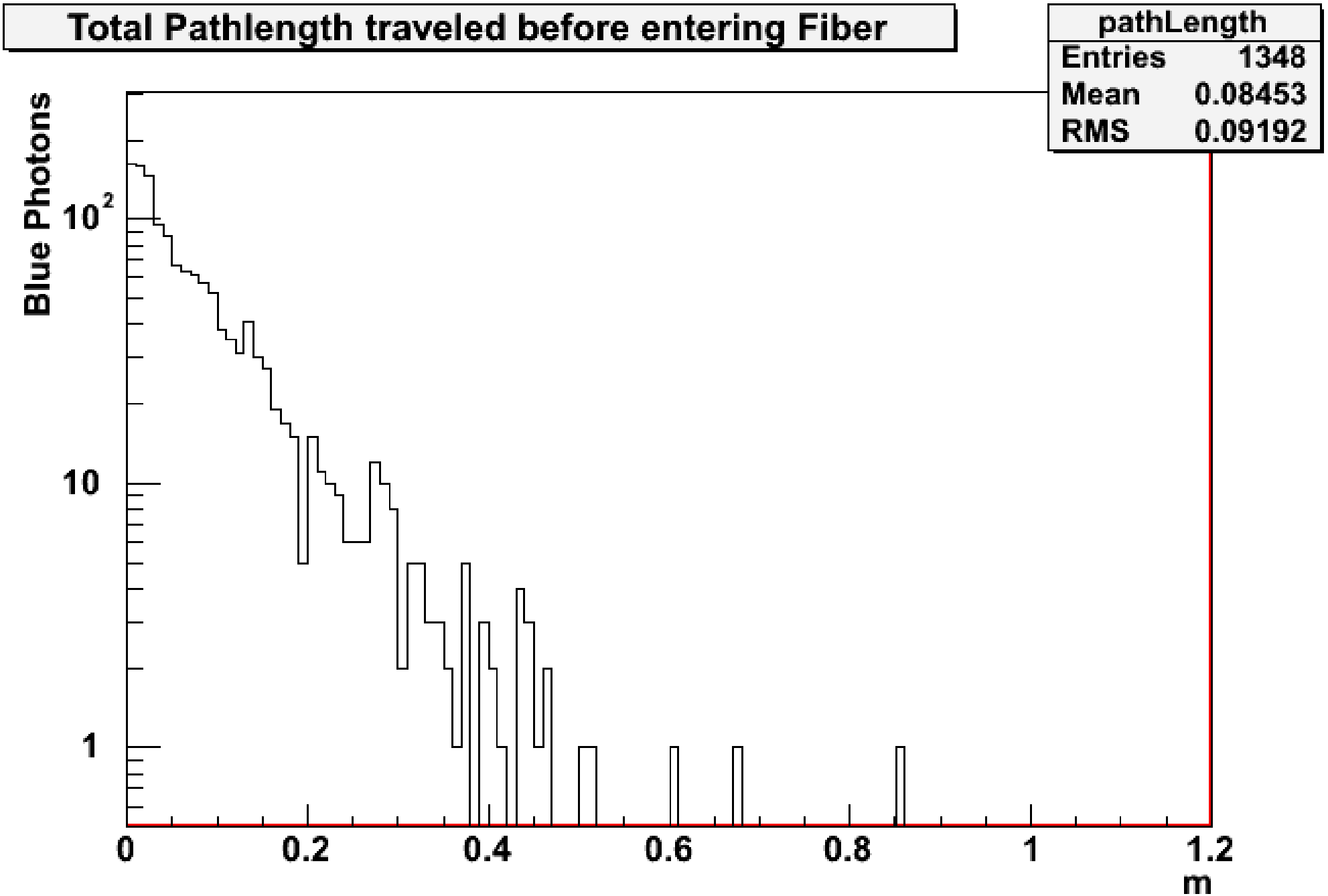


Blue Photon Tracker:

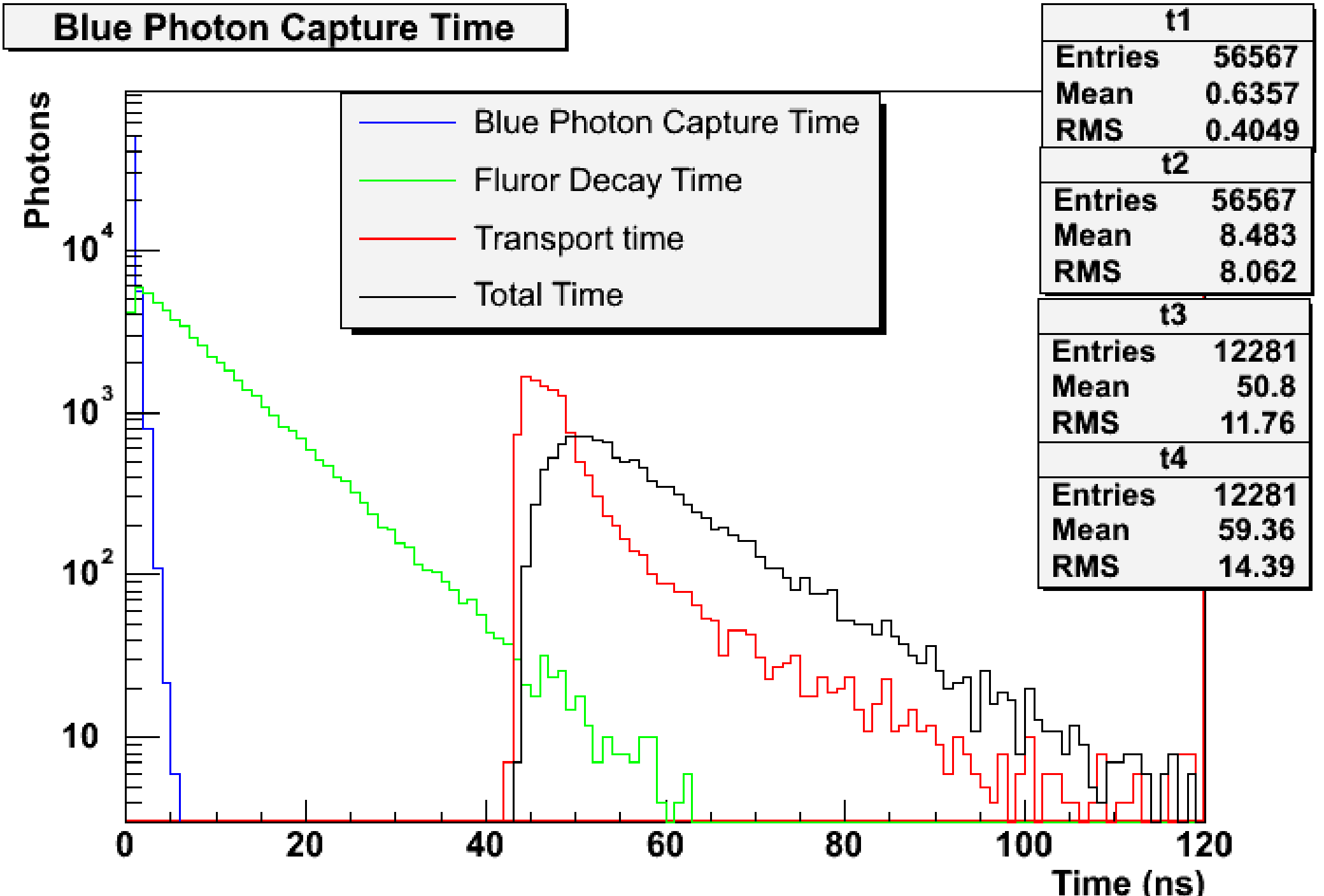




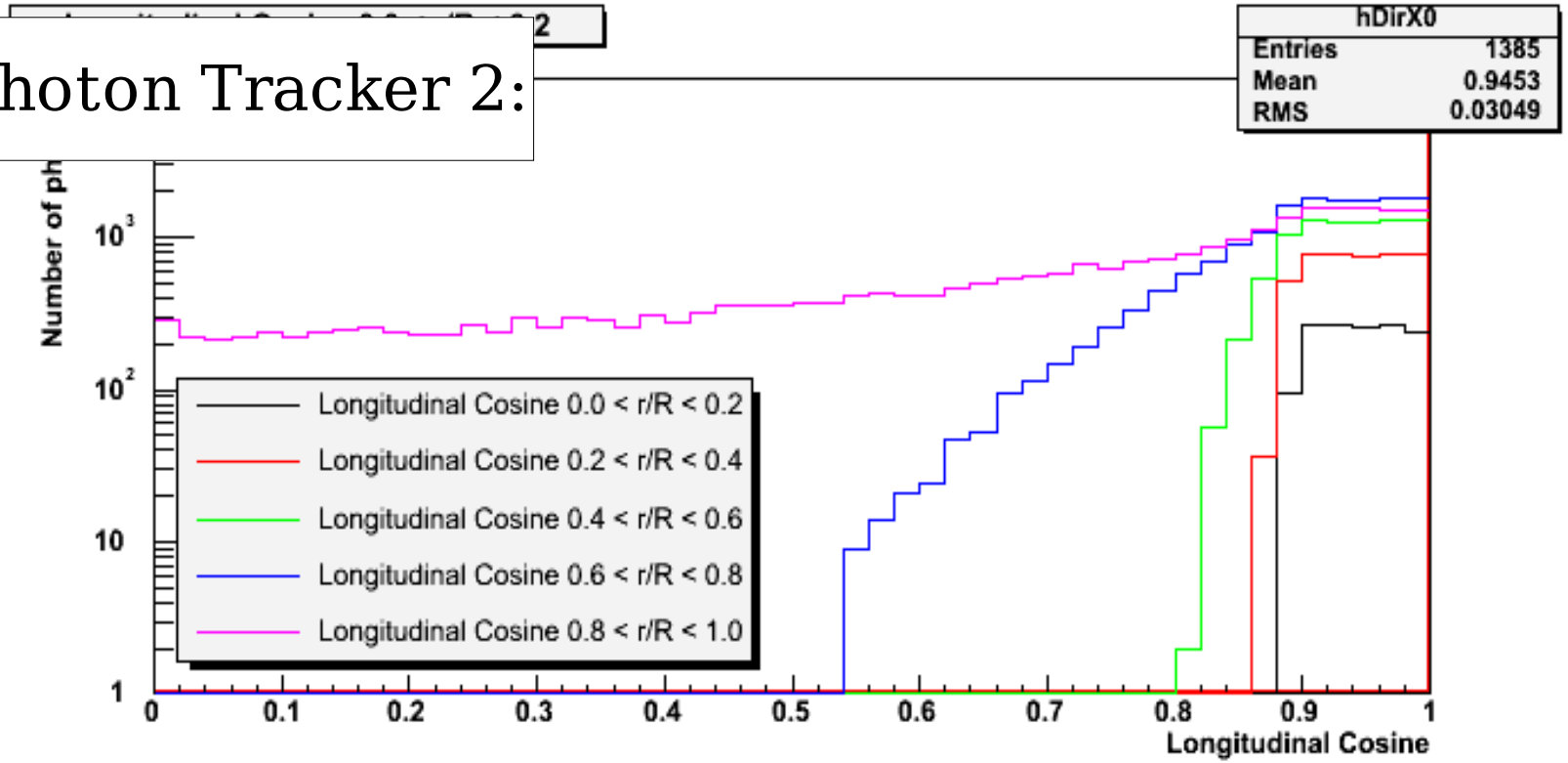
Blue Photon Tracker:

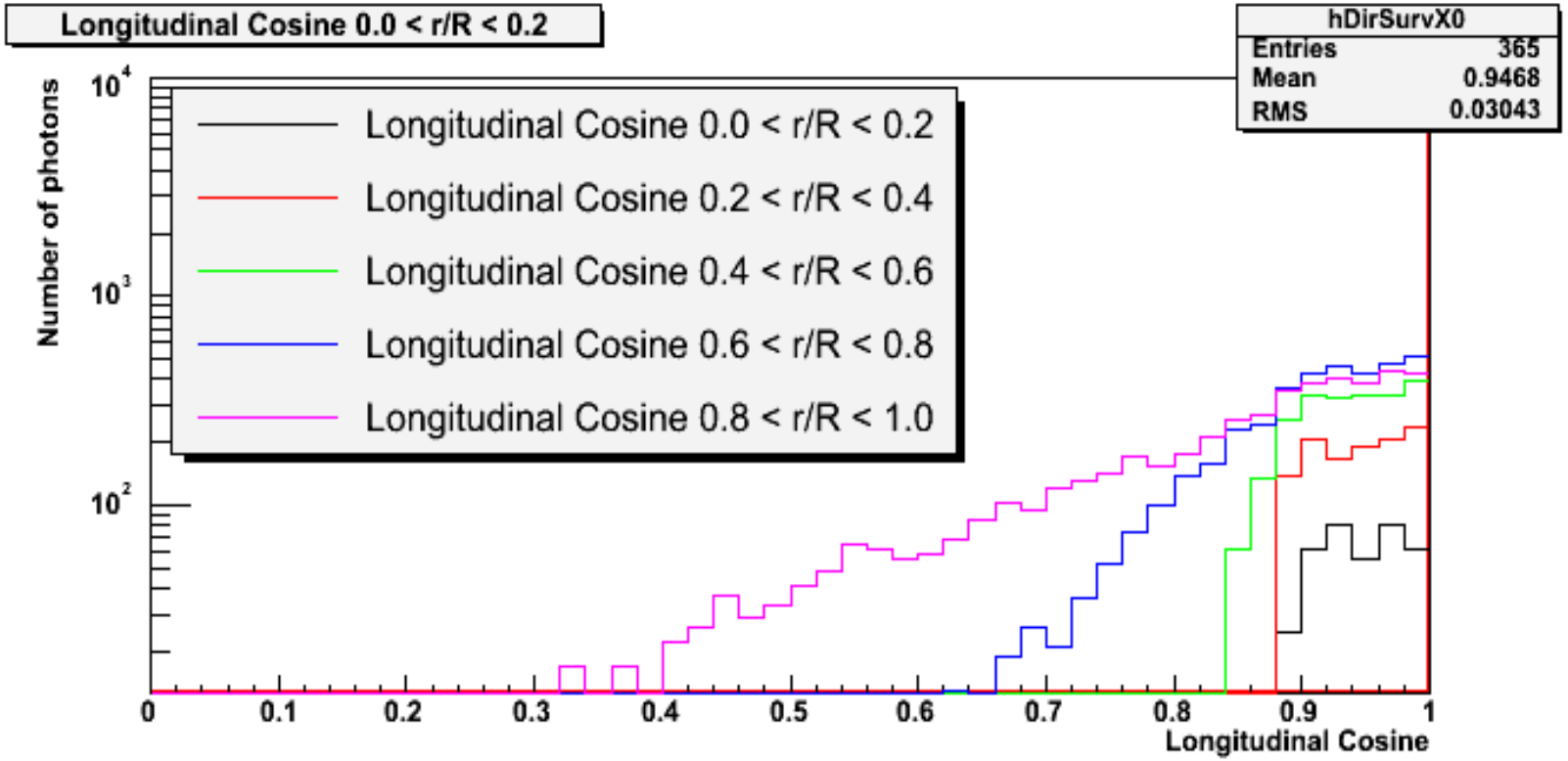


WLS/Green Photon Tracker 1: For ½ way down a 2 m FD strip:



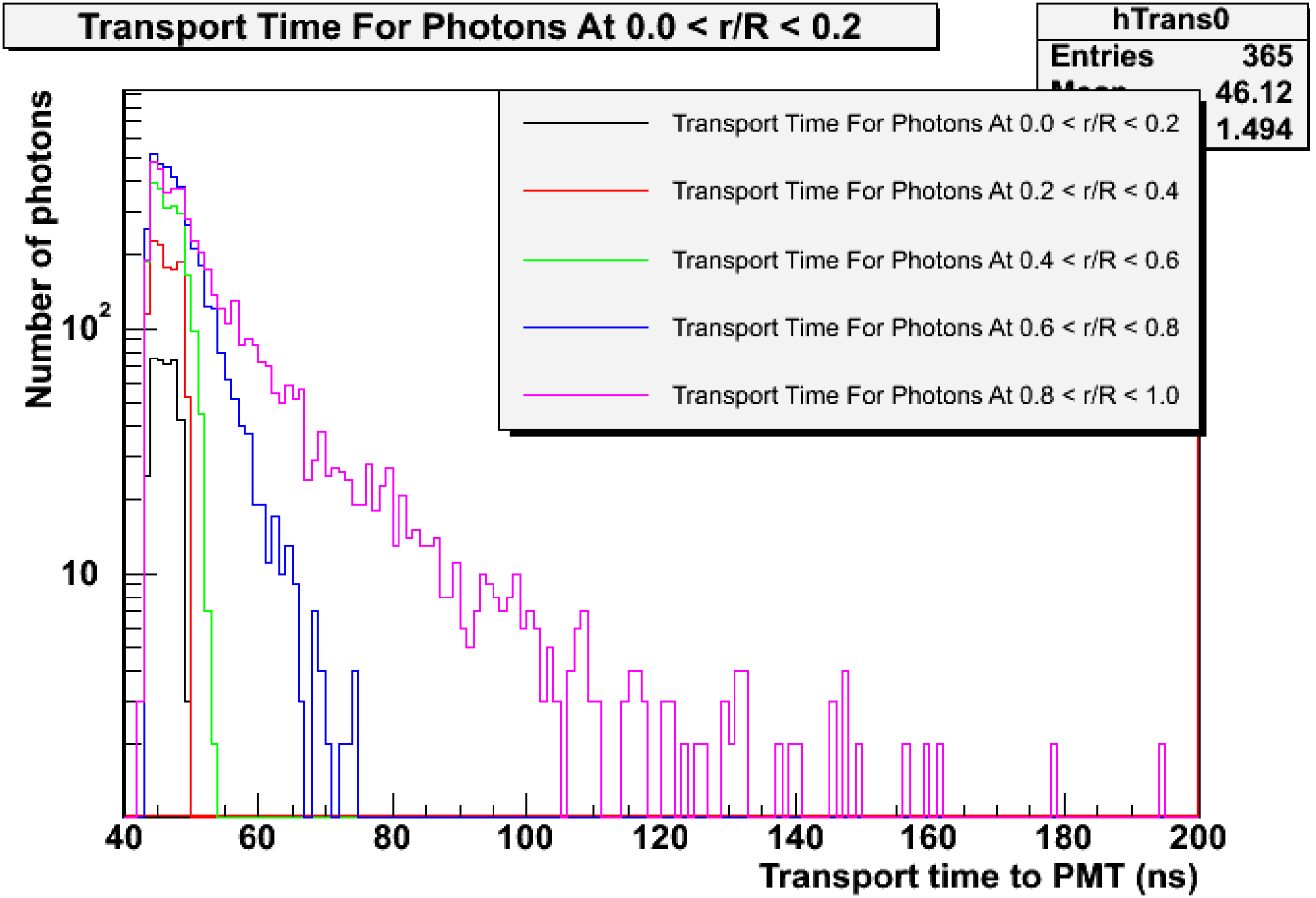
WLS/Green Photon Tracker 2:





For ½ way down a 2 m FD strip

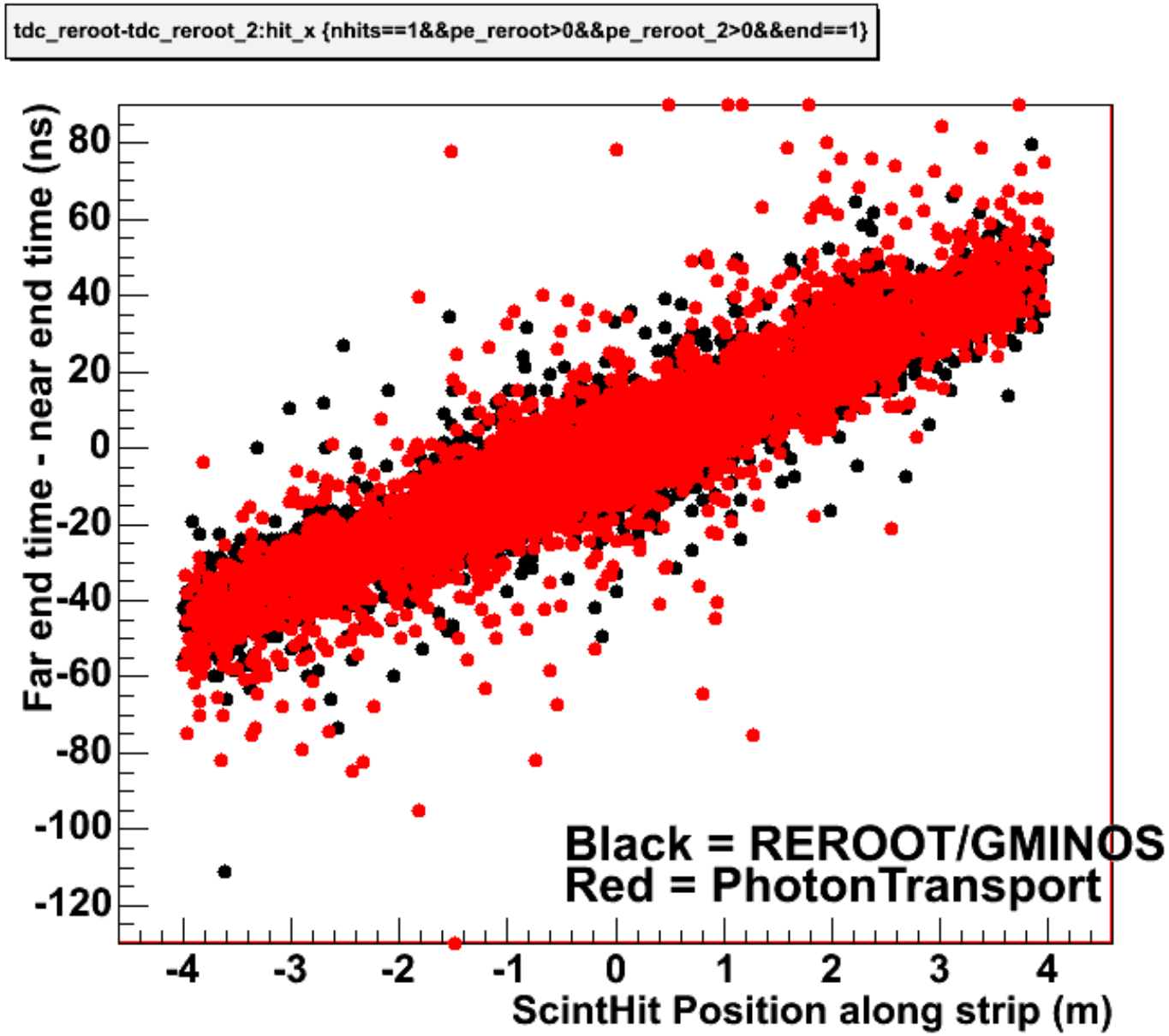
WLS/Green Photon Tracker 3: For ½ way down a 2 m FD strip:



Hits from first 100 events of
gm_far_le_noosc_1_hjun02.root

Compare stripend-by-stripend

Compare t1-t2
for GMINOS
and Photon:
same basic result



To Do / Problems:

- Doesn't work on Near Detector due to UgliGeom problems (FD, CalDet seem OK).

- Build new PhotonComputer that uses strip-to-strip calibrations

- Get green fibre penetration right

- Add wavelength dependent blue photon model

- Needs documentation

- Needs testing.

Longer term:

- Add PDF-based blue photon model for speed