Subject: [SOLVED] PVertexFile
Posted by Michael Kunkel on Wed, 13 Jun 2012 21:01:24 GMT
View Forum Message <> Reply to Message

Greetings,

I had a question on the format of PVertexFile.
Is it one leaf of array vX:vY:vZ, or seperate leafs?

I ask because I am trying to smear vertices along a target that is 400mm in length and 20mm in radius.
I create a uniform distribution of 50M events of a cylinder equally the dimensions mentioned, called vertex.root with separate leaves vX, vY, vZ, tree name "vertex" as prescribed in http://www-linux.gsi.de/~hadeshyp/pluto/v5.40/PVertexFile.html

When I attempt to use this

```
//#include "loadPluto.h";
//Program to generate multiple PLUTO root file
//Author Michael C. Kunkel
#include "TH1.h"
#include "TH2.h"
#include "TH3.h"
#include "TChain.h"
#include "TCanvas.h"
#include "TF1.h"
#include "/w/hallb/clasg12/mkunkel/PLUTO/pluto_v5.40/src/PParticle.h"
#include "/w/hallb/clasg12/mkunkel/PLUTO/pluto_v5.40/src/PReaction.h"
#include "/w/hallb/clasg12/mkunkel/PLUTO/pluto_v5.40/src/PBeamSmearing.h"
#include "/w/hallb/clasg12/mkunkel/PLUTO/pluto_v5.40/src/PVertexFile.h"


void Vertex_Simulation(){

   gROOT->Reset();

   char nam1[60] = "/volatile/clas/clasg12/mkunkel/GG_ETA_SIM/PLUTO_GEN/eta_";

   char nam2[25] = "_gammagamma_vertex";

   for(int ij = 1; ij<=1; ij++){
  char c[10];
  sprintf(c,"%d",ij);
  char creater[75];
  sprintf(creater,"%s%s%s",nam1,c,nam2);
  cout << creater<<endl;


   double ebeam_min = 1.1725;
   double ebeam_max = 5.44575;
   PBeamSmearing *beam_smear = new PBeamSmearing("beam_smear", "Beam smearing");
```

```
    TF1* beam_smear_fn = new TF1("beam_smear_fn", "1./x", ebeam_min, ebeam_max);

    beam_smear->SetReaction("g + p");
    beam_smear->SetMomentumFunction(beam_smear_fn);
    makeDistributionManager()->Add(beam_smear);


    PReaction my_reaction("_P1 = 2.2","g","p","p eta [g g]",creater,1,0,1,0);

       //Construct the vertex container:
    PVertexFile *vertex = new PVertexFile();

vertex->OpenFile("/w/hallb/clasg12/mkunkel/PLUTO/ETA_GAMMAGAMMA_SIM/VERTICES/v
ertex.root");
    //add to prologue action
    my_reaction.AddPrologueBulk(vertex);
    // my_reaction.Print();   //The "Print()" statement is optional
    my_reaction.Loop(50000);

    }

}
```

I receive a segmentation fault, if I remove the vertex the code runs correctly.

Subject: Re: PVertexFile
Posted by Michael Kunkel on Thu, 14 Jun 2012 20:07:44 GMT
View Forum Message <> Reply to Message

I have tried various ways to fill a ntuple with vertex information.
Named the leafs vX, vY, xZ as prescribed in the link previously given. I have made it into array of vertex[3], tried with and without seqNr and still when I run the PLUTO code previously given I see this error

```
Info in <PParticle::operator+>: (ALLOCATION) The composite g + p has been added
Info in <PStaticData::AddDecay>: (ALLOCATION) Decay index 14001: g + p --> p + eta
Info in <PStdModels::GetModels>: Read std models
Info in <PDistributionManager::Attach>: Re-iteration of std plugin done
OPENING VERTEX FILE <<DONE with a COUT statement from code>>
PReaction: calculating widths in PData...

 *** Break *** segmentation violation
```

Any help is appreciated.
Thanks
Michael C. Kunkel

Subject: Re: PVertexFile
Posted by Ingo Froehlich on Fri, 15 Jun 2012 10:45:40 GMT
View Forum Message <> Reply to Message

Dear Michael,

just for testing I have created a small script for generating the vertex events as follows:

```
TFile *f = new TFile("Vertex.root", "RECREATE");
TNtuple *ntuple = new TNtuple("vertex", "Vertex data", "vx:vy:vz");

PReaction my_reaction;
my_reaction.Do("vz = sampleFlat() * 400 - 200;");
my_reaction.Do("loop: vx = sampleFlat() * 20 - 10; vy = sampleFlat() * 20 - 10; ");
my_reaction.Do("if ((vx*vx + vy*vy) > 100); goto loop");
my_reaction.Output(ntuple);

cout << my_reaction.Loop(10000) << " vertex events created" << endl;

f->cd();
ntuple->Write();
f->Close();
```

At least running a part of your script:

```
PReaction my_reaction("_P1 = 2.2","g","p","p eta [g g]","delme",1,0,1,0);

//Construct the vertex container:
PVertexFile *vertex = new PVertexFile();
vertex->OpenFile("Vertex.root");
//add to prologue action
my_reaction.AddPrologueBulk(vertex);
// my_reaction.Print();   //The "Print()" statement is optional
my_reaction.Loop(10000);
```

seems to work for me...

Subject: Re: PVertexFile
Posted by Ingo Froehlich on Fri, 15 Jun 2012 10:51:51 GMT
View Forum Message <> Reply to Message

Just for additional info:

one can of course combine the scripts and access the vertex info directly via the variables _event_vertex_*:

```
PReaction my_reaction("_P1 = 2.2","g","p","p eta [g g]","delme",1,0,1,0);
my_reaction.Do("vz = sampleFlat() * 400 - 200;");
my_reaction.Do("loop: vx = sampleFlat() * 20 - 10; vy = sampleFlat() * 20 - 10; ");
my_reaction.Do("if ((vx*vx + vy*vy) > 100); goto loop");
my_reaction.Do("_event_vertex_x=vx; _event_vertex_y=vy; _event_vertex_z=vz; ");
my_reaction.Loop(10000);
```

Subject: Re: PVertexFile
Posted by Michael Kunkel on Fri, 15 Jun 2012 11:46:09 GMT
View Forum Message <> Reply to Message

I will attempt this. One more question that might seem silly. What units are required and outputted for vertex information? I seem to not seem this in documentation.

Thanks Again

Subject: Re: PVertexFile
Posted by Ingo Froehlich on Fri, 15 Jun 2012 11:50:17 GMT
View Forum Message <> Reply to Message

I also had to search, and found in PParticle.h that it should be in mm