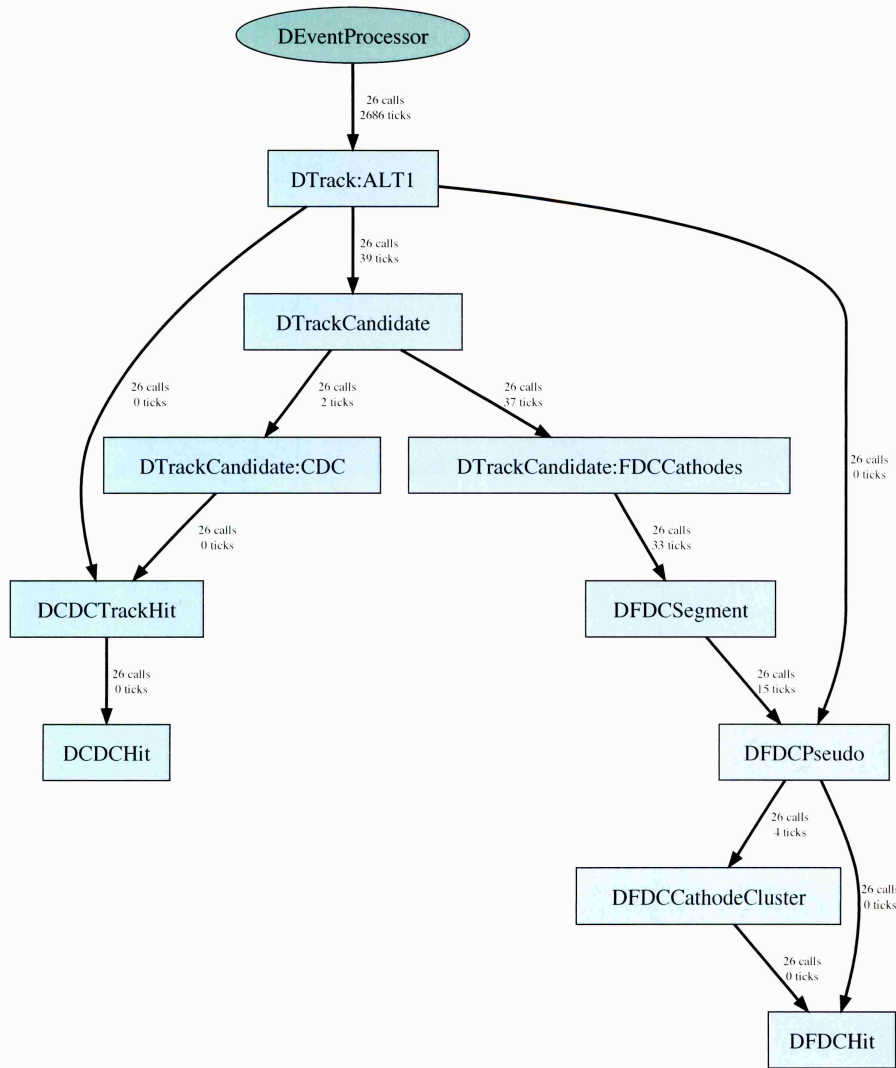


Highlights of Tracking Related Work Done Since March Tracking Review

May 16, 2008 David Lawrence

Modularization of Tracking Code

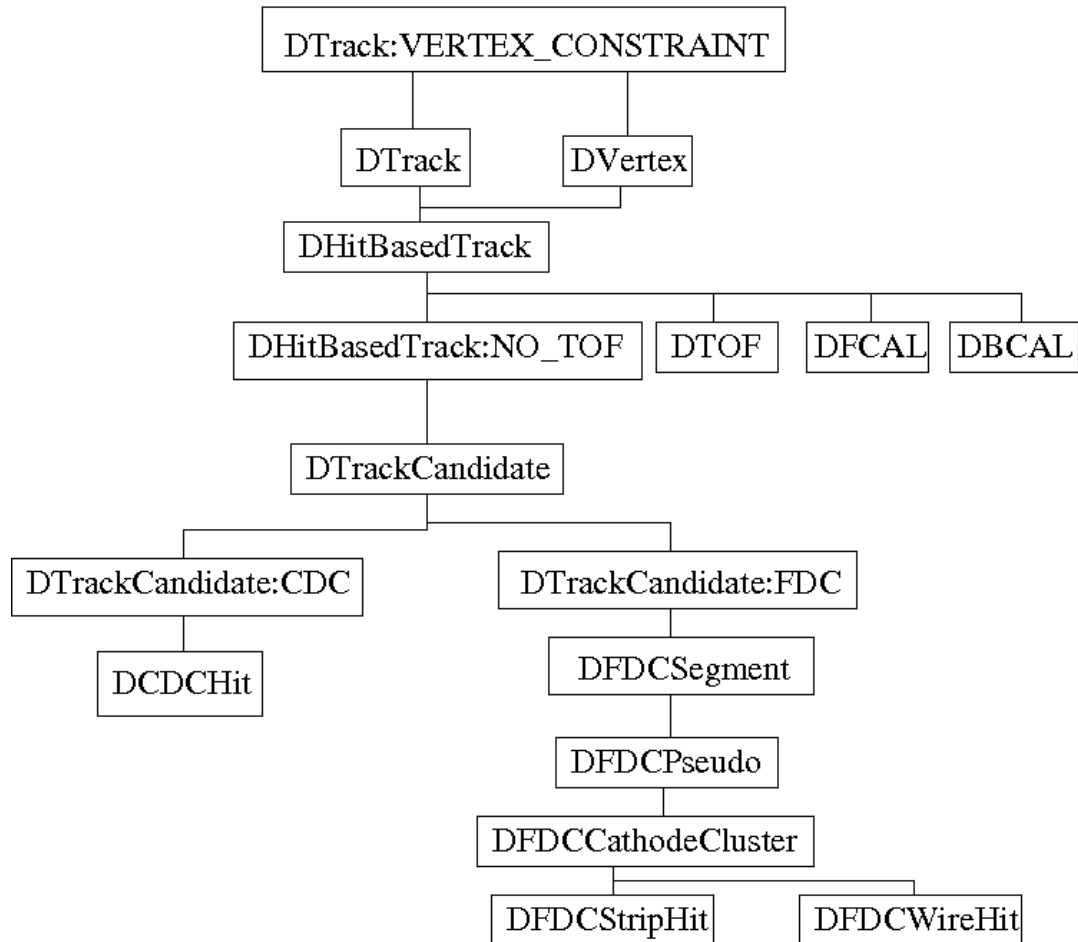


On April 4th Simon, Mark and Dave met to discuss how the information flow for track reconstruction could best be broken up into “factories”.

The diagram to the left shows the current state of the code as observed by JANA.

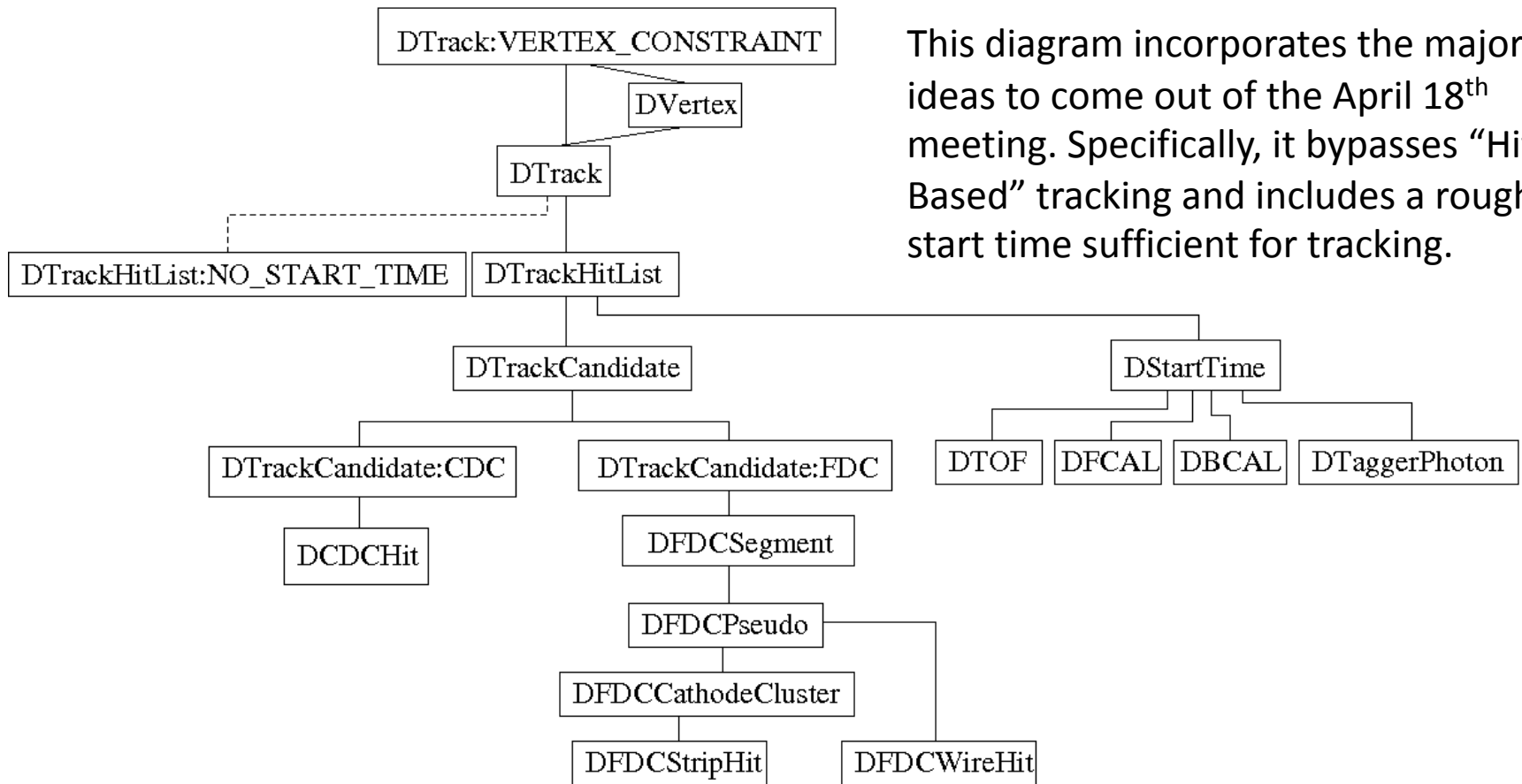
The main problem with the current design is that it does not allow a clean and obvious way to include the start time that will be needed before the final tracking (using drift times) can be done.

Modularization of Tracking Code cont.



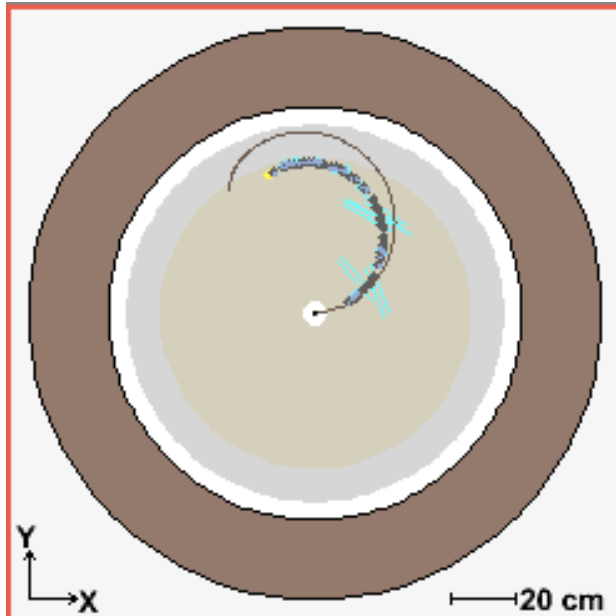
This diagram was presented by David as a starting point to the discussion. It was scribbled on a bit and at the end, several significant suggestions were made on how it might be improved....

Modularization of Tracking Code cont.



This diagram incorporates the major ideas to come out of the April 18th meeting. Specifically, it bypasses “Hit-Based” tracking and includes a rough start time sufficient for tracking.

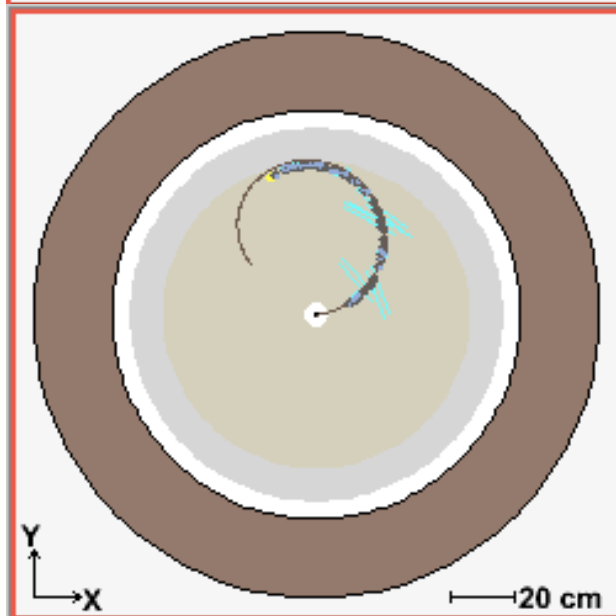
Utilize B-Field Map in CDC Track Finder



***Uniform
B-field***

One of the priorities for tracking reconstruction is to improve the robustness of the current finder and fitter.

Part of this includes improving the quality of the candidate parameters coming from the CDC track finder.

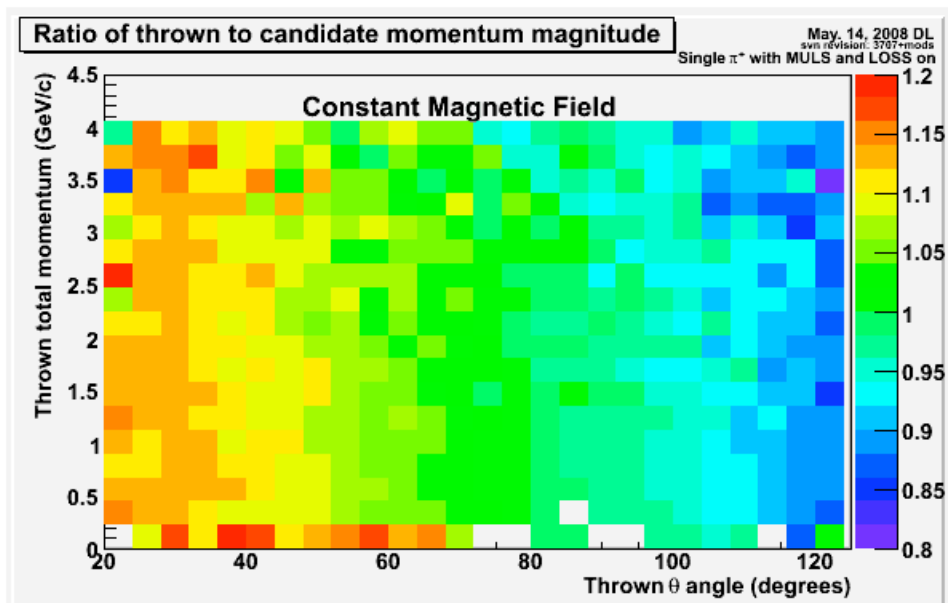


***Space-Point
Averaged
B-field***

The quality is somewhat limited by the “large” field gradient in the CDC. Until now, the parameters assumed a uniform field of -2.0T.

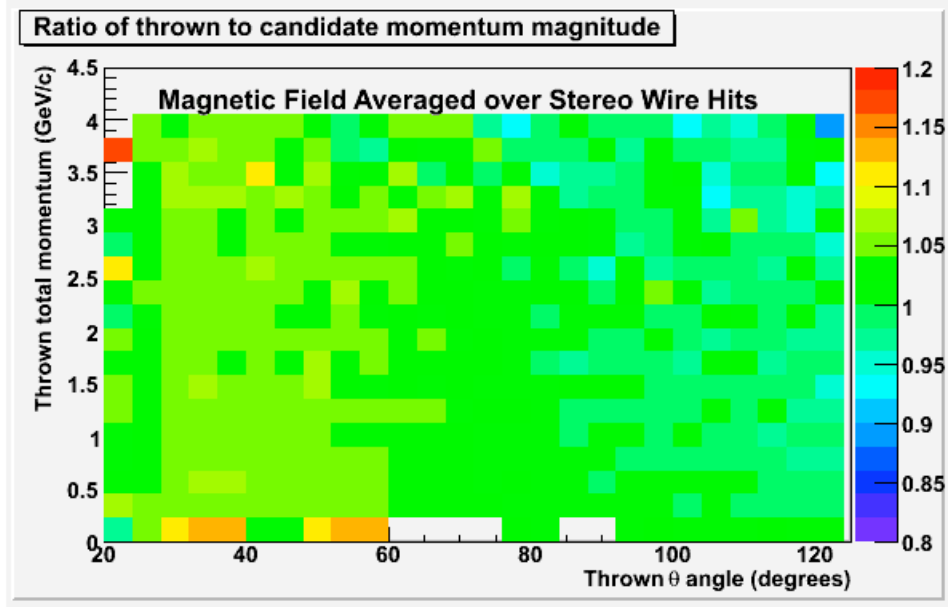
Recent changes scale the momentum by the average field at each of the stereo hits.

Utilize B-Field Map in CDC Track Finder



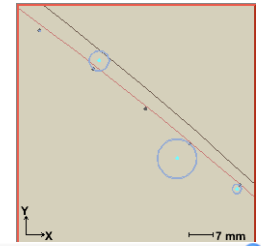
*Uniform
B-field*

These two plots indicate the determination of the total momentum in the CDC track candidates when using a uniform and space-point averaged value for the magnetic field.



*Space-Point
Averaged
B-field*

New Track Inspector in *hdview2*



Source: hidgeant.hddm

Hall-D Event Viewer

View Controls: -X, X+, -Y, Y+, -Z, Z+, ZOOM, Transverse Coordinates (x/y, r/phi), Event Controls (continuous, delay: 0.25), Info (Run: -----, Event: 2), Inspectors (Track Inspector, TOF Inspector, BCAL Inspector, FCAL Inspector), Quit

Track Draw Opt: DTrackCar, DTrack, DMCThrow, DMCTrajec

Hit Draw Opt: CDC, CDC Drift, CDCTruth, FDC Wire, FDC Pseu, FDC Inters, FDCTruth, TOF, TOFTruth, FCAL, FCALTruth, BCAL, BCALTruth

Track Info:

Thrown	trk:	type:	p:	theta:	phi:	z:
1	pi+	0.4754	0.8003	1.048	65	
---	---	---	---	---	---	---
---	---	---	---	---	---	---
---	---	---	---	---	---	---

Reconstructed:

trk:	type:	p:	theta:	phi:	z:
1	q=+	0.4359	0.8451	1.078	65.25
---	---	---	---	---	---
---	---	---	---	---	---
---	---	---	---	---	---

Hall-D Event Viewer:Track Inspector

Hit: 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0

Controls: Pan, Zoom, Event (Prev, Next), Info, reset

Track 1 (s-axis source): Data type: DTrack, Factory Tag: <default>, Track: 0

Track 2: Data type: DTrackCandidate, Factory Tag: CDC, Track: Best Match

Track 3: Data type: DMCThrown, Factory Tag: <default>, Track: Best Match

Track 4: Data type: <none>, Factory Tag: , Track:

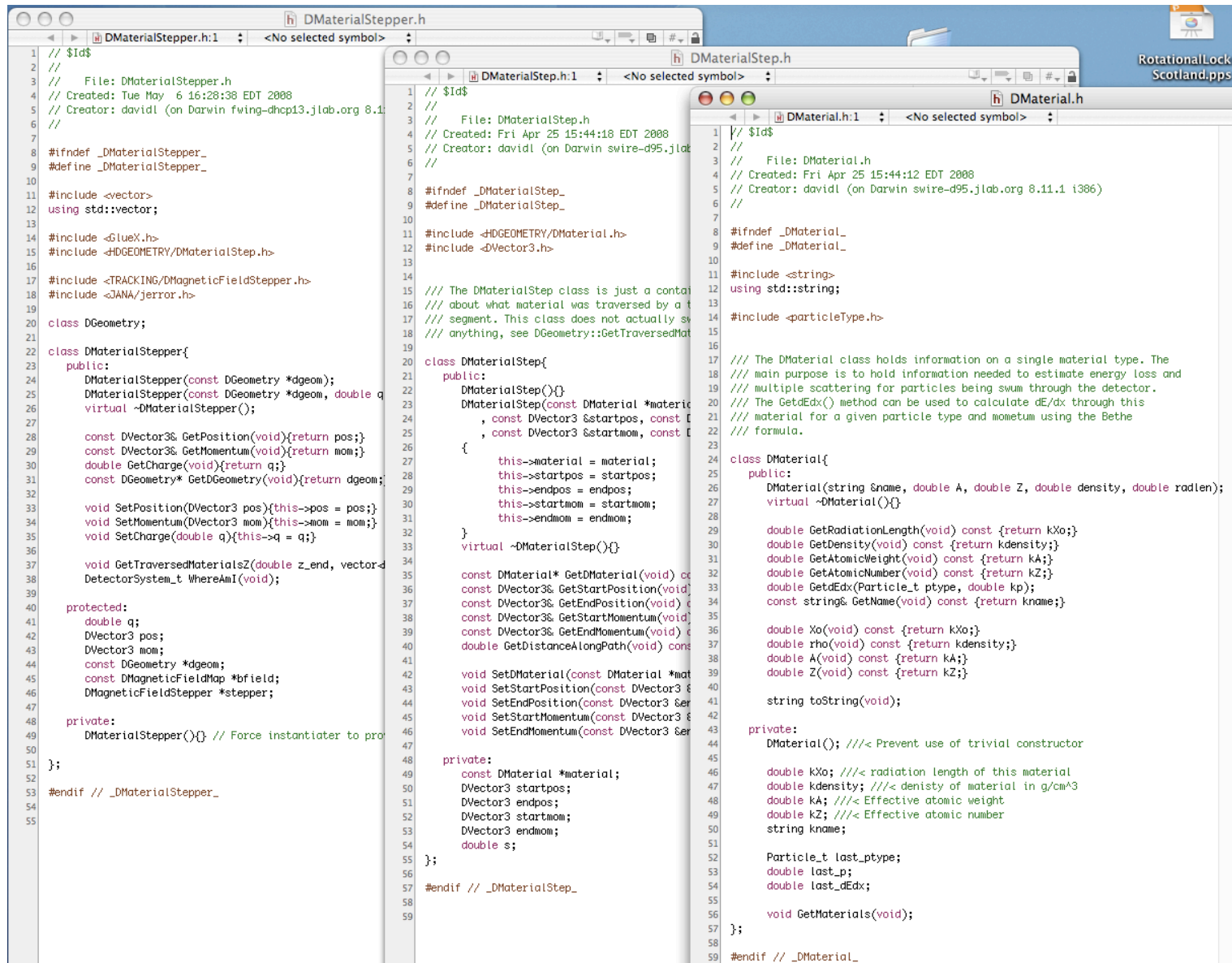
Track Info

Hit Info

residual plot: s (cm) vs resi (μm)

private, Kid's Videos, AlexNotes, Linux ISO, RHEL4, FC6, Microsoft Windows XP, RHEL5, FC8, Ubuntu 6.06.1

Started Implementing DMaterial classes



```
1 // $Id$
2 //
3 // File: DMaterialStepper.h
4 // Created: Tue May 6 16:28:38 EDT 2008
5 // Creator: davidl (on Darwin fwing-dhcp13.jlab.org 8.1
6 //
7
8 #ifndef _DMaterialStepper_
9 #define _DMaterialStepper_
10
11 #include <vector>
12 using std::vector;
13
14 #include <GlueX.h>
15 #include <HDGEOMETRY/DMaterialStep.h>
16
17 #include <TRACKING/DmagneticFieldStepper.h>
18 #include <JANA/jerror.h>
19
20 class DGeometry;
21
22 class DMaterialStepper{
23 public:
24     DMaterialStepper(const DGeometry *dgeom);
25     DMaterialStepper(const DGeometry *dgeom, double q);
26     virtual ~DMaterialStepper();
27
28     const DVector3& GetPosition(void){return pos;}
29     const DVector3& GetMomentum(void){return mom;}
30     double GetCharge(void){return q;}
31     const DGeometry* GetDGeometry(void){return dgeom;}
32
33     void SetPosition(DVector3 pos){this->pos = pos;}
34     void SetMomentum(DVector3 mom){this->mom = mom;}
35     void SetCharge(double q){this->q = q;}
36
37     void GetTraversedMaterialsZ(double z_end, vector<
38     DetectorSystem_t WhereAmI(void);
39
40 protected:
41     double q;
42     DVector3 pos;
43     DVector3 mom;
44     const DGeometry *dgeom;
45     const DMagneticFieldMap *bfield;
46     DMagneticFieldStepper *stepper;
47
48 private:
49     DMaterialStepper(){} // Force instantiator to pro
50
51 };
52
53 #endif // _DMaterialStepper_
54
55
```

```
1 // $Id$
2 //
3 // File: DMaterialStep.h
4 // Created: Fri Apr 25 15:44:18 EDT 2008
5 // Creator: davidl (on Darwin swire-d95.jlab
6 //
7
8 #ifndef _DMaterialStep_
9 #define _DMaterialStep_
10
11 #include <HDGEOMETRY/DMaterial.h>
12 #include <DVector3.h>
13
14
15 /// The DMaterialStep class is just a container
16 /// about what material was traversed by a
17 /// segment. This class does not actually store
18 /// anything, see DGeometry::GetTraversedMat
19
20 class DMaterialStep{
21 public:
22     DMaterialStep(){}
23     DMaterialStep(const DMaterial *material
24     , const DVector3 &startpos, const D
25     , const DVector3 &startmom, const D
26     {
27         this->material = material;
28         this->startpos = startpos;
29         this->endpos = endpos;
30         this->startmom = startmom;
31         this->endmom = endmom;
32     }
33     virtual ~DMaterialStep(){}
34
35     const DMaterial* GetDMaterial(void) const;
36     const DVector3& GetStartPosition(void) const;
37     const DVector3& GetEndPosition(void) const;
38     const DVector3& GetStartMomentum(void) const;
39     const DVector3& GetEndMomentum(void) const;
40     double GetDistanceAlongPath(void) const;
41
42     void SetDMaterial(const DMaterial *mat);
43     void SetStartPosition(const DVector3 &
44     void SetEndPosition(const DVector3 &e
45     void SetStartMomentum(const DVector3 &
46     void SetEndMomentum(const DVector3 &e
47
48 private:
49     const DMaterial *material;
50     DVector3 startpos;
51     DVector3 endpos;
52     DVector3 startmom;
53     DVector3 endmom;
54     double s;
55 };
56
57 #endif // _DMaterialStep_
58
59
```

```
1 // $Id$
2 //
3 // File: DMaterial.h
4 // Created: Fri Apr 25 15:44:12 EDT 2008
5 // Creator: davidl (on Darwin swire-d95.jlab.org 8.11.1 i386)
6 //
7
8 #ifndef _DMaterial_
9 #define _DMaterial_
10
11 #include <string>
12 using std::string;
13
14 #include <particleType.h>
15
16
17 /// The DMaterial class holds information on a single material type. The
18 /// main purpose is to hold information needed to estimate energy loss and
19 /// multiple scattering for particles being swum through the detector.
20 /// The GetdEdx() method can be used to calculate dE/dx through this
21 /// material for a given particle type and momentum using the Bethe
22 /// formula.
23
24 class DMaterial{
25 public:
26     DMaterial(string sname, double A, double Z, double density, double radlen);
27     virtual ~DMaterial(){}
28
29     double GetRadiationLength(void) const {return kXo;}
30     double GetDensity(void) const {return kdensity;}
31     double GetAtomicHeight(void) const {return kA;}
32     double GetAtomicNumber(void) const {return kZ;}
33     double GetdEdx(Particle_t ptype, double kp);
34     const string& GetName(void) const {return kname;}
35
36     double Xo(void) const {return kXo;}
37     double rho(void) const {return kdensity;}
38     double A(void) const {return kA;}
39     double Z(void) const {return kZ;}
40
41     string toString(void);
42
43 private:
44     DMaterial(); ///< Prevent use of trivial constructor
45
46     double kXo; ///< radiation length of this material
47     double kdensity; ///< density of material in g/cm^3
48     double kA; ///< Effective atomic weight
49     double kZ; ///< Effective atomic number
50     string kname;
51
52     Particle_t last_ptype;
53     double last_p;
54     double last_dEdx;
55
56     void GetMaterials(void);
57 };
58
59 #endif // _DMaterial_
60
```