Subject: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 12:55:40 GMT
View Forum Message <> Reply to Message

Hi,

when I replied to Ola's thread, I stumbled over something... In the wirepoint policy, the detPlane method is not implemented. But if you inherit from RecoHitIfc<WirePointHitPolicy> you should implement, hot to get the detPlane in this place. I know it is again virtual in RecoHitIfc, and indeed you implement this feature in PndSttRecohit. But this is not the idea of genfit. If someone else wants to use you policy, he has to reimplement this code, which is of course the only complicated code for a wire hit.
If you put the code to WirepointHitPolicy::detPlane(), and dont further overwrite it in PndSttRecoHit, it will be called through RecoHitIfc (please have a look at the tiny file genfit/RecoHitIfc.h).

So, could you please change that? Or is there a problem here I overlooked. If it is changed in all file, I would like to make this method in RecoHitIfc non-virtual, to avoid these situations in the future.

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 13:16:59 GMT
View Forum Message <> Reply to Message

Hi again,

I forgot to mention that the other place where this appears is in PndDchRecoHit2. It looks like the same code was used in both cases.

If you implement this code in WirepointHitPolicy, then also you you dont need to reimplement getDetPlane at all. The detPlane method of the policy is called in RecoHitIfc.

Again, I want to beg your pardon that this i so complex and you dont have any docu. I am working on this! Please help me to get some consistency in the package, by moving this implementation.

If you have any more questions, please do not hesitate to bug me all you want on this!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 16 Mar 2009 14:42:02 GMT
View Forum Message <> Reply to Message

Hi Christian,
there is a problem with moving the detPlane function implementation from the reco hit to the WirepointHitPolicy: in fact in that function (for stt and dch) we find the point of closest approach

to the wire and to do that we use geane (you see this part of code is written for GeaneTrackRep), while genfit should be independent from it.

By looking to SpacepointHitPolicy I see that the detPlane function uses getVirtualDetPlane. Maybe we need in AbsTrackRep a function to find also the virtual detector plane in the point of closest approach to a wire.

Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 14:54:21 GMT
View Forum Message <> Reply to Message

Hi Lia and Ola,

thanks for pointing out the problems with my idea. I have to think about whether it would be wise to have such a function like the AbsTrackRep::extrapolateToPoca (which is called by virtualDetectorPlane)...
Actually it might well be that I remove getVirtualDetectorPlane all together, because it just calls AbsTrackRep::extrapolateToPoca....

Probably the best way to solve this would be to add a method to AbsTrackRep which is extrapolateToLine (which is clear what it does) and then call this from WirepointHitPolicy::detplane(). Your Poca code would then be implemented in GeaneTrackRep.

What do think about this solution?

If there is still slight modifiactions to the standard behavior necessary in DchHit, one could override detPlane from the policy, call the super-class methid, and then modify the result before returning the result.

Please let me know what you think!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 16 Mar 2009 15:16:49 GMT
View Forum Message <> Reply to Message

From my side, I think this could be a good solution.
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 15:49:01 GMT

View Forum Message <> Reply to Message

Hi,

Ola and Lia: Could you help me a little bit, by explaining the differences between Stt and Dch hits? I would lie to understand to maybe provide a better structure to do this.

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 16 Mar 2009 17:31:59 GMT
View Forum Message <> Reply to Message

Hi,

basically the differences are:

1) the fact that in PndSttRecoHit we make a check on the distance of the poca from the wire:

```
Double_t distance;
distance =
TMath::Sqrt(fabs(((wire1-vpf).Mag2()*(wire2-wire1).Mag2()-pow((wire1-vpf).Dot(w$
// check vpf inside tube
if(distance>0.5) {
  cout << "vpf outside the firing tube" << endl;
  FitterException exc("distance vpf-wire > 0.5", __LINE__,__FILE__);
  throw exc;
}
```

while the PndDchRecoHit does not need it;

2) in PndSttRecoHit the origin of the detector plane is set in the center of the tube:
TVector3 O = (wire1 + wire2) * 0.5;
while in PndDchRecoHit it is in the point of closest approach on the wire:
TVector3 O = vwi;

Please Ola correct me if I missed something.
Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 17:46:18 GMT
View Forum Message <> Reply to Message

Hi,

thanks for the info. I Have to think about that distance problem. But as for the choice of the

origin for the plain, it is completely arbitrary, isnt it? That would mean that one could just pick one of the two choices for both detectors.

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 16 Mar 2009 18:09:37 GMT
View Forum Message <> Reply to Message

In principle yes, but with a warning: originally we used the poca on wire too, because we put _hitCoord[7][0] = 0.0 (no z information), as the dch does now; later when we inserted the z coordinate, it has been necessary to move the origin to a fixed point along the tube (the center for example) in order to be able to set the z coordinate in the detector frame: currenthit->GetZ() - z_of_the_origin = currenthit->GetZ() - currenthit->GetZcen().

If you leave the origin in the poca on wire when you fill the coordinates you don' t know yet where the origin is and so you cannot fill the z correctly ("z_of_the_origin" is unknown)... ok, maybe the H matrix could be used, but in that case you need to have the poca on wire info inside it... it was more simple to change the origin.

So from the stt side, I would keep this origin of the plane (or as well one of the two extremities of the wire).
I don' t know if this is ok for the dch too...
Lia.

Subject: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Anonymous Poster on Mon, 16 Mar 2009 22:10:27 GMT
View Forum Message <> Reply to Message

Hi,

I introduced a new virtual method in AbsTrackRep, which is

```
virtual TVector3 extrapolateToLine(const TVector3& point1,
      const TVector3& point2,
      TMatrixT<double>& statePred,
      TMatrixT<double>& covPred,
      DetPlane& planePred);
```

It has a default implementation in AbsTrackRep.cxx, so all TrackReps which dont have this feature are still fine.

Could you please put your extrapolate to line code in GeaneTrackRep::extrapolateToLine, which you would have to put in GenaeTrackRep.h and .cxx? The return value TVector3 would be the point (of closest approach).

I still have to think about your distance problem, but I am sure we solve that special case for

the straws to still share the policy::detPlane with Dch.

Putting this extrapolateToLine to GeaneTrackRep is just the first step.

Why I am doing all this: To combine code which does the same all in one place, and make the structure cleaner and more understandable. I am sorry for all the work I cause you.

Thank you, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 16 Mar 2009 22:25:05 GMT
View Forum Message <> Reply to Message

Hi again,

I have a very simple solution for your distance cut: Put a

protected:
double _maxDistance

in WirepointHitPolicy. Then make a WirepointHitPlicy which loks something like this

```
const DetPlane&
SpacepointHitPolicy::detPlane(AbsRecoHit* hit, AbsTrackRep* rep)
{
  TMatrixT<double> rawcoord = hit->getRawHitCoord();
  //I dont know which of the 8 params is for the wire points
  TVector3 point1(rawcoord[0][0],rawcoord[1][0],rawcoord[2][0]);
  TVector3 point2(rawcoord[3][0],rawcoord[4][0],rawcoord[5][0]);
  int dimension = rep->getDim();
  TMatrixT<double> statePred(dimension,1);
  TMatrixT<double> covPred(dimension,dimension);
  //note that _plane is defined in AbsTrackRep
  TVector3 poca=rep->extrapolateToLine(point1,point2,statePred,covPred,_plane);

  /*C. Hoeppner, March 09: I am not entirely sure that these two calls
    are needed, but they dont hurt for sure. Something happens here
    to the orientation of u and v, so keep it.*/
  TVector3 m=_plane.getNormal();
  _plane.setNormal(m);

  //now calculate distance of poca to line between point1 and point2
double distance;
if(distance>_maxDistance) {
      cout << "vpf greater than maxValue" << endl;
      FitterException exc("distance vpf-wire larger than maxValue", __LINE__,__FILE__);
      throw exc;
    }
```

```
  return _plane;
}
```

For the DCH where you dont want this distance cut, just set the _maxDistance (do this in the contructors of the recoHits) to a very large value. Or you can use a bool flag to deactivate this feature. This you would also be set in the ctors.

About the origin of the plane: can you come to a common solution? Is Lia's solution OK for you Ola?

Thanks for your support!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Aleksandra Wronska on Tue, 17 Mar 2009 09:21:30 GMT
View Forum Message <> Reply to Message

Dear Lia and Christian,

I have no objections about introducing _maxDistance in WirepointHitPolicy. It's the easiest solution, isn't it?

Concerning the origin of detPlane, the answer for now is "I do not know", I am afraid. I need a few hours to look for a reasonable solution for dch to use with a fixed plane origin. Any suggestions from Lia (as the original author of the code) are welcome

What I have in mind (without a clear idea of implemenation yet) is that I could set the Z coordinate of the DchRecoHit2 (checking previously that it was not set before, such that the same code works for you, too) to the Z of poca on the wire, calculated with respect to the wire centre. Does it have any chance to work, Lia?
This would require an extra function for setting Z of a RecoHit, but that's a minor thing, isn't it?

cheers,
ola

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Tue, 17 Mar 2009 10:34:13 GMT
View Forum Message <> Reply to Message

Hi Ola,
as far as I understand you don' t have a z coordinate for your reco hit at the beginning of the Kalman procedure and so you would set the z coordinate to the value found by the propagation, after the propagation to find the detPlane is performed... is this correct?
Well, in this case you would consider as a measured value the extrapolated value and so the Kalman would use the extrapolated value twice when doing its weighted mean...

Wouldn' t it be better to set the z coordinate to 0.0 and give a very large error to it, so that the Kalman practically does not consider it when doing the weighted mean and would take only the extrapolated value? (I think this could work, but I' m not 100% sure)

But let me understand one thing: now the z coordinate is set to 0.0, but will you be able to set it to a different value in future, or will you never have that coordinate? Can' t you get some info on that from the prefit?

Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Aleksandra Wronska on Tue, 17 Mar 2009 10:53:47 GMT
View Forum Message <> Reply to Message

Dear Lia,

thanks a lot for your suggestions.

To make things clearer:
indeed, I have no information about the z-coordinate (=along the wire) in my RecoHit at the moment and I use your old method with setting detPlane origin to poca on the wire and z-coord of the hit to zero.

I can test the method with large errors on Z, let's see if it works at all.

In principle my prefitter constructs a 3d point on each of the chambers, so I could try to modify my CylinderHits in the PreFitter assigning the value of coordinate along the wire there. I have to warn you (or more Christian), however, that it will take me some time. Since tomorrow till the end of the week I am totally occupied with teaching

regards,
ola

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Lia Lavezzi on Tue, 17 Mar 2009 17:13:41 GMT
View Forum Message <> Reply to Message

Hi,
I am trying to "move" the code from the stt reco hit to the extrapolateToLine, but there are some problems:

1) in the stt reco code we only find the point of closest approach to the wire and from this we build the detector plane, but looking at the extrapolateToLine function I see that we want a complete extrapolation, with the state/cov/detPlane predictions... and this brings me to the second problem...

2) the easiest way to do this is to use the geane PropagateToVirtualPlaneAtPCA: this function calculates the point of closest approach to the wire, builds the plane (in the STT usual way) and performs the propagation to this plane. This would give the right result, but doing things this way we would have the propagation from the starting point to the detPlane twice for each kalman step (one here when the plane is built, the other when the Kalman extrapolation is performed). I had also a quick look into the extrapolateToPoca and I see the same problem there...

Given these two considerations, wouldn' t it be better to have here (to be used in the detector plane determination) two findPocaToPoint/ToLine functions, which only find the poca, in addition to (or in substitution of) the two extrapolateToLine/ToPoca functions ? They could also give the detector plane or this could be left in the detPlane(...) functions.

Another solution could be to leave the extrapolateToPoca/ToLine, but to avoid to repeat it with the usual extrapolate call into Kalman::processHit.

What do you think?

I could also write the function with the PropagateToVirtualPlaneAtPCA solution for now and leave the decision to a later stage...

Ciao,
Lia.

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Anonymous Poster on Tue, 17 Mar 2009 17:23:28 GMT
View Forum Message <> Reply to Message

Hi Lia,

I have to leave now, but I will take care of it tomorrow. I wanted to remove the full extrapolation from the extrapolateToPoca and Line anyway to make things nicer. I will do so tomorrow.

Bye, Christian

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Anonymous Poster on Wed, 18 Mar 2009 10:51:51 GMT
View Forum Message <> Reply to Message

Hi Lia,

you are absolutely right that we do some extrapolations twice. I thought a lot about this problem, and I can not come up with a general solution to it. Here is the biggest problem with it:

If you would only want to do it once, you would have to return the statePred and CovPred together with the DetPlane in AbsRecoHit::getDetPlane(). This is not yet a problem. But, in your hit policy::detPlane, where you implement this, you have to make some call to an extrapolateToLine or so of AbsTrackRep. Now if this is supposed to give you also the state and Cov, it needs to fix already the DetPlane, because state and cov can only be defined in a plane.
This is in general impossible, since it would mean that everytime you want to change your definition of the DetPlane for a wire hit or so, you would have to change track reps, and then different detectors would need extrapolate1,2,3,... functions. In short it would be a mess!

My opinion is this: We make a an extrapolateToLine and also to Poca, which does not give the state,cov and plane results and just returns the POCA. We live with the fact, that we do some extrapolations twice. But keep in mind that they are not really the same, in the final extrapolation you fixed your plane geometry, and it is clear that then you have to do another extrapolation.

Please remember, that we are paying this minor price for a great deal of flexibility which we gain! And I think that is exactly what we need at this early stage of our experiment. We can still easily test and interchange different track models and hit geometries without any changes to the core code of the tracking! To my knowledge (and I did quite a lot of research on finding something) there is no other tracking system which could even handle the STT and TPC together with everything else in geometry we have it.

Lia, what do you think of my proposal for changing what the extrapolateToLine should do?

Cheers, Christian

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Lia Lavezzi on Wed, 18 Mar 2009 12:23:58 GMT
View Forum Message <> Reply to Message

Hi,
ok, please, let me shortly summarize things, because I' m getting lost in all this discussion and I want to understand things well

Let' s consider for now the extrapolateToPoca (just because it is already implemented, and the extrapolateToLine will be almost the same):
- NOW it does the following (by calling the PropagateToVirtualPlaneAtPCA): it extrapolates to a very large track length, finds the point of closest approach, builds here its own detector plane, finally re-extrapolates to this detector plane and gives the results, which are the state/cov/detPlane/poca.

- Your idea is to rewrite the extrapolateToPoca in order not to call the PropagateToVirtualPlaneAtPCA and to just have one extrapolation (the one to a very large

track length), which only finds the point of closest approach, without filling any PREDICTED state/cov/detPlane.

If so, this is the same idea I have, so I agree

In this case we will have (that' s true!) an extra extrapolation, but is really necessary in order to find the poca, so we just have to keep it.

Please correct me if I misunderstood something...

Ciao,
Lia.

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Anonymous Poster on Wed, 18 Mar 2009 12:37:09 GMT
View Forum Message <> Reply to Message

Hi Lia,

yes this exactly what I meant! Just one more question before I make the changes:

When I change the argument list of extrapolateToPoca and ToLine, could you please make the proper modifications to GeaneTrackRep::extrapolateToPoca to remove the not anymore needed functionality. You are much more of an expert than me in Genae and dont think it will be a lot of work for you, will it?

So, I will change the argument list now. After you had a chance to remove the functionality from extrToPoca I will test immediately if everything still works.

Sounds like a plan?

Thanks for the discussion, Christian

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Lia Lavezzi on Wed, 18 Mar 2009 14:17:47 GMT
View Forum Message <> Reply to Message

Hi Christian,
ok, I will do it tomorrow (sorry, but today I can' t have a look to that, I hope this is not a problem...).

Ciao,
Lia.

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Anonymous Poster on Wed, 18 Mar 2009 14:26:44 GMT
View Forum Message <> Reply to Message

Hi Lia,

one more question, before I can submit my changes. In GeaneTrackRep::exptrapolateToPoca we set the origin of the resulting plane to result.GetOrigin(). Can you tell me what this point means?

Thanks, Christian

Subject: changes in PndDchKalmanQA
Posted by Anonymous Poster on Wed, 18 Mar 2009 14:32:05 GMT
View Forum Message <> Reply to Message

Hi Ola,

I am changing the argument list for AbsTrackRep::extrapolateToPoca. One of your files, PndDchKalmanQATask.cxx will require a change. I think I can not submit it, but if I submit the rest of the changes, the trunk wont compile anymore. Are you there to check in a diff I can send you (shortly) after I check in my code?

Cheers, Christian

Subject: Re: changes in PndDchKalmanQA
Posted by Anonymous Poster on Wed, 18 Mar 2009 15:13:08 GMT
View Forum Message <> Reply to Message

Hi Ola,

Mohammad was so kind to give me write access to dch/DchFitting. I promise I will give these rights up as soon as this restructuring is done. I wont change the functionality of your code, just the formalities.

Lia, I also have write access temporarily to stt/sttreco for the same reason.

Cheers, Christian

Subject: Re: new in AbsTrackRep: extrapolateToLine and consequences on WirepointHitPolicy
Posted by Lia Lavezzi on Wed, 18 Mar 2009 15:44:40 GMT
View Forum Message <> Reply to Message

Quote: In GeaneTrackRep::exptrapolateToPoca we set the origin of the resulting plane to result.GetOrigin(). Can you tell me what this point means?

It is the origin of the virtual detector plane as calculated during the propagation procedure (see FairGeanePro::FindPCA and FairGeanePro::Propagate(from parabola to parabola)) and for the propagation to closest approach to a POINT it is just the point itself (with respect to which the PCA is calculated).

Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Wed, 18 Mar 2009 16:21:54 GMT
View Forum Message <> Reply to Message

Hi Lia,

the changes are now commited. You can overwrite extrapolateToLine in GenaeTrackRep.

You have to return two values by reference: the poca and the direction of the track in the poca. This is needed I needed to calculate the detPlane, at least in my case. If you dont need it for the STT, please implement it in the trackRep anyways, for future use, afterall it should be easy.

Could you also have a look in extrapolateToPoca? Maybe it can be cleaned up a little bit. I already changed it to do what I need, and I tested that it is still doing fine.

I am out for today. Thanks for all your help!

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Thu, 19 Mar 2009 12:35:37 GMT
View Forum Message <> Reply to Message

Hi Christian,
I implemented the GeaneTrackRep::extrapolateToLine but before putting it on the svn I have one question: in STT case we need to know the point of closest approach on wire in addition to the point of closest approach on the track (the usual poca) to build the plane, while we don' t care about the direction of the track in poca, can we change the TVector3 from dirInPoca to poca_onwire (or something like that) ?

Concerning the extrapolateToPoca I have to look at it more deeply, but since you don' t need the state/cov at the poca something like the extrapolateToLine (but with pca = 1) should work (to only find the poca, without the need to call the "Propagate" function which performs the actual propagation)... I only see a problem in finding the direction of the track at the point of closest approach... I need to think about this a little, to see if it is possible to get it without any extra extrapolation (a part the one to find the poca). I will let you know.

For the extrapolateToLine, my implementation is ready, with the change dirInPoca --> poca_onwire: I tested it with my stt reco hit and it works fine, so if you give me the "ok" I can

put it on the svn.

Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Thu, 19 Mar 2009 13:11:07 GMT
View Forum Message <> Reply to Message

Hi Lia,

well the direction in the POCA, I think I already have in now in extrapolateToPoca. Maybe you can take a look, but since all my fist work, I pretty satisfied with the situation.

I see you point on needing poca_onwire, and I want to add it. But one question: Do you have the info about the direction of the track in the POCA (maybe you want to look in extrapolateToPoca)? If so, I would like to add your parameter as a third return value and not replace it (so the argument list would stay with an additional TVector3& poca_onwire).

Chhers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Thu, 19 Mar 2009 15:42:09 GMT
View Forum Message <> Reply to Message

As the code is written now, I don' t have the info on the track direction in poca, but I think I could get it by writing a code more similar to the extrapolateToPoca (i.e. by adding the additional propagation): do we really need this info?

That' s true, in the extrapolateToPoca you have the track direction because you extrapolate the track to the poca, but I thought we wanted to get rid of the statePred (and covPred) in the extrapolateToPoca/ToLine in order to avoid to call the Propagate function there, did we? ...I' m a little confused...

I attach to this message the GeaneTrackRep.cxx with my implementation of the extrapolateToLine, just to show you how I would do things (at least for the extrapolateToLine)... but anyway it would not be so difficult for me to change the code and make it similar to the extrapolateToPoca.

I' d like to find the track direction without the need of performing the propagation, but I admit the most simple way is (as you did) to perform the additional propagation after having found the poca; in this case, however, we could do things in a slighlty different way: now we use the PropagateToVirtualPlaneAtPCA and then a virtual detector plane is built internally (in Propagate): this would be ok if we wanted to consider this plane as detPlane, but we don' t want to do this! We want to build our own detPlane (in the appropriate function). So actually, to find the direction, it would be enough to perform a propagation to the track length (and not to

the plane) which corresponds to the found poca (without the need to build the plane)... I could look in FairGeanePro, if this is already feasible.

So if we want the direction also for the extrapolateToLine, I could provide a function similar to the extrapolateToPoca in order to have a preliminary implementation and then think about some modifications to make it faster... what do you think?

Lia.

File Attachments

1) GeaneTrackRep.cxx, downloaded 392 times

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Thu, 19 Mar 2009 15:54:46 GMT
View Forum Message <> Reply to Message

Hi Lia,

yes I think that we need this info. At least I know we do in the Poca case, because other wise I have no idea on how to orient the normal vector for the plane. I think that it is also useful to have it in the extrapolateToLine method.

I will add your third return value and commit it in a few minutes.

If you would find a way to speed things up that would be great. I think right now my top priority is still the consistent structure. But yes you are very welcome to improve things in Poca also!!

I will eave shortly.

CU tomorrow!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Thu, 19 Mar 2009 15:56:20 GMT
View Forum Message <> Reply to Message

Ah, and sorry I didnt have time to look over the code. I am sure it's fine, just check it in and try your fits whether they still work.

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Thu, 19 Mar 2009 15:57:28 GMT
View Forum Message <> Reply to Message

Sorry, and another:

If you dont have the direction right now, just leave it empty. We can fix it in the next weeks.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Fri, 20 Mar 2009 11:58:41 GMT
View Forum Message <> Reply to Message

Hi,
I just uploaded the GeaneTrackRep with the implemented extrapolateToLine: now the dirInPoca is empty.

I also changed the PndSttRecoHit in order to use this function (I know the detPlane determination will be moved to the WirepointHitPolicy, but I wanted to test it) and I obtain the same results I got before, so it seems to work

I will think about the dirInPoint...

Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Fri, 20 Mar 2009 13:36:11 GMT
View Forum Message <> Reply to Message

Hi Lia,

thanks a lot for the help in restructuring things. If you come up with something on the direction or want to discuss it, I am here....

I read again throught the argument about the difference in origin in STT and DCH. I understoof the difference now. In the STT you also have a measurement of the coordinate along the wire (by the way for curiosity: How do you plan doing that - charge sharing, timing on double sided readout?), and in the DCH case you dont have it.

But these are indeed two different things and I propose to have two policies for that.

WirepointHitPolicy for the STT
WireHitPolicy for the DCH

I guess the naming is self-explanatory. Lia, do you have write access in genfit? If so, could you implement your changed WirepoitHitPolicy and also the WireHitPolicy in which you put the code for Ola's choice of origin placement?

Then after that Ola could use WireHitPolicy and get rid of her own implementation of getDetPlane.

When we are done with this process, I will remove the virtual from getDetPlane in the RecoHitIfc to make this method non-overwriteable.

Lia, if I ask too much of you, please let me know and I will try to do some things as far as I can. Overall I think the process should be easy though.

Chers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Fri, 20 Mar 2009 17:11:18 GMT
View Forum Message <> Reply to Message

Hi Christian,
I uploaded the WirepointHitPolicy (and the PndSttRecoHit), please have a look if they are correct.

Quote:I read again throught the argument about the difference in origin in STT and DCH. I understoof the difference now. In the STT you also have a measurement of the coordinate along the wire (by the way for curiosity: How do you plan doing that - charge sharing, timing on double sided readout?), and in the DCH case you dont have it.

But these are indeed two different things and I propose to have two policies for that.

WirepointHitPolicy for the STT
WireHitPolicy for the DCH

Would it be possible to have only one policy and use a "strategy" like the one you proposed to handle the "distance from wire" problem? I mean, we could add to the WirepointHitPolicy a variable that must be filled with the detectorID in order to discriminate between the dch and the stt when the origin of the plane has to be set or the distance cut has to be applied. What do you think? Could this be a solution?

Concerning the stt z: now we reconstruct it using the skewed tubes, during the local STT reconstruction.

Have a nice week end!
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Fri, 20 Mar 2009 17:18:18 GMT
View Forum Message <> Reply to Message

Hi Lia,

that sounds like a good solution. Inside the WirepointHitPolicy you would save a bool value, which you fill at construction time of your recoHit, which is a switch between detectors which can and which can not measure the position along the wire. That sounds good!

You have a nice weekend, too!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Fri, 20 Mar 2009 19:28:02 GMT
View Forum Message <> Reply to Message

Hi Lia,

I just checked if everything is fine now structurally and it looks great! To make it usable for the Dch, we just need the two variables maxDistance and alongWirePosFlag (or better names) and then Ola could try out to remove the getDetPlane method from PndDhcRecoHit2 and fill the two new variables in the policy in the constructor.
I think the maxDistance should be set by default to something huge (1.E50 or whatever), so it will work by default for the DCH and needs to be set to 0.5 in the SttRecoHit ctor. For the flag for the z-pos.... I dont know what should be the default. Maybe a detector which doesnt measure it is more general. Either is fine for me though.

Bye, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 23 Mar 2009 12:32:25 GMT
View Forum Message <> Reply to Message

Hi Ola & Lia,

on second thought, I think it is necessary to have two hit policies for detectors which can measure a position along the wire and those which can not. The reason is the dimensionality of the hit. The DCH hit is only a 1-D information. There for the correct HMatrix (OK, this has nothing to do with the policy) would be 5x1 and would be [0,0,0,1,0] assuming that the U-vector points perpendicular to the wire, like it does.

Also in the policy, where you also fix the dimensionality when you calculate the hit coordinates and covariances this should be dont correctly.

It is true that just putting the coordinate to zero with a huge errors will lead to an unbiased fit, but the number of degrees of freedom for the chi2 calculation will be wrong.

If you dont have any objections I will introduce a new policy WireHitPolicy (without the point) which will do this correctly. I would then still ask you to check the detPlane() method for your choice of detector plane before you use it.

Does that make sense? Would you have some time to test this new implementation?

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Aleksandra Wronska on Mon, 23 Mar 2009 12:39:22 GMT
View Forum Message <> Reply to Message

Dear Christian,

no objections from my side, go ahead. I can start with testing it tomorrow morning.

cheers,
ola

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 23 Mar 2009 12:55:01 GMT
View Forum Message <> Reply to Message

Hi,
I added the _maxdistance and _wireCoordFlag to the WirepointHitPolicy. By default I put:
- _maxdistance = 1.E50,
- _wireCoordFlag = false.

I also added the settings of the stt value in PndSttRecoHit for the two variables, so for stt everything works.

I think now the policy should be usable also by the dch ...Ola, if you find something wrong please tell me!

Ciao,
Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Lia Lavezzi on Mon, 23 Mar 2009 13:01:45 GMT
View Forum Message <> Reply to Message

Ooops!
Sorry, I didn' t see your message in time and I posted mine without having read this!!

Ok, if you manage also to change the dimensions I agree we can use two different policies...

Feel free to remove my last changes to the WirepointHitPolicy if needed... but I would keep the _maxdistance variable, to better handle it...

Lia.

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 23 Mar 2009 14:33:16 GMT
View Forum Message <> Reply to Message

Hi Ola & Lia,

I committed my changes:

- new WireHitPolicy which gives 1-D hitCoord and hitCov and uses poca_onwire as DetPlane origin
- removed wirepoint flag
- changed PndDchRecoHit2 to seven dimensions (no more 0.0 z-ccordinate) and the HMatrix to 5x1

Please test this new version and give me some feedback. I am positive that it will work, but you never know....

Thanks for your support!

Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Mon, 30 Mar 2009 14:21:41 GMT
View Forum Message <> Reply to Message

Hi Ola,

I was wondering if there are any problems after my changes to your PndDchRecoHit2?

Cheers, Christian

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Aleksandra Wronska on Tue, 31 Mar 2009 06:16:33 GMT
View Forum Message <> Reply to Message

Hi Christian,

there is basically one problem, namely that dch kalman task exits with:

PndDchKalmanTask2::Exec
---
extrapolateToLine() as implemented in /d/panda02/awronska/pandaroot/genfit/AbsTrackRep.cxx was called. This means that this feature was used in a track rep which didnt overwrite this method.
C++ throw;

cheers,
ola

Subject: Re: WirepointHitPolicy detplane implementation
Posted by Anonymous Poster on Wed, 01 Apr 2009 10:34:20 GMT
View Forum Message <> Reply to Message

Hi everybody,

the issue with PndDchKalmanTask2 has been solved. It was just a misunderstanding.

Cheers, Christian