LArSoft - Necessary Maintenance #11871

Feature # 11870 (Assigned): Have a uniform way to identify decay/stopping position of a track

Decay products in pandora do not start from the same vertex, while Projection Matching Algorithm's do

03/03/2016 09:23 AM - Gianluca Petrillo

Status:	Accepted	**Start date:**	03/03/2016
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Reconstruction	**Estimated time:**	0.00 hour
Target version:	2017-4-quarter	**Spent time:**	0.00 hour
Experiment:	LArSoft		

Description

In an interaction or decay, it is expected that all the decay products start from the same vertex.
The recob::PFParticle collection can describe a hierarchy of particles and their decay/interaction relations.
Each particle has a starting vertex associated with it and a list of *daughter particles* outgoing, each of them associated with a vertex as well.

Apparently pandora saves a different starting vertex for each daughter, possibly related to the start of the reconstructed track (a pattern-recognition point of view). The Projection Matching Algorithm saves the same vertex for all daughters (a physics point of view).

After we agree to a uniform prescription, at least one of the two will need to be modified to follow it.

History

#1 - 03/03/2016 09:57 AM - John Marshall

As commented above, Pandora attempts to provide a full representation of a particle decay/interaction hierarchy.

For a neutrino interaction, there will be one PFParticle that represents the neutrino, with a vertex representing the neutrino interaction vertex. There will then be PFParticles representing primary daughters of the neutrino, e.g. a muon and a proton, and each of these will also have a vertex (typically will be close the neutrino vertex, but not always the case, e.g. case of two photons produced by pi0 decay). The muon may subsequently decay and an electron may be reconstructed as a daughter of the muon (a secondary daughter of the neutrino). The PFParticle representing the electron will have a vertex at the point of muon decay / electron creation.

For cosmic ray muons, there will typically be a primary PFParticle that represents the muon, with a vertex at the end of the track that has the highest y-coordinate. The muon will likely have a number of daughter PFParticles representing delta rays. These will each have a vertex at the start of the delta ray shower, typically the closest 3D point to the muon track

For a pictorial representation, please see slides 4 and 5 in the .pdf file linked below:
http://microboone-docdb.fnal.gov:8080/cgi-bin/ShowDocument?docid=5387

For a more thorough grounding in the ambitious use of the fine granularity imaging capabilities of LAr TPCs by Pandora, please see the more comprehensive slides linked below (esp. slide 16 onwards):
https://indico.fnal.gov/getFile.py/access?contribId=5&sessionId=0&resId=0&materialId=slides&confId=10394

It would certainly seem to be a shame to drop the information provided by secondary vertices at the point of translating the event outcome from the Pandora EDM to the LArSoft EDM. My understanding is that the full list of Pandora vertices (neutrino, plus all daughters in the full hierarchy) is used by the MicroBooNE CC inclusive group in their identification of neutrino induced events in the MicroBooNE data.

Any/all comments most welcome!

#2 - 03/05/2016 07:23 AM - Robert Sulej

- File pim-pandora-ev1.jpg added

- File pim-cc-pma-ev1.jpg added

Hi,

Sorry for delay, I needed to make my larsoft installation working again to make some pictures to explain better my point. So the attached event is pi- (~horizontal track incoming from the left), it interacts and produces some secondary tracks.

I do not want to discuss efficiencies and how many times the correct topology was found, just what convention to expect in downstream algorithms.

After some thinking and John comments I would say that for parsing such hierarchy in downstream analysis I need one vertex in the interaction point
- it can be associated to the incoming PFParticle, but there needs to be a way to distinguish it from the vertex tagging the beginning of the particle track if such vtx is associated as well;
- neutrino PFParticle, as I understand and use data products now, does have such interaction vertex at the end of its (invisible) track, but it is a special case and there is no starting vertex.

Then daughters would need vertices at beginnings of their tracks:
- reasonable and useful for daughters like gammas converting at some distance from interaction, as John pointed out
- may seem to be redundant info for hadronic or electron tracks starting from the vertex, but actually can make life easier when parsing the event and very usefull if one needs to check signal gap presence between interaction vertex and daughter.

Is the above convention used in Pandora output? How to distinguish starting vertex from those tagging an interaction? For muon it is not that important to have "interaction" vertices for each delta ray, starting points associated to delta tracks are enough, and all other cases I can think of will have one or two vertices: starting and interaction at the end (if any).

One more comment: the first 3D point of the trajectory can be easyli used as a visible track beginning, so one can check the gap as a distance to the end vertex of the parent etc, so another option, maybe more uniform with neutrino, can be: use trajectory points (or space points) to get the track beginning and associate vertex to the track if it dies in the interaction.

If you have idea for other solution, please, share. I can adjust output from pma module to whatever we decide. For now I add vertex at the particle start and in it is a common vertex associated to many tracks if they are produced in the same interaction.

I though it is an option that also contains all the information in a minimal way: gap vertex - track beginning is easy to parse, if no daughters then the last trajectory point is the stopping point (see example on the second attached picture).

Cheers,
Robert

#3 - 03/06/2016 05:05 AM - John Marshall

Hi Robert,

Thanks for your reply! The convention used in the Pandora output is for the PFParticles to handle all aspects of the particle hierarchy.

Incoming PFParticles, whether they are neutrinos or cosmic rays, are distinguished by having no associated parent PFParticles (you can ask a PFParticle whether it IsPrimary()). The daughters of the neutrino will be the leading lepton (and maybe a number of protons, etc.). The daughters of a cosmic ray will be the delta rays. It is possible to navigate upstream along the PFParticle hierarchy, via parent indices (or downstream, via vectors of daughter indices), with a lack of parents (or daughters) indicating termination of the hierarchy in the relevant direction.

To each PFParticle is then associated a vertex. This vertex could indeed take-on some additional, descriptive labels (in Pandora we have labels for: interaction vertices, start, end, apex, corners and more generic "features"). The convention is that the PFParticle vertex is the start vertex, except for the neutrino as you mentioned above, where the vertex represents the interaction point.

This should all mean there is no ambiguity in the Pandora output. In essence, Pandora tries to exactly mirror the well-defined output produced by e.g. event generation steps in HEP: it provides a navigable hierarchy of particles, each with a particle type, a four-momentum, a start-vertex position and an associated collection of hits. This is very much physics-motivated, rather than pattern-recognition motivated (as labelled in the opening description for this thread).

It seems sensible, from the Pandora point of view, to maintain this well-defined convention, as other approaches would seem to throw-away information reconstructed by Pandora (any maybe instead push the onus onto the spacepoints, provided either by Pandora or a downstream track fitting package). We think that the EDM should support full translation of information from Pandora (so actually some new vertex labels would be useful!) and also support persistence of any downstream alternative approaches or refinements e.g. track fits to Pandora "found tracks".

Maybe we should have a quick Skype chat if we are still at cross-purposes following this discussion?

Best wishes,

John

#4 - 03/09/2016 02:11 PM - Robert Sulej

Hi John,

You are right, a way to distinguish various types of vertices is needed to store a more complete description of reconstructed events.
We had some discussion with Gianluca today and we have a potential solution:
vertices of different types are saved in separate, named collections (like interaction, endpoint, ...)
- partile is associated to the vertex in the "interaction" collection, if such interaction vtx is reconstructed, so neutrino is treated in the same way as e.g. interacting hadron
- particle is associated to the vertex in the "startpoints" (or any appropriate name) collection,
- and in the same way for other useful features...
- many vertices with different meanings can be associated to one particle, and downstream algorithms can easyli select vertex of the interesting type.

Art supports such associations to objects in the collection selected by instance name. Would it be OK for you to adopt this solution in the Pandora-LArSoft interface (or for Andy if he wrote the interface code)?

Cheers,
Robert

#5 - 03/10/2016 10:37 AM - John Marshall

Hi Robert,

This sounds very good. We're happy to adopt a clean solution that allows us to persist all the information reconstructed by Pandora, and which provides good clarity of usage for downstream software/users.

Andy may actually be best-placed to take this on, but we can discuss amongst ourselves. When I recently restructured LArPandoraInterface, I basically just moved Andy's ProduceArtOutput functionality, and changed the interface. I deliberately left this aspect of implementation untouched.

[Translation from Pandora back to LArSoft is handled by the static helper functions in LArPandoraOutput. For more information on the structure of LArPandoraInterface, please see:
https://indico.fnal.gov/getFile.py/access?contribId=2&resId=0&materialId=slides&confId=11554]

If you and/or Gianluca could maybe provide some e.g. pseudocode examples for use at the point of Pandora -> LArSoft translation and for some typical downstream use-cases, it would definitely help to expedite these changes.

Thanks and best wishes,

John

#6 - 03/10/2016 12:49 PM - Robert Sulej

Hi John, All,

Great. I just have collected statistics of users (N=2), both saying that mutiple named collections can be less user friendly. But after short discussion one user agreed that it is not true and actually collections can make analysis easier. I will convince the other one with code examples.

Please, give me some time to prepare such code examples and think a bit of scheme of collections that we wont need to change completely after a week of using... I'll come back with a proposal.

Cheers,
Robert

#7 - 03/22/2016 10:16 AM - Robert Sulej

- File save_reco_results.cc added

- File read_event.cc added

Hi John, All,

Maybe I have found a way that allows smoothly introduce named collections of vertices. Code examples below are from the larreco branch rsulej_MultipleVtxCollections where kinks on tracks are saved in the named collection, kinks are assigned to tracks also using instance name, while interaction vertices, I think most often needed for analysis, are saved in the old way with no instance names.

I assign 1 interaction vertex to each particle / track to indicate where the particle was created.

Any additional information, like kinks in this case, but also start points or other reconstructed features can go to separate collections.

I am going now to add small modification to the event display to show vertices and kinks in a distiguishable way.

So, to save all the data products, please see attached file:
save_reco_results.cc

...and much more simple way to read these results downstream:
read_event.cc

Cheers,
Robert

#8 - 10/31/2016 04:50 PM - Katherine Lato

From: Robert Sulej <robert.sulej@cern.ch>
Date: Wednesday, October 12, 2016 at 3:28 AM
To: Katherine Lato <klato@fnal.gov>
Cc: Erica Snider <erica@fnal.gov>
Subject: ODP: redmine issue

Andy Blake wrote Pandora-LArSoft interface. I don't know who should maintain it now (not me, I only have impressions what else may be reconstructed in Pandora but is not yet translated to LArSoft data products).

It is improtant to have some common policy for event structure description. We are now working on related codes in PMA, but I think it would be good to have the decision maker with a view from "outside", not connected directly to one or another reco approach.

#9 - 12/14/2016 11:29 AM - Katherine Lato

- Status changed from New to Accepted

- Target version set to 2017-2-quarter

At the experiment level, need a discussion about agreeing on what the content of a data product means. LArSoft will facilitate this discussion.

#10 - 02/21/2017 01:56 PM - Katherine Lato

- Target version changed from 2017-2-quarter to 2017-4-quarter

Files

pim-pandora-ev1.jpg	347 KB	03/05/2016	Robert Sulej
pim-cc-pma-ev1.jpg	228 KB	03/05/2016	Robert Sulej
save_reco_results.cc	2.84 KB	03/22/2016	Robert Sulej
read_event.cc	726 Bytes	03/22/2016	Robert Sulej