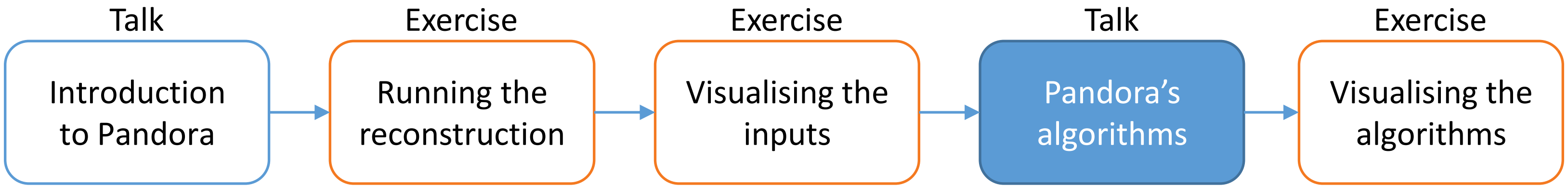
Reconstruction algorithms

Andy Chappell for the Pandora team

06/06/2024 DUNE-France Analysis Workshop

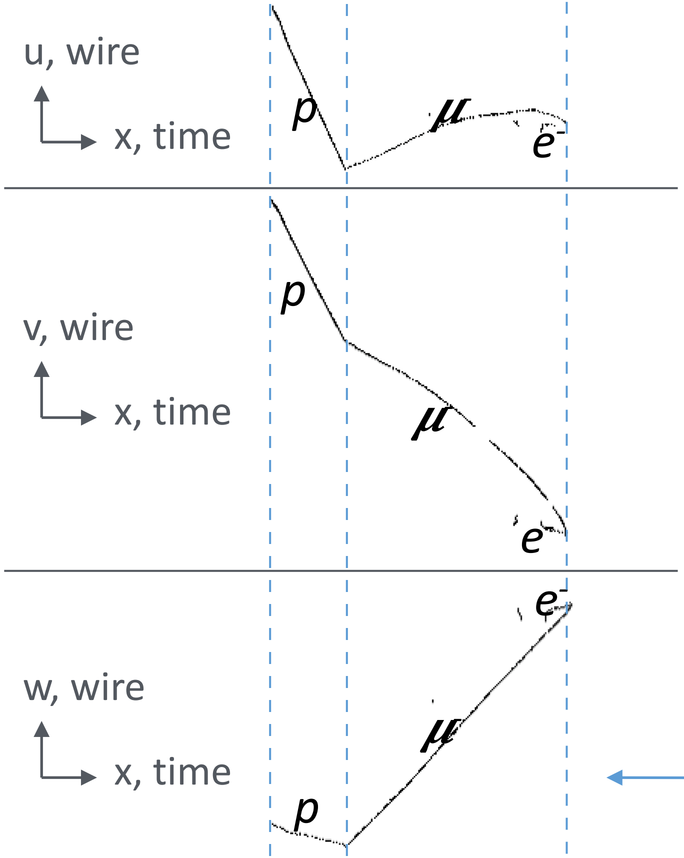
Reconstruction session



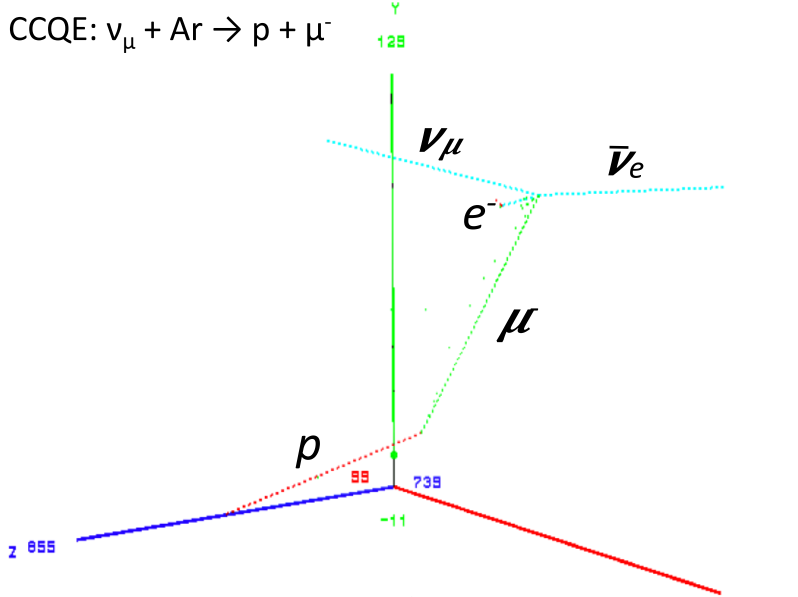
Credit: These slides are based on previous LArSoft workshop slides by John Marshall

Key references: [Pandora ProtoDUNE paper](#)
[Pandora MicroBooNE paper](#)

Inputs to Pandora



CCQE: $\nu_\mu + Ar \rightarrow p + \mu^-$

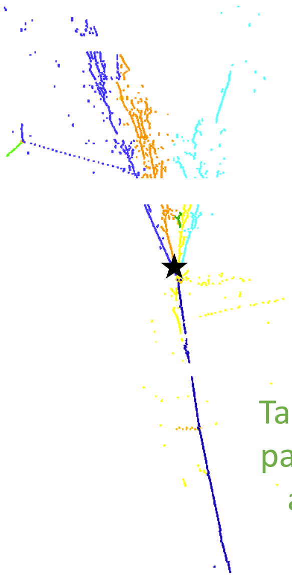


3D truth information

Three 2D representations with common x coordinate, derived from drift time

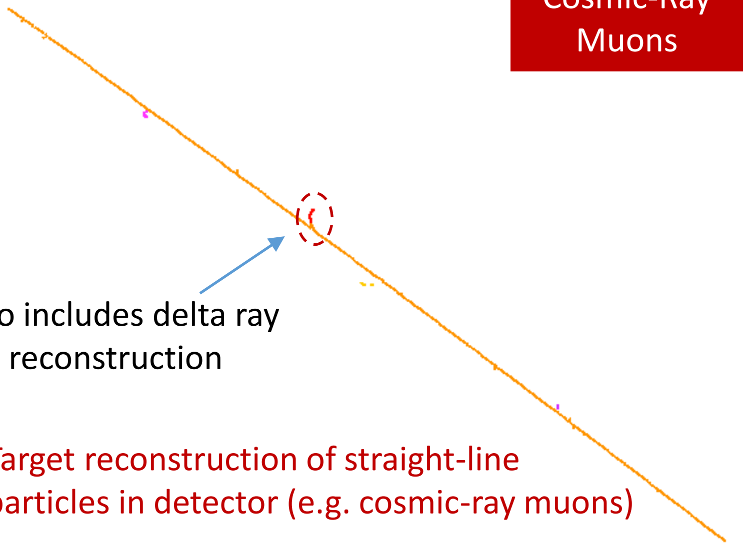
Consolidated reconstruction

- We use a multi-algorithm approach to create two algorithm chains:
- Consolidated reconstruction uses these chains to guide reconstruction for all use cases:
- Cosmic rays ✓, Multiple drift volumes ✓, Arbitrary wire angles ✓, 2 or 3 wire planes ✓



Neutrino (or TestBeam)

Target reconstruction of particles emerging from an identified vertex ★

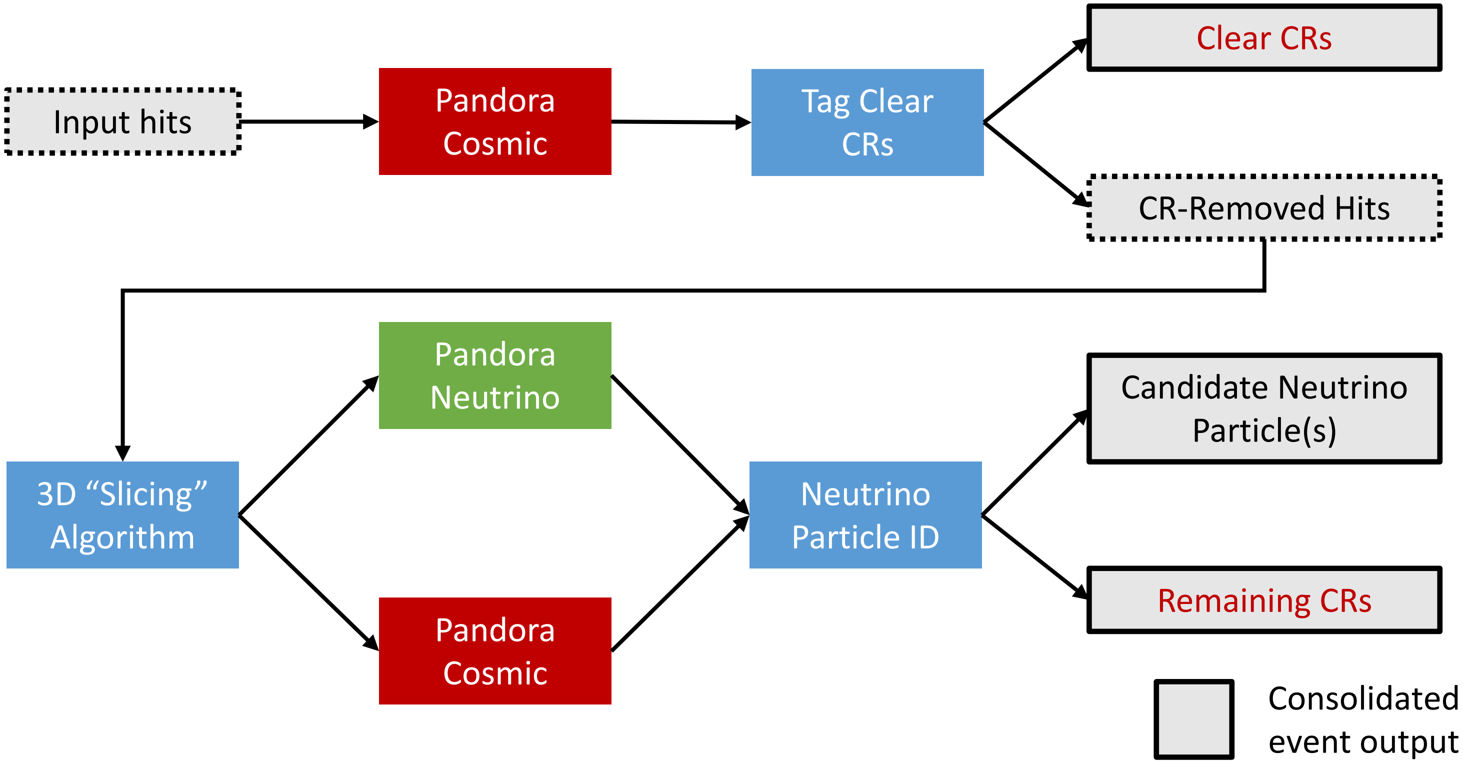


Cosmic-Ray Muons

Also includes delta ray reconstruction

Target reconstruction of straight-line particles in detector (e.g. cosmic-ray muons)

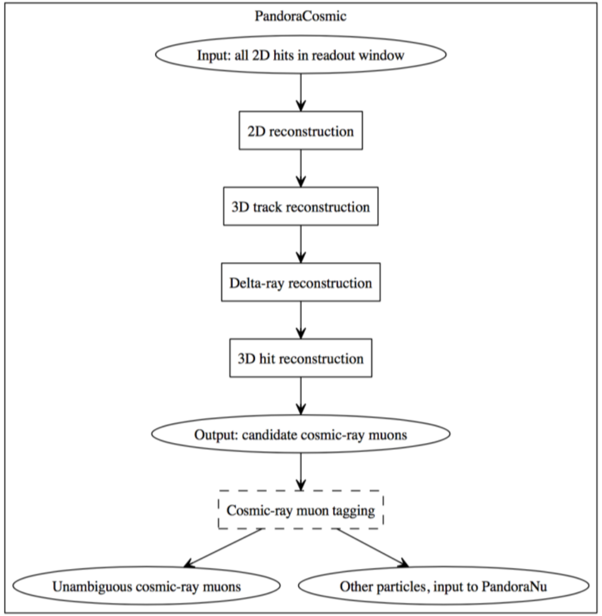
Consolidated reconstruction

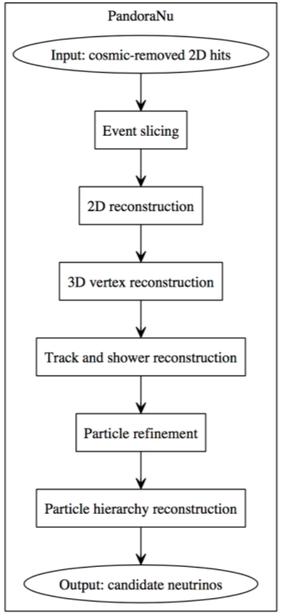


Consolidated reconstruction - Algorithm chains

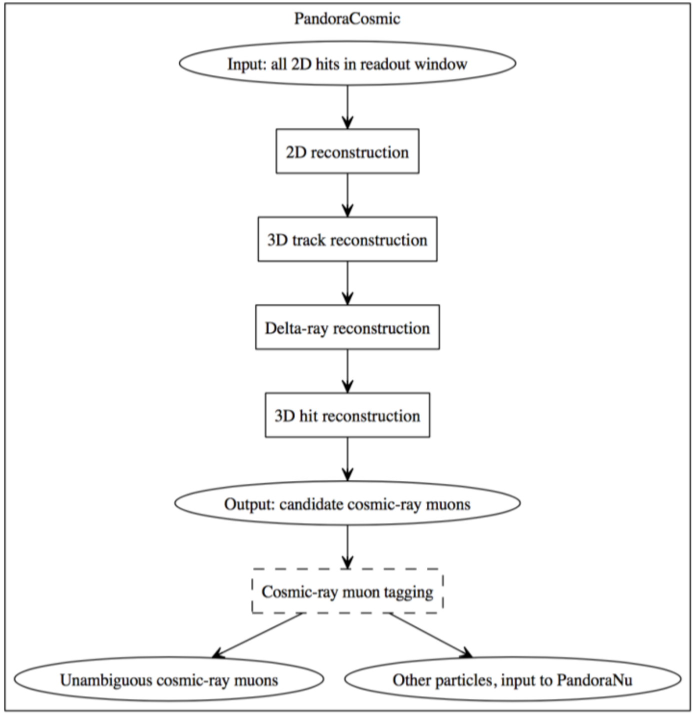
- Two Pandora algorithm chains created for LArTPC use, with many algs in common:
  - PandoraCosmic: strongly track-oriented; showers assumed to be delta rays, added as daughters of primary muons; muon vertices at track high-y coordinate.
  - PandoraNu: finds neutrino interaction vertex and protects all particles emerging from vertex position. Careful treatment to address track/shower tensions.

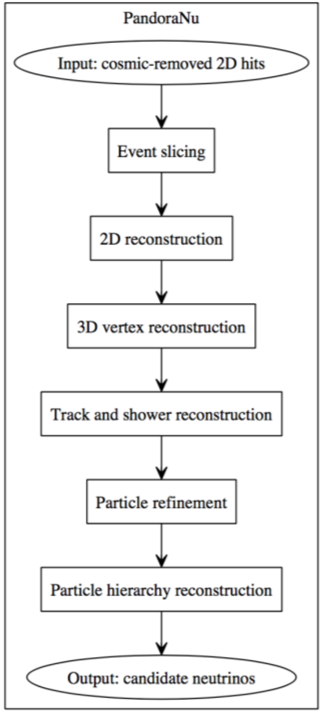
Initially use a two-pass approach:
Input to PandoraNu excludes hits from unambiguous cosmic rays.

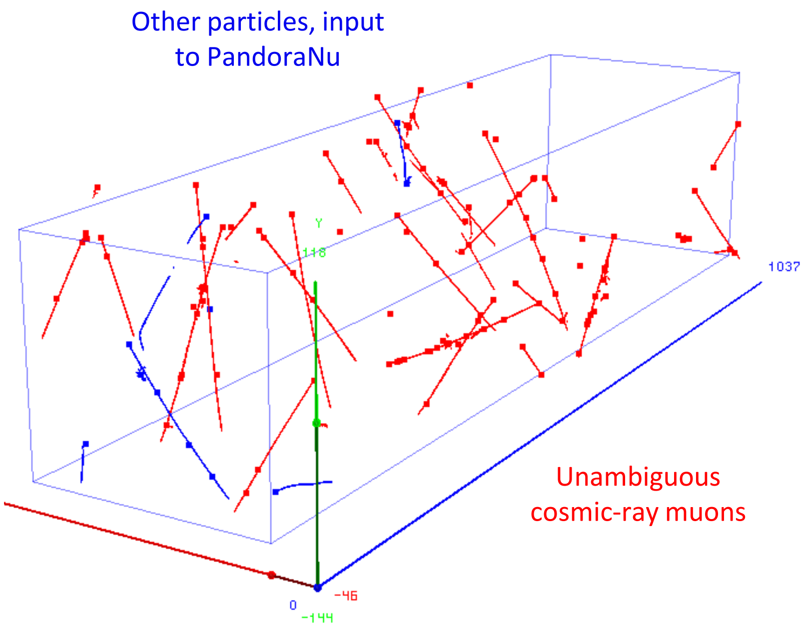




PandoraCosmic → PandoraNu



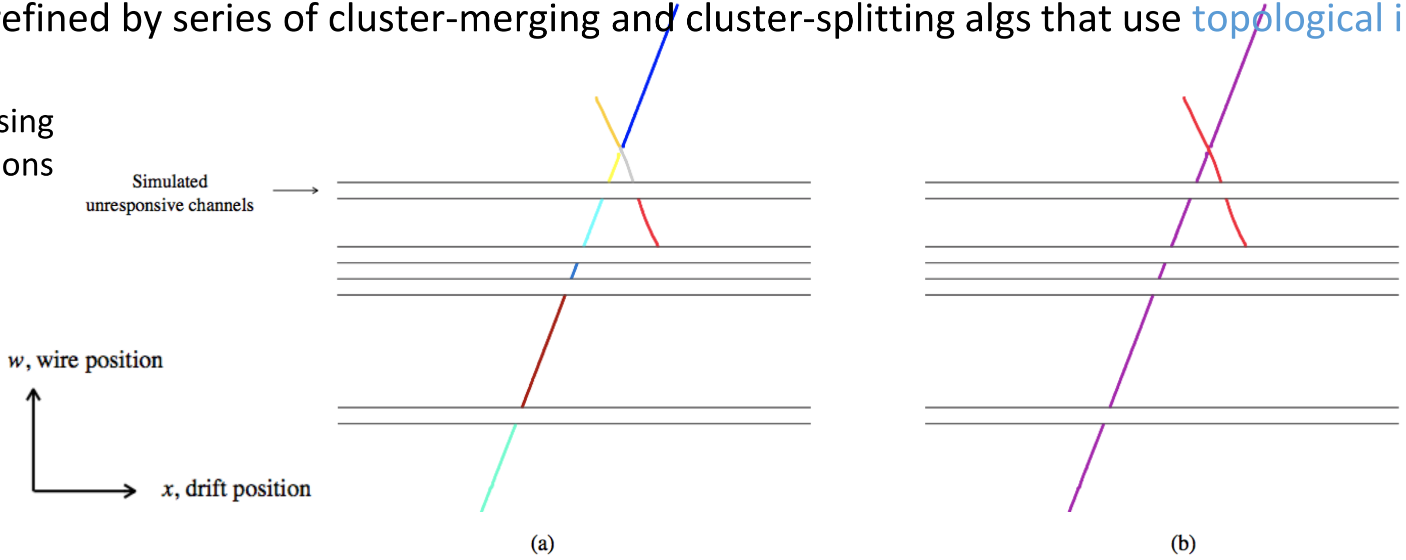




Cosmic-Ray Muon Reconstruction - 2D

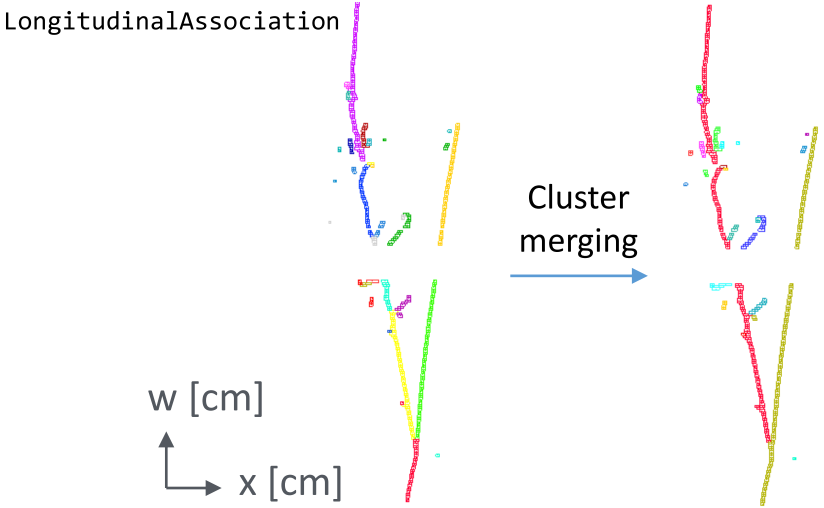
- For each plane, produce list of 2D clusters that represent continuous, unambiguous lines of hits:
  - PandoraCosmic: strongly track-oriented; showers assumed to be delta rays, added as daughters of primary muons; muon vertices at track high-y coordinate.
- Clusters refined by series of cluster-merging and cluster-splitting algs that use topological info.

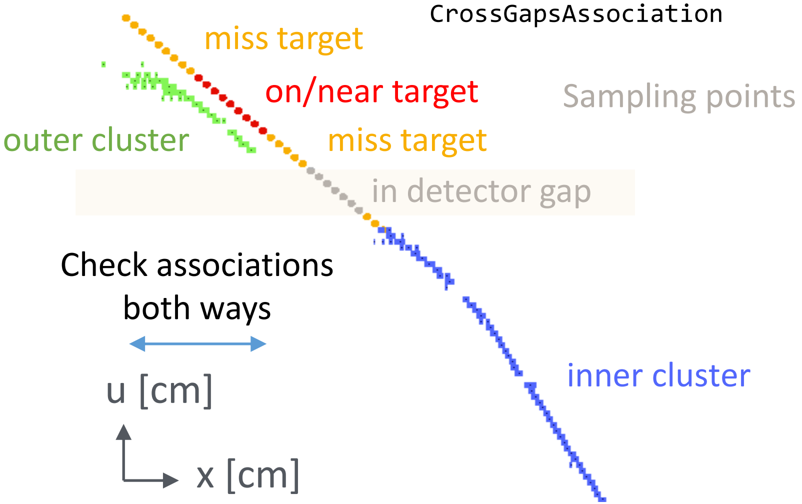
Example: Crossing cosmic-ray muons



Topological Association - 2D

- Cluster-merging algorithms identify associations between multiple 2D clusters and look to grow the clusters to improve completeness, without compromising purity.
  - The challenge for the algorithms is to make cluster-merging decisions in the context of the entire event, rather than just considering individual pairs of clusters in isolation.
  - Typically need to provide a definition of association (for a given pair of clusters), then navigate forwards and backwards to identify chains of associated clusters that can be safely merged.

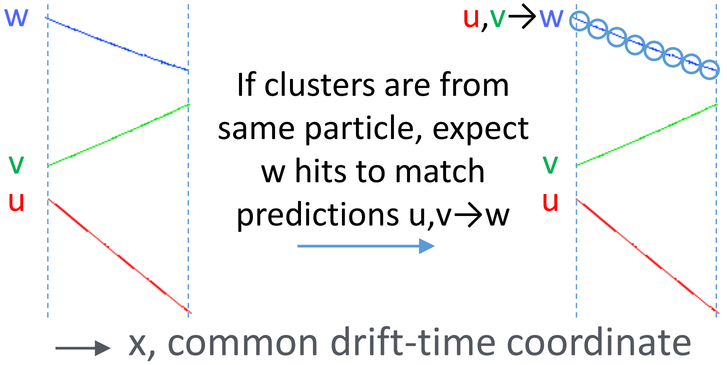




Track Pattern Recognition - 3D

- Our original input was 3x2D images of charged particles in the detector.
- Should now have reconstructed three separate 2D clusters for each particle:
  - Compare 2D clusters from u, v, w planes to find the clusters representing same particle.
  - Exploit common drift-time coordinate and our understanding of wire plane geometry.
  - At given x, compare predictions {u,v→w; v,w→u; w,u→v} with cluster positions, calculating χ^2

Candidate 2D Clusters



Close agreement: predictions sit right upon real hits here

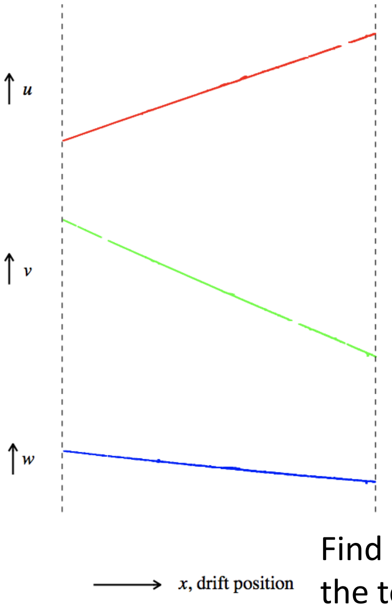
Sample Cluster consistency across common x-overlap region

Store all results in a "tensor", recording x-overlap span, no. of sampling points, no. of "matched" sampling points and χ2. Documents all 2D cluster-matching ambiguities.

Track Pattern Recognition - 3D

Tensor stores overlap details for trios of 2D clusters. Tools make 2D reco changes to resolve any ambiguities. If a tool makes a change (e.g. splits a cluster), all tools run again.

`ClearTracksTool`

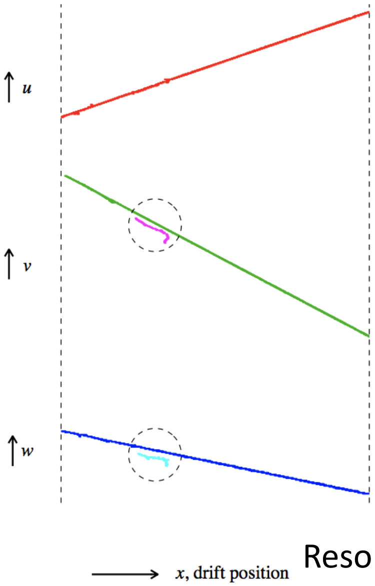


u:*v*:*w*
1:1:1

No. of associated 2D Clusters u:v: w

Find unambiguous elements in the tensor, demanding that the common x-overlap is 90% of the x-span for all three clusters.

`LongTracksTool`

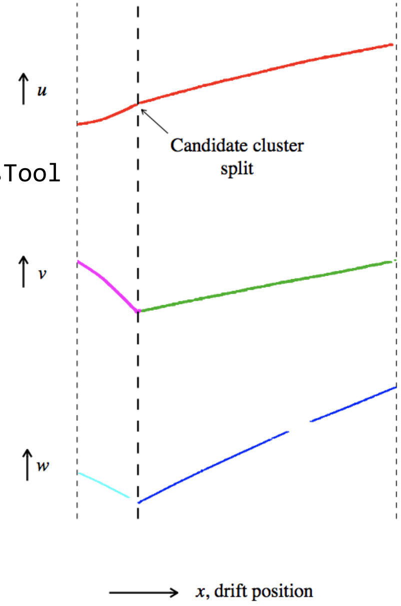


u:*v*:*w*
1:2:2

Resolve obvious ambiguities: clusters are matched in multiple configurations, but one tensor element is much “better” than others.

Track Pattern Recognition - 3D

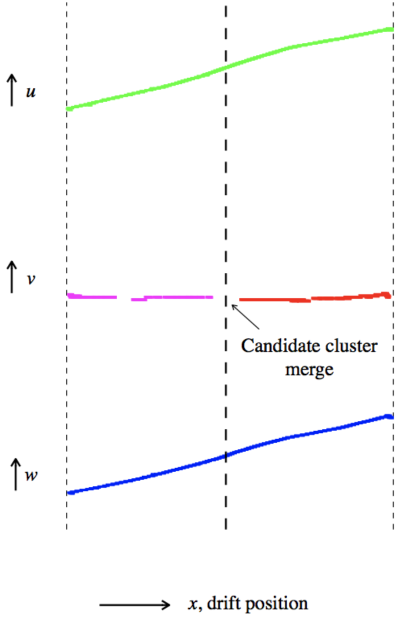
`OvershootTracksTool`



u:*v*:*w*
1:2:2

Two clusters in w and v, matched to common u cluster. Split ***u*** cluster.

`UndershootTracksTool`



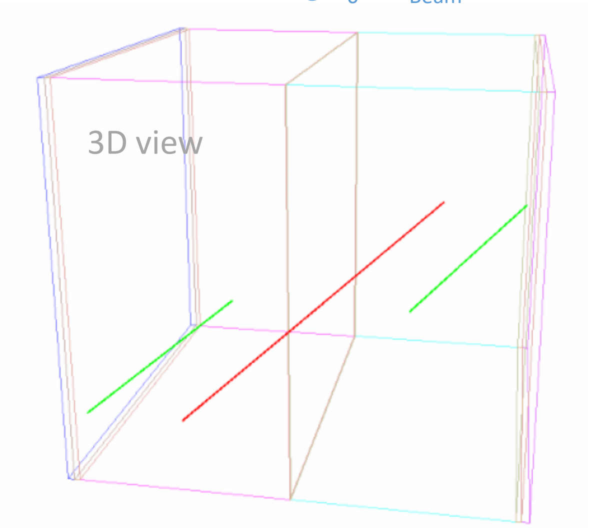
u:*v*:*w*
1:2:1

Two clusters in v view, matched to common clusters in u and w views. Merge ***v*** clusters.

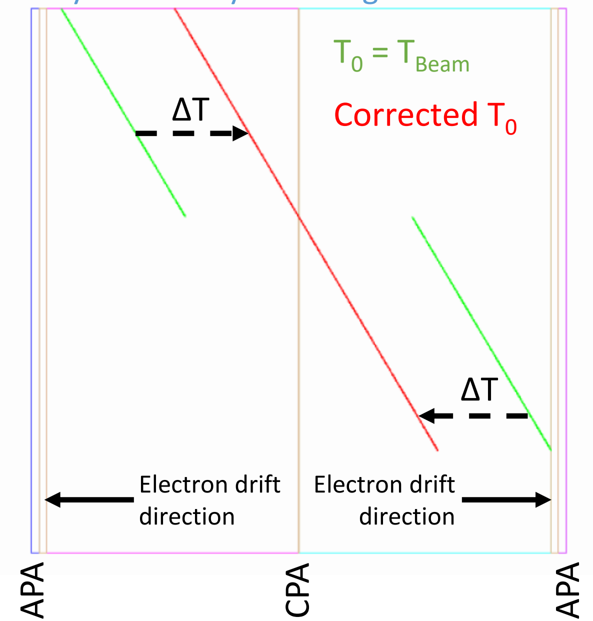
- Use all connected clusters to assess whether this is a true 3D kink topology.
- Modify 2D clusters as appropriate (i.e. merge or split) and update cluster-matching tensor.
- Initial ClearTracks tool then able to identify unambiguous groupings of clusters and form particles.

Stitching and T_0 Identification

- In a LArTPC image, one coordinate derived from drift times of ionisation electrons:
  - But, only know electron arrival times, not actual drift times: need to know start time, T_0
  - For beam particles, can use time of beam spill to set T_0, but unknown for cosmic rays
  - Place all hits assuming $T_0 = T_{Beam}$, but can identify T_0 for any cosmic rays crossing volumes

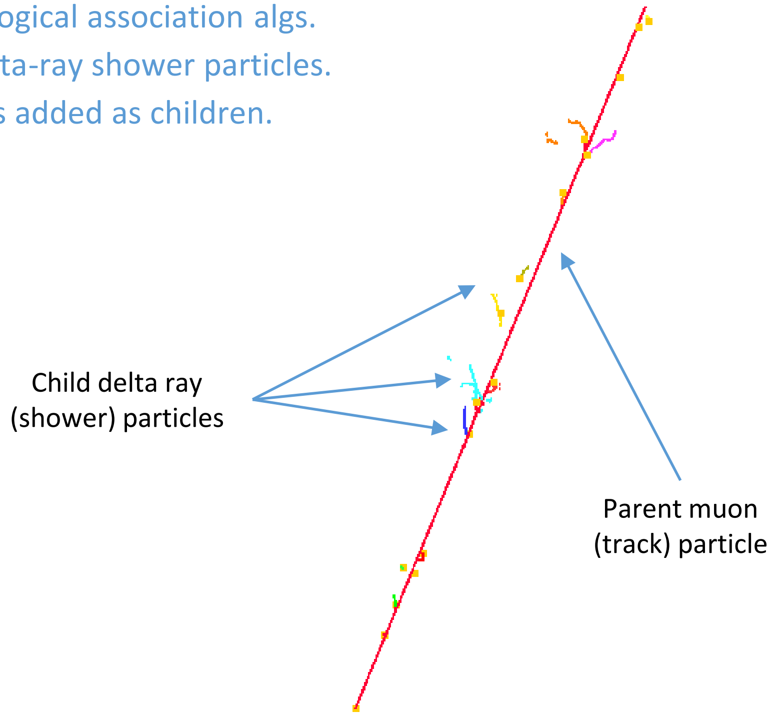


Stitch together any cosmic rays crossing between volumes, identifying T0



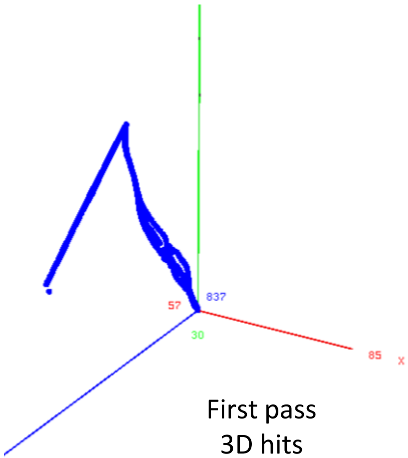
Delta-Ray Reconstruction - 2D, 3D

- Assume any 2D clusters not in a track particle are from delta-ray showers:
  - Simple proximity-based re-clustering of hits, then topological association algs.
  - Delta-ray clusters matched between views, creating delta-ray shower particles.
  - Parent muon particles identified, and delta-ray particles added as children.

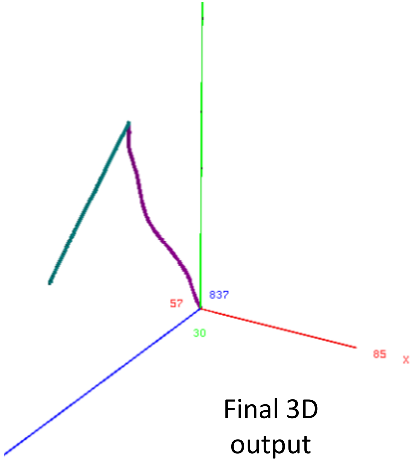


3D Hit/Cluster Reconstruction

- For each 2D Hit, sample clusters in other views at same x, to provide u_{in}, v_{in} and w_{in} values
- Provided u_{in}, v_{in} and w_{in} values don't necessarily correspond to a specific point in 3D space
- Analytic expression to find 3D space point that is most consistent with given u_{in}, v_{in} and w_{in}
  - $\chi^2 = (u_{out} - u_{in})^2 / \sigma_u^2 + (v_{out} - v_{in})^2 / \sigma_v^2 + (w_{out} - w_{in})^2 / \sigma_w^2$
  - Write in terms of unknown y and z, differentiate wrt y, z and solve
  - Can iterate, using fit to current 3D hits (extra terms in χ^2) to produce smooth trajectory

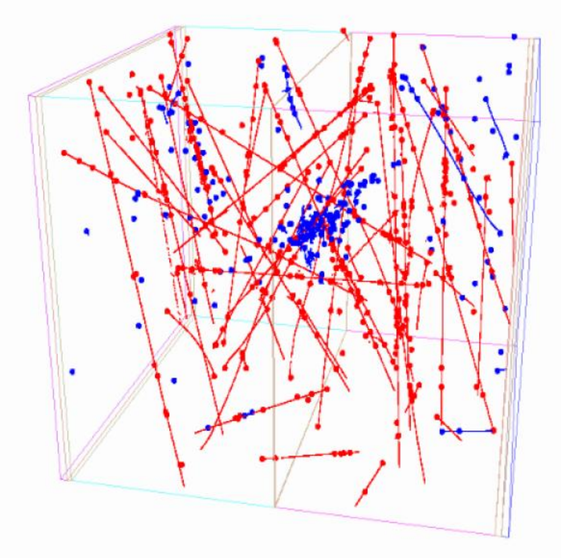


First pass
3D hits



Final 3D
output

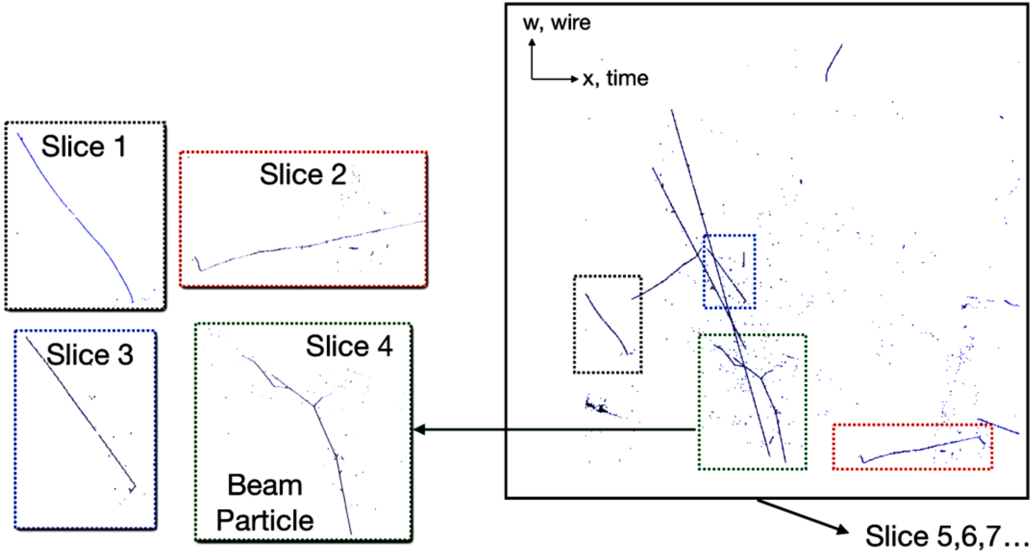
Cosmic Ray Tagging and Slicing



Identify clear cosmic rays (red) and hits to reexamine under test beam hypothesis (blue)

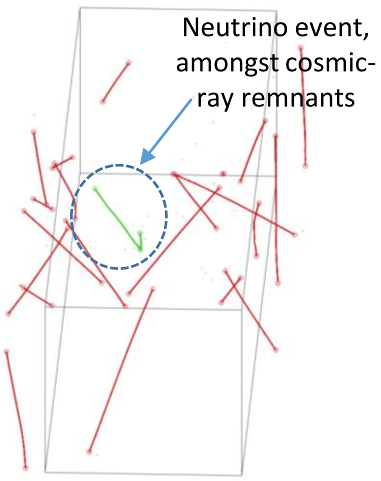
- Clear cosmic rays:
  - Particles appear to be "outside" of detector if $T_0 = T_{Beam}$
  - Particles stitched between volumes using a $T_0 \neq T_{Beam}$
  - Particles pass through the detector: "through going"

- Slice/divide blue hits from separate interactions
- Reconstruct each slice as test beam particle
- Then choose between cosmic ray or test beam outcome for each slice



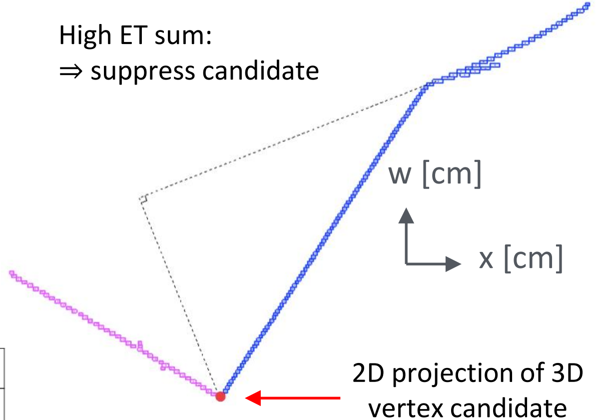
Neutrino Reconstruction

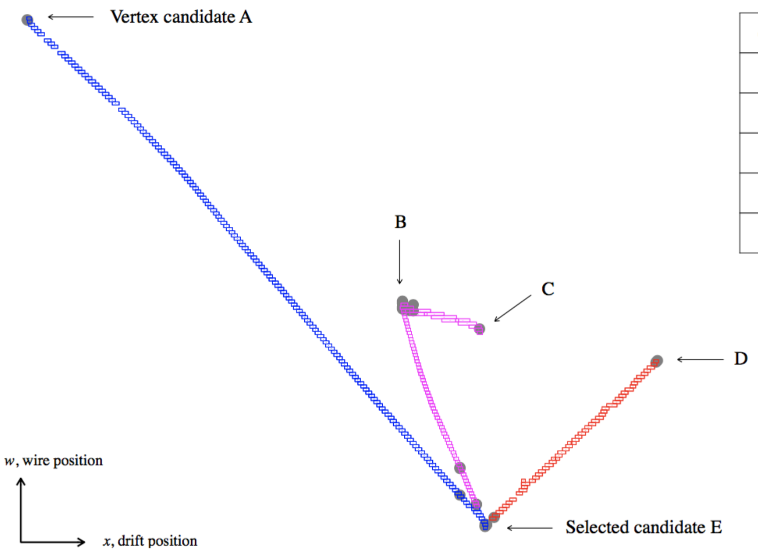
- Must be able to deal with presence of any cosmic-ray muon remnants.
  - Run fast version of reconstruction, up to 3D hit creation
  - “Slice” 3D hits into separate interactions, processing each slice in isolation.
  - Each slice ⇒ candidate neutrino particle.
- Neutrino pass reuses track-oriented clustering and topological association.
  - Topological association algs must handle rather more complex topologies.
  - Specific effort to reconstruct neutrino interaction vertex.
  - More sophisticated efforts to reconstruct showers.



Vertex Reconstruction – BDT version

- Search for neutrino interaction vertex:
  - Use pairs of 2D clusters to produce list of possible 3D vertex candidates.
  - Examine candidates, calculate a score for each and select the best.





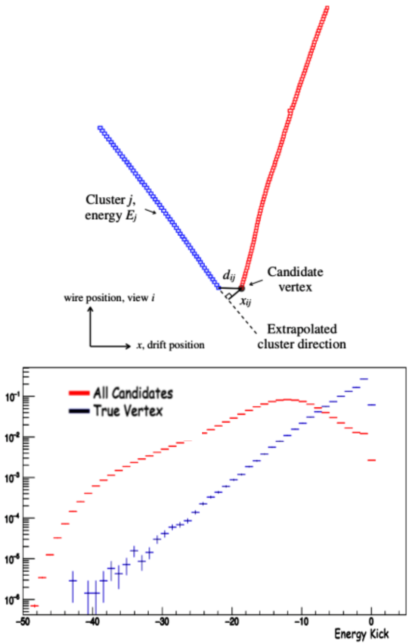
Candidate	S	$S_{\text{energy kick}}$	$S_{\text{asymmetry}}$	$S_{\text{beam deweight}}$
A	4.9E-07	3.5E-06	1.00	0.14
B	1.3E-02	3.1E-02	0.99	0.42
C	1.1E-03	2.4E-03	0.95	0.46
D	5.7E-10	1.1E-09	1.00	0.52
E	9.0E-01	9.0E-01	1.00	0.99

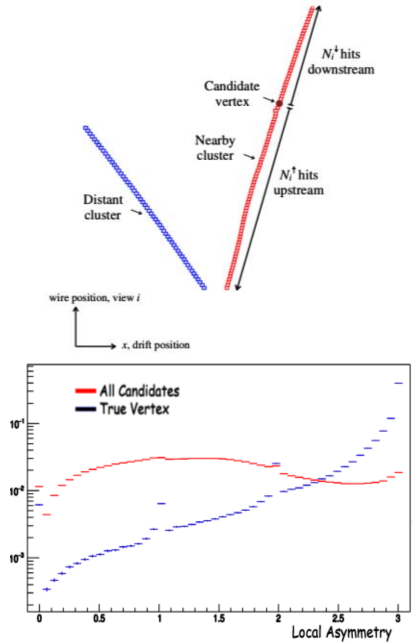
Scores for labelled candidates, with breakdown into component parts:

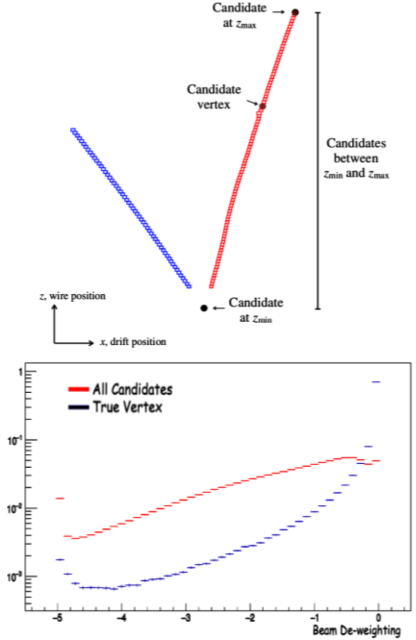
- Downstream usage:
  - Split 2D clusters at projected vertex position.
  - Use vertex to protect primary particles when growing showers.

Vertex Reconstruction – BDT version

- Interaction vertex is an important feature point in our LArTPC images:
  - Continued development, ever-more sophisticated approaches to finding 3D vertex position
  - Boosted Decision Trees (BDTs) or Support Vector Machines (SVMs) to select best candidate







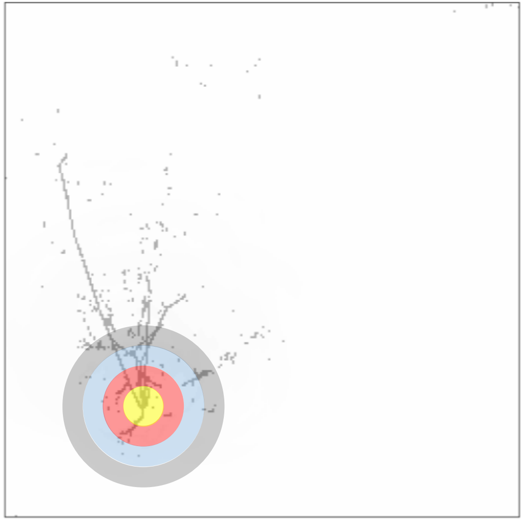
E.g BDT/SVM “features”

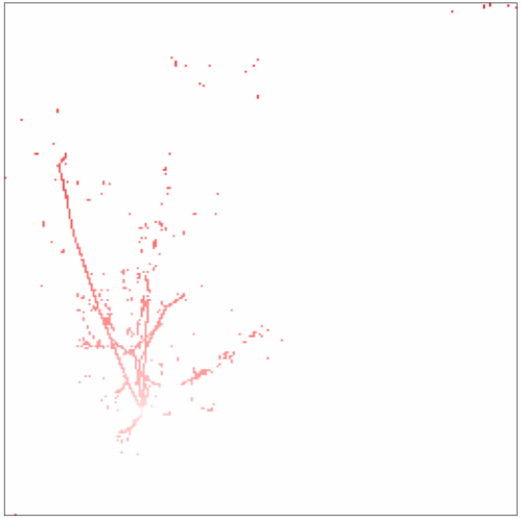
Vertexing reconstruction – U-Net version

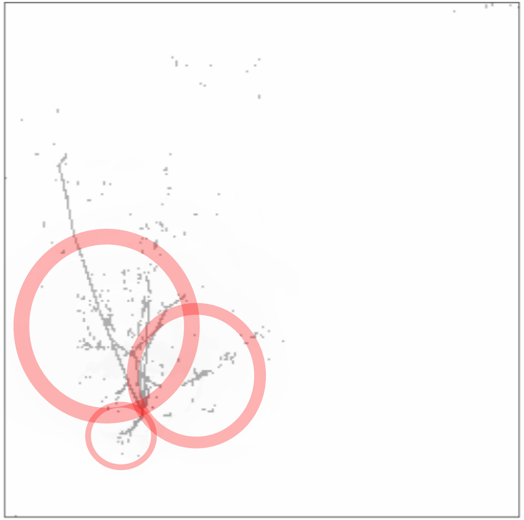
In training hits are assigned a class according to distance from true vertex

Network trained to learn those distances from input images

Network infers hit distances and resultant heat map isolates candidate vertex

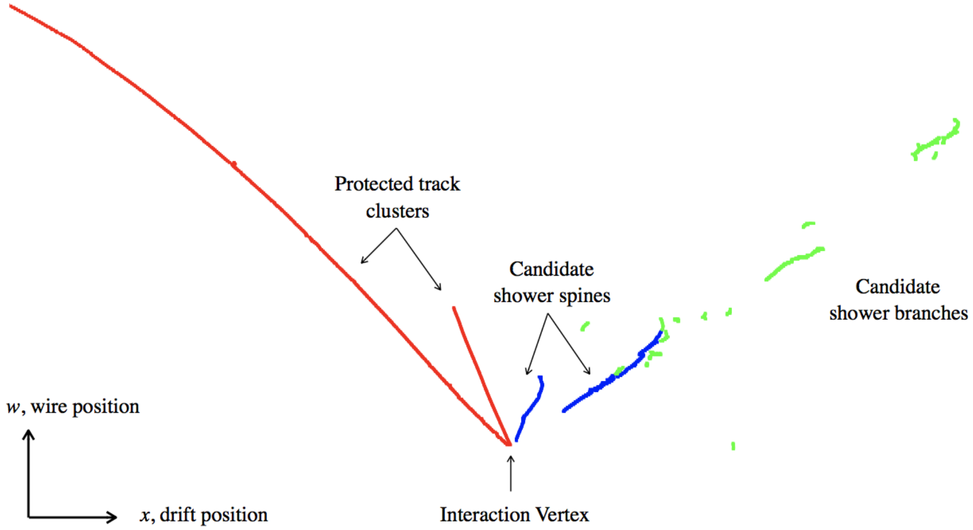






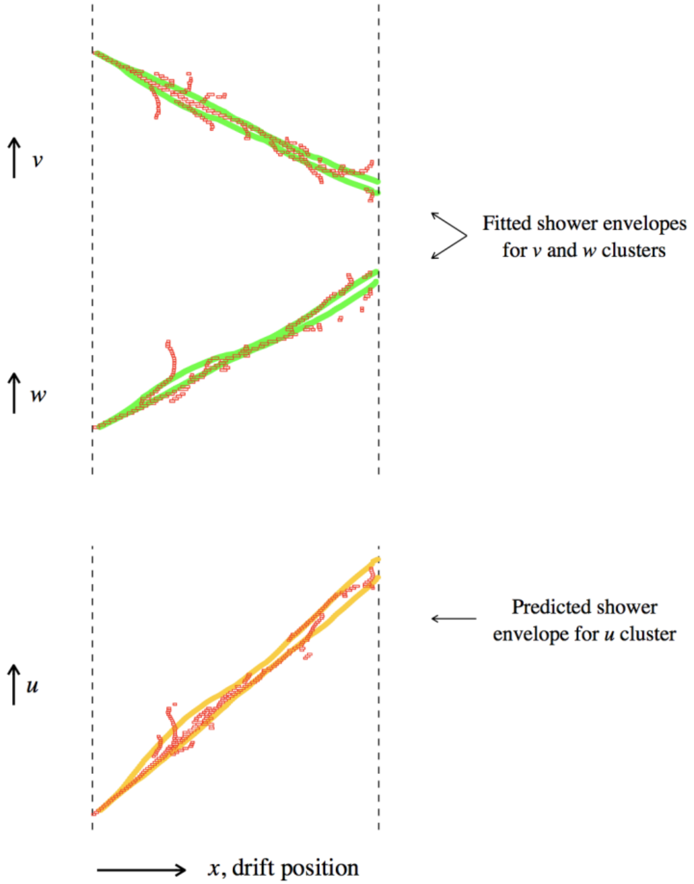
Shower Reconstruction - 2D

- Track reconstruction exactly as in PandoraCosmic, but now also attempt to reconstruct primary electromagnetic showers, from electrons and photons:
  - Characterise 2D clusters as track-like or shower-like and use topological properties to identify clusters that might represent shower spines.
  - Add shower-like branch clusters to shower-like spine clusters. Recursively identify branches on the top-level spine candidate, then branches on branches, etc.



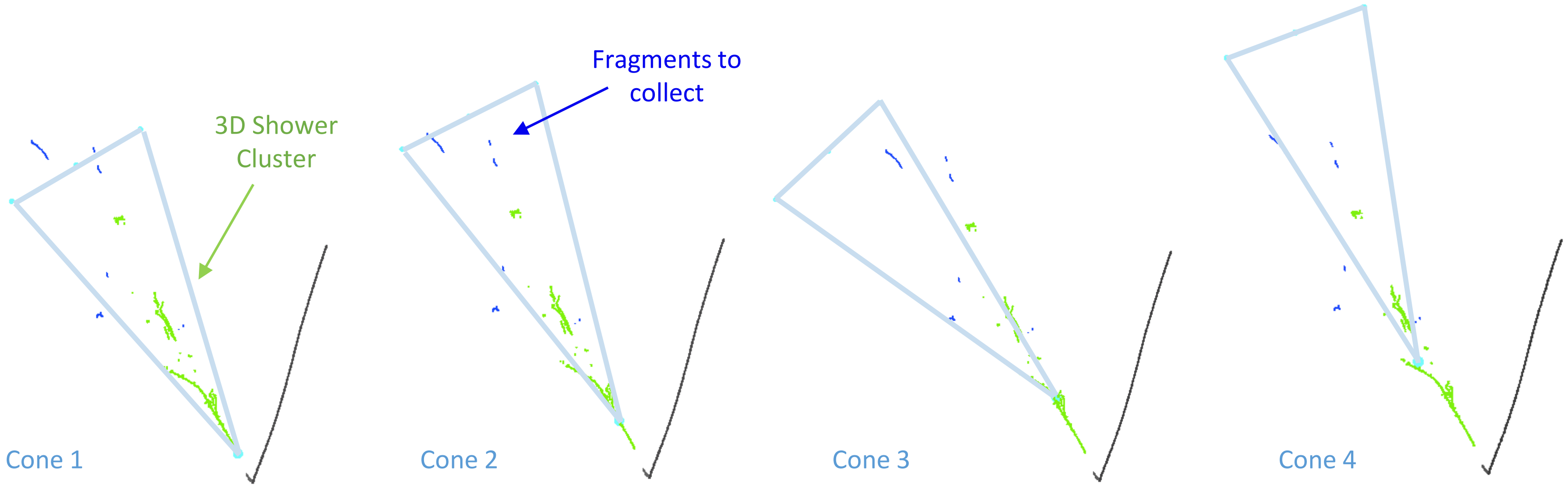
Shower Reconstruction - 3D

- Reuse ideas from track reco to match 2D shower clusters between views:
  - Build a tensor to store cluster overlap and relationship information.
  - Overlap information collected by fitting shower envelope to each 2D cluster.
  - Shower edges from two clusters used to predict envelope for third cluster.



Particle Refinement - 2D, 3D

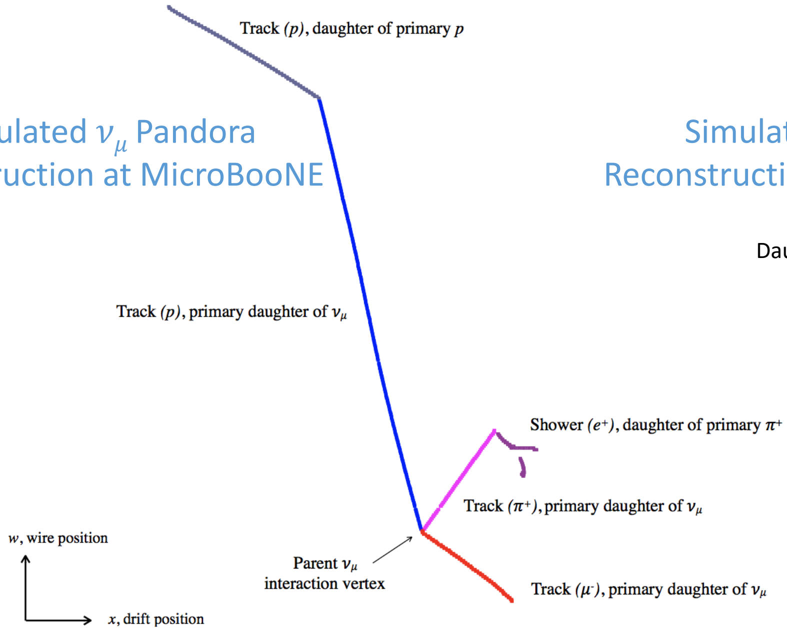
- Series of algs deal with remnants to improve particle completeness (esp. sparse showers):
  - Pick up small, unassociated clusters bounded by the 2D envelopes of shower-like particles.
  - Use sliding linear fits to 3D shower clusters to define cones for merging small downstream shower particles or picking up additional unassociated clusters.
  - If anything left at end, dissolve clusters and assign hits to nearest shower particles in range.



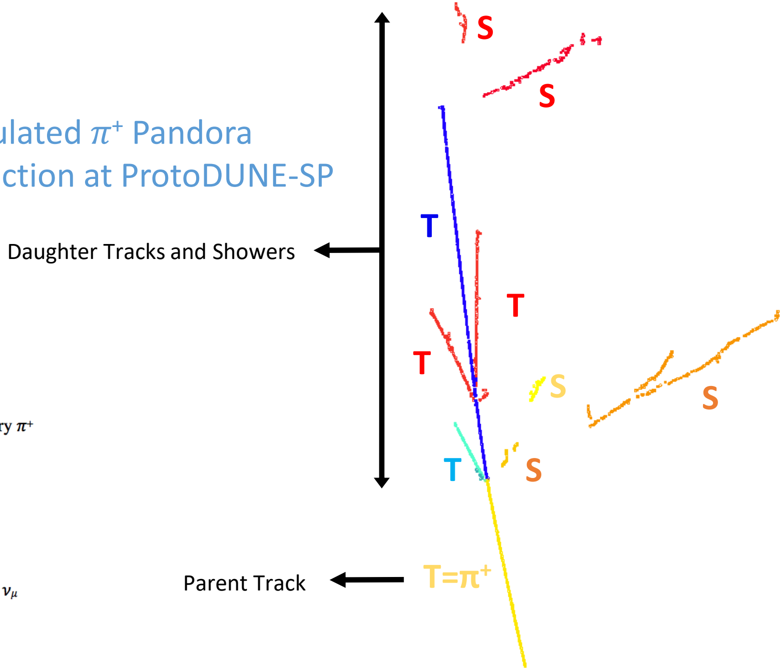
Particle Hierarchy Reconstruction - 3D

- Use 3D clusters to organize particles into a hierarchy, working outwards from interaction vtx
  - Use the hierarchy to access particles in analyzers

Simulated ν_μ Pandora Reconstruction at MicroBooNE

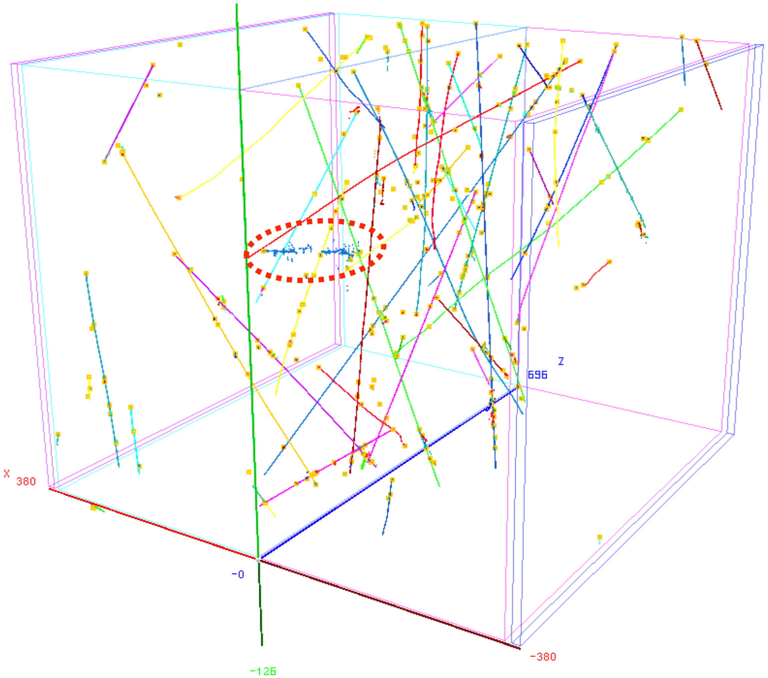


Simulated π^+ Pandora Reconstruction at ProtoDUNE-SP



EPJC (2018) 78:82

Consolidated output



E.g. Reconstruction output: test beam particle (electron) and: N reconstructed cosmic-ray muon hierarchies

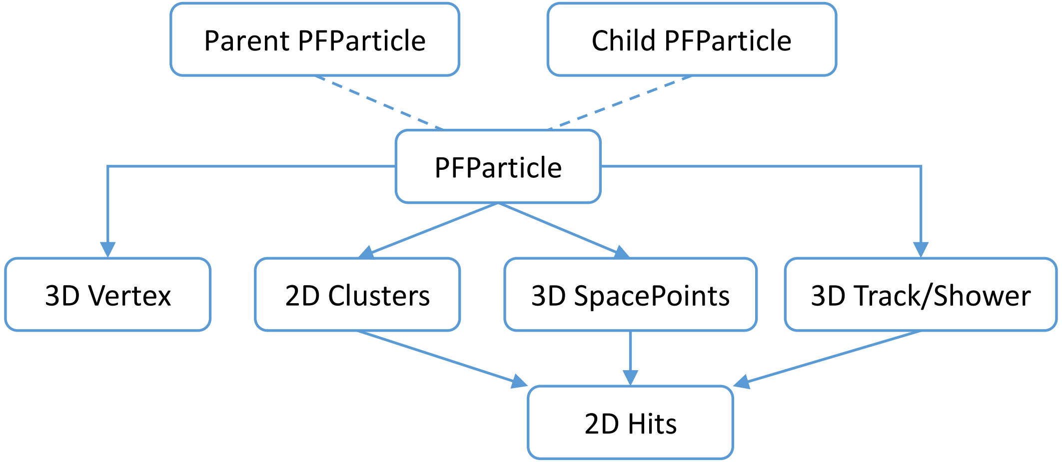
Child tracks and showers

Parent track

E.g. Test beam particle: charged pion

Reconstruction Output

- Must translate output from Pandora Event Data Model to LArSoft Event Data Model. The key output is the PFParticle (PF ⇒ Particle Flow):
  - Each PFParticle corresponds to a distinct track or shower and is associated to 2D clusters.
  - 2D clusters group hits from each readout plane, and are associated to the input 2D hits.
  - PFParticles also associated to 3D spacepoints and a 3D vertex.
  - PFParticles placed in a hierarchy, with identified parent-daughter relationships.
  - PFParticles flagged as track-like or shower-like (both outcomes are persisted).



Just the most important outputs shown here

Overall summary

- The use of Liquid Argon technology is one of the cornerstones of the current and future neutrino programmes.
- High-performance reconstruction techniques are required in order to fully exploit the imaging capabilities offered by LArTPCs:
  - Pandora multi-algorithm approach uses large numbers of decoupled algorithms to gradually build up a picture of events.
  - Output is a carefully-arranged hierarchy of reconstructed particles, each corresponding to a distinct track or shower.

We will now try visualizing actions of individual algorithms