



# **LAr Event Reconstruction with the PANDORA Software Development Kit**

---

**Andy Blake, John Marshall, Mark Thomson  
(Cambridge University)**

LArSoft Reconstruction Meeting,  
December 12<sup>th</sup>, 2012.

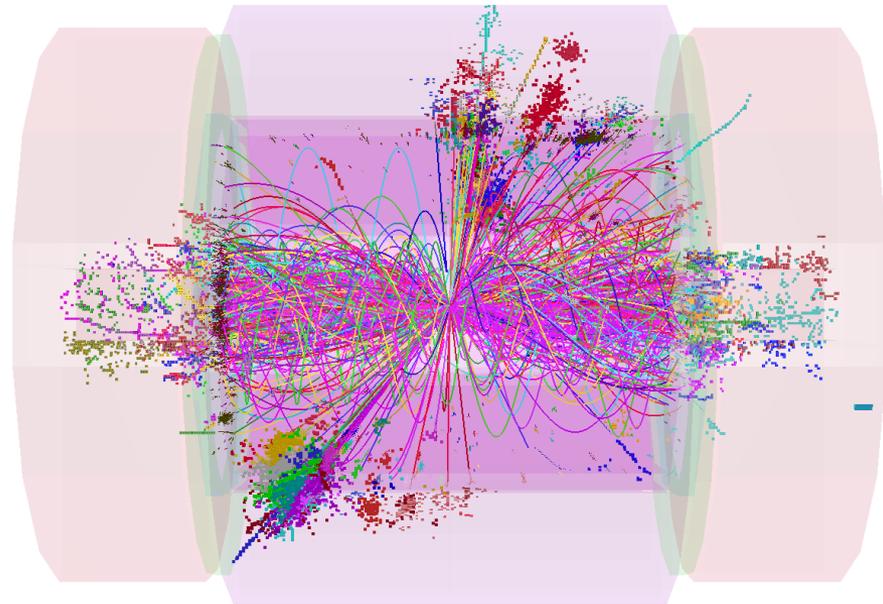
# Overview

- **Over the past couple of months, Cambridge has begun an effort to develop LAr reconstruction algorithms.**
  - Using expertise in reconstruction at fine-grain particle detectors, having developed particle flow calorimetry for ILC and CLIC.
- **We are using the Pandora Software Development Kit, written by Mark Thomson and John Marshall at Cambridge.**
  - Contains a set of generic analysis tools for fine-grain detectors, developed during the ILC/CLIC effort.
  - Designed for multi-purpose and multi-experiment applications. (currently in use by ILC/CLIC and ATLAS).
  - The Pandora tools are readily applicable to LAr detectors!
- **Our LAr efforts have now reached a fairly advanced stage.**
  - Using the Pandora tools, we have developed a chain of 2D pattern recognition algorithms for LAr reconstruction.
  - We are now making the transition from 2D to 3D algorithms.
  - Our approach looks promising! But there's still much to do...

# Pandora

- **Pandora is a software toolkit for developing and running pattern recognition algorithms. It provides the following:**
  - **Tools** for analysing the topology of particle interactions.
  - **Template algorithms** for reconstructing tracks and showers, using topological information.
  - An **environment** for building reconstruction algorithms.
  - **Visualisation** tools (see right).
  - A set of robust **APIs for running reconstruction tasks**.
  - A set of **reconstruction objects**, managed using **STL containers**.
- **A single-library C++ framework.**
  - No dependencies (other than ROOT-based event display).

$H^+H^-$  production in CLIC detector, reconstructed by PANDORA.

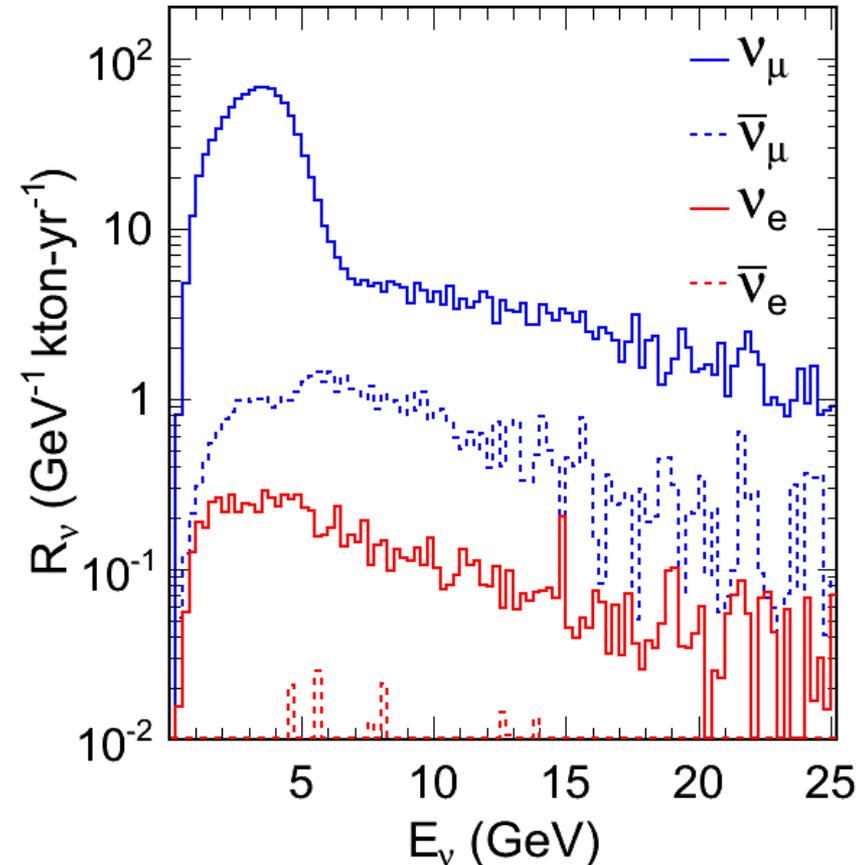


<https://svnsrv.desy.de/viewvc/PandoraPFANew/>

$e^+e^- \rightarrow H^+H^- \rightarrow t\bar{b}b\bar{t} \rightarrow 8 \text{ jets}$

# Reconstruction Software

- **Have integrated Pandora into LArSoft as an external package.**
  - Using LArSoft, S2012.05.09.
- **Simulate two samples of beam neutrino interactions using MicroBooNE geometry.**
  - LBNE nominal beam spectrum.
    - ◇ Events with primary tracks.
  - LBNE spectrum with  $\nu_\mu$  and  $\nu_e$  neutrinos interchanged.
    - ◇ Events with primary showers.
- **Run hit-finder over each sample.**
  - Run FFT hit-finding algorithm.
- **Pass the reconstructed hits to Pandora for pattern recognition.**
  - By writing an ART 'producer'.



Nominal LBNE beam spectrum  
(2010 LBNE Flux files + GENIE).

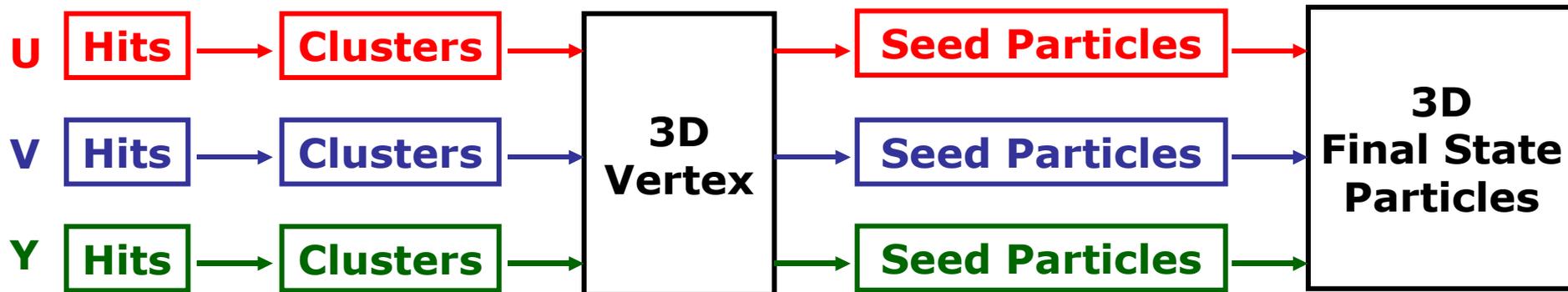
# Reconstruction Strategy

## 2D Reconstruction (Prototype reconstruction, using one view):



Strategy: Reconstruct 2D tracks and showers simultaneously.

## 3D Reconstruction (Full reconstruction, using all three views):



Strategy: Use 3D information as much and as early as possible.

**Pandora was written to solve this type of problem!**

# 2D Reconstruction

- **Current status**: so far, we have used the Pandora tools to develop a full chain of 2D (i.e. single-view) algorithms.
  - Each algorithm is a prototype for further development.
    - ◇ Need to work out the transition from 2D to 3D.
- **Basic strategy**: use **topological associations** to merge hits into clusters and then final-state particles.
  - Pandora provides tools for forming topological associations between hits and clusters.
  - Pandora also provides generic algorithms for merging hits based on these associations.
- **Pandora philosophy**: break down the problem into **many separate algorithms**, each with a specific purpose.
  - Each algorithm grows the event in a particular way.
  - Try to be conservative, and only make good merges.
    - ◇ Hard to undo bad merges!

# 2D Reconstruction

## **1. Start by merging together hits to form clusters.**

- Use nearest-neighbour clustering algorithm to join together contiguous hits.

## **2. Use topological associations (proximity and pointing) to grow clusters.**

- Making end-to-end merges, creating extended clusters. Identify 'spine' of event.

## **3. Use resulting extended clusters to reconstruct vertex.**

- Find the location that best 'connects' the event.

## **4. Identify final-state particles emitted from vertex.**

- Apply 'length-growing' algorithms to grow tracks.
- Apply 'branch-growing' algorithms to grow showers.

## **5. Use topological associations to add remaining clusters.**

- 'Box' algorithm for enclosed clusters.
- 'Parallel' algorithm for neighbouring clusters.
- 'Cone' algorithm for downstream clusters.

# 2D Reconstruction

- **So far, we've developed 18 algorithms that implement these steps.**

- Each algorithm performs a particular role in the reconstruction.
  - ◇ The names roughly indicate the purpose!
- We're continuing to tune and improve them.
  - ◇ And developing more.
- We currently don't use:
  - ◇ 3D information.
  - ◇ Pulse height.

**ClusterCreation**

**ClusterAssociation**

**ClusterExtension**

**KinkSplitting**

**VertexFinding**

**VertexParticleFinding**

**ParticleLengthGrowing**

**ParticleFinding**

**ParticleBranchGrowing**

**ParticleConsolidation**

**ParticleRelegation**

**ParallelMerging**

**BoundedBoxMerging**

**ConeBasedMerging**

**RemnantClustering**

**RemnantConsolidation**

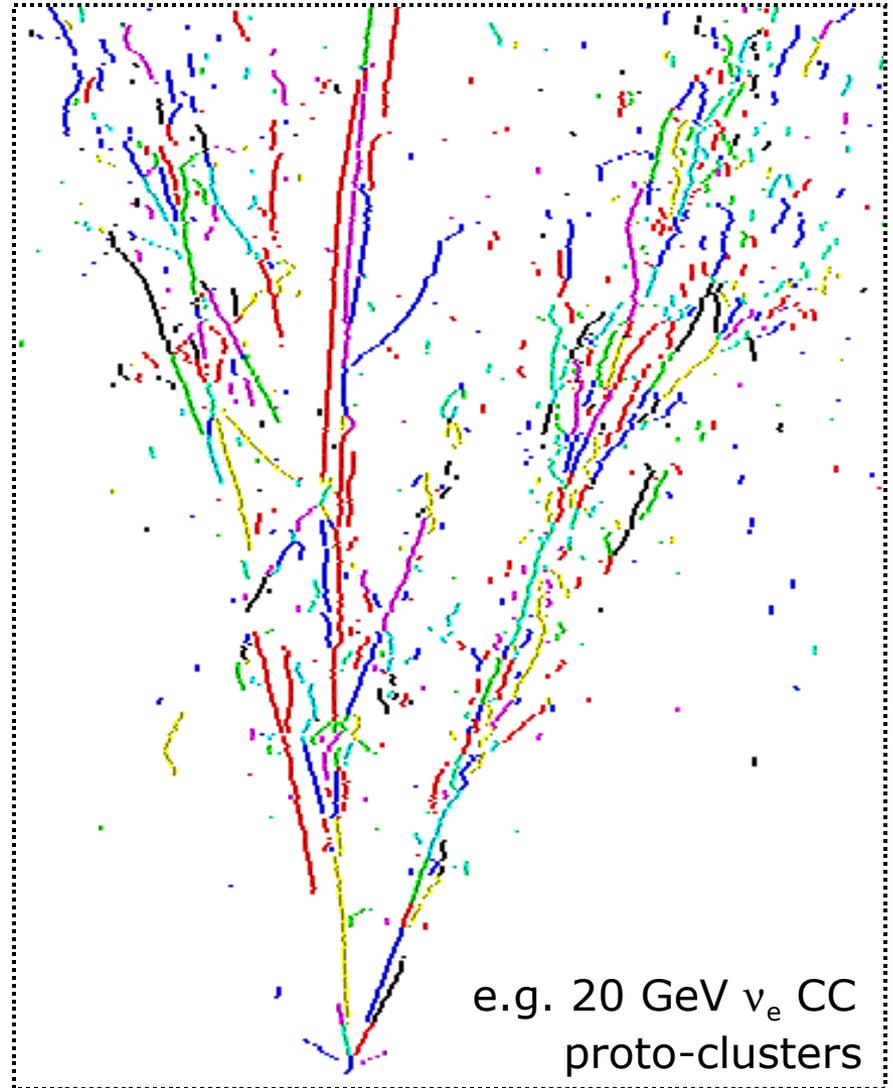
**IsolatedHitMerging**

**2DParticleCreation**

1. Create clusters
2. Extend clusters
3. Identify vertex
4. Find final-state particles (tracks and showers)
5. Add remaining hits and clusters to final-state particles

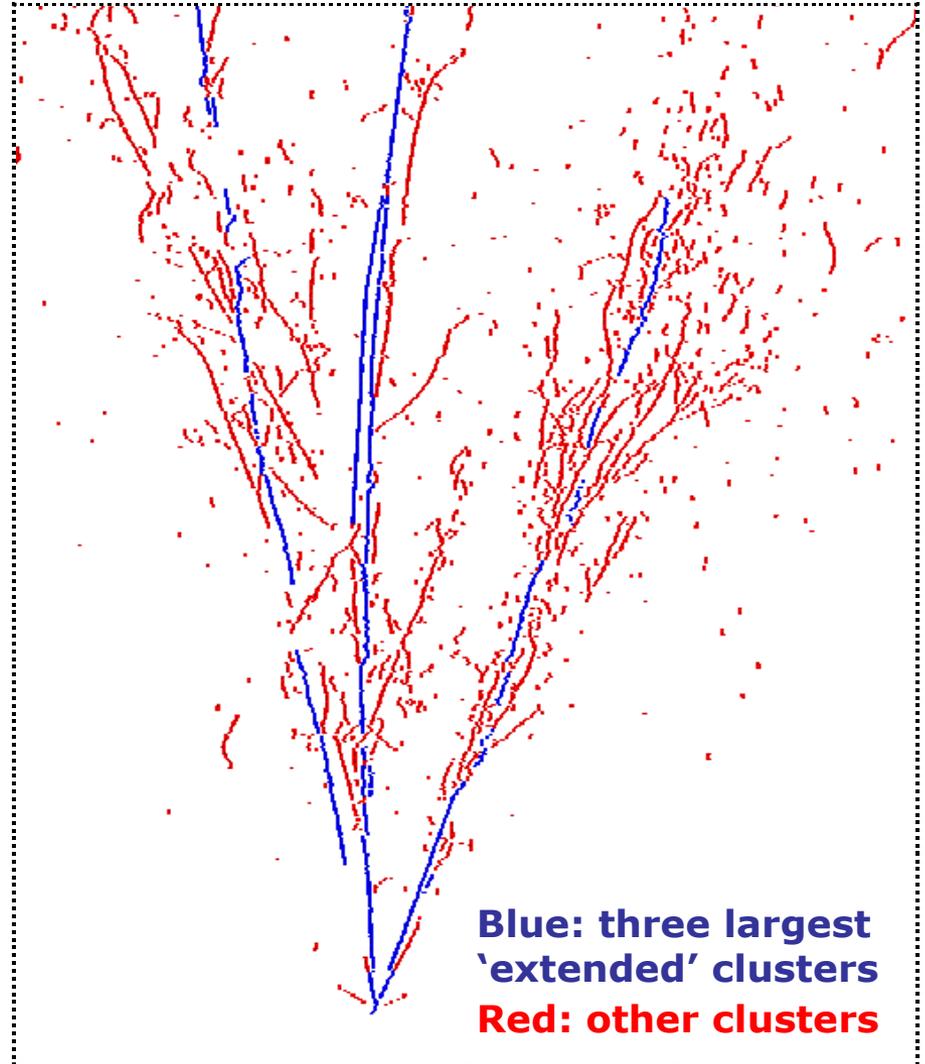
# 1. Cluster Creation

- **Start by merging together groups of hits to form an initial set of proto-clusters.**
- **Method:**
  - Join together contiguous hits using a custom-built nearest neighbour algorithm.
- **Results:**
  - Obtain a patchwork of clusters that can be joined together to form candidate particles.
    - ◊ We just need to make the correct joins!
  - Note that most proto-clusters have good pointing information.
    - ◊ Valuable in determining which joins are correct.



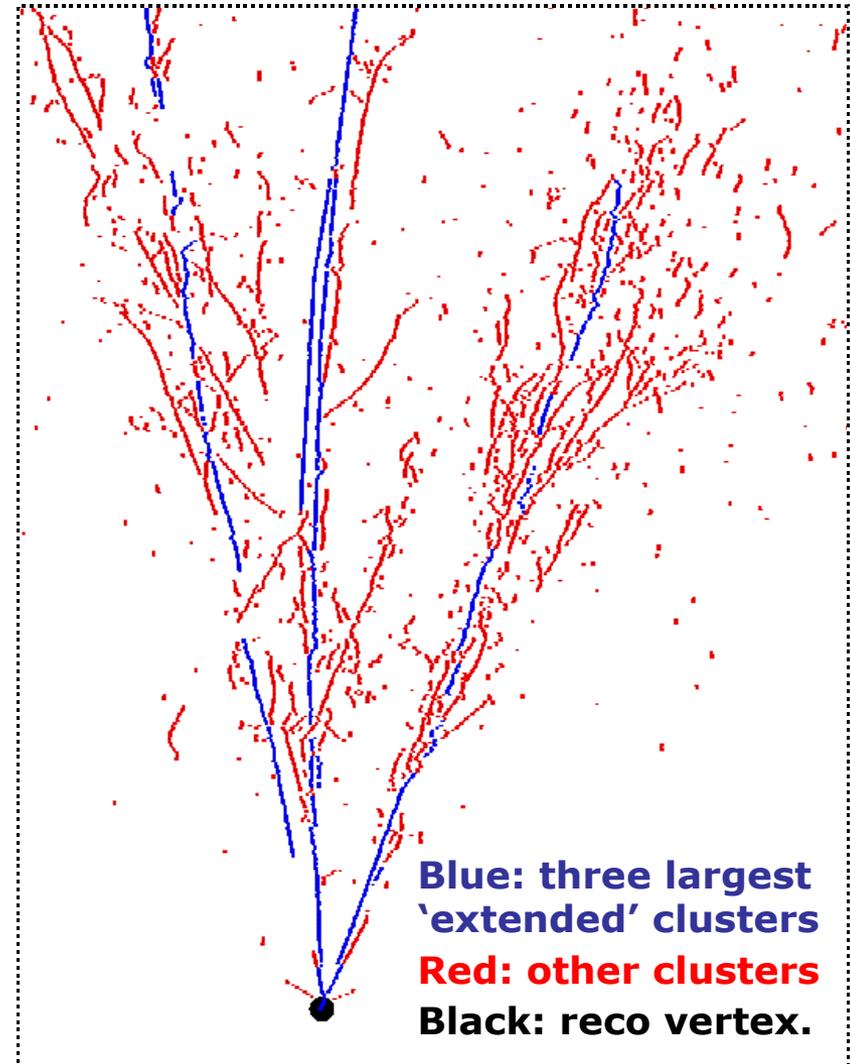
## 2. Cluster Extension

- **Next, reconstruct the 'spine' of each event by making end-to-end joins between proto-clusters.**
- **Method:**
  - Compare all possible pairs of clusters and generate a matrix of possible end-to-end joins.
    - ◇ Small angular deviation.
    - ◇ Small impact parameter.
  - Then make 'good' joins that maximise 'length'.
    - ◇ Treat ambiguities with care.
- **Results:**
  - Does the job, but needs work.  
Crucial to do this step well !!



# 3. Vertex Reconstruction

- **Extended clusters provide enough information to identify vertex.**
- **Method:**
  - For a given vertex hypothesis, apply a 'fast reconstruction', which joins up extended clusters according a simple physics model.
  - Calculate a likelihood function which evaluates the degree of 'connectedness'.
    - ◇ Favours: nodes, emissions, prongs, branches etc...
    - ◇ Disfavours: bad pointing, large angles, isolation etc...
  - Able to find displaced vertices (i.e.  $\pi \rightarrow \gamma\gamma$ ) as well as secondary decays/interactions.



# 3. Vertex Reconstruction

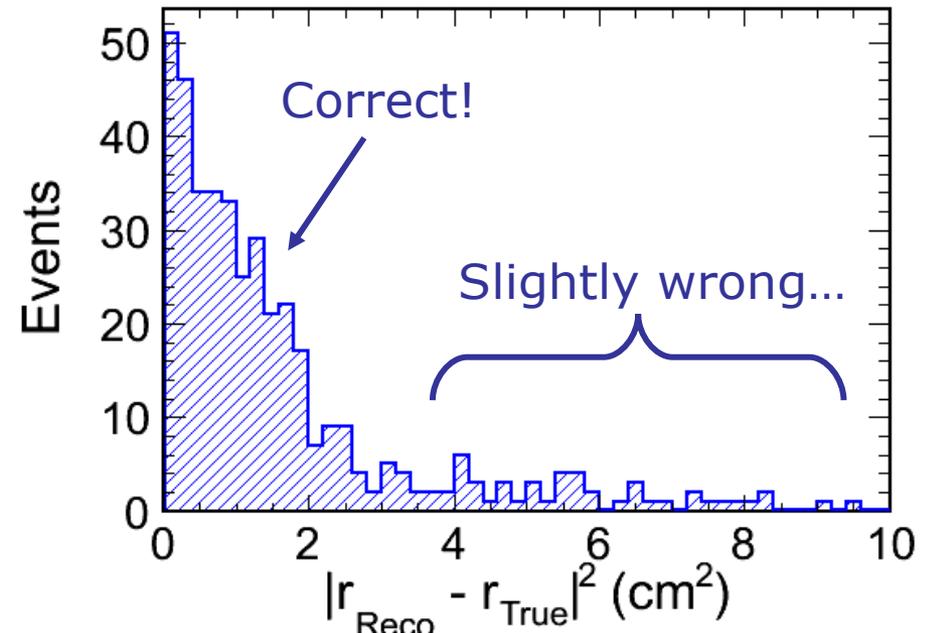
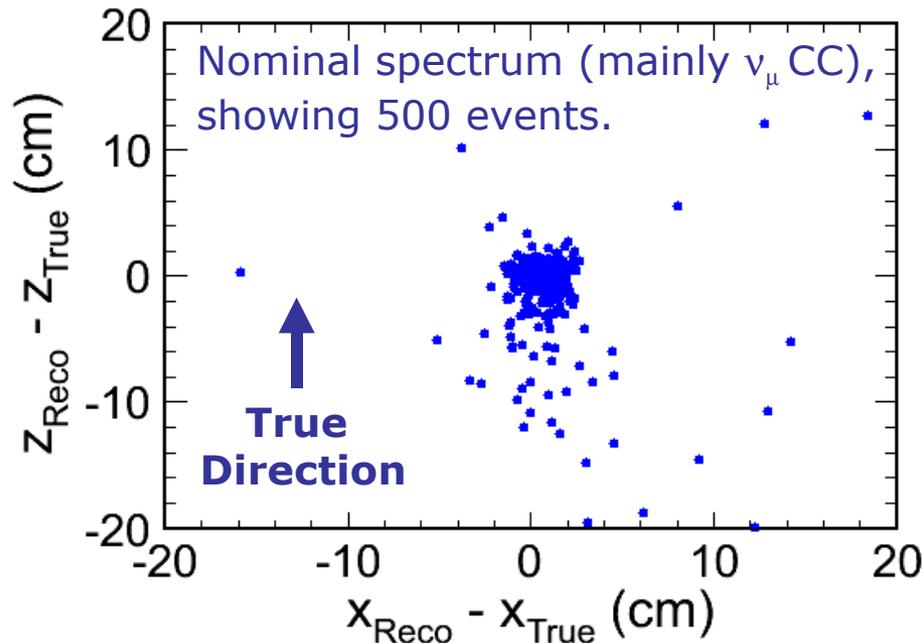
## • Vertex Finding Algorithm (contd.)

### Results:

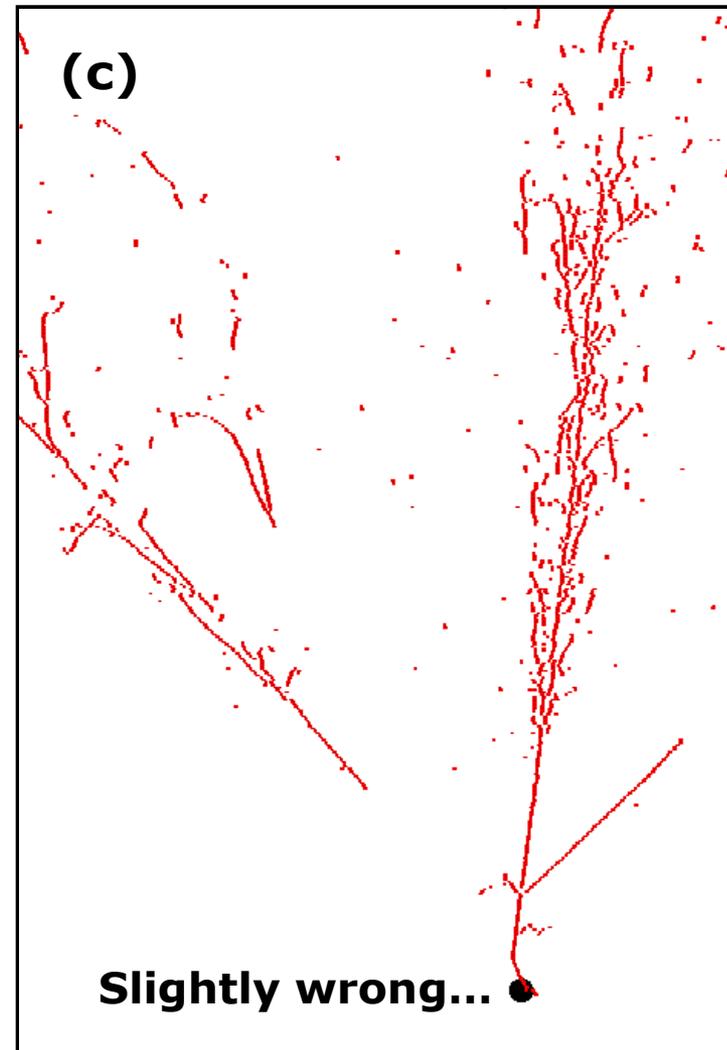
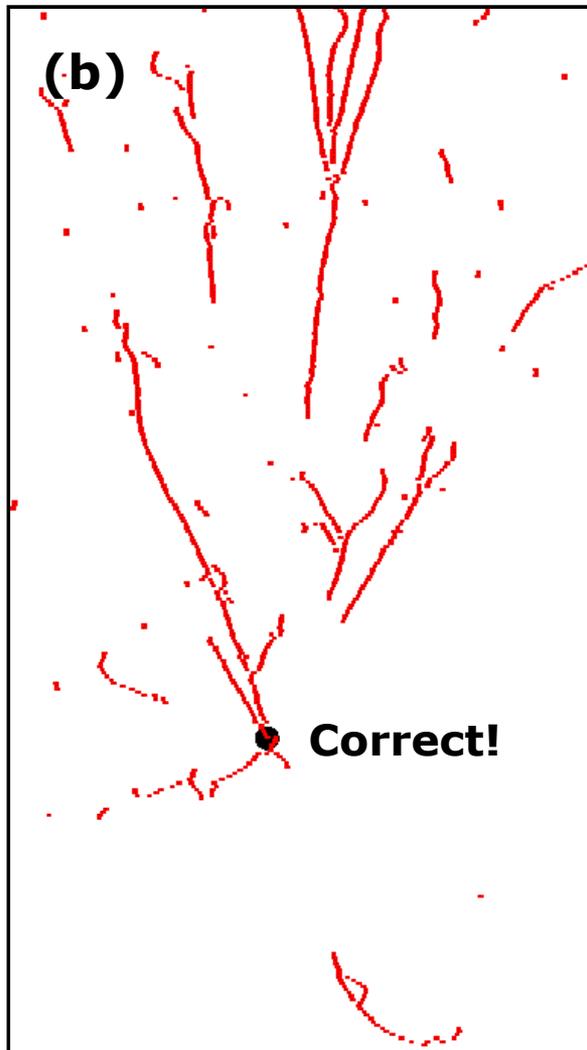
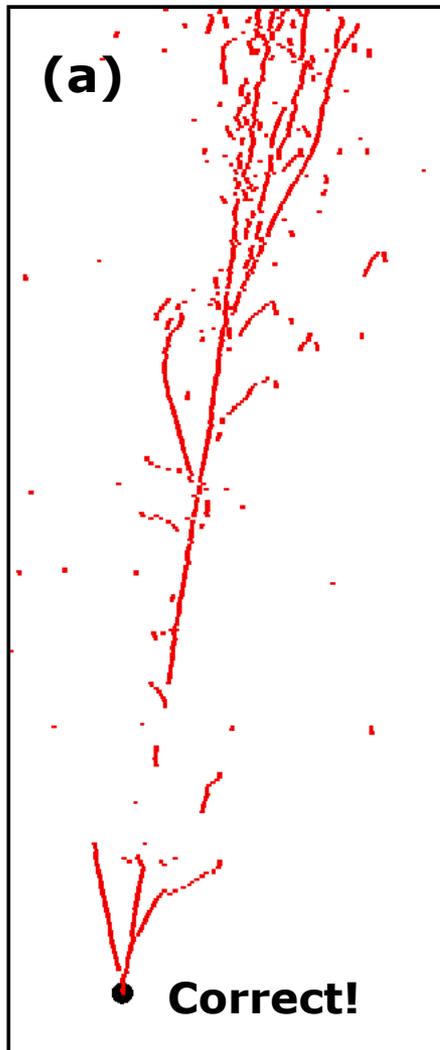
- ◇ Most events well-reconstructed (68% of events within 1.5 cm).
- ◇ However, a substantial tail (10%) have overshot the true vertex.

### Summary of vertex resolution

Resolution	Nominal	$\nu_{\mu} \leftrightarrow \nu_e$
68%	1.5 cm	1.6 cm
90%	7 cm	6 cm

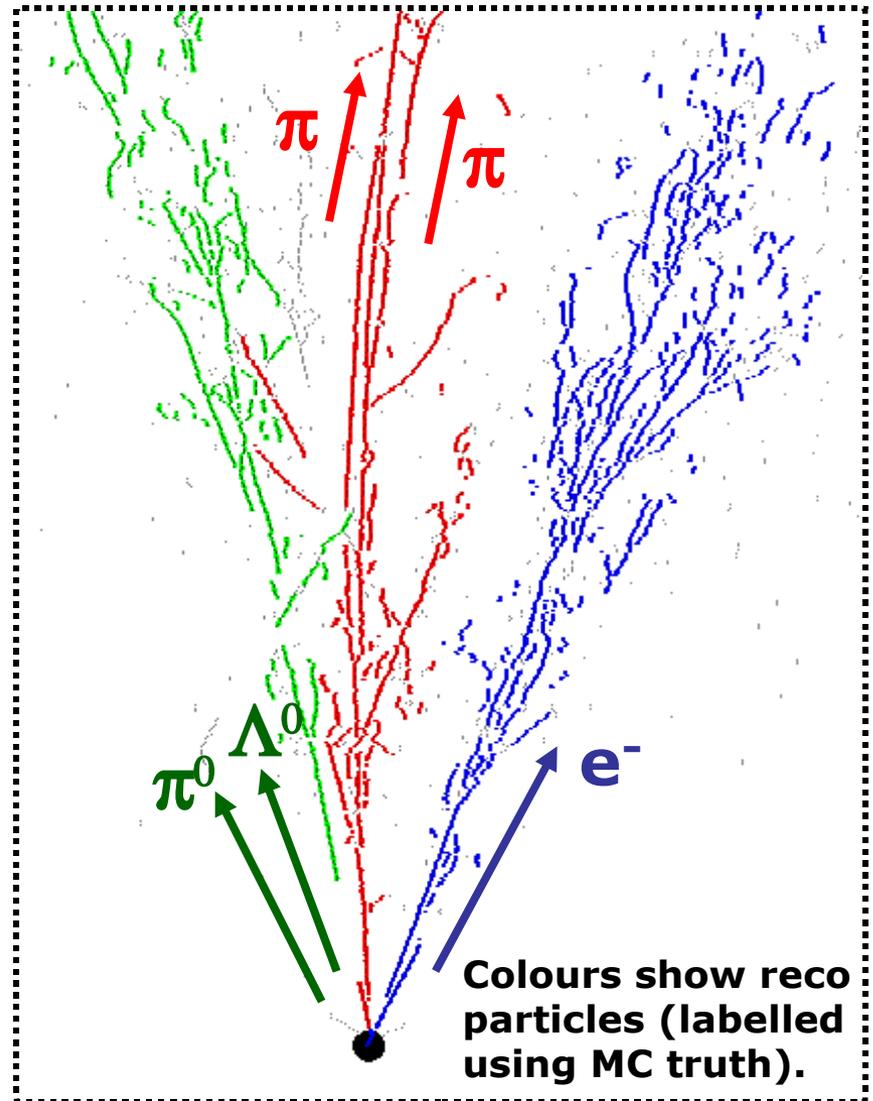


# 3. Vertex Reconstruction

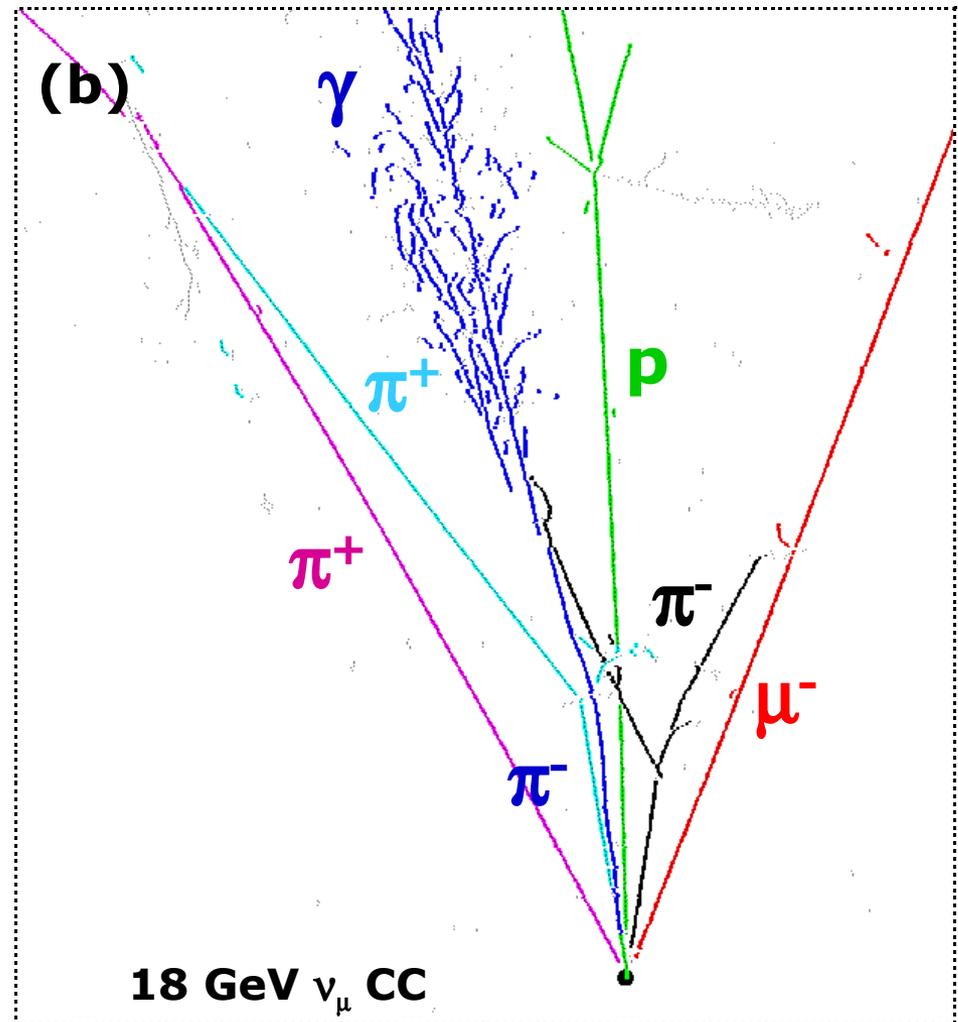
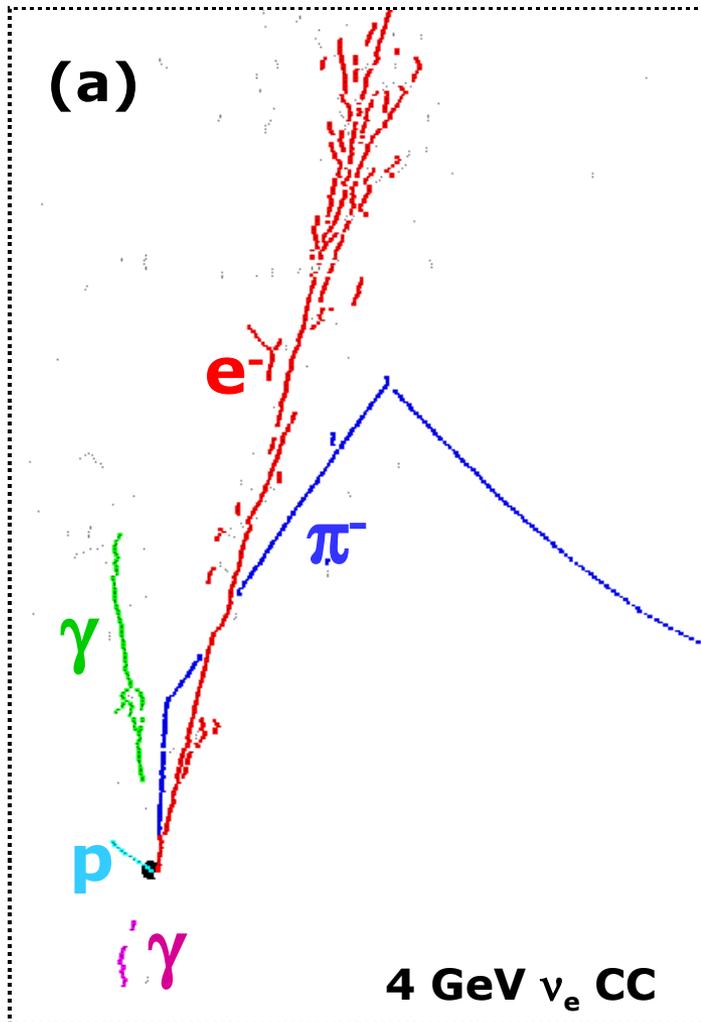


# 4. Final-State Particles

- Use extended clusters, along with vertex, to reconstruct final-state particles.
- **Method:**
  - Select **particle seeds** based on length and proximity to vertex.
  - Use **length-growing** algorithms to grow clusters longitudinally.
    - ◊ Builds track-like clusters.
  - Use **branch-growing** algorithms to grow showers transversely.
    - ◊ Builds shower-like clusters.
- **Results:**
  - The next slides show some illustrative examples.
    - ◊ Multi-particle final states.

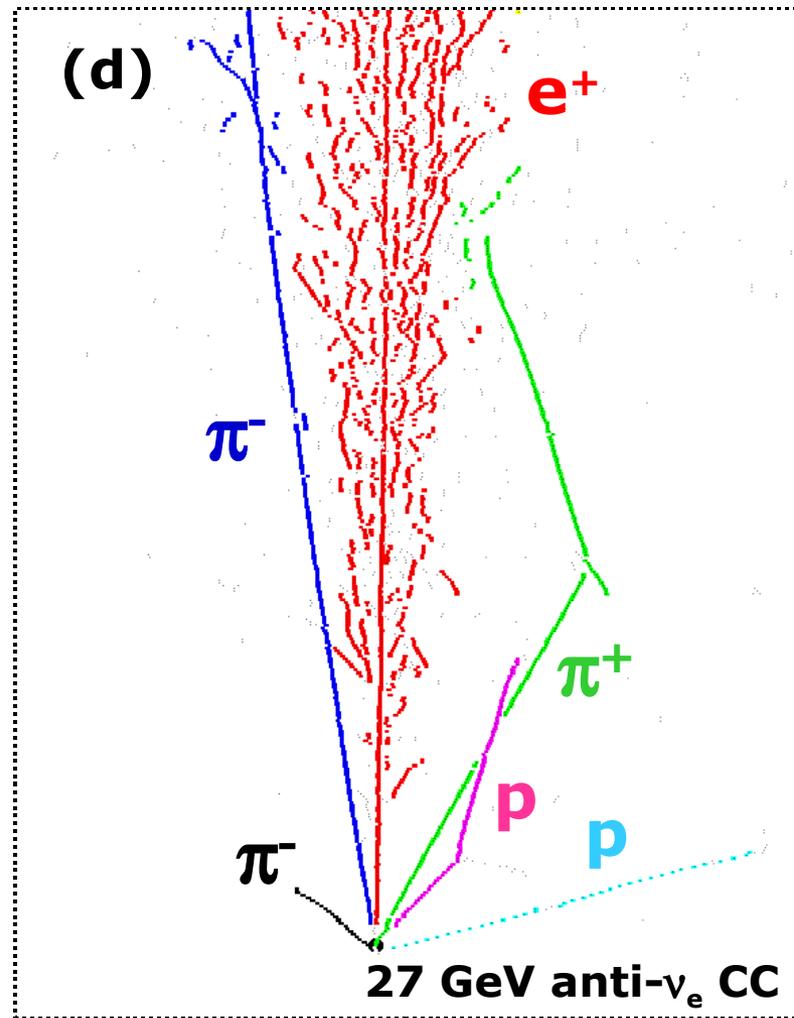
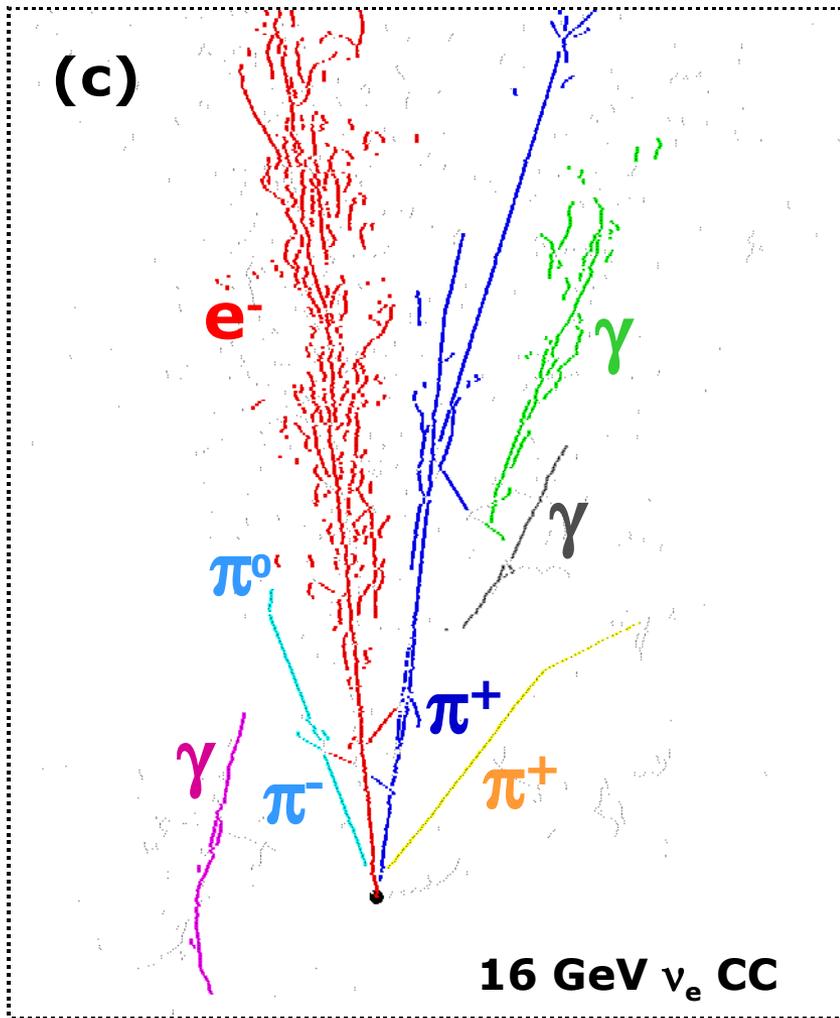


# 4. Final-State Particles



**Coloured lines show reconstructed particles (labelled using MC truth)**

# 4. Final-State Particles



**Coloured lines show reconstructed particles (labelled using MC truth)**

# Thoughts and Future Work

- **Our efforts on LAr event reconstruction look promising.**
  - Have integrated Pandora into LArSoft software framework, and developed a chain of 2D pattern recognition algorithms.
  - Able to handle quite complex multi-particle final states!
- **We're now starting to make transition from 2D to 3D.**
  - Crucial to compare views as early and as much as possible!
  - Pandora will be an ideal tool for developing 3D reconstruction.
- **Don't have finished product yet. Need more development. However, hopefully this talk indicates 'proof of principle'.**
- **Some near-term objectives:**
  - (1) Distribute the 'core' Pandora code base. Make sure this can be cleanly integrated into NuSoft.
  - (2) Tidy up the ART 'Producer' module (doesn't yet pass the clusters/tracks/showers back into the ART framework!)
  - (3) Tidy up, and distribute, the 2D reconstruction algorithms.