Anode Faces in Wire-Cell Toolkit

Brett Viren

February 3, 2025

Topics

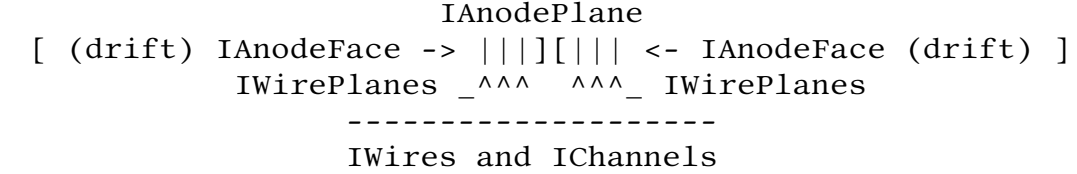
- Review/summarize detector representations in Wire-Cell Toolkit.
- Review/summarize how “faces” apply to WCT imaging and clustering.
- Describe my view of essential cluster “stitching” info to exploit.
- Face representation (lack of) in current PC tree.
- Proposal to update PC tree to represent and use “faces”.

The “anode plane” data hierarchy

WCT describes a TPC detector with a tree (forest) of objects with layers of type:

1. `IAnodePlane`
2. `IAnodeFace`
3. `IWirePlane`

These provide structure to overall set of `IWire` and `IChannel`.



IAnodePlane

A TPC sub-detector with channel-level sensitivity to shared drift volumes

- A DUNE APA is represented by one `IAnodePlane`
- Wrapped or jumpered wires conflate sensitivity over **two** drift volumes.
- The “plane” refers to the entire assembly, not a single wire plane.

Main data:

- “ident” number
- collection of `IWireFace`.

IAnodeFace

A TPC sub-sub-detector with wire-level sensitivity to a single drift volume

- Wrapped wires: a face on either side of the anode plane.
- Jumpered wires: faces are side-by-side across one anode plane.
- Roughly corresponds to a LArSoft “TPC”.

Main data:

- “ident” number
- sensitive volume bounding box
- “ray grid” coordinate system
- collection of `IWirePlane`

IWirePlane

A collection of co-planar of electrodes.

- One IAnodeFace mother and one IAnodePlane grandmother.

Main data:

- "ident" number
- WirePlaneId (apa/face/plane numbers as packed int)
- "Pimpos" wire-plane coordinate system
- collection of IWire and IChannel.
  - IWires in wire-in-plane order.
  - IChannels in wire-attachment-number order.

Bring `IAnodePlane` to data

Typically, configuration arranges for a **coincidence**:

- A data-flow graph node is configured with the **type/name** of an `IAnodePlane`
  - eg the `"anode"` config parameter.
- The flow graph is constructed so that the node receives data that is associated with the same `IAnodePlane`.

So far, components tend to use exactly 0 or 1 `IAnodePlane`.

- In general, a component may access multiple `IAnodePlane`'s.
- Or, may be limited to use a single `IAnodeFace` or `IWirePlane`.

IAnodeFace and WCT imaging

An IBlob is naturally a **per-face** object because collection wire planes (at least) are per-face.

- IAnodeFace::pointer IBlob::face() captures/represents this association.

IAnodeFace and WCT clustering

Initial clusters are points sampled from blobs so are **implicitly** per-face.

- MicroBooNE has only 1 face so multi-face is not an issue.

Next and novel clustering stage is to **stitch clusters**.

- 3 stitching cases: across **anode** faces, **cathode** faces and drift volume **sides**.

How do we handle the “face issue”?

Essential stitching algorithm exploits (imo)

Anode stitching - time/drift dimension

Anode-crossing track puts ADC in **common ticks** on both faces.

Cathode stitching - time/drift dimension

Cathode-crossing track puts ADC in **common ticks** after **adjusting** for:

- drift velocity/duration over known drift distance.

Side stitching - time/drift dimension

Side-crossing track puts ADC in **common ticks** after **adjusting** for:

- drift velocity/duration over unknown drift distance (need Q-L match)

All cases - transverse dimensions

Must model the track the location/direction.

- Side stitching may require space-charge effect corrections.

Faces and point-cloud tree

PC tree layers represent: **grouping**, **cluster**, **blob**.

- The blob nodes have a `Dataset` representing blob sample points.
- All nodes may carry additional `Datasets`.

Currently, no face information in the PC tree (?)

- No "face" ID recorded on "blob nodes" (?)
- No "blob node" $\leftrightarrow$ `IBlob` connection (?) so no `IBlob::face()` call.

Proposal to represent and use “faces” in next-stage clustering

- Form a **drift volume ID** (DVID) from the pair of numbers: `(anode ident,face)`.
  - Save DVID to each blob node as a `Dataset` (PC).
  - An initial (per-face) cluster has DVID that of any of its blobs.
- Form a **drift volume neighbor map**.
  - Dense 3D array of DVID values, elements correspond to detector layout.
  - Store in PC tree as a `Dataset` on the **grouping** node.
- Family of DVID lookup functions.
  - DVID pairs by “anode”, “cathode” and “side” neighboring.
  - Must be parameterized by “wrapped” or “jumpered” assumption.
  - And by layout (eg, DUNE C-A-C-A-C vs PD A-C-A vs ICARUS C-A-C|C-A-C).

Stitching of “anode”, “cathode” or “side” type starts by finding sets of “initial” (per-anode) clusters of the corresponding neighboring.

FIN

Backups