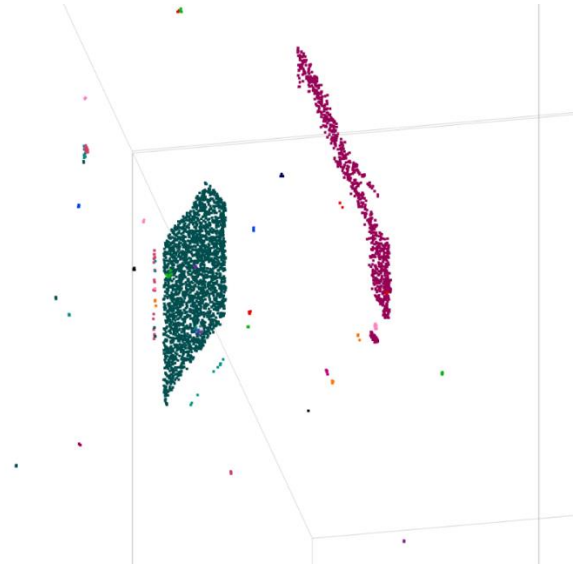
Label cells to build training datasets



- For some Blobs in imaging step, the blob can be big.
  - Intrinsic ambiguity
  - Provide more information like topology to ML to reduce the ambiguity

- Blobs is defined by strips overlap, contains several cells.
- For each cell, label the cell is real or not based on simulation.
  - A Cell here may not be strictly a single cell, limitation of the computation, divide each blobs according to requirement: Coarse or fine

- Use labeled cells from Blobs to build training datasets

- Using protodune-vd simulation.
- Compare vd-simulation imaging and true energy deposit from simulation.

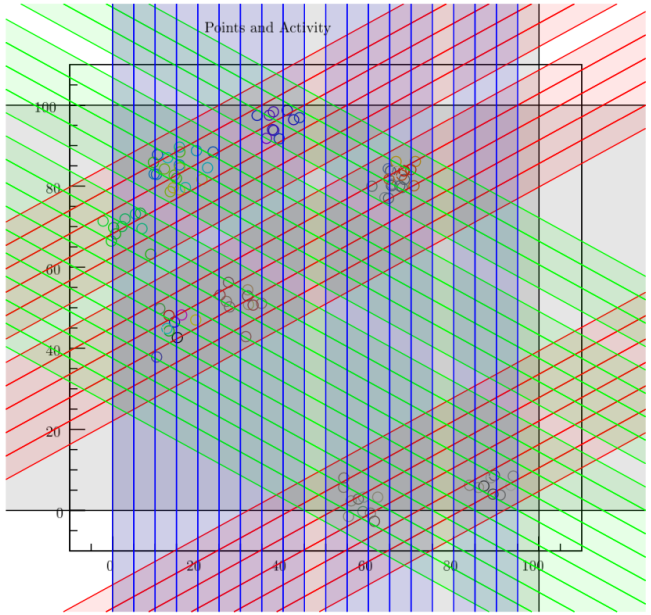
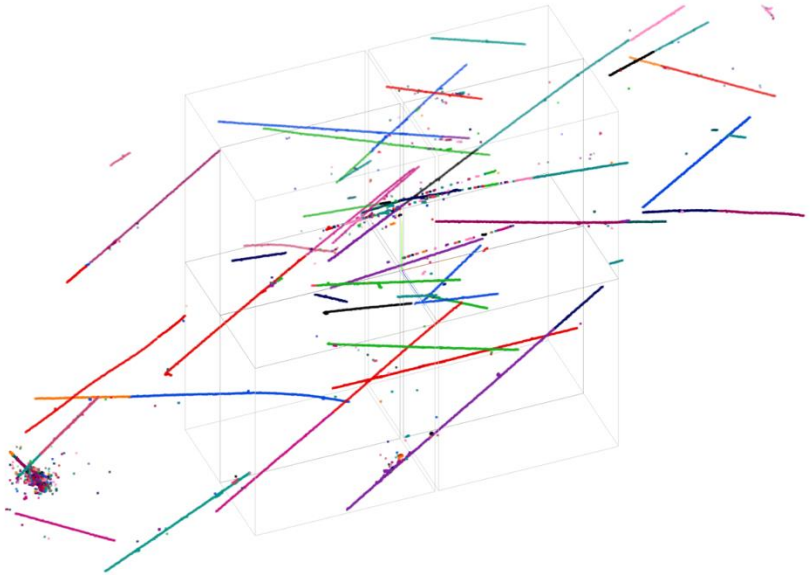


Figure 2: Depiction of the location of generated "electrons" in a toy simulation and the resulting activity arrays which colored based on the ray grid layer the span.

Imaging vs true energy deposit

- vd-simulation imaging:

https://www.phy.bnl.gov/twister/bee/set/ac7fd044-31f0-4c35-8cc6-645f615c29ba/event/0/



- simulation true energy deposit:
  - sim::SimEnergyDeposit
  - cellTree_module.cc

https://www.phy.bnl.gov/twister/bee/set/ac7a6634-e6c4-4325-938c-21a3bafd00ff/event/0/

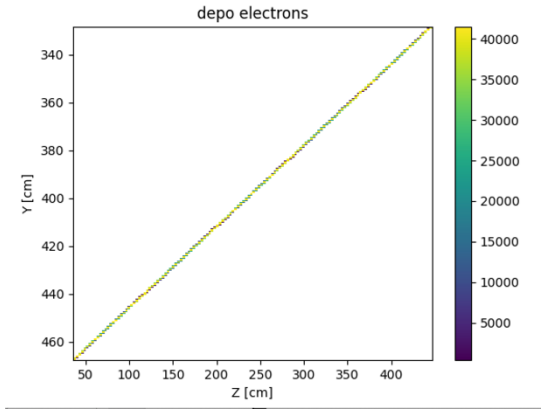
Next thoughts:

- Make sure these two x-axis is consistent
- In each slice, cut Blob to cells
  - Merge a cell to its neighbor if it is too small
- Label if the cell is true or not

Thanks Xin, Wenqiang, Jay, Chao, Hokyeong for all the discussion, information and suggestions

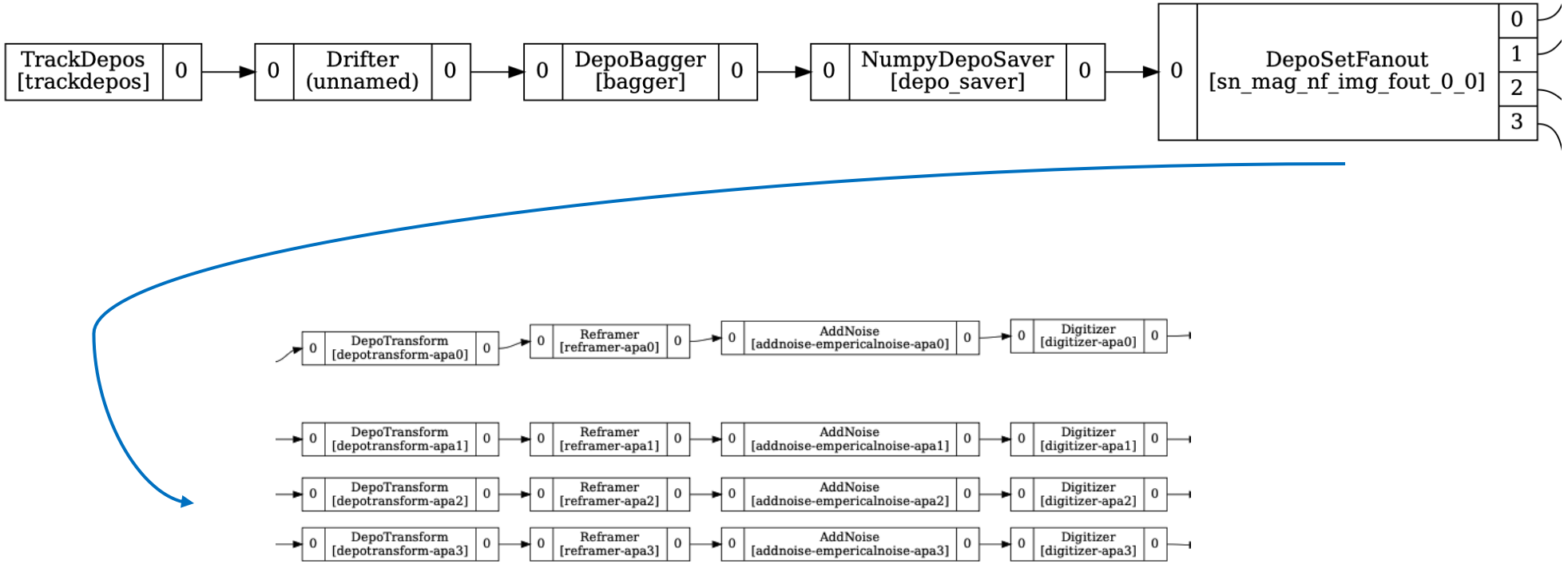
Start a simple simulation

- Based on pdhd wct standalone simulation:



will be used in BlobDepoFill

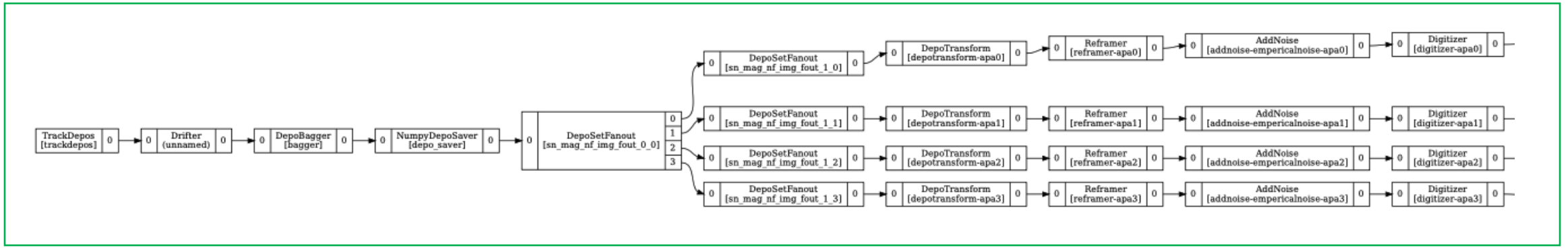
Track simulation, Charge Deposition

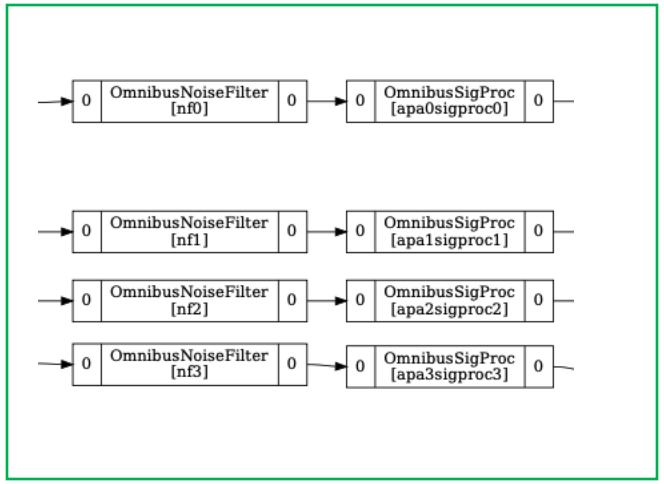


Simulate detected signal and bkg

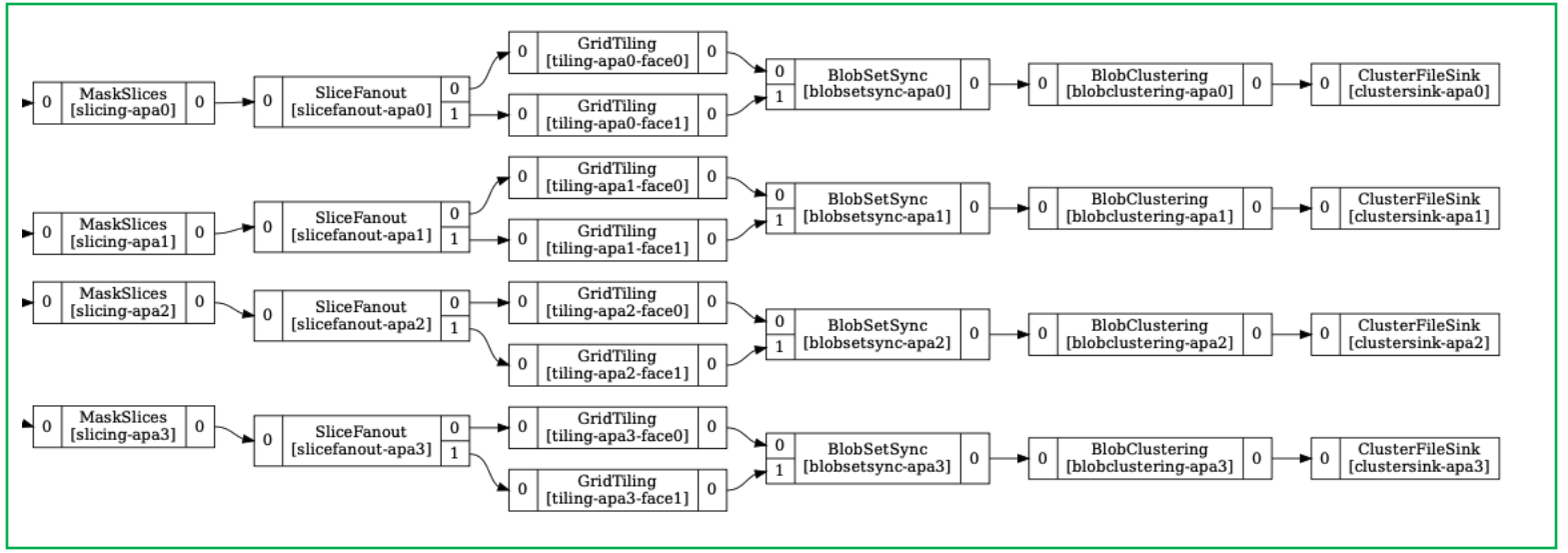
Start a simple simulation

Simulation:





Signal processing



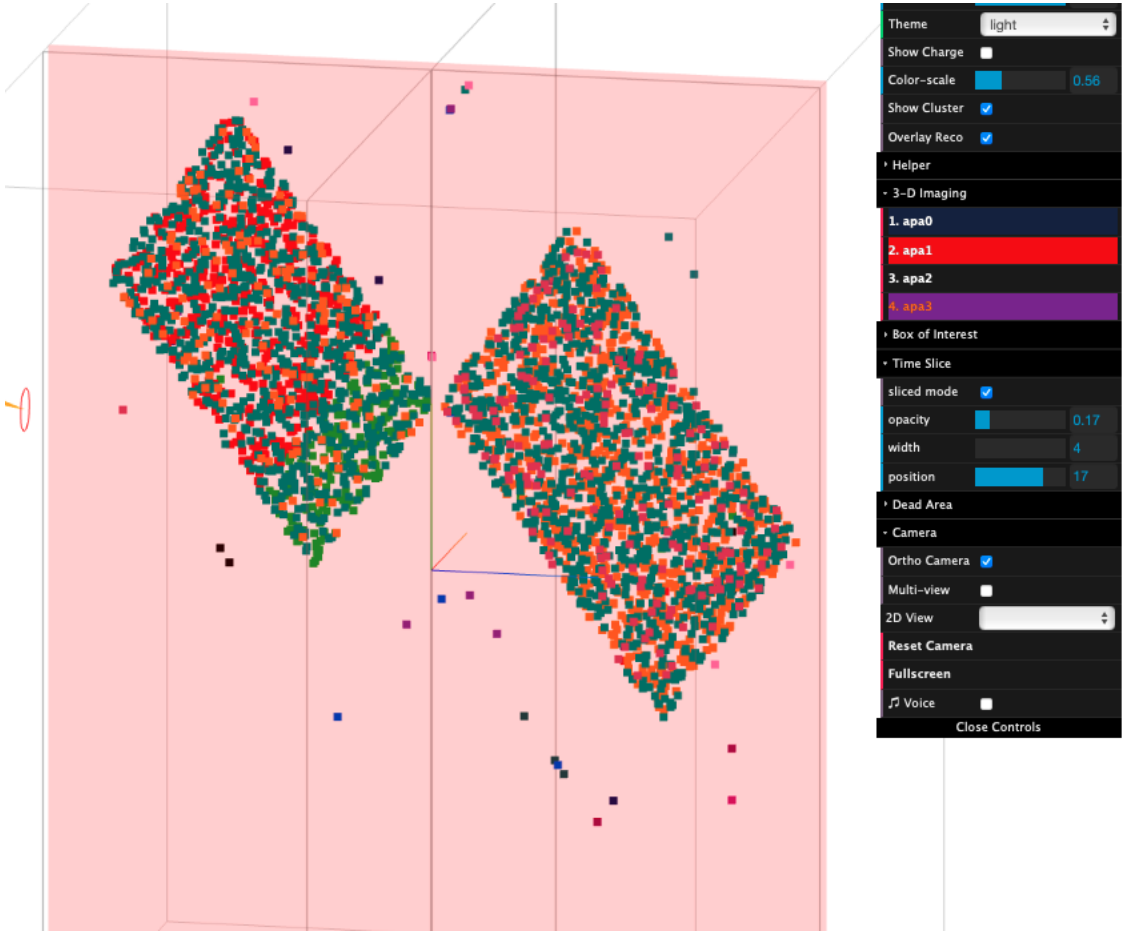
Imaging:
Only to Form a Blob, no grouping, no charge solving or deghosting

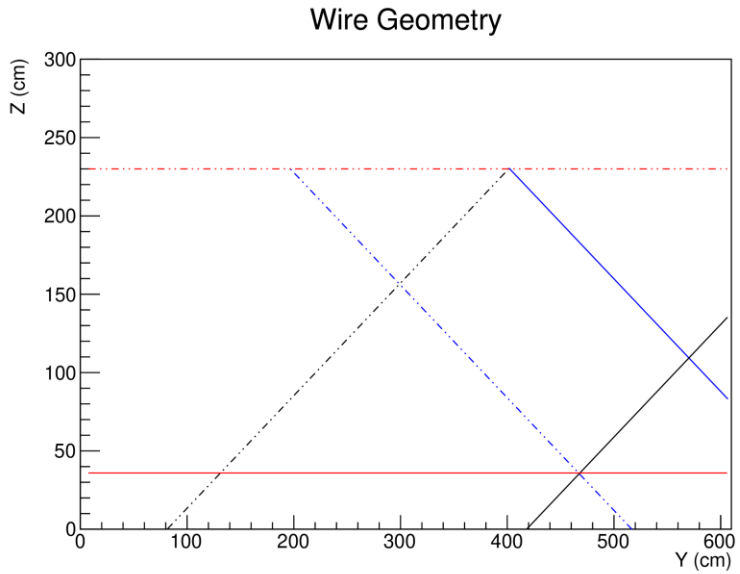
Blob check

https://www.phy.bnl.gov/twister/bee/set/f5a6cb0f-96ab-4a80-8e90-3d55445ffc40/event/0/

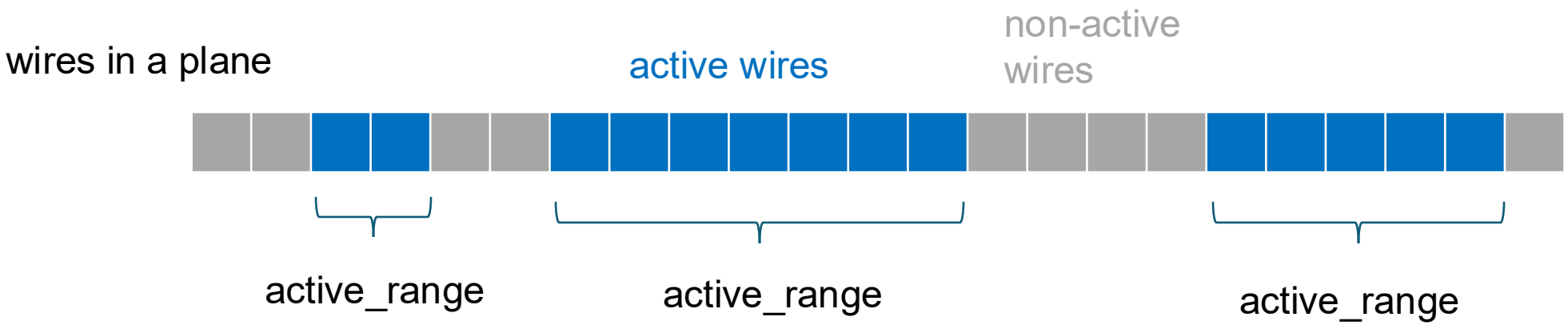
```
// };
local track= {
head: wc.point(10, 467.694, 35.7493, wc.cm),
tail: wc.point(10, 328.196, 446.613, wc.cm),
};
```

- Simulate an isochronous track
- remove *geom_clustering*
- dump blobs then draw separately





Blob separation



- active_range from 3 planes overlap each other will form a blob
- Blob size depend on size of the active_range
- Simple cut the active_range at the beginning, we can form the blobs at the size we want.

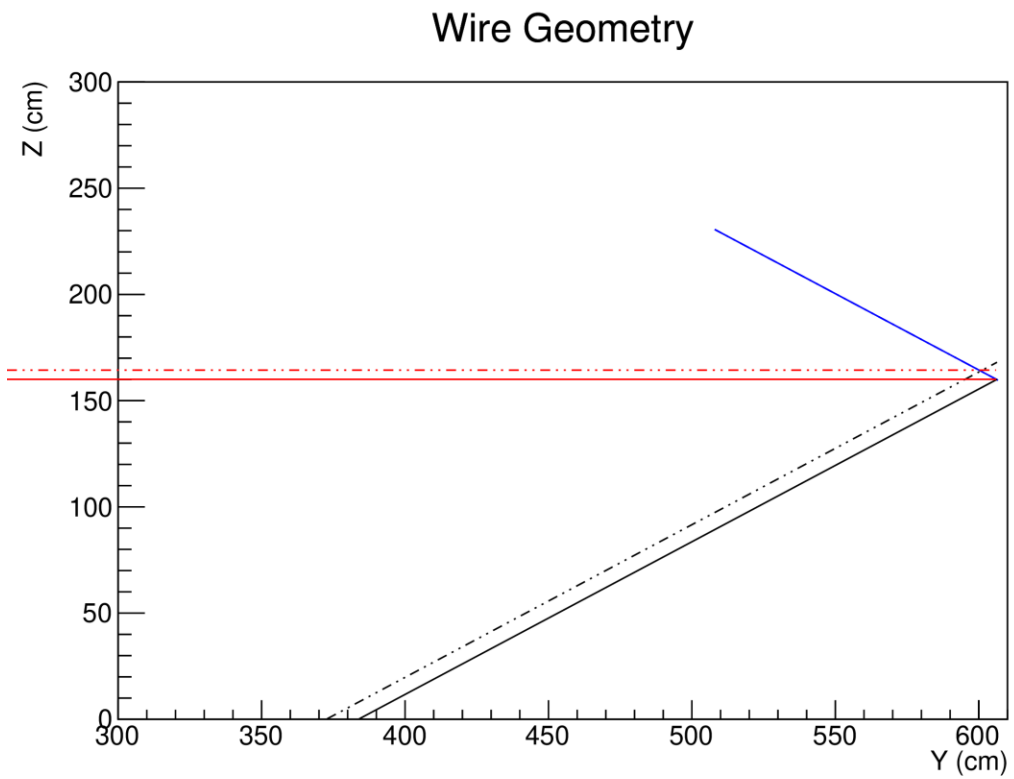
- A simple test:
- When active_range is generated, if range_length>15, then divide the active_range into [range_length/15] pieces

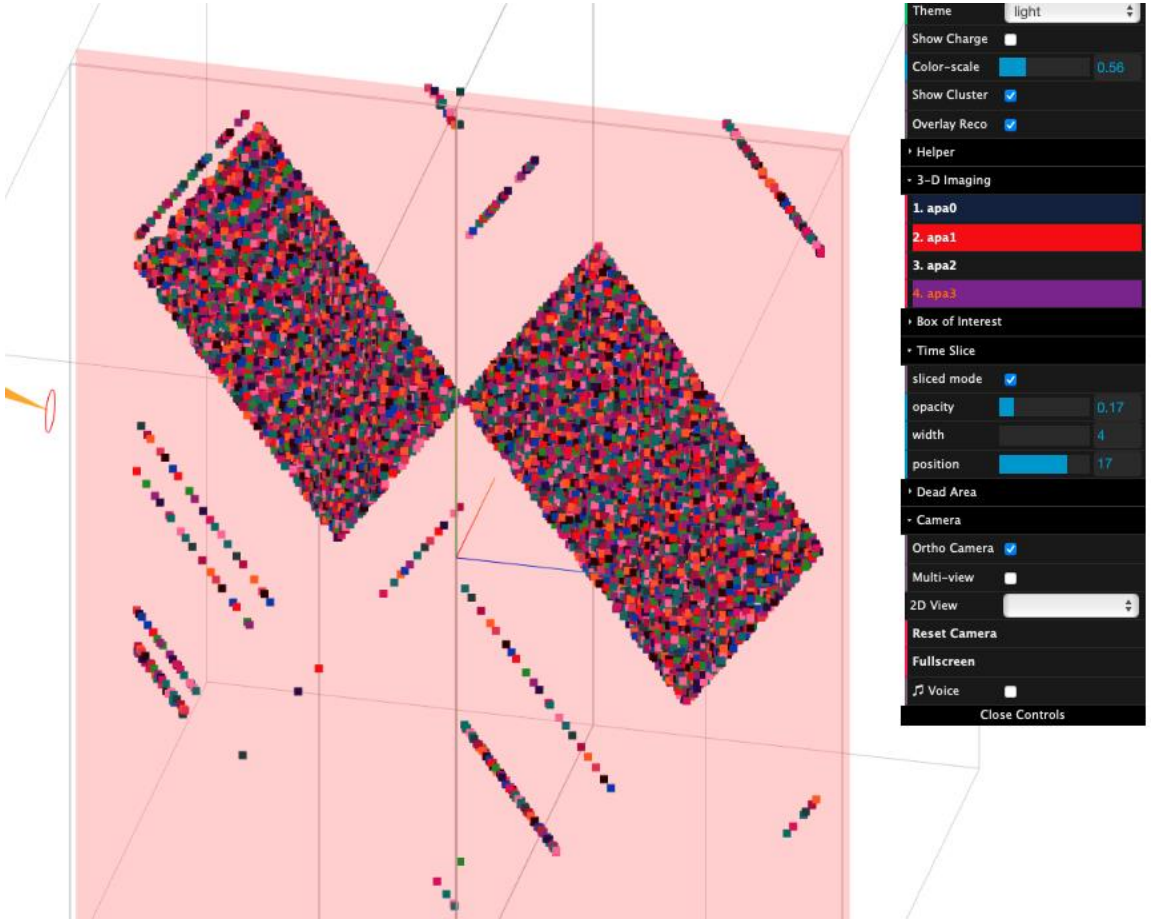
- Cons: if we need to divide to medium blob then small blobs, then need an additional way to build hierarchy.
  - Maybe can rely on *geom_clustering*

- I can further figure out how to actually separate blobs if needed.

```
Activity::ranges_t Activity::active_ranges() const
{
    ranges_t ret;
    range_t current{end(), end()};

        // Helper lambda to split and add ranges
    auto add_range = [&](const range_t& range) {
        int range_length = range.second - range.first;
        if (range_length > 15) {
            // Calculate how many sub-ranges we need
            int num_subranges = (range_length + 14) / 15; // Ceiling division
            int subrange_size = range_length / num_subranges;
            int remainder = range_length % num_subranges;

            auto start_it = range.first;
            for (int i = 0; i < num_subranges; ++i) {
                auto end_it = start_it + subrange_size;
                // Distribute remainder across first few subranges
                if (i < remainder) {
                    ++end_it;
                }

                ret.push_back({start_it, end_it});
                start_it = end_it;
            }
        } else {
            // Range is <= 15, add as-is
            ret.push_back(range);
        }
    };

    for (auto it = begin(); it != end(); ++it) {
        // entering active range
        if (current.first == end() and *it > m_threshold) {
            current.first = it;
            continue;
        }
        // exiting active range
        if (current.first != end() and *it <= 0.0) {
            current.second = it;
            // ret.push_back(current);
            add_range(current);
            current.first = end();
        }
    }
```

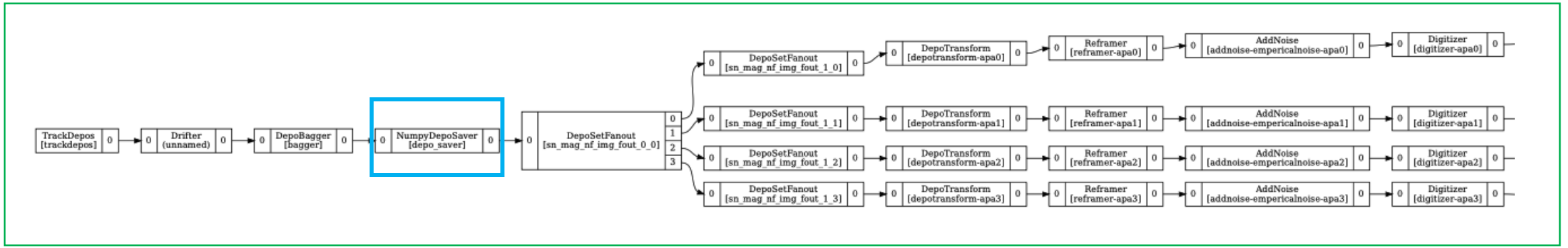
A demo

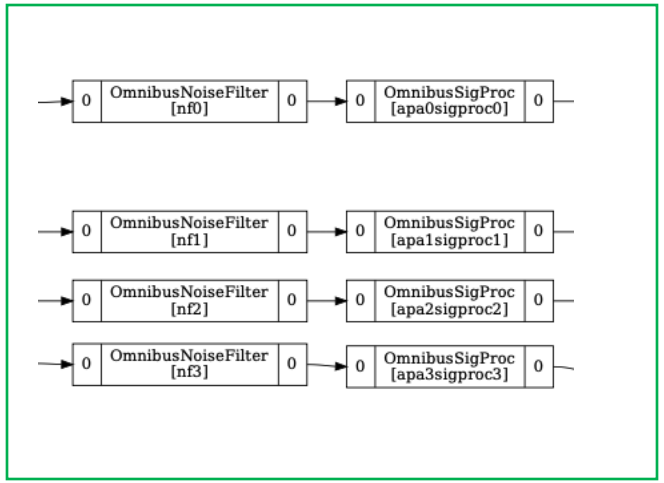

Wire Geometry
Z (cm)
300
250
200
150
100
50
0
300
350
400
450
500
550
600
Y (cm)


Theme
light
Show Charge
Color-scale
0.56
Show Cluster
Overlay Reco
Helper
3-D Imaging
1. apa0
2. apa1
3. apa2
4. apa3
Box of Interest
Time Slice
sliced mode
opacity
0.17
width
4
position
17
Dead Area
Camera
Ortho Camera
Multi-view
2D View
Reset Camera
Fullscreen
Voice
Close Controls

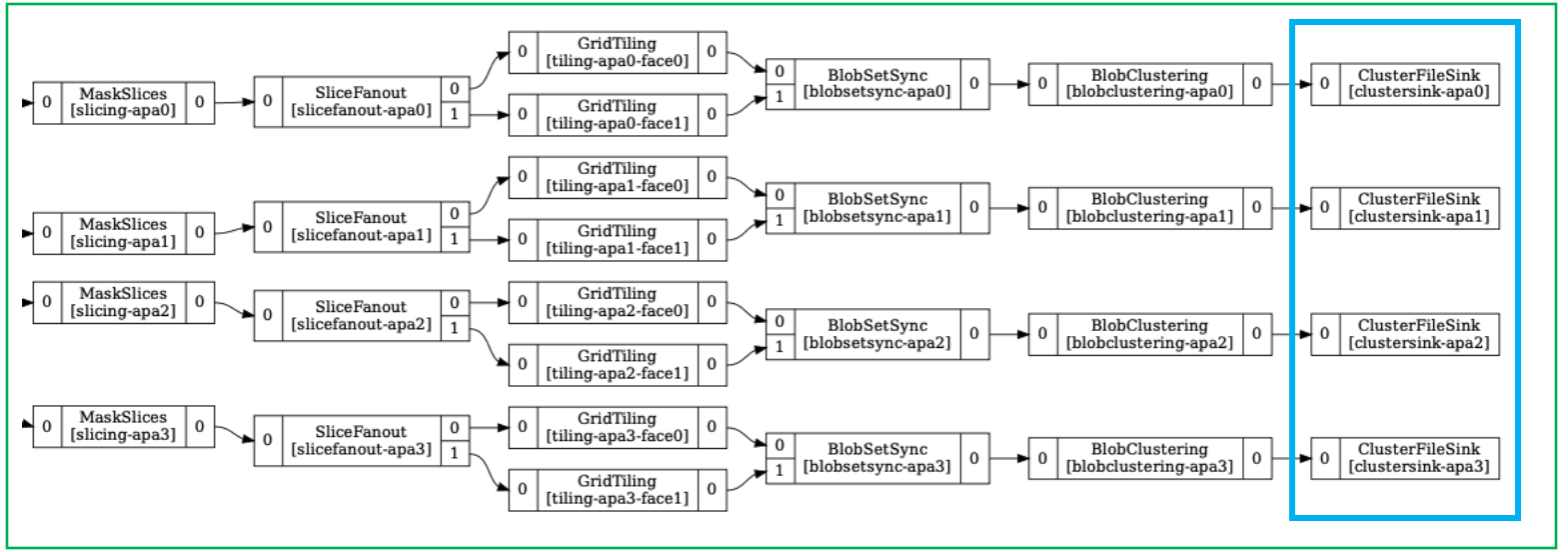
Truth information: BlobDepoFill

Simulation:



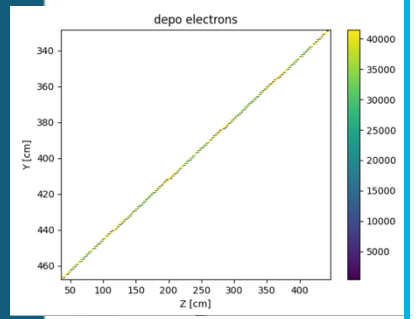


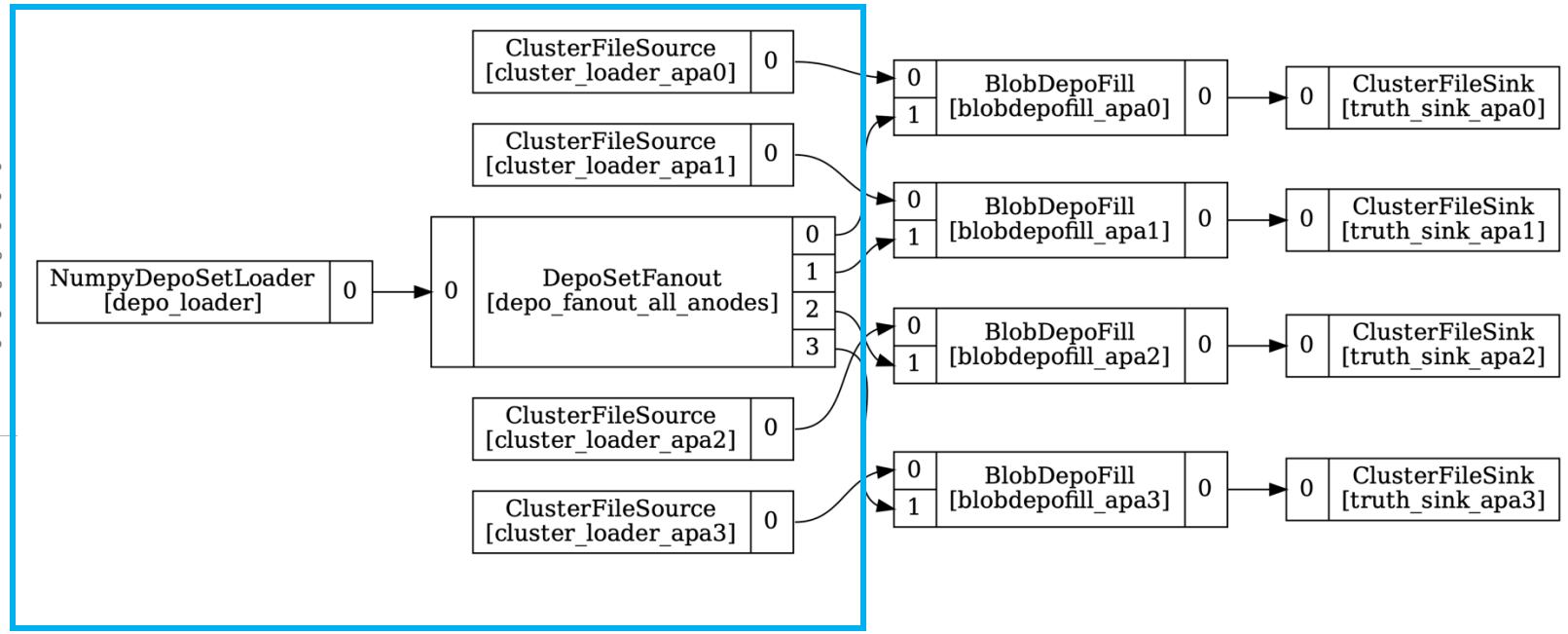
Signal processing



Imaging:
Only to Form a Blob, no grouping, no charge solving or deghosting

Truth information: BlobDepoFill





- integrates of depos over blobs to get the true charge of a blob
- Still working on it…debugging…

What is the true charge of a blob?

The integral of the simulated, drifted ionization electron distribution over the pre-determined volume of one blob.

https://www.phy.bnl.gov/~bviren/tmp/wctimg/blob-depo-fill.pdf