SLAC NATIONAL ACCELERATOR LABORATORY

Inverse-Imaging with larnd-sim

D. Douglas

Neutrino ML meeting

U.S. DEPARTMENT OF ENERGY | Stanford University

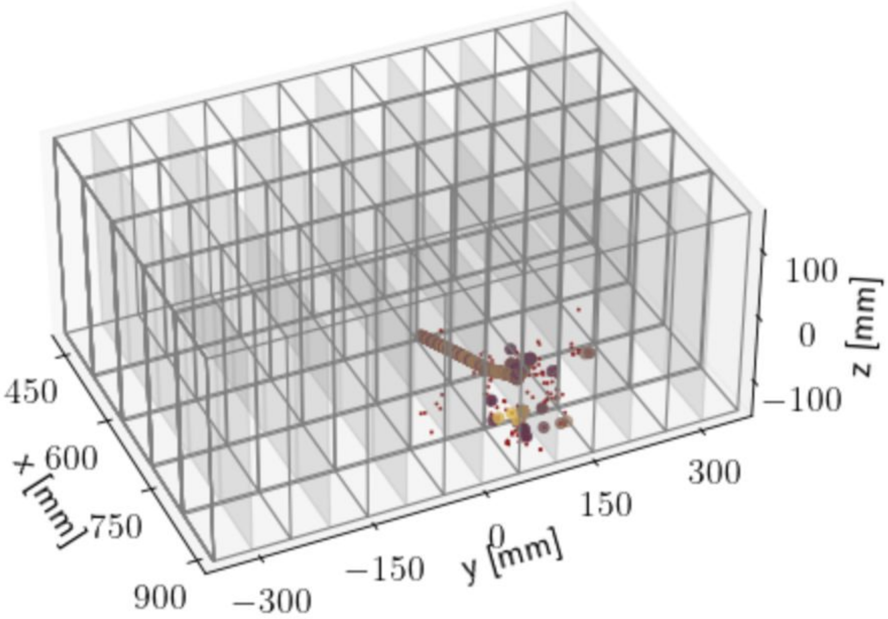
larnd-sim

Larnd-sim models the mapping of:

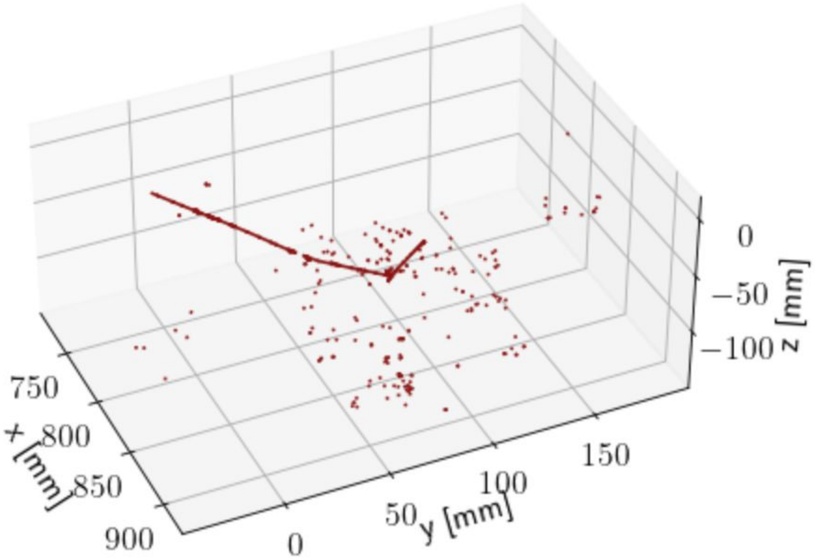
- Ionizing energy deposits by energetic particles to the

To

- Detector ASIC packets corresponding to charge induction on 2D segmented anode planes



edep-sim



[Edep-sim](#) is a wrapper around [Geant4](#). It models the energy deposition of charged particles in matter based on some input momentum profiles and geometry descriptions

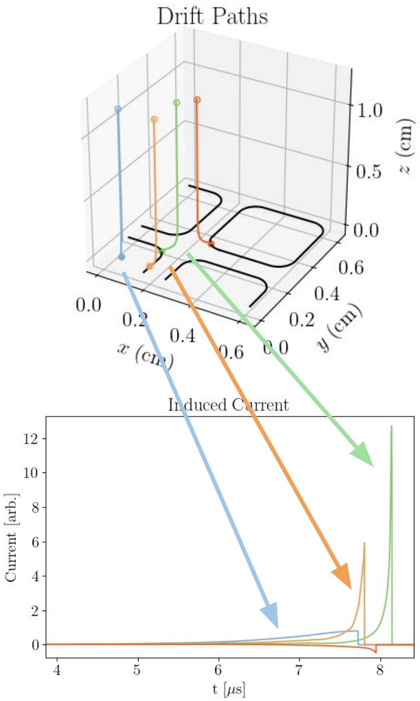
Hit Formation

Charge clouds drift to the anode plane

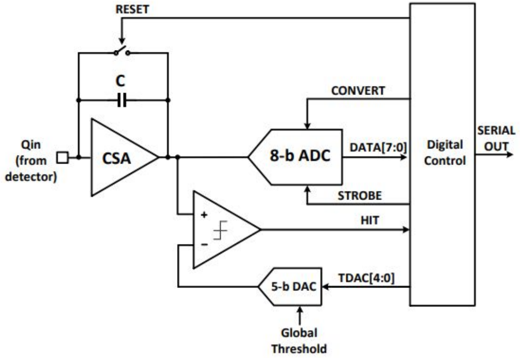
Voltage is induced on the surfaces of electrodes

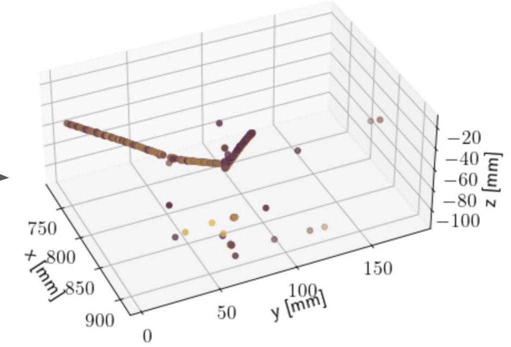
Pixel electronics register a “hit” and digitize charge after a threshold is reached + 8 clock cycles

Measurement is (pixel address, timestamp, ADC value)

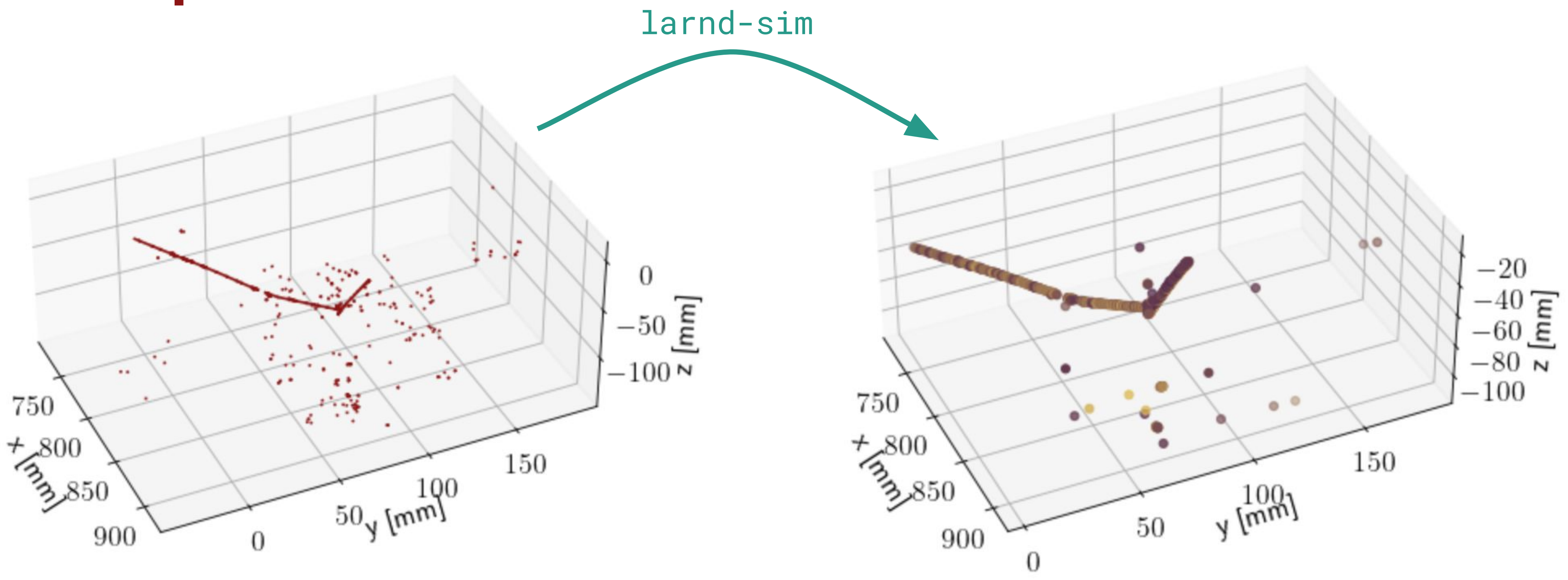


Σ

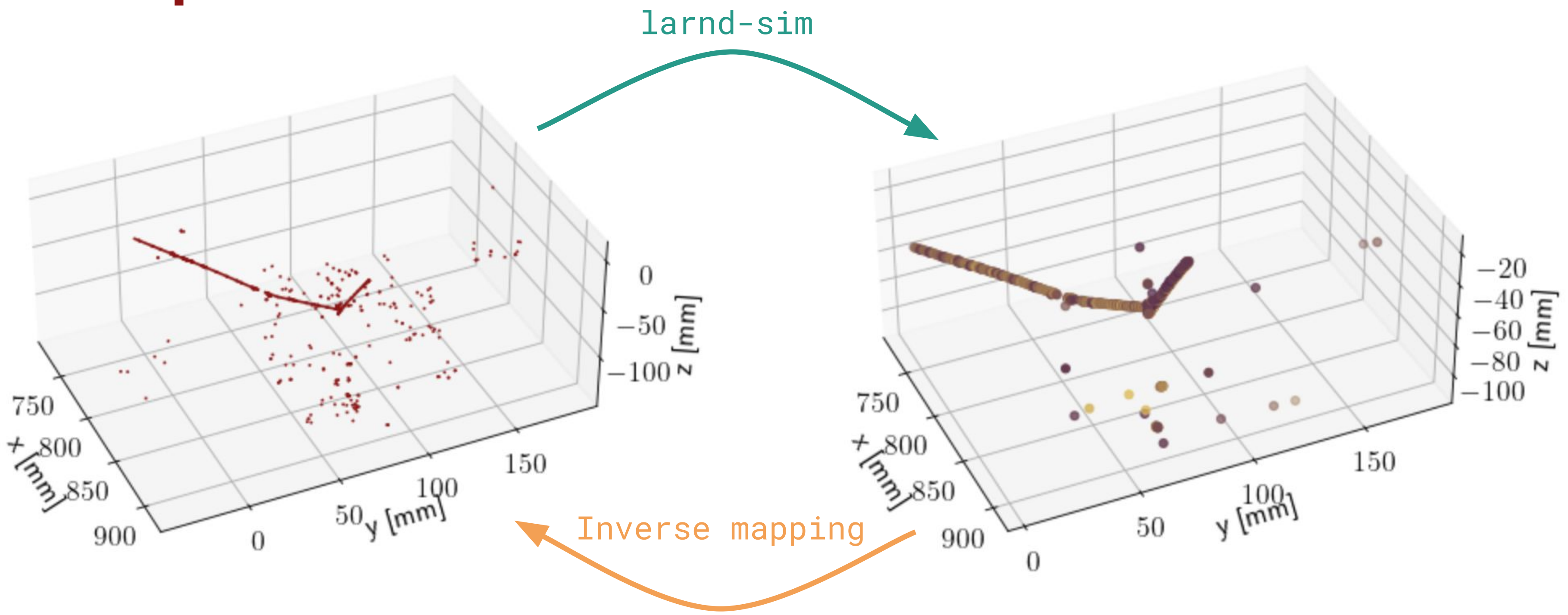




The Spaces


larnd-sim
0
−50
−100
z [mm]
750
800
850
900
x [mm]
0
50
100
150
y [mm]
−20
−40
−60
−80
−100
z [mm]
750
800
850
900
x [mm]
0
50
100
150
y [mm]

The Spaces

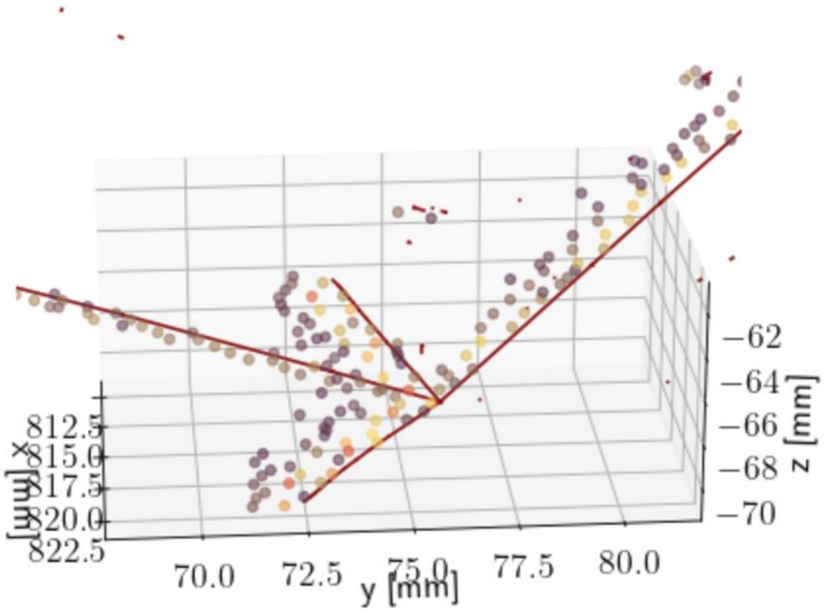

larnd-sim
Inverse mapping
0
−50
−100
z [mm]
750
800
850
900
x [mm]
0
50
100
150
y [mm]
−20
−40
−60
−80
−100
z [mm]
750
800
850
900
x [mm]
0
50
100
150
y [mm]

Correspondence and Lossiness

Z positioning is not perfect, relies on accurate trigger and drift model

Trigger structure is not perfect, can fail for low energy events

Drift model in larnd-sim assumes perfect uniformity. In the real detector, there are observed imperfections in the performance of field shaping devices

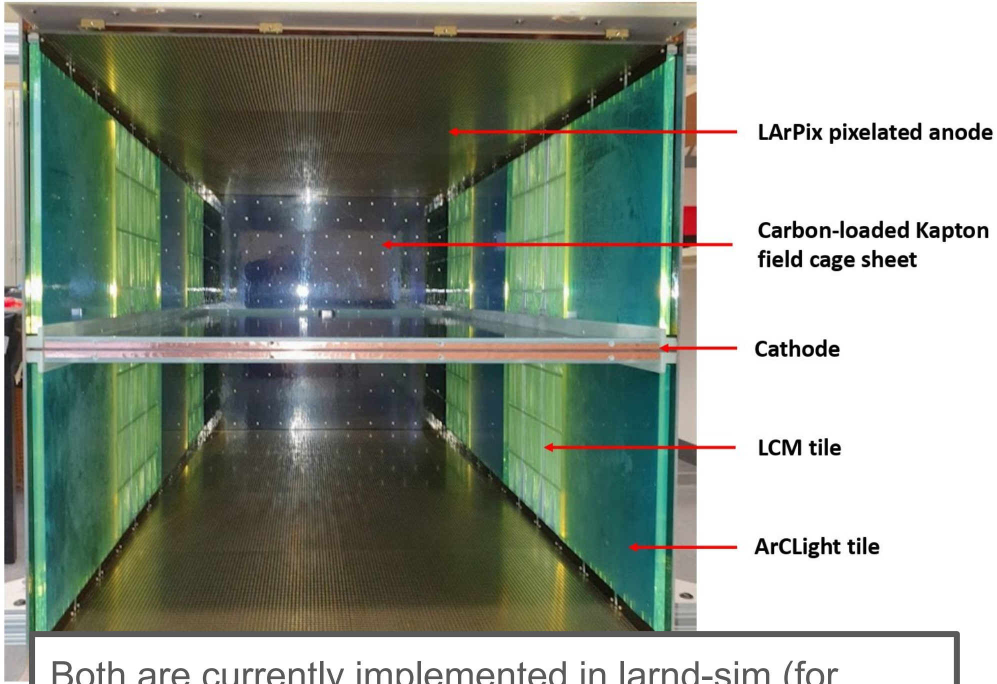


Triggering

There are two main types of triggers in this detector:

- "Self trigger": the anode plane itself registers a hit
- "External trigger": external systems (light detectors, external muon taggers) register a signal

These trigger packets mark a t_0 against which drift time is measured



Both are currently implemented in larnd-sim (for module0!) (not in ynashed's fork!)

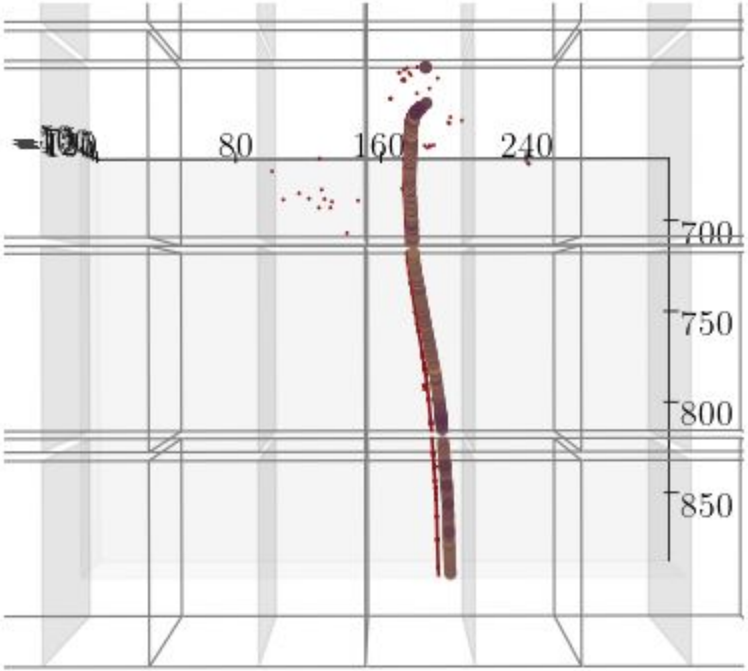
Detection Effects

ADC hits are a sub-sample of the actual charge distribution due to threshold, absptn.

Timing is imperfect: hit time ≠ charge arrival time (induction happens prior to charge arrival, worse for larger charge clouds)

Z-placement is imperfect because

- Hit timing
- Trigger timing
- (in data) drift non-uniformity not modeled



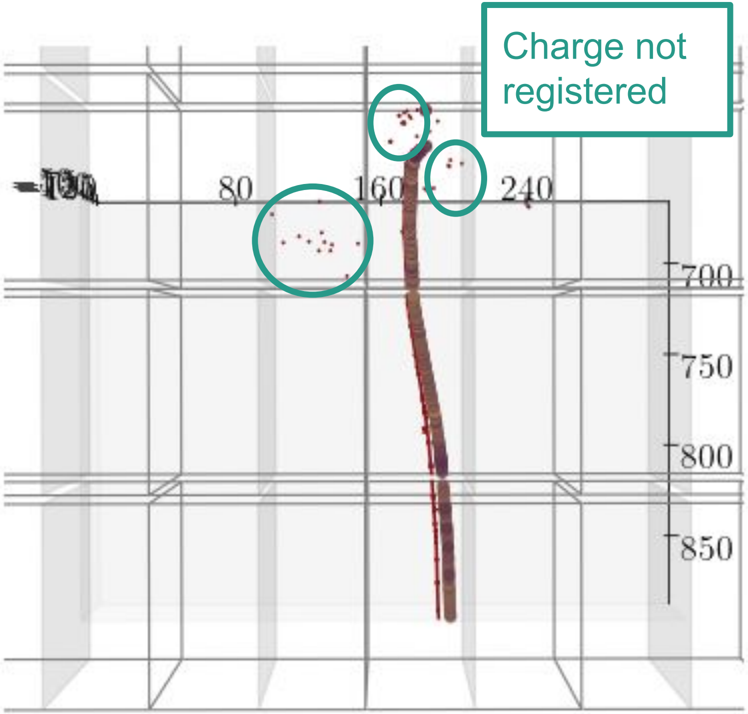
Detection Effects

ADC hits are a sub-sample of the actual charge distribution due to threshold, absptn.

Timing is imperfect: hit time ≠ charge arrival time (induction happens prior to charge arrival, worse for larger charge clouds)

Z-placement is imperfect because

- Hit timing
- Trigger timing
- (in data) drift non-uniformity not modeled



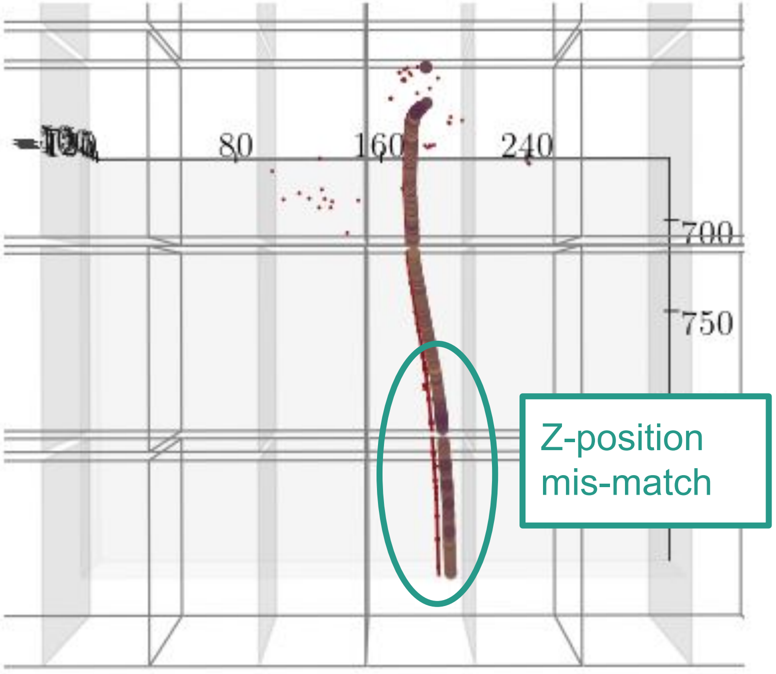
Detection Effects

ADC hits are a sub-sample of the actual charge distribution due to threshold, absptn.

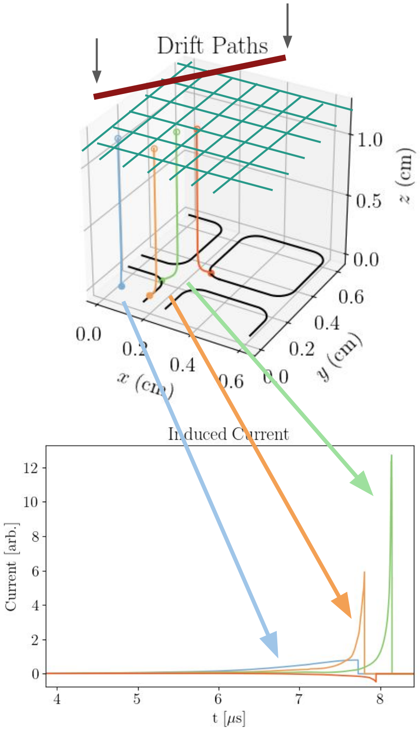
Timing is imperfect: hit time ≠ charge arrival time (induction happens prior to charge arrival, worse for larger charge clouds)

Z-placement is imperfect because

- Hit timing
- Trigger timing
- (in data) drift non-uniformity not modeled



Voxelized dE/dx



As it is currently, the edep-sim toolchain produces segments of energy deposition

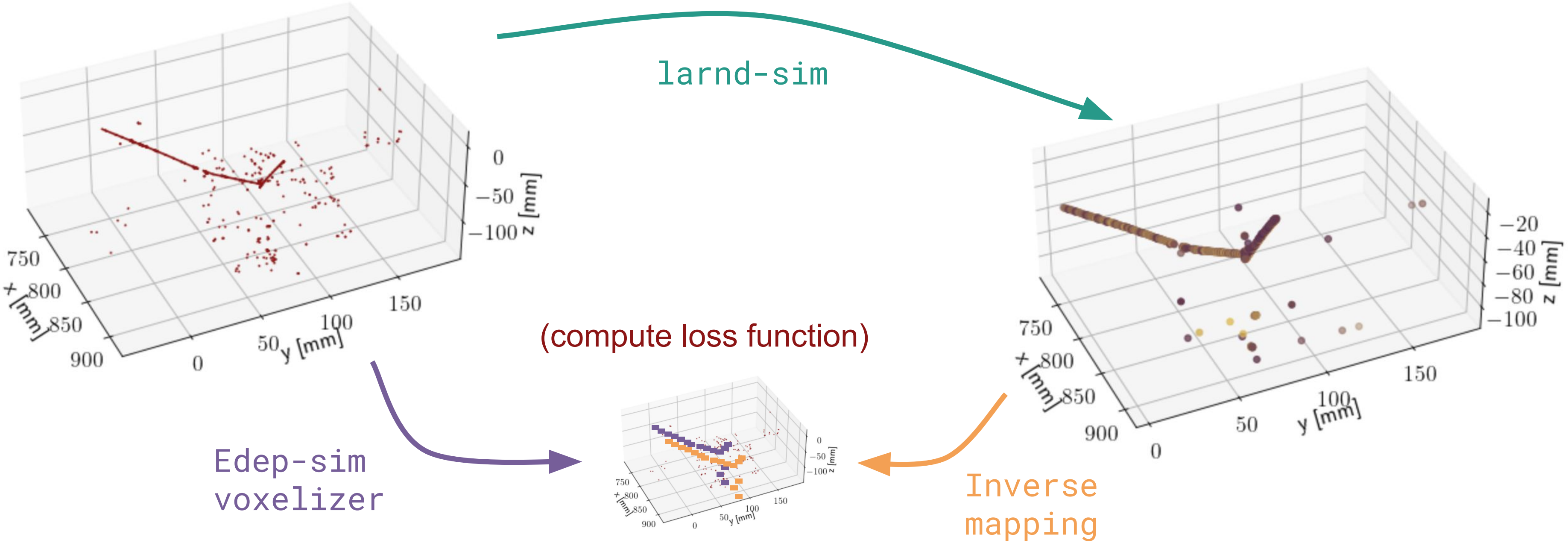
Larnd-sim transports these segments (applying attenuation and diffusion)

Pixel response is implemented on a sub-grid of the pixel pitch, so voxelization of edep inputs is not a huge task

This will allow our network architecture to map voxel-to-voxel (with edep voxels being 10x finer than larnd-sim voxels). This is in the preliminary stages!

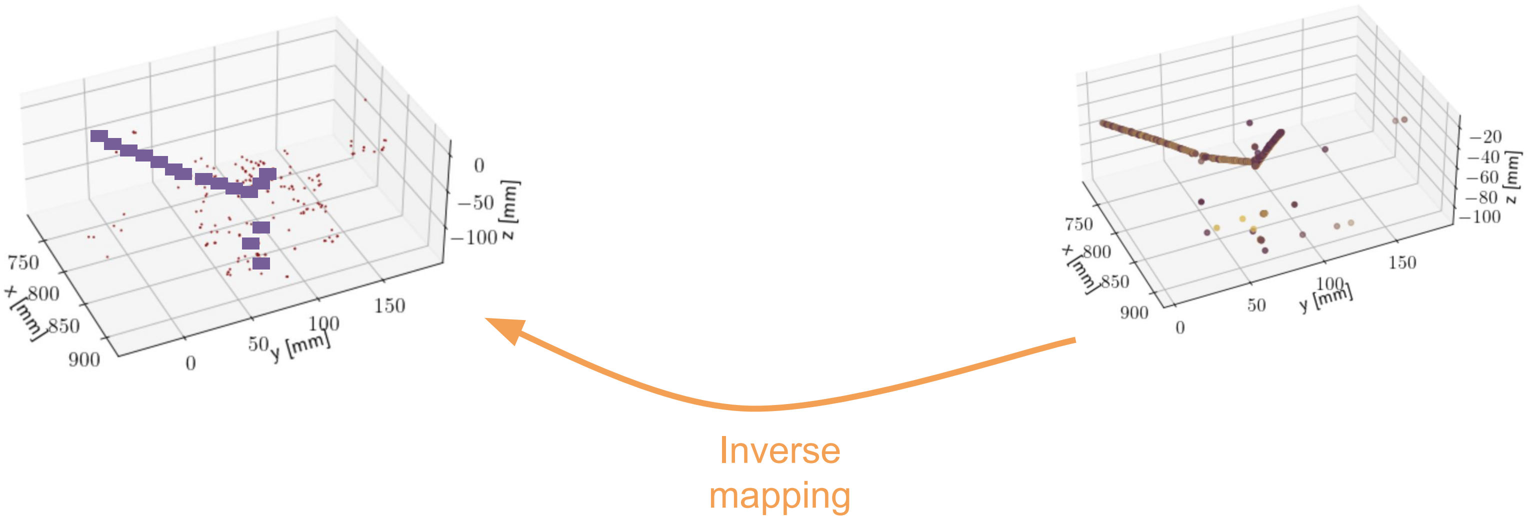
Voxelized dE/dx

For now, we can apply a voxelization in parallel



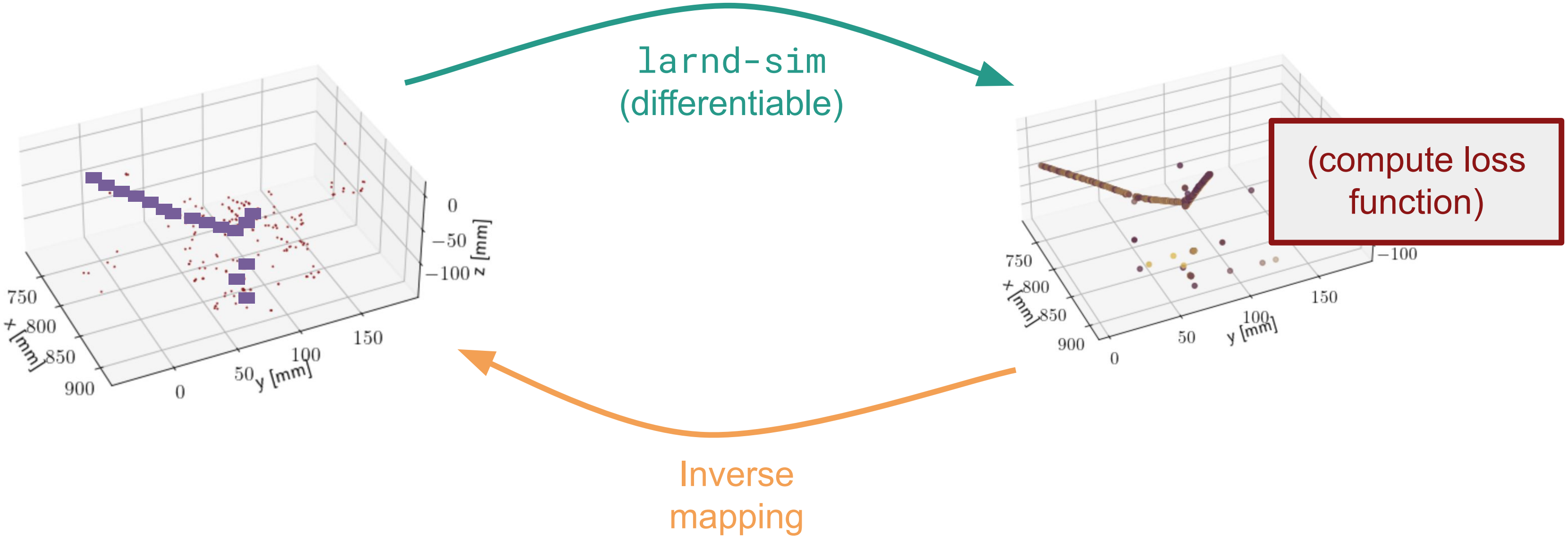
Using the Differentiable Simulation (FUTURE!)

A differentiable implementation of larnd-sim would allow back-propagation of a loss through the detector simulation, allow us to train on data directly



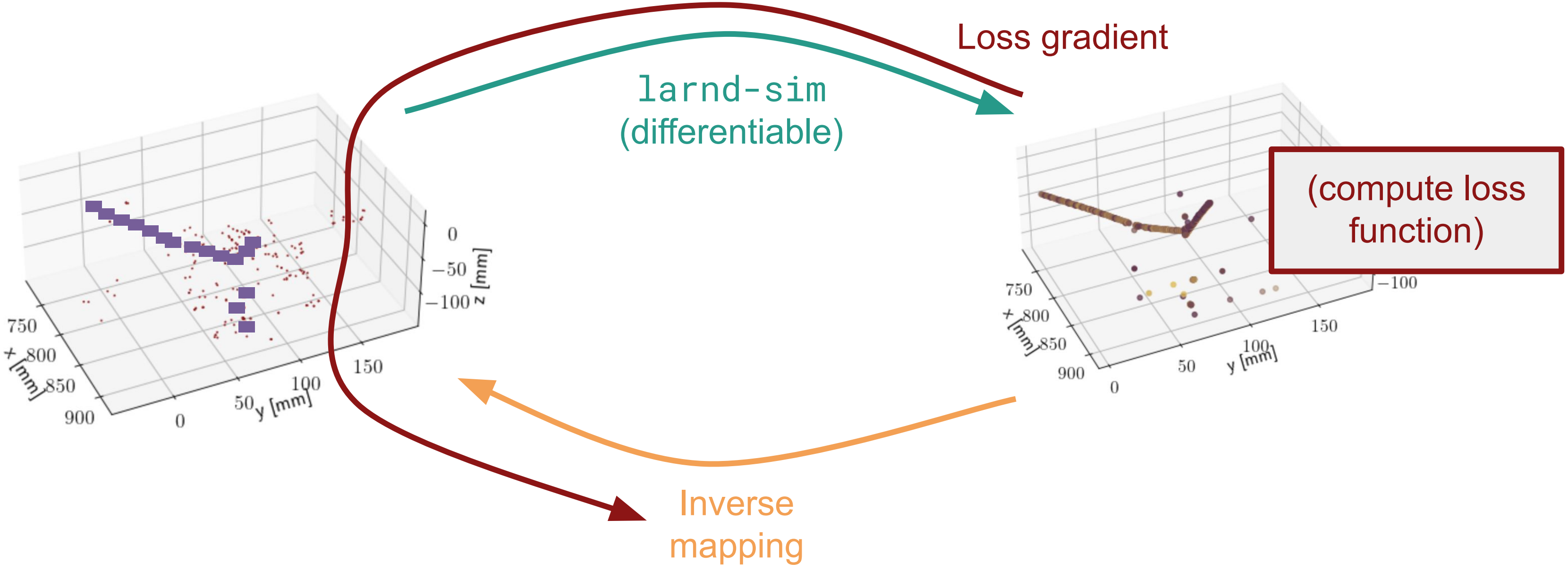
Using the Differentiable Simulation (FUTURE!)

A differentiable implementation of larnd-sim would allow back-propagation of a loss through the detector simulation, allow us to train on data directly



Using the Differentiable Simulation (FUTURE!)

A differentiable implementation of larnd-sim would allow back-propagation of a loss through the detector simulation, allow us to train on data directly

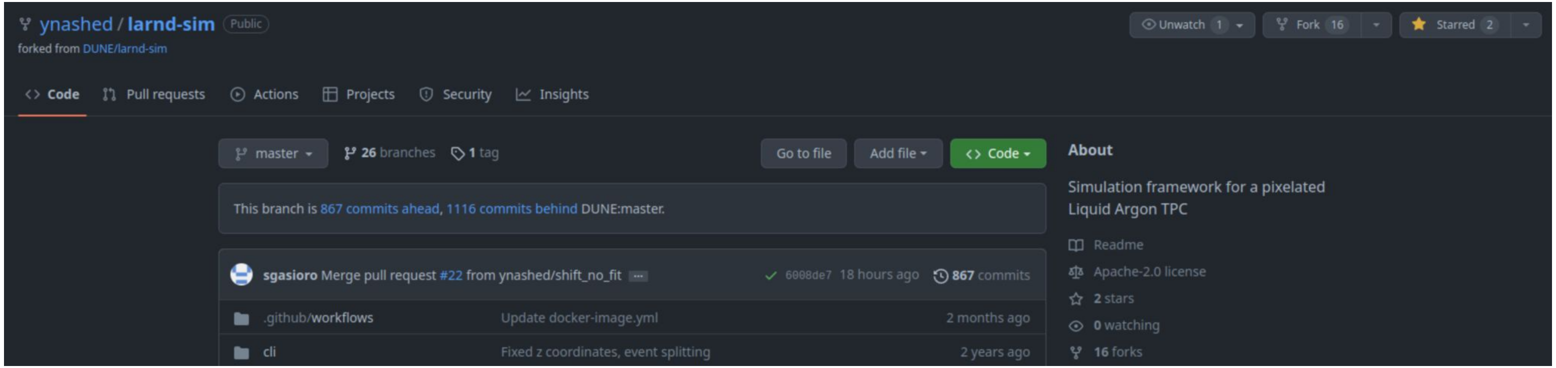


Using the Differentiable Simulation

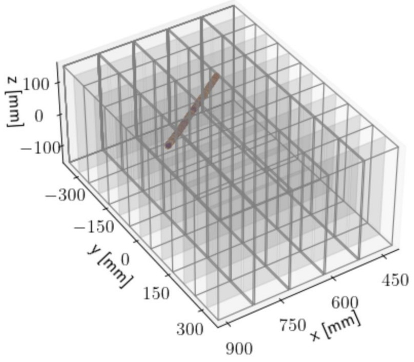
The differentiable fork of larnd-sim is currently only configured for module0

Updating it to use larnd-sized geometry may be possible, but requires some work.

Will proceed with the upstream fork of larnd-sim for now, but it will be nice to have differentiability for the future

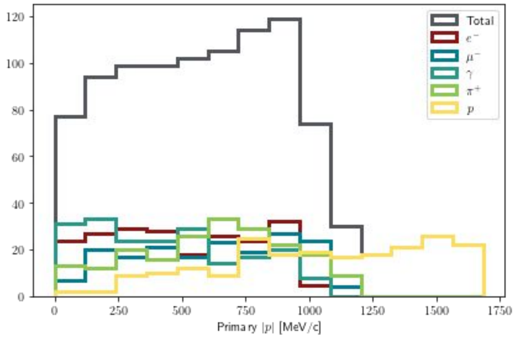


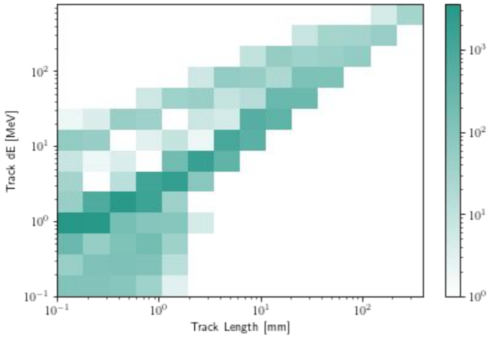
Sample Preparation

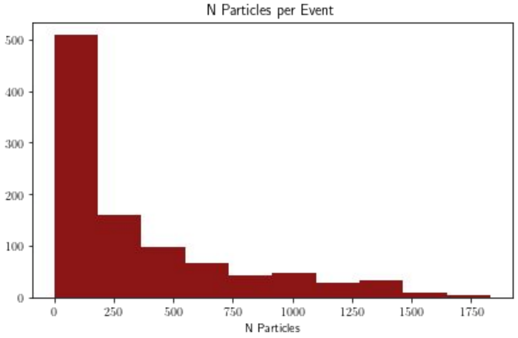


~100k single primary particle events are prepared on SDF at /sdf/group/neutrino/dougl215/singleParticle with MPV

An example of how to parse these outputs into x,y,z,dQ (in the form of a simple plotting utility) can be grabbed from here: https://github.com/DanielMDouglas/NDeventDisplay







Next Steps

- Generate Train/Test samples ✔
- Preliminary reconstruction ✔
- Data loader ✔
- Configurable network ✔
- Loss function 🤔 some ideas… start with MSE
- Train!
- Voxelization
  - Edep-sim voxelizer ✔
  - Larnd-sim voxelized input fork