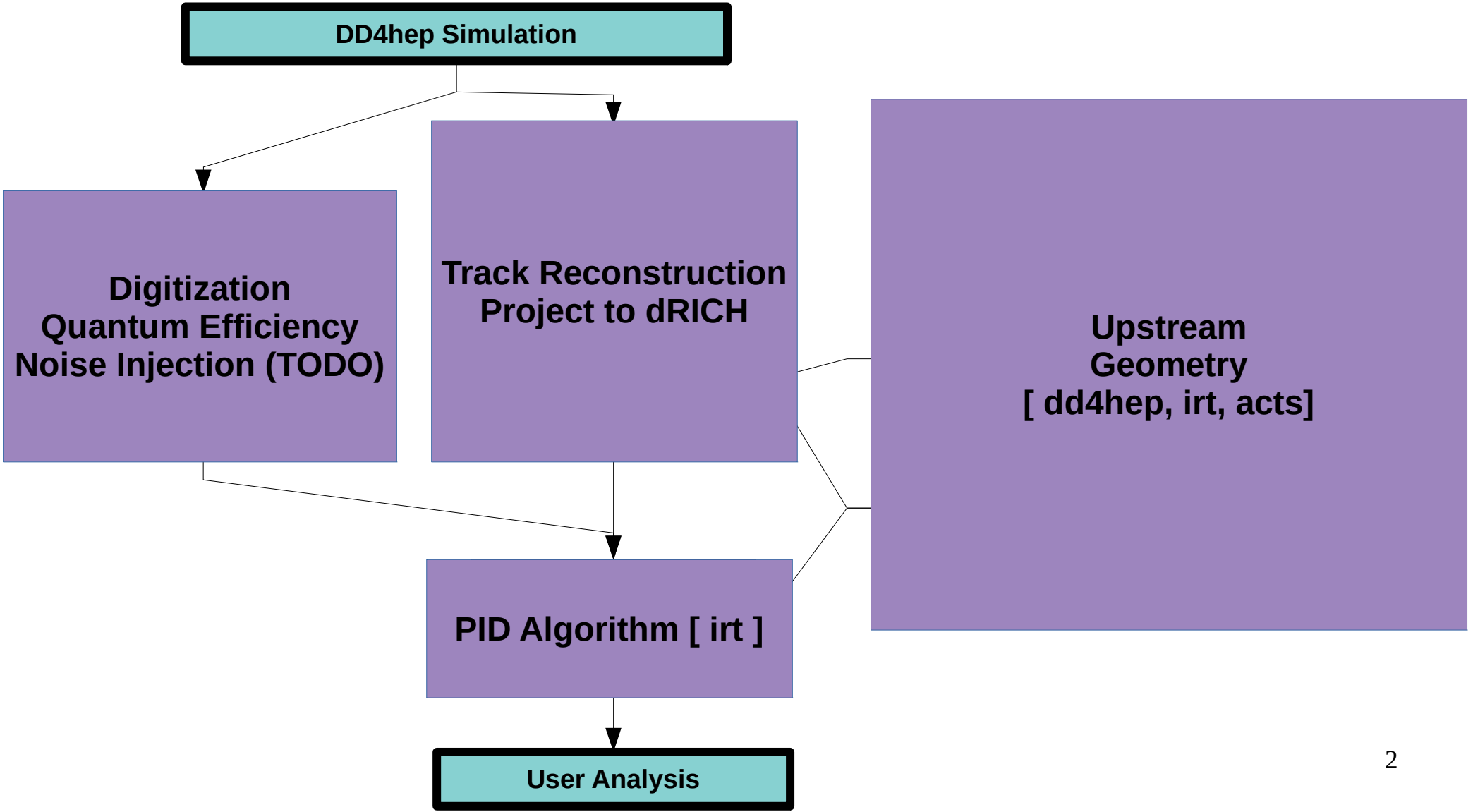
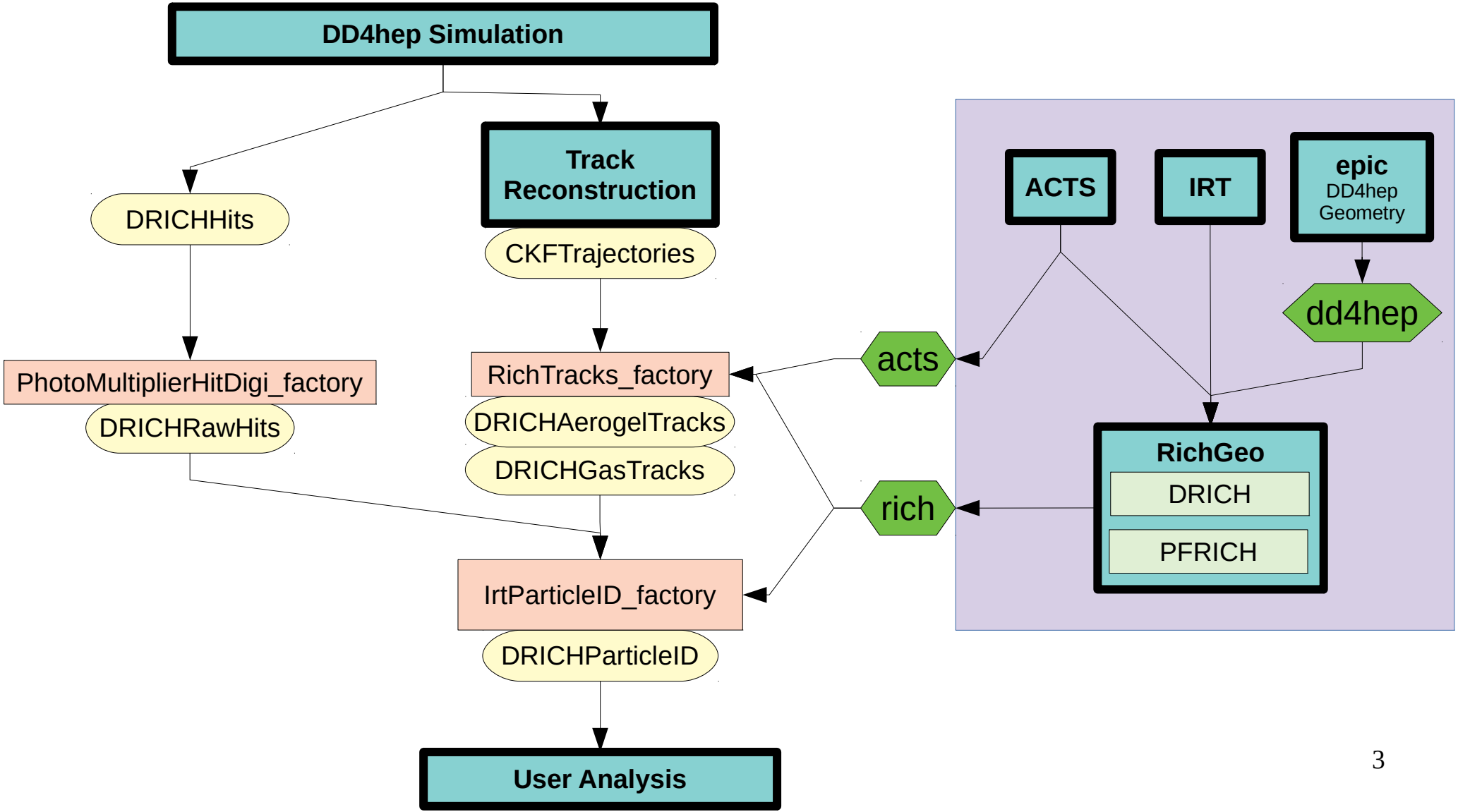
dRICH Reconstruction Update

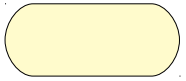
Christopher Dilks
dRICH Meeting
14 December 2022


DD4hep Simulation
Digitization
Quantum Efficiency
Noise Injection (TODO)
Track Reconstruction
Project to dRICH
Upstream
Geometry
[dd4hep, irt, acts]
PID Algorithm [irt]
User Analysis

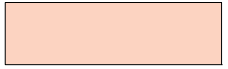

DD4hep Simulation
Track Reconstruction
DRICHHits
CKFTrajectories
PhotoMultiplierHitDigi_factory
DRICHRawHits
RichTracks_factory
DRICHAerogelTracks
DRICHGasTracks
IrtParticleID_factory
DRICHParticleID
User Analysis
ACTS
IRT
epic
DD4hep
Geometry
dd4hep
acts
rich
RichGeo
DRICH
PFRICH

EICrecon (JANA-based) dRICH Reconstruction

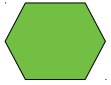
External EIC-recon Independent code (upstream, downstream, etc.)



Collection of objects, such as sensor hits or reconstructed track points



Factory, that turns input collection(s) into a single output collections. Typically comes with a JANA-independent algorithm



Service: define once, used in many places; handles common needs such as geometry and I/O

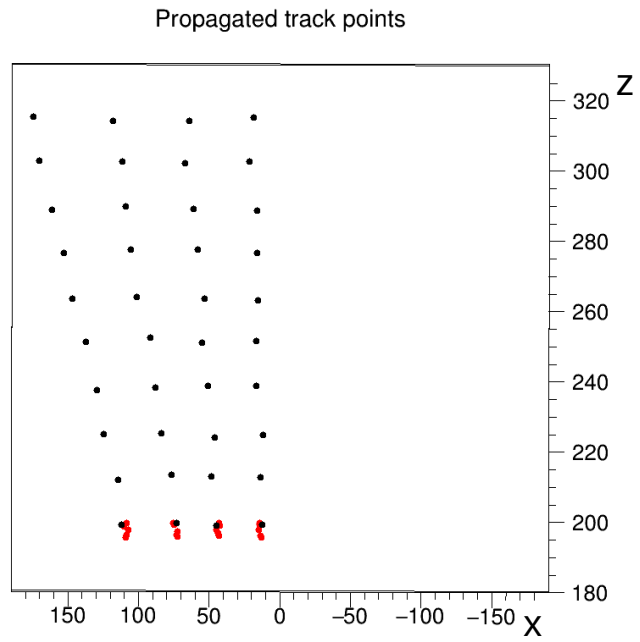
Blocker

- EICrecon currently lacks a unit system…
- Proposed DD4hep units and usage of its parser
- Coupled with the need to refactor the configuration

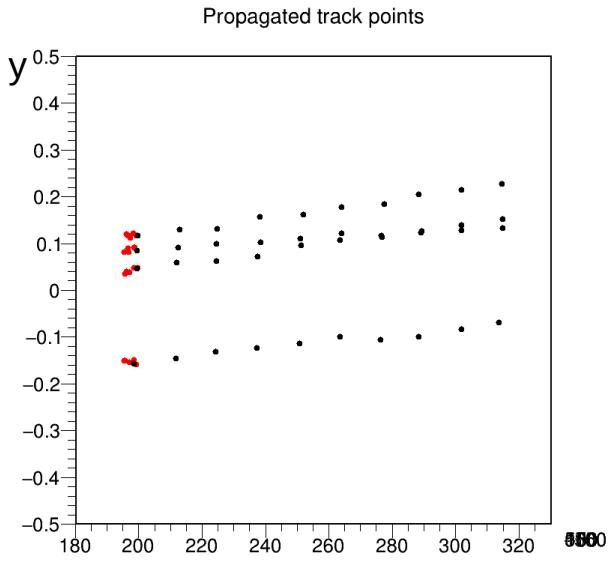
Track Propagation

5 z-planes in aergoel
10 z-planes in gas

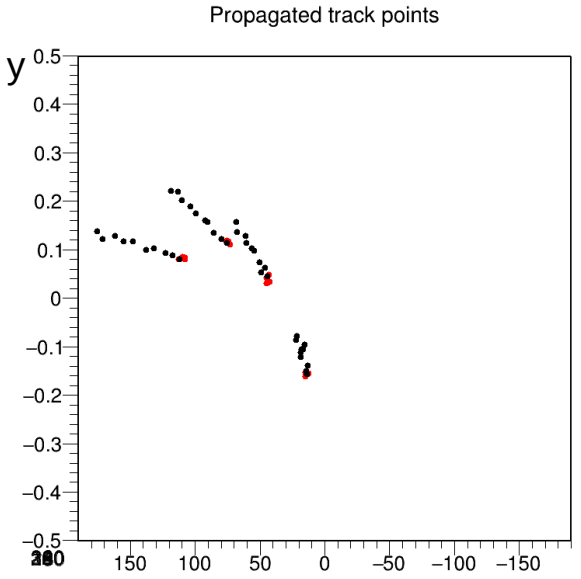
Thrown pions in y=0 plane
Reconstructed track points in **Aerogel** and **Gas**



Top view (toward +y)



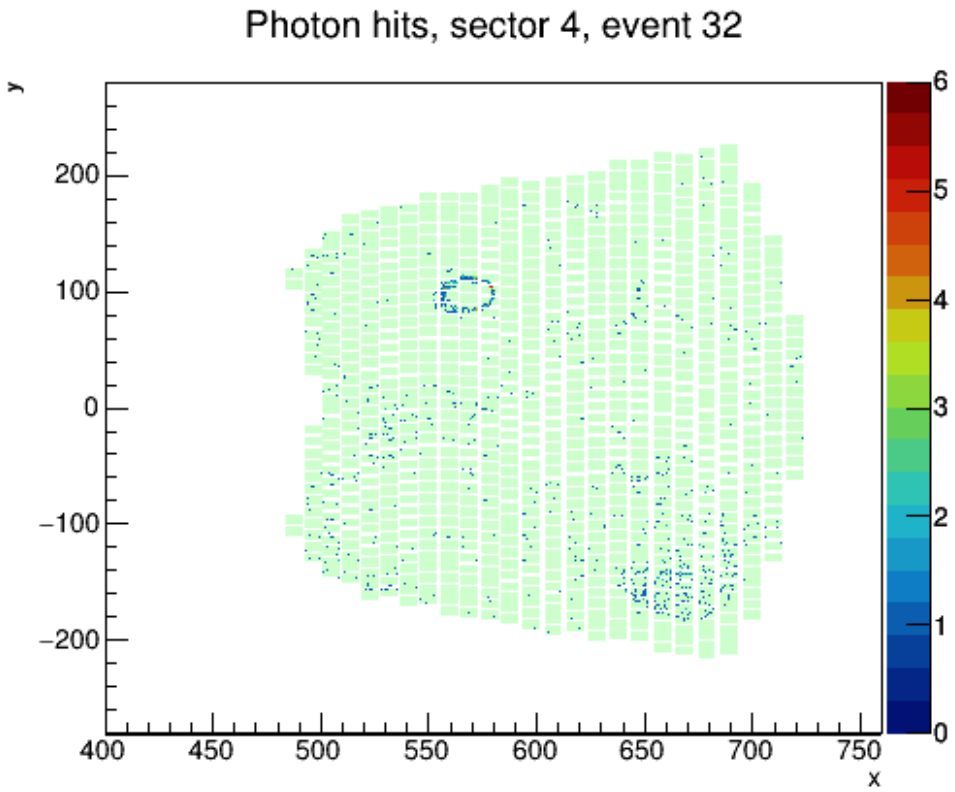
Side view (toward +x)

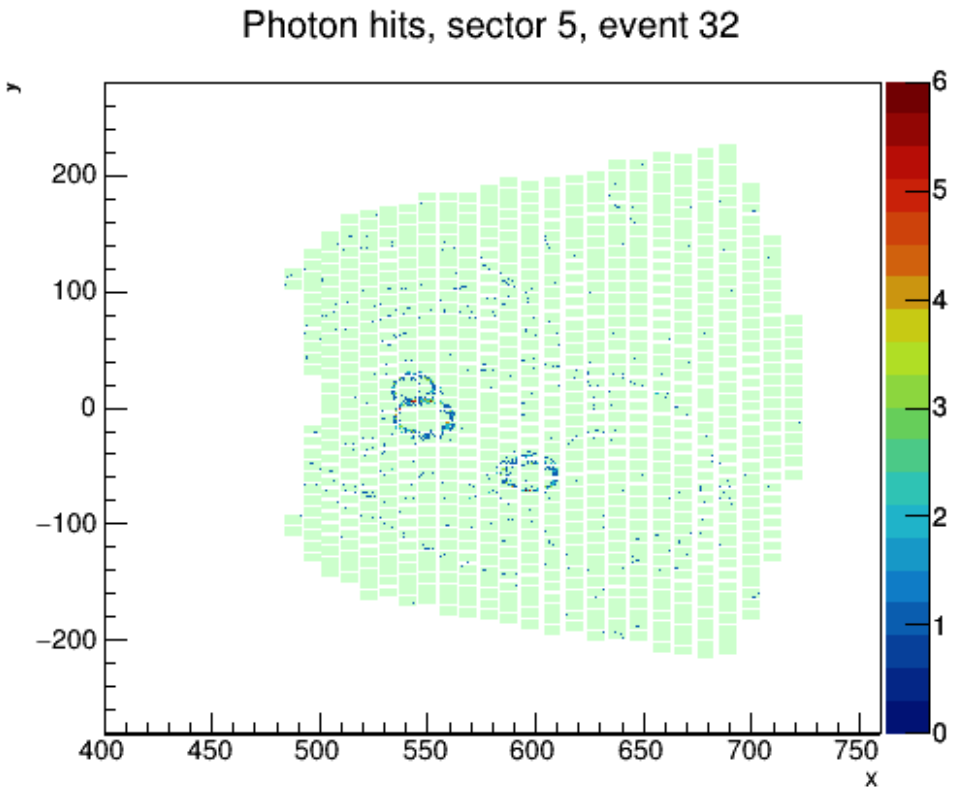


Front view (toward +z)

Note: Small variations in point positions may not be real (more likely an artifact of drawing)

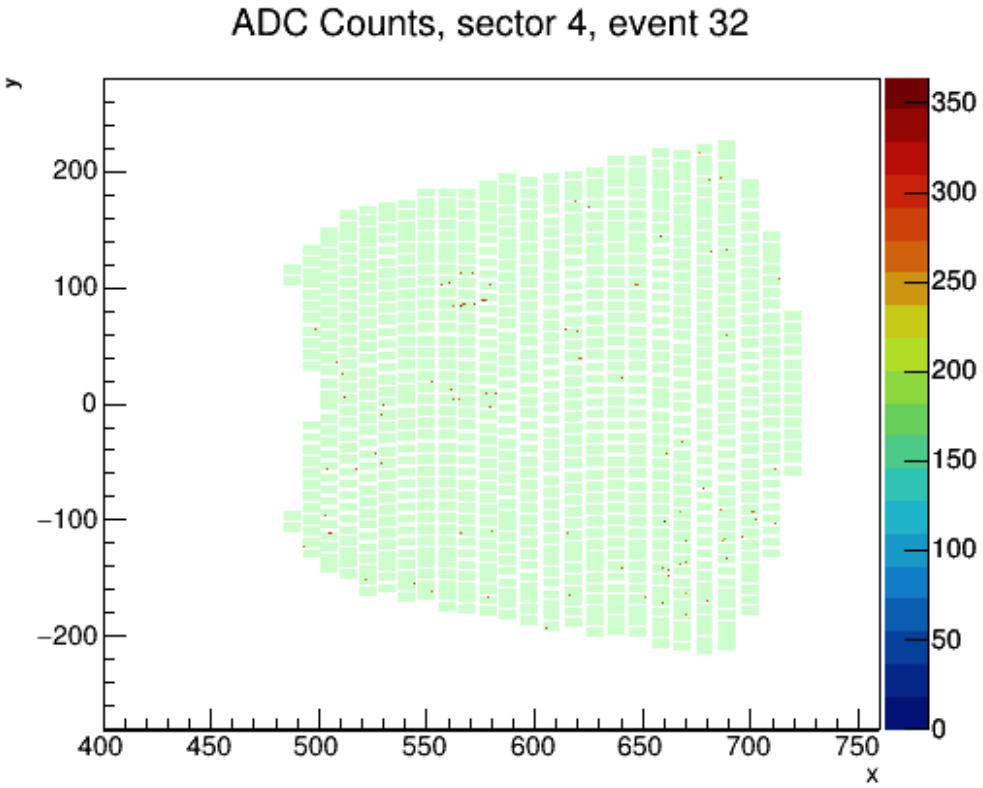
Event Display

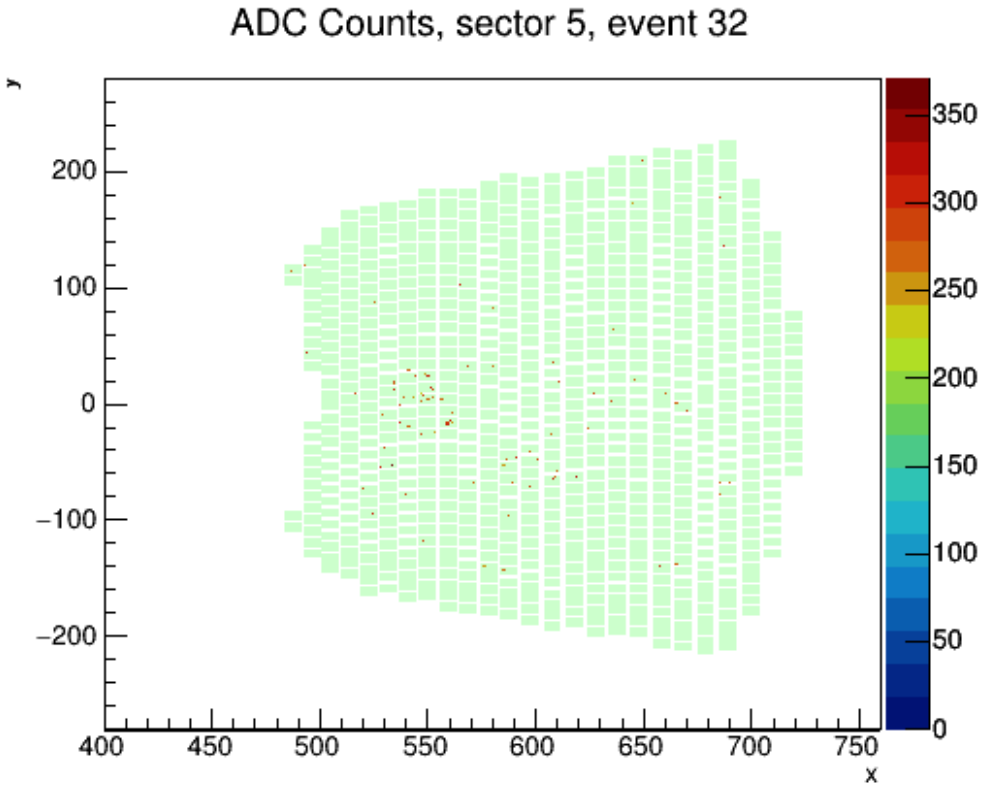




Event Display

ADC-weighted pixel hits
→ visualize effects of noise modeling



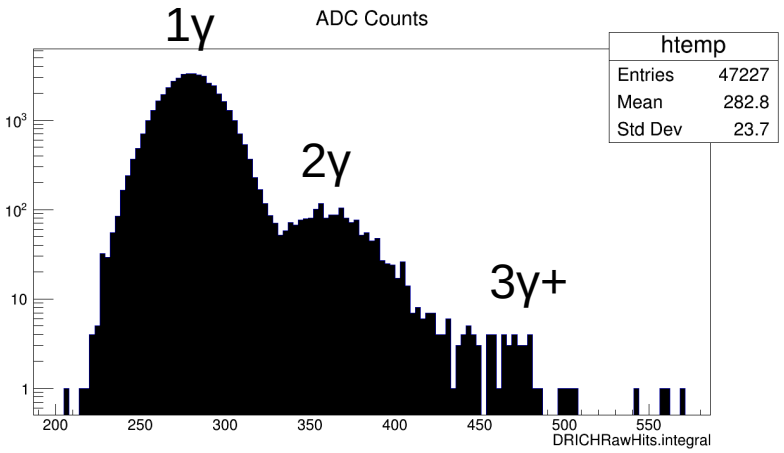


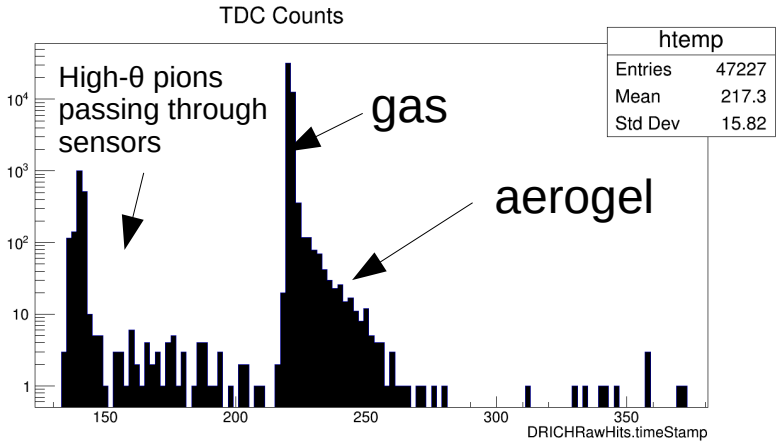
Digitization

- **TODO: validate configuration parameters**
  - **Trigger parameters (gate, pedestal, etc.)**
  - **Quantum Efficiency**
  - **Safety Factor 70%**
  - **Pixel Gap cuts**
  - **TODO: Noise injection**
  - **TODO: add Time over Threshold**

```
std::vector<std::pair<double, double> > quantumEfficiency = {
  {325*dd4hep::nm, 0.04},
  {340*dd4hep::nm, 0.10},
  {350*dd4hep::nm, 0.20},
  {370*dd4hep::nm, 0.30},
  {400*dd4hep::nm, 0.35},
  {450*dd4hep::nm, 0.40},
  {500*dd4hep::nm, 0.38},
  {550*dd4hep::nm, 0.35},
  {600*dd4hep::nm, 0.27},
  {650*dd4hep::nm, 0.20},
  {700*dd4hep::nm, 0.15},
  {750*dd4hep::nm, 0.12},
  {800*dd4hep::nm, 0.08},
  {850*dd4hep::nm, 0.06},
  {900*dd4hep::nm, 0.04}
};
```

```
// triggering
double hitTimeWindow = 20.0*dd4hep::ns;
double timeStep      = 0.0625*dd4hep::ns;
double speMean       = 80.0;
double speError      = 16.0;
double pedMean       = 200.0;
double pedError      = 3.0;
```

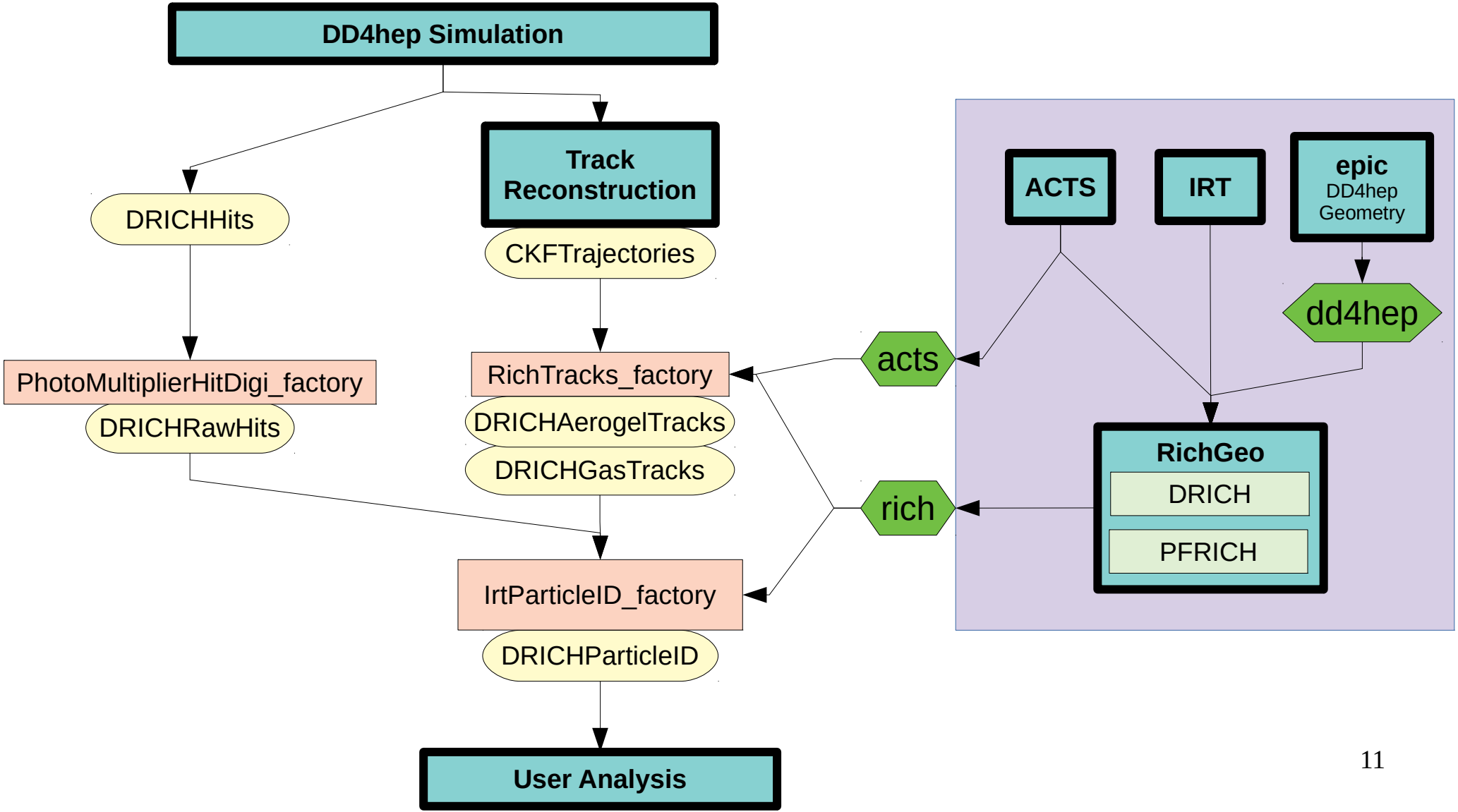


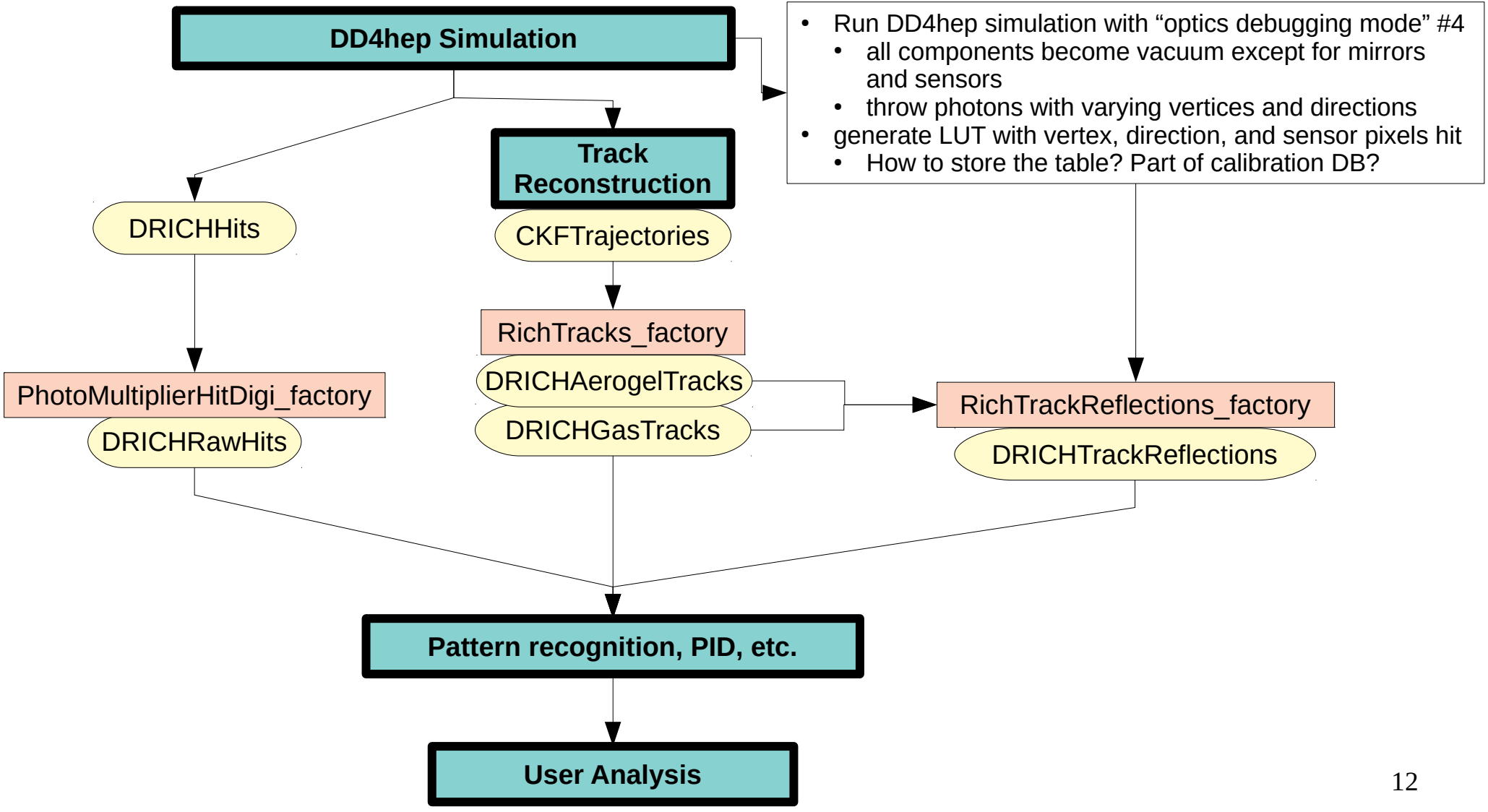


Noise Injection

- SiPM Noise injection could be added to PhotoMultiplierHitDigi
  - https://github.com/eic/EICrecon/issues/352
  - See G4SiPM for ideas and models
  - Longer term beyond dRICH scope: DDDigi

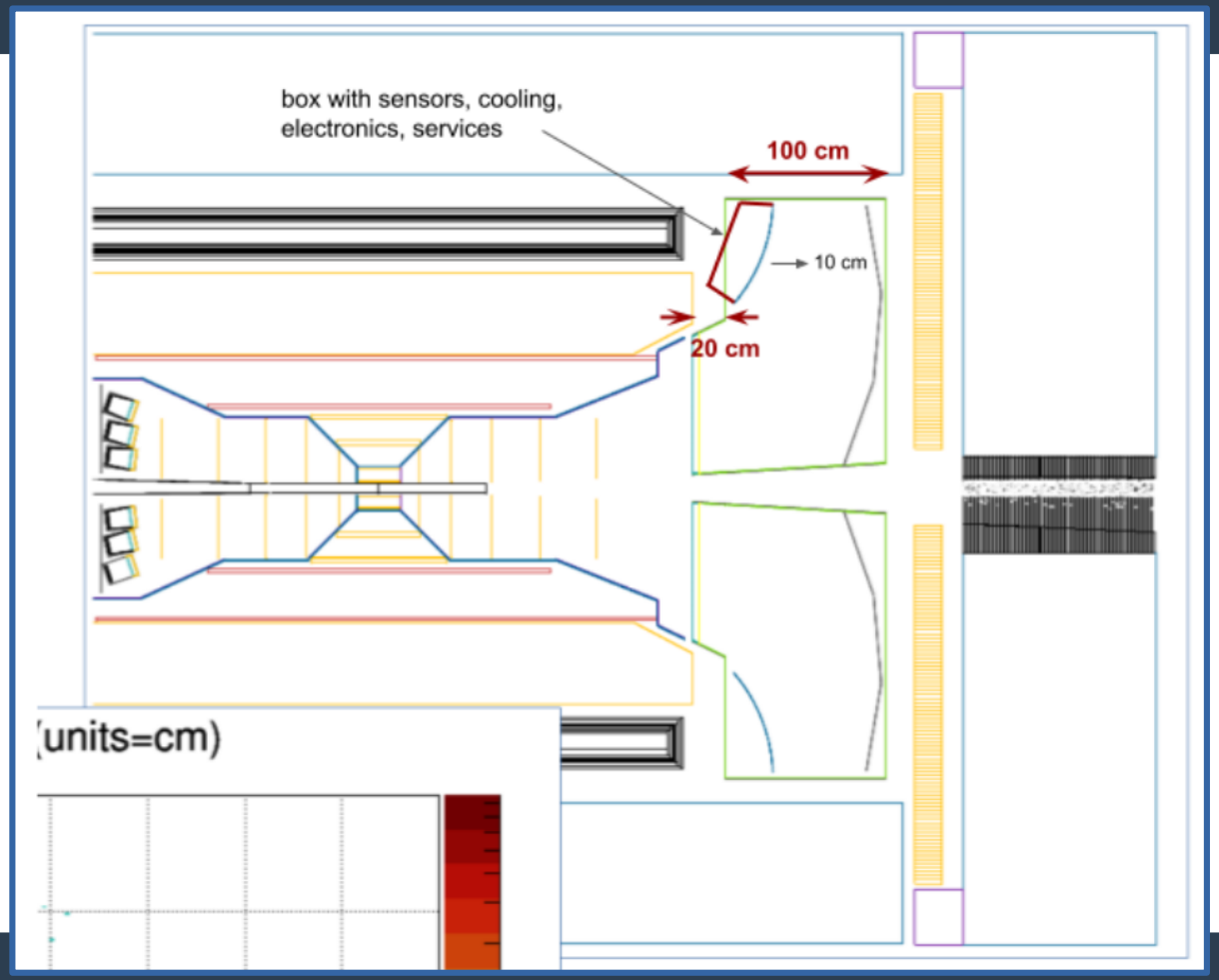
Keep in mind other sources of noise, in both the simulation and reconstruction levels…


DD4hep Simulation
Track Reconstruction
DRICHHits
CKFTrajectories
PhotoMultiplierHitDigi_factory
DRICHRawHits
RichTracks_factory
DRICHAerogelTracks
DRICHGasTracks
IrtParticleID_factory
DRICHParticleID
User Analysis
ACTS
IRT
epic
DD4hep
Geometry
dd4hep
acts
rich
RichGeo
DRICH
PFRICH


DD4hep Simulation
Run DD4hep simulation with “optics debugging mode” #4
all components become vacuum except for mirrors and sensors
throw photons with varying vertices and directions
generate LUT with vertex, direction, and sensor pixels hit
How to store the table? Part of calibration DB?
Track Reconstruction
DRICHHits
CKFTrajectories
RichTracks_factory
PhotoMultiplierHitDigi_factory
DRICHAerogelTracks
DRICHGasTracks
RichTrackReflections_factory
DRICHRawHits
DRICHTrackReflections
Pattern recognition, PID, etc.
User Analysis

Modeling Services

- https://github.com/eic/epic/issues/175
- Add service material for the sensors
  - Resin substrates, Bases, Support, Cooling, …
  - **What materials?**
  - **What thickness?**
- Need to shift the sensors forward, and re-focus the mirror



PID Parameterization

- Parameterization generator code:
  - Efficiency in bins of (η,p), for pairs in {π,K,p}
  - Configuration for Delphes fast simulation (screenshot)
  - Could make a format usable in full simulation analysis

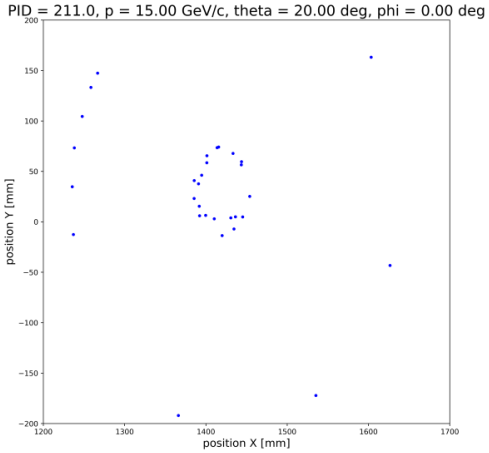
```
add EfficiencyFormula {211} {321} {
  (eta<  1.20 || eta>=  3.60 || pt * cosh(eta) <    0.90 || pt * cosh(eta) >=   27.00) * ( 0.00 ) +
  (  1.20 <= eta && eta <   1.60) * (   0.90 <= pt * cosh(eta) && pt * cosh(eta) <    1.40) * (0.000000) +
  (  1.20 <= eta && eta <   1.60) * (   1.40 <= pt * cosh(eta) && pt * cosh(eta) <    2.90) * (0.000000) +
  (  1.20 <= eta && eta <   1.60) * (   2.90 <= pt * cosh(eta) && pt * cosh(eta) <    4.20) * (0.000000) +
  (  1.20 <= eta && eta <   1.60) * (   4.20 <= pt * cosh(eta) && pt * cosh(eta) <    5.50) * (0.000000) +
  (  1.20 <= eta && eta <   1.60) * (   5.50 <= pt * cosh(eta) && pt * cosh(eta) <   10.00) * (0.000000) +
  (  1.20 <= eta && eta <   1.60) * (  10.00 <= pt * cosh(eta) && pt * cosh(eta) <   15.00) * (0.000381) +
  (  1.20 <= eta && eta <   1.60) * (  15.00 <= pt * cosh(eta) && pt * cosh(eta) <   20.00) * (0.026793) +
  (  1.20 <= eta && eta <   1.60) * (  20.00 <= pt * cosh(eta) && pt * cosh(eta) <   27.00) * (0.140689) +
  (  1.60 <= eta && eta <   2.00) * (   0.90 <= pt * cosh(eta) && pt * cosh(eta) <    1.40) * (0.000000) +
```

Machine Learning – for Reconstruction

AI4EIC 2022 Hackathon:
https://indico.bnl.gov/event/16586/page/435-hackathon

eventID	PID	momentum	theta	phi	X0	...	X59	Y0	...	Y59	Z0	...	Z59
0	211 or 321	p [GeV/c]	θ (deg)	ϕ (deg)	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]
1	211 or 321	p [GeV/c]	θ (deg)	ϕ (deg)	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]
..	211 or 321	p [GeV/c]	θ (deg)	ϕ (deg)	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]
..	211 or 321	p [GeV/c]	θ (deg)	ϕ (deg)	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]
N	211 or 321	p [GeV/c]	θ (deg)	ϕ (deg)	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]

Table 1: Table summarizing the data format for the hackathon problems



- Challenge: use ML and these data to classify between pions and kaons, under scenarios of increasing difficulty
- The most difficult scenario involved varying momenta, noise hits, and B-field
- Future idea: integrate into EICrecon and compare to baseline IRT