BERKELEY LAB
Bringing Science Solutions to the World

U.S. DEPARTMENT OF ENERGY
Office of Science

Data Model for Background and Noise Study

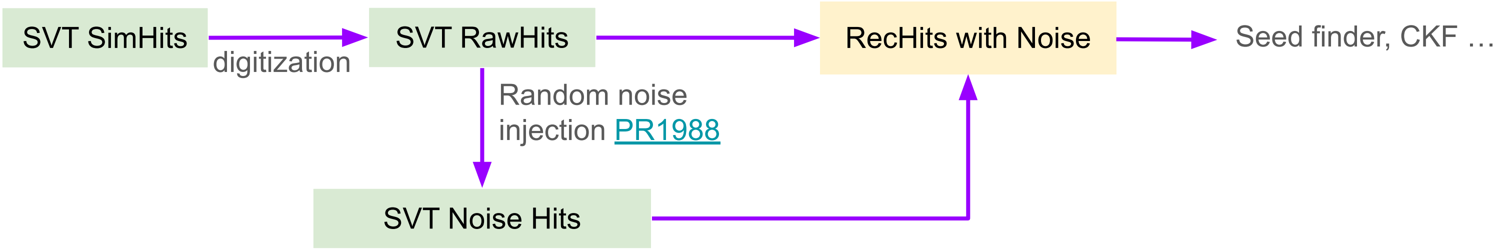
Shujie Li, Barak Schmookler

ePIC Reconstruction WG meeting

Aug 25, 2025

SVT Pixel Noise and Beam Background in simulation

- **Noise injection:** (See Minjung's talk)



SVT SimHits → SVT RawHits (digitization)

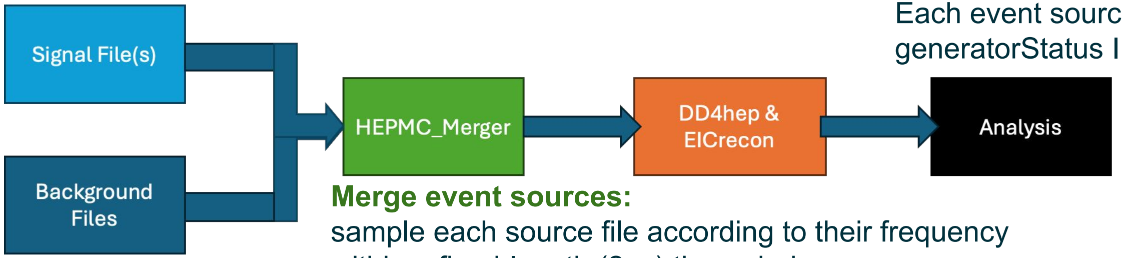
SVT RawHits → SVT Noise Hits (Random noise injection PR1988)

SVT RawHits → RecHits with Noise

SVT Noise Hits → RecHits with Noise

RecHits with Noise → Seed finder, CKF …

- **Signal + background merge:**



Signal File(s) → HEPMC_Merger

Background Files → HEPMC_Merger

HEPMC_Merger → DD4hep & EICrecon → Analysis

Each event source has distinct generatorStatus ID in MCParticle

Merge event sources:
sample each source file according to their frequency within a fixed-length (2us) time window

One event = one collision → **merged** → One event = one 2us time slice that contains ("forced DIS")

- one Q^2>1 GeV^2 NC DIS events
- Beam background at calculated freq. (SR, electron Bremsstrahlung, Coulomb, Touschek, proton beam gas)

Noise and Background Hit Rate

# of hits are shown in per 2us slice

		# of noise hits	**# of bkgrd hits (18x275)**
Detector name	**Barrel layers**	FHR = 5e-7/pixel/2us	
VertexBarrel	L0	76	371
	L1	102	220
	L2	254	113
SagittaSiBarrel	L3	1,145	43
OuterSiBarrel	L4	2,639	10

Scale with sensitive area

Higher rate when closer to beampipe

Detector name	**Disks**	**# of noise hits**	**# of bkgrd hits (18x275)**
InnerTrackerEndcapN	E-Disk0	405	99
MiddleTrackerEndcapN	E-Disk1	1,442	175
OuterTrackerEndcapN	E-Disk2	1,442	169
	E-Disk3	1,440	112
	E-Disk4	1,435	17
InnerTrackerEndcapP	H-Disk0	405	93
MiddleTrackerEndcapP	H-Disk1	1,442	115
OuterTrackerEndcapP	H-Disk2	1,441	34
	H-Disk3	1,429	9
	H-Disk4	1,414	8

Noise and Background Impact on Tracking

- **Simulation:**
  Noise hits / Background particle → contamination hits

- **Impact**:
  - For seed finder:
    - More potential triplets
    - Slower performance
  - For track finding:
    - Low purity (mix hits from signal and contamination)
    - Ghost track (track assembled from contamination hits)
  - For track fitting:
    - Worse resolutions
    - Low efficiency (if fit quality too bad)

Example:

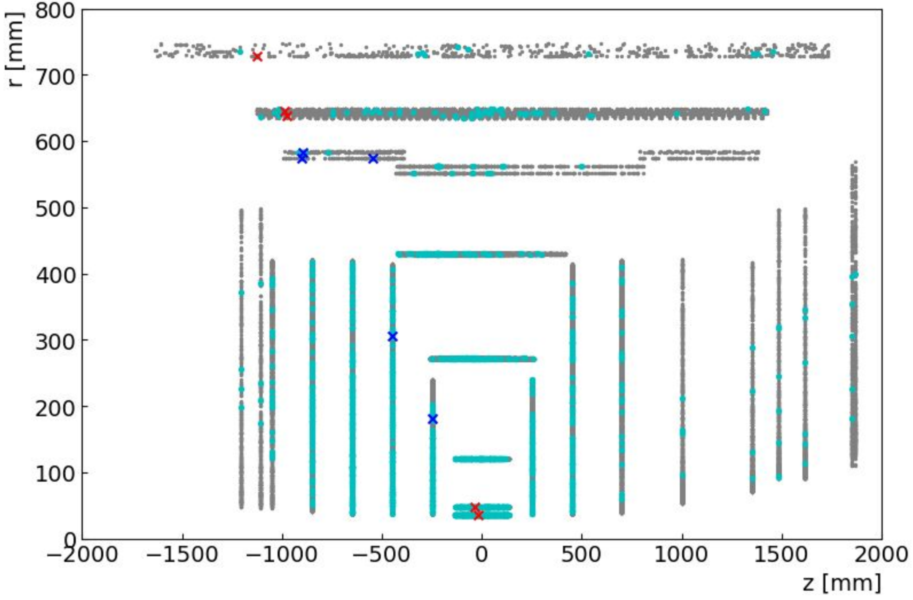
Hit map from one 2us slice of signal+background

Grey: all hits from 1000 events

Cyan: all hits from a given event

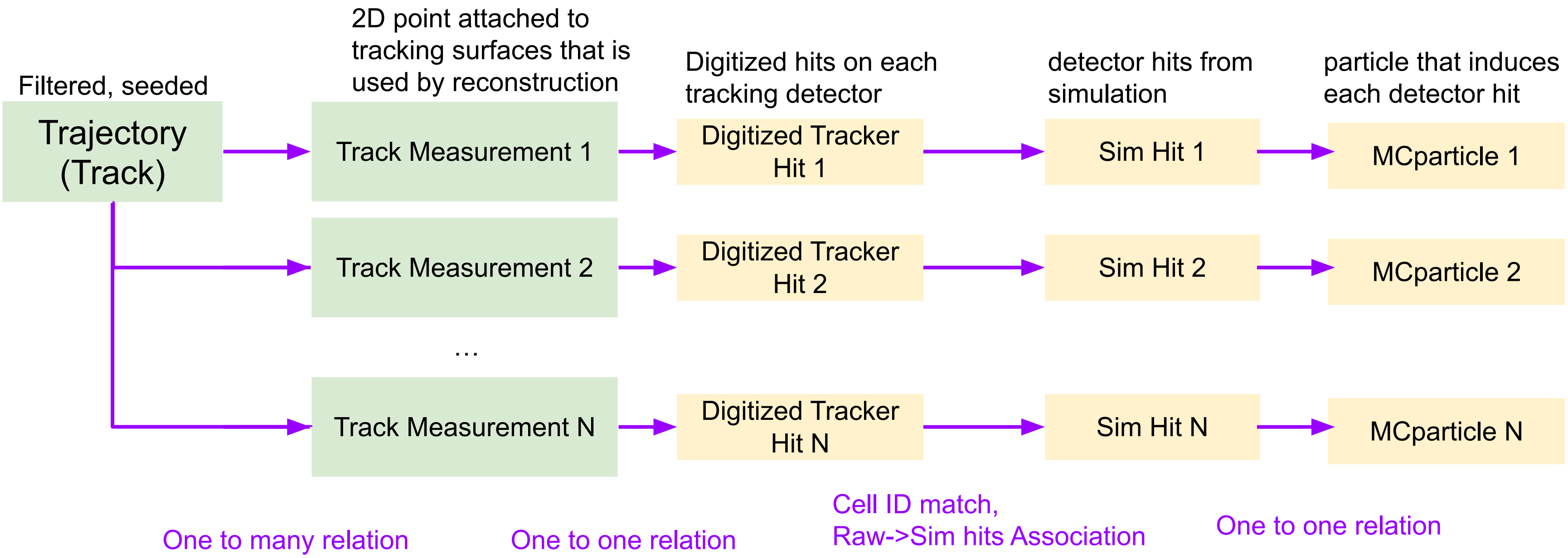
Red: good measurements used in tracking

Blue: outliers (didn't pass chi2 cut)



Hit-based Tracking Study Workflow

- **Match trajectory, detector hits, and simulated particles**



Filtered, seeded

Trajectory (Track)

2D point attached to tracking surfaces that is used by reconstruction

Track Measurement 1

Track Measurement 2

…

Track Measurement N

Digitized hits on each tracking detector

Digitized Tracker Hit 1

Digitized Tracker Hit 2

Digitized Tracker Hit N

detector hits from simulation

Sim Hit 1

Sim Hit 2

Sim Hit N

particle that induces each detector hit

MCparticle 1

MCparticle 2

MCparticle N

One to many relation

One to one relation

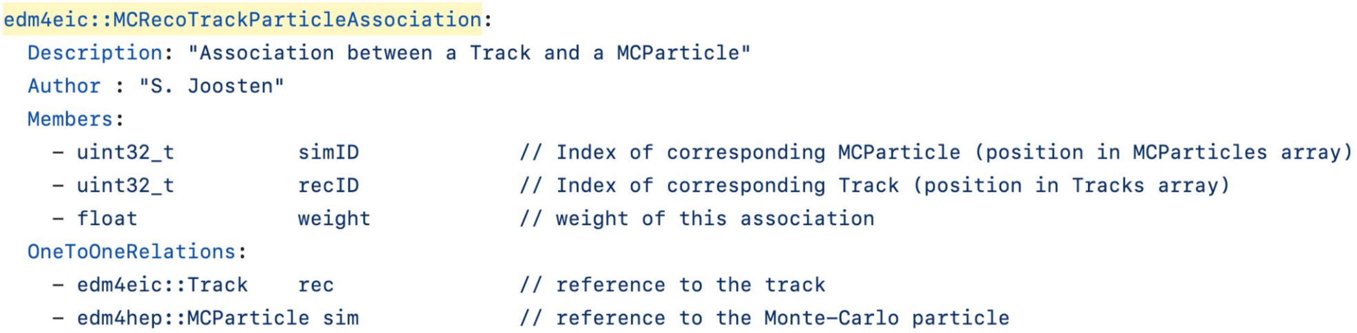
Cell ID match,
Raw->Sim hits Association

One to one relation

Efficiency: fraction of primary particles that are associated with tracks.
Purity: for a given track, fraction of hits from one particle.

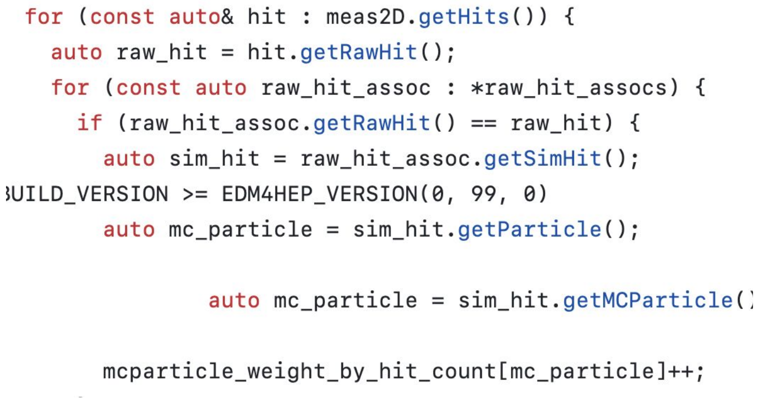
Tracking Study Workflow

- **Existing Track → Particle association**



```
edm4eic::MCRecoTrackParticleAssociation:
  Description: "Association between a Track and a MCParticle"
  Author : "S. Joosten"
  Members:
    - uint32_t          simID           // Index of corresponding MCParticle (position in MCParticles array)
    - uint32_t          recID           // Index of corresponding Track (position in Tracks array)
    - float             weight          // weight of this association
  OneToOneRelations:
    - edm4eic::Track    rec             // reference to the track
    - edm4hep::MCParticle sim           // reference to the Monte-Carlo particle
```

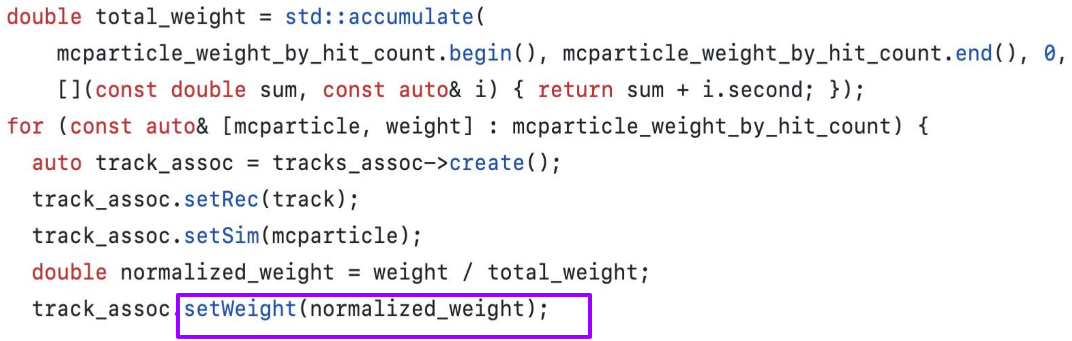
- **Existing code in Acts2Tracks.cc**



```
for (const auto& hit : meas2D.getHits()) {
  auto raw_hit = hit.getRawHit();
  for (const auto raw_hit_assoc : *raw_hit_assocs) {
    if (raw_hit_assoc.getRawHit() == raw_hit) {
      auto sim_hit = raw_hit_assoc.getSimHit();
UILD_VERSION >= EDM4HEP_VERSION(0, 99, 0)
      auto mc_particle = sim_hit.getParticle();

          auto mc_particle = sim_hit.getMCParticle(

      mcparticle_weight_by_hit_count[mc_particle]++;
```

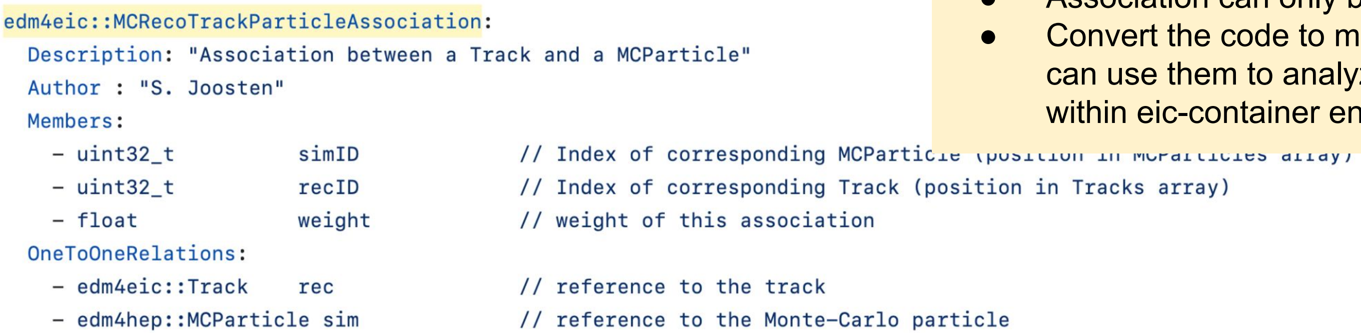


```
double total_weight = std::accumulate(
    mcparticle_weight_by_hit_count.begin(), mcparticle_weight_by_hit_count.end(), 0,
    [](const double sum, const auto& i) { return sum + i.second; });
for (const auto& [mcparticle, weight] : mcparticle_weight_by_hit_count) {
  auto track_assoc = tracks_assoc->create();
  track_assoc.setRec(track);
  track_assoc.setSim(mcparticle);
  double normalized_weight = weight / total_weight;
  track_assoc.setWeight(normalized_weight);
```

purity

Tracking Study Workflow

Propose to add (1): track-based relation to hits

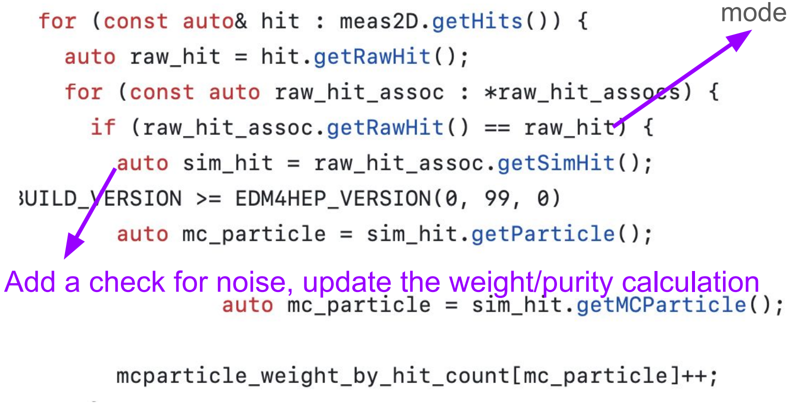


```
edm4eic::MCRecoTrackParticleAssociation:
  Description: "Association between a Track and a MCParticle"
  Author : "S. Joosten"
  Members:
    - uint32_t          simID             // Index of corresponding MCParticle (position in MCParticles array)
    - uint32_t          recID             // Index of corresponding Track (position in Tracks array)
    - float             weight            // weight of this association
  OneToOneRelations:
    - edm4eic::Track    rec               // reference to the track
    - edm4hep::MCParticle sim             // reference to the Monte-Carlo particle
```

OneToManyRelations:

- edm4eic::RawTrackerHit rawHits

Rawhits already accessible, need to save into the data model



```
for (const auto& hit : meas2D.getHits()) {
  auto raw_hit = hit.getRawHit();
  for (const auto raw_hit_assoc : *raw_hit_assocs) {
    if (raw_hit_assoc.getRawHit() == raw_hit) {
      auto sim_hit = raw_hit_assoc.getSimHit();
3UILD_VERSION >= EDM4HEP_VERSION(0, 99, 0)
      auto mc_particle = sim_hit.getParticle();

          auto mc_particle = sim_hit.getMCParticle();


    mcparticle_weight_by_hit_count[mc_particle]++;
```

Add a check for noise, update the weight/purity calculation

Suggestion from the meeting:

- Association can only be one2one
- Convert the code to methods, so that the user can use them to analyze eicrecon rootfiles within eic-container environment.

Example output:

MCParticle ID	Track ID	weight	rawHits
12	1	0.67	[VertexBarrel #1, MPGDDisk #3]
22	1	0.33	[SiDisk #1]

- Provides clear connection from track to the source of majority or contamination hits
- Allows various detector/physics WGs to estimate the detector- or physics-specific impact from contamination in a consistent way

Tracking Study Workflow

Propose to add (2): particle-based relation to hits Inspired by this example from STAR

edm4eic::MCParticleSimHitsAssociation:

```
            Description: "Association between SimHits and a MCParticle"
 Members:
            - uint32_t        particleID     // Index of corresponding MCParticle
            - uint32_t        nhits          // number of sim hits associated with this particle
OneToOneRelations:
             - edm4eic::SimTrackerHit  simHit            // reference to SimHits
             - edm4hep::MCParticle     particle            // reference to the Monte-Carlo particle
```

Example output:

MCParticle ID	nhits	simHits
12	10	[VertexBarrel #1, VertexBarrel #3, SiBarrel #1, MPGDDisk #1, …]
22	8	[Vertex Barrel #2, SiDisk #1, SiDisk #2, …]

- Identify source of detector hits in a noise+background sample
- Combine (1) and (2) for a comprehensive tracking analysis, e.g.:
    - Particle #12 left 10 hits through ePIC detector system, among which 2 hits are used to reconstruct track #1, 5 are used for track #2, 3 hits are not used in tracking (all in xx detector)
    - Track #3 has 4 hits from 2 particles and one noise hit, it’s a ghost track that should be removed from efficiency calculation.