MPGD Digitization (and Simulation) in EICrecon

Yann Bedfer

CEA/DPhN Saclay

19 March 2025

Present MPGD Workflow

- `ddsim` → Collections of **simulated hits** = `edm4hep::SimTrackerHit`

  $= (\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{\vec{P}}, \boldsymbol{E_{dep}}, \boldsymbol{t},$ **cellID, pathLength**. . .)

  MPGDs = `MPGDBarrelHits`, `OuterMPGDBarrelHits`,
  `BackwardMPGDEndcapHits`, `ForwardMPGDEndcapHits`.

- `eicrecon`'s **digitization** → Collections of **raw hits** = `edm4eic::RawTrackerHit`

  $= (\boldsymbol{iQ}, \boldsymbol{iT},$ **cellID**$)$

  → X-reference SimHit ↔ RawHit

  - `SiliconTrackerDigi` (*all but CyMBaL and Outer, as of now*):

    1 SimHit → **1** RawHit , **cellID** $=$ **cellID**

  - `MPGDTrackerDigi` (*CyMBaL, Outer, as of now*):

    1 SimHit → **2** RawHits, **cellID's** $\neq$ **cellID**

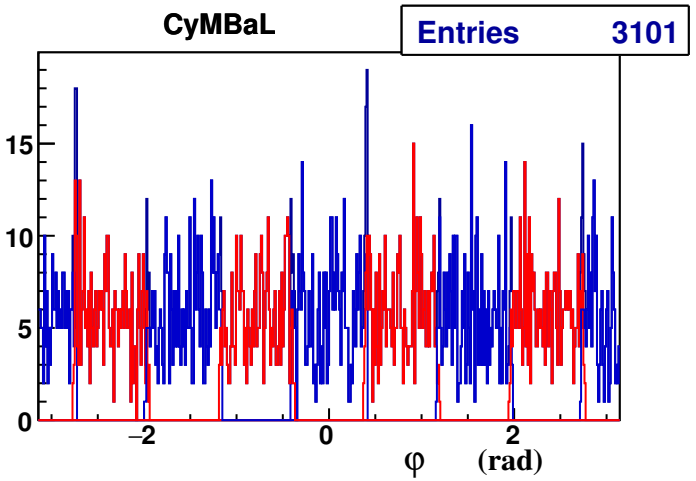
  RawHit/SimHit impacted by **Merging** of hits falling on same **cellID** ($SiTD$) or strip **cellID** (MTD).

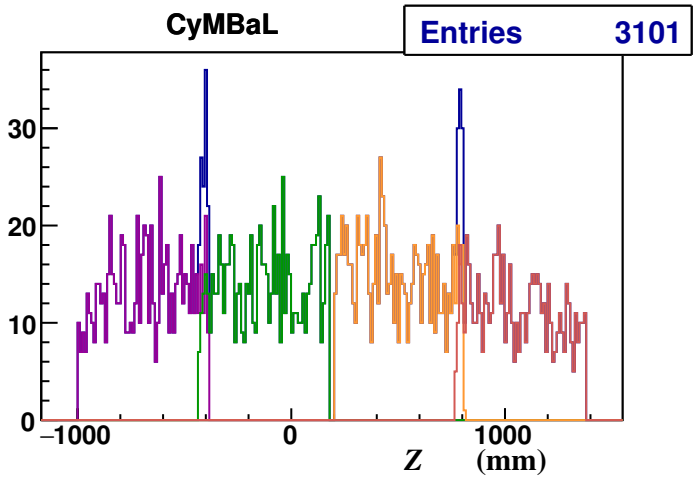
- `eicrecon`'s **reconstruction** → Collections of **reconstructed hits** `edm4eic::TrackerHit`

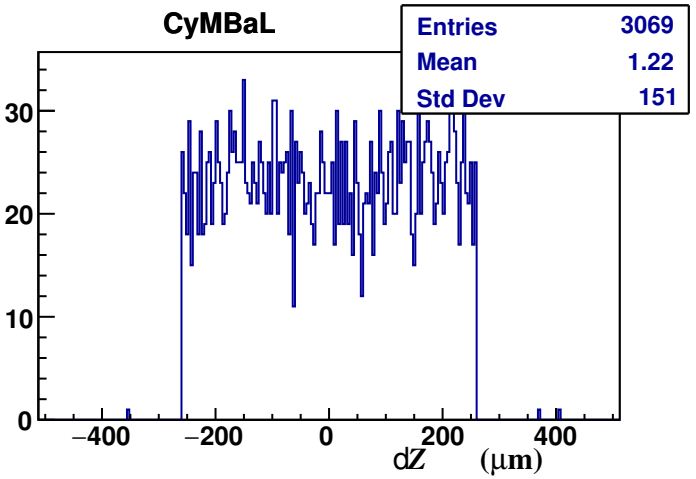
  $=$ (**position,positionError**, $\boldsymbol{E_{dep}}, \boldsymbol{\Delta E_{dep}}, \boldsymbol{t}, \boldsymbol{\Delta t},$ **cellID**)

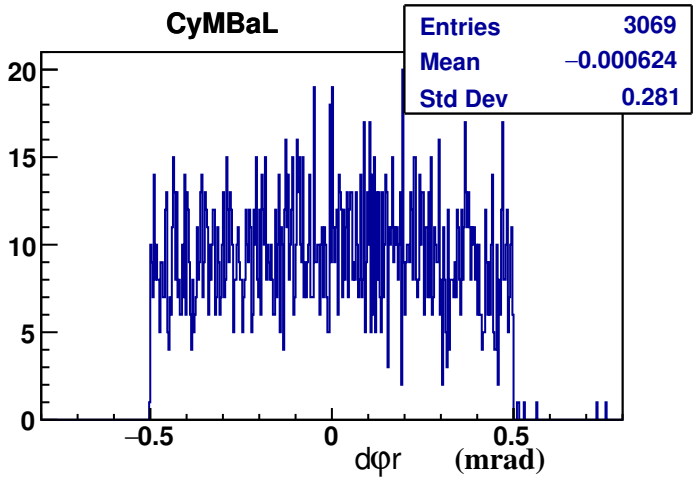
CyMBaL w/ SiliconTrackerDigi

- `ddsim: SIM.gun` μ^- $15 < \theta < 165$ deg. $P = 10$ GeV
- `eicrecon -PMPGD:SiFactoryPattern=0x3 ...` → SimHits, Residuals (RawHit-SimHit).





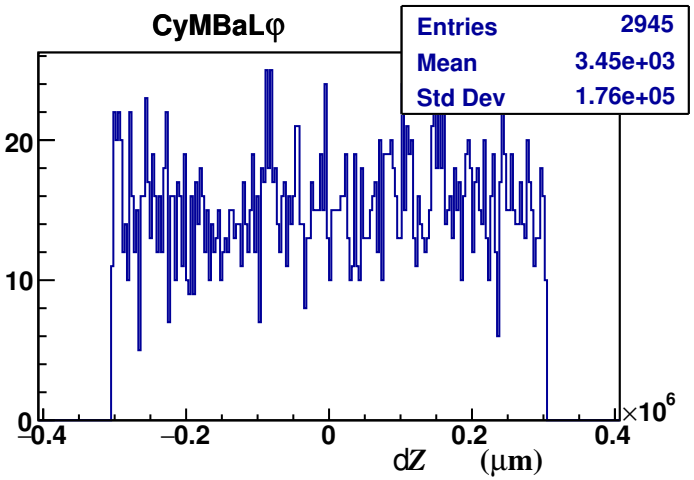


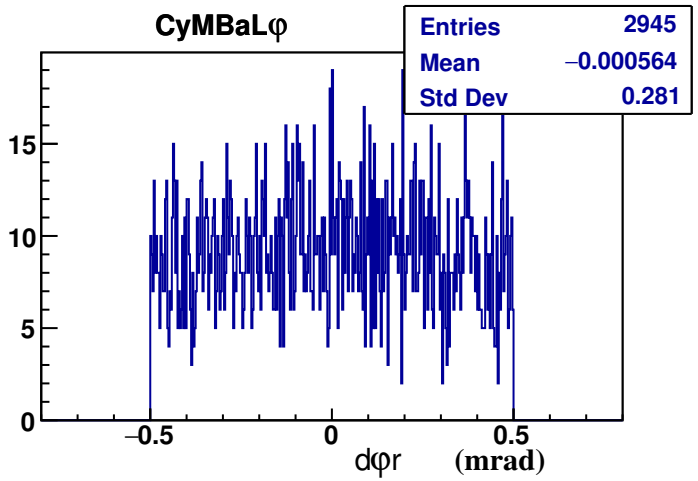


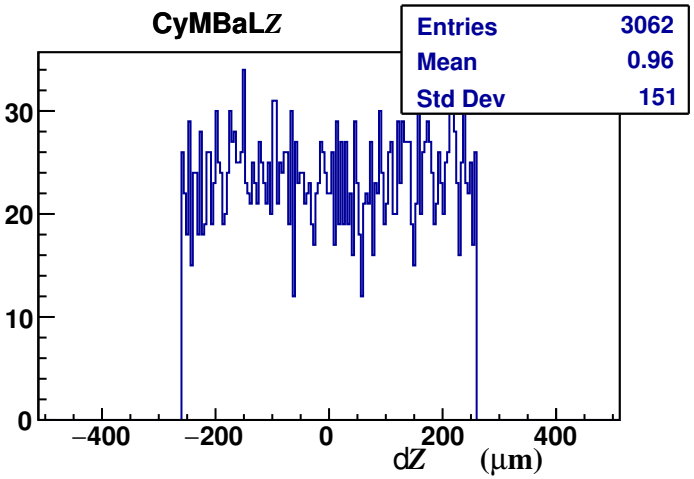
- *Note: Some inefficiency.*

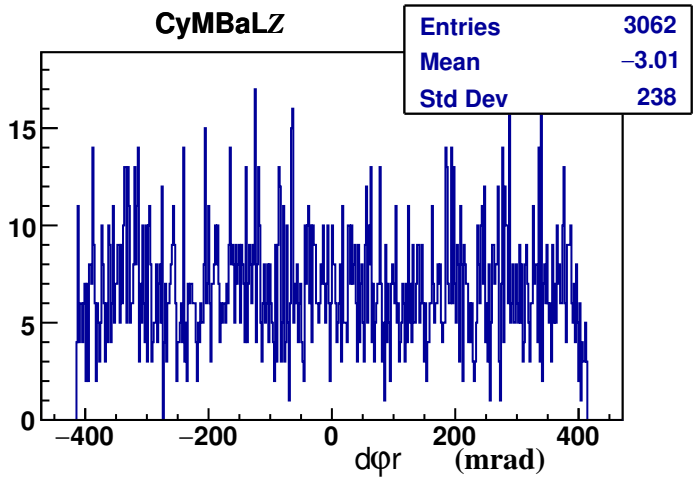
CyMBaL w/ MPGDTrackerDigi

- Same simulation as before.
- `eicrecon -PMPGD:SiFactoryPattern=0x0`(=*default*) . . . → Residuals (RawHit-SimHit).





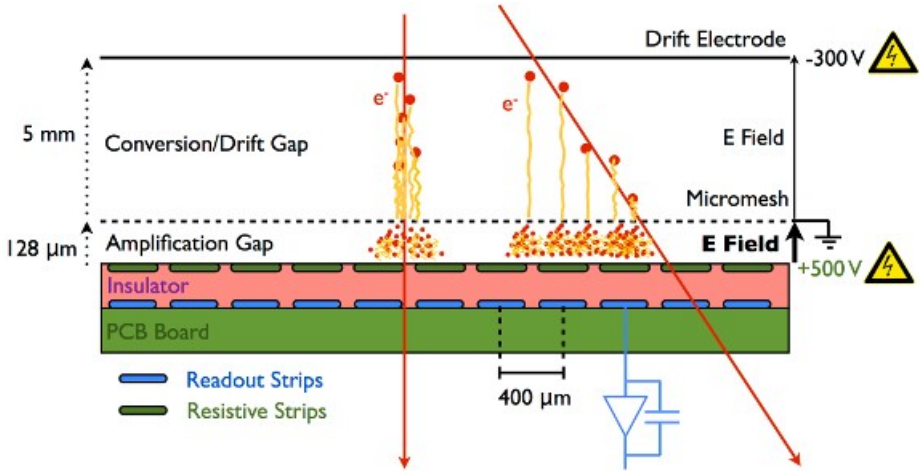




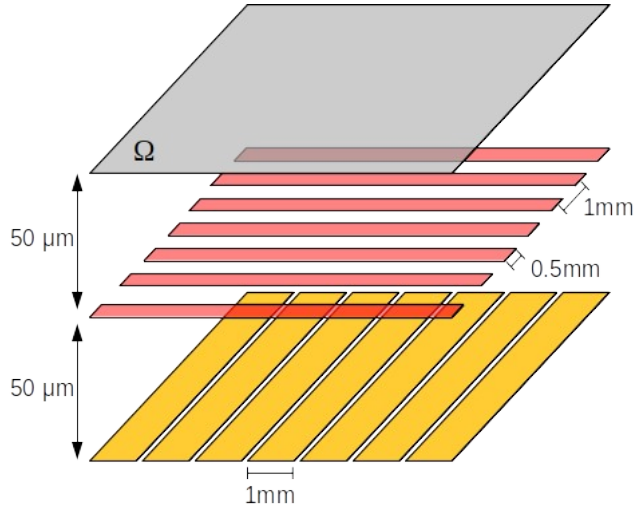
- *Note: φ-Entries* **!=** *Z-Entries: could be secondary particles.*

MPGDTrackerDigi

- **Current** `MPGDTrackerDigi`:
  - **Re-segmentation**: Overwriting what's done in simulation. → Strips instead of Pixels.
  - **Merging**: Based on newly segmented strip **cellID**.
  - **For the rest**: **exact same** as `SiliconTrackerDigi`.
- **Planned** `MPGDTrackerDigi`:



Class12 Micromegas: 1D-strip Readout



Charge Sharing ⇒ 2D-strip Readout

- **One I**onization × **A**mplification $\xrightarrow[\textbf{Sharing}]{\textbf{Charge}}$ **Two proto** Hits, along strips $\boldsymbol{u} = \boldsymbol{p}, \boldsymbol{n}$

  Amplitude/Time **S**mearing $= \boldsymbol{S}(\textbf{I}\times\textbf{A}) + \boldsymbol{S}(\textbf{ChSh}|\boldsymbol{u})) + \boldsymbol{S}(\textbf{F}\text{ront}\textbf{E}\text{nd})$.

  Do we need the **F**ront**E**nd term? Or negligible?
- **Charge Spreading** (*can only be a posteriori*): **proto** Hit → **cluster** of Hits.

Issues?

- Separate Simulation from Digitization?
  - Put forward by **Dmitry Kalinkin**: so as to have a **unique digitization method**.
    I.e.: Factorize `MPGDTrackerSimulation` × `SiliconTrackerDigi`.
    `MPGDTrackerSimulation` **input collection** of SimHits, **output new collection** of SimHits.
  - Problem: Any **specificity of MPGDs** preventing this?
    - Is FE electronics same for the two coordinates (*including parameters*)?
      If not, **output two collections**, one per coordinate, w/ **each its own parameters** `config`?
    - Amplitude smearing at **F**ront**E**nd? (*not in current* `SiliconTrackerDigi`)
    - **Conclusion: No showstopper**. But some amount of work. . .

- Issue: Threshold **discrimination** applied **too early**, see EICrecon #1722.
  - **Discrimination first**, then **Merging**.
    ⇒ Prevents merging of small energy from helping pass threshold.
  - **Merging first**, then **Discrimination**. **Time assigned** to Merged Hit**?**
    Earliest? ⇒ Sensitivity to whatever accidental background.
    Largest E_{dep}? Not fully satisfactory.
  - Shouldn't we have a `config::deadTime`?
    Does this imply inheritance of deadTime across time frames?

Backup

Question to experts

- I am looking for a **Strip Reference System** (*for OuterBarrel, which has U, V strips*).

- In `Acts::Surface`:

```
virtual Result<Vector2> globalToLocal(
    const GeometryContext& gctx, const Vector3& position,
    const Vector3& direction,
    double tolerance)
```

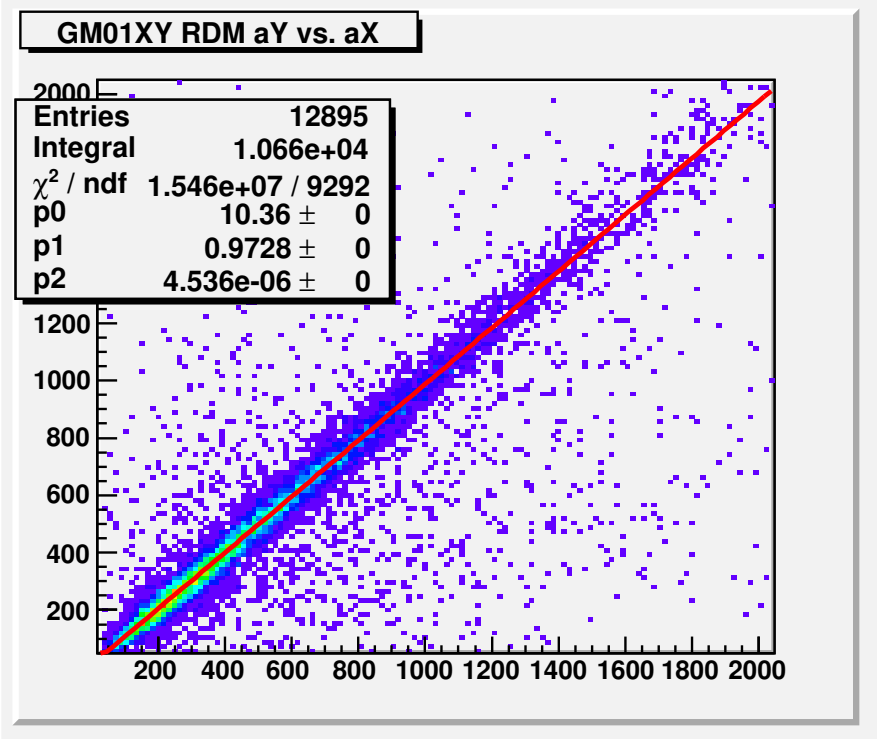
- In `DD4hep::DDrec::Surface`:

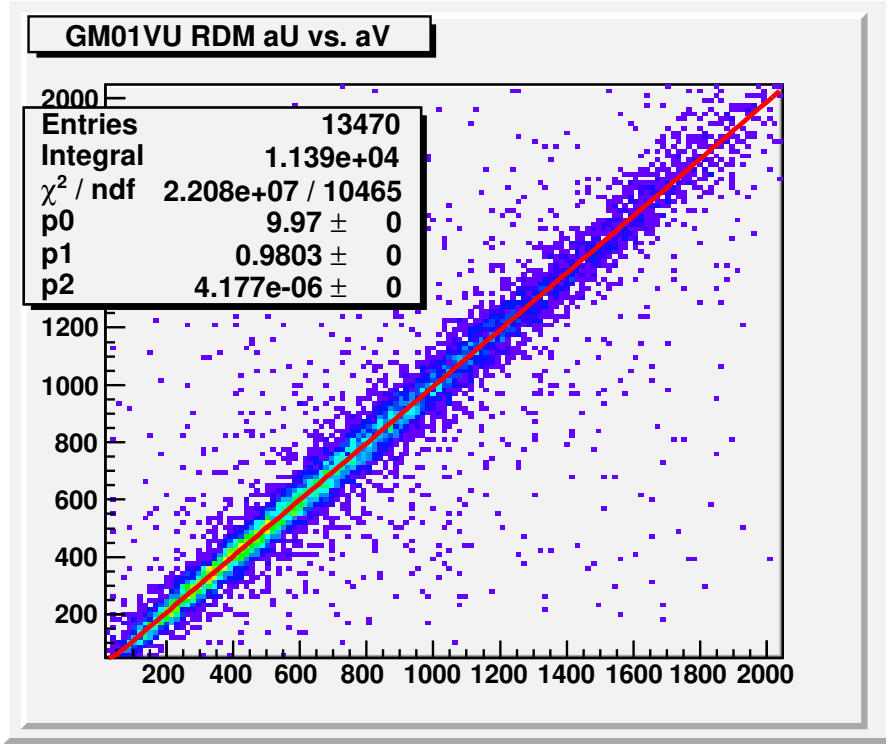
```
VolPlaneImpl(
    SurfaceType typ, double thickness_inner ,double thickness_outer,
    Vector3D u_val ,Vector3D v_val ,Vector3D n_val , . . . )
```

- `globalToLocal` turns to be insensitive to `Vector3D u_val ,Vector3D v_val`.

Amplitude Correlation

- Can be **exploited in Tracking**.
- As an **example**: amplitude correlation obtained in **COMPASS GEMs** (*after some calibration*):





Class definitions (in /opt/local/include)

- edm4hep:SimTrackerHit = (X, Y, Z, $\vec{P}$, E_{dep}, t, **cellID**, **pathLength**. . .)

  std::uint64_t **cellID**;
  float **EDep**; // energy deposited [GeV].
  float **time**; // time in lab frame [ns].
  float **pathLength**; // path length in sensitive material
  std::int32_t **quality**; // quality bit flag.
  ::edm4hep::Vector3d **position**; // position [mm].
  ::edm4hep::Vector3f **momentum**; // 3-momentum [GeV]

- edm4eic::RawTrackerHit = (iQ, iT, **cellID**)

  std::uint64_t **cellID**;
  std::int32_t **charge**; // ADC value
  std::int32_t **timeStamp**; // TDC value.

- edm4eic::TrackerHit = (**position**,**positionError**, E_{dep}, ΔE_{dep}, t, Δt, **cellID**)

  std::uint64_t **cellID**;
  ::edm4hep::Vector3f **position**; // Hit (cell) position [mm]
  ::edm4eic::CovDiag3f **positionError**; // Covariance Matrix
  float **time**; // Hit time [ns]
  float **timeError**; // Error on the time
  float **edep**; // Energy deposit in this hit [GeV]
  float **edepError**; // Error on the energy deposit [GeV]

(*Note:* ***cellID*** *comment omitted, because misleading imho.*)