中国科学院高能物理研究所
Institute of High Energy Physics
Chinese Academy of Sciences

Development of CEPC Drift Chamber Software

Weidong Li (IHEP)
on behalf of the CEPC DC software working group

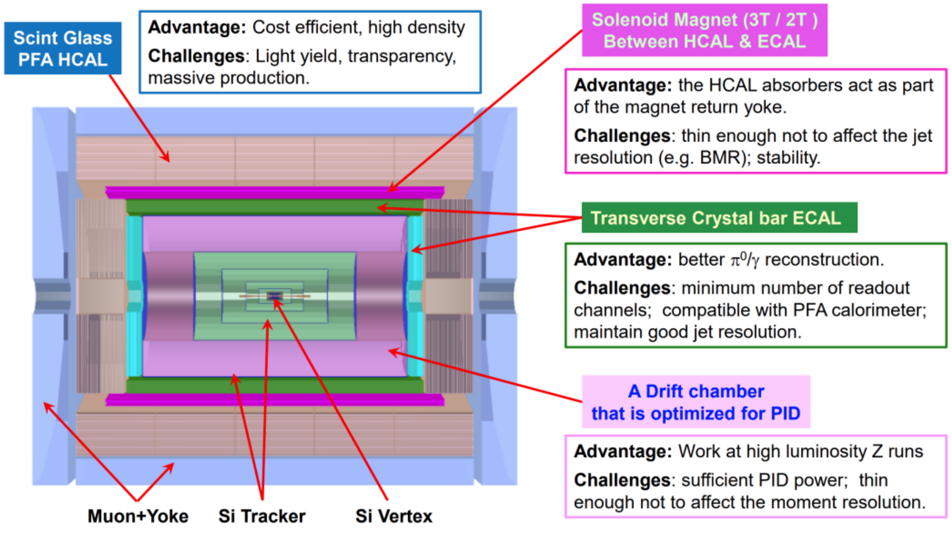
IAS workshop, Feb-13-2023

Contents

- Introduction
  - Detector design
  - Event data model
- Software implementations
  - Data processing flow
  - Detector simulation
  - Track reconstruction
- Summary

Detector Design

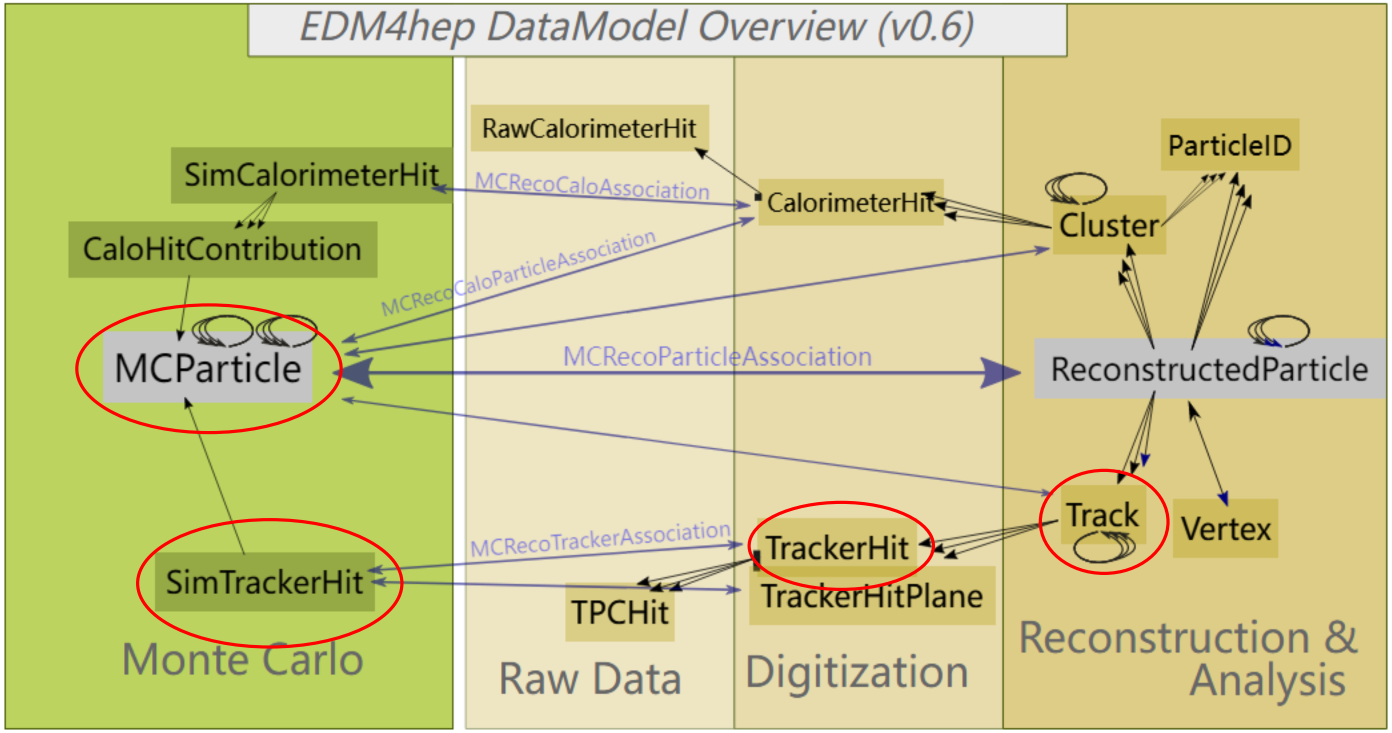
- The CEPC experiment mainly aims to precisely measure the property of the Higgs boson.
- Physics requirements: high track efficiency (~100%), momentum resolution (<0.1%), PID (2σ p/K separation at P < ~ 20 GeV/c), etc.
- For the 4th conceptual detector, silicon detector and drift chamber (DC) are designed to provide both tracking and PID for charged particles.
- Both detector design and physics potential studies needs strong support of simulation and reconstruction software.



Half length	2980 *mm*
Inner and outer radius	800 to 1800 *mm*
# of Layers	100/55
Cell size	~10x10mm/18x18mm
Gas	He:C_4H_{10}=90:10
Single cell resolution	0.11 *mm*
Sense to field wire ratio	1:3
Total # of sense wire	81631/24931
Stereo angle	1.64~3.64 *deg*
Sense wire	Gold plated Tungsten ϕ=0.02*mm*
Field wire	Silver plated Aluminum ϕ=0.04*mm*
Walls	Carbon fiber 0.2 *mm*(inner) and 2.8 *mm*(outer)

Event Data Model (1)

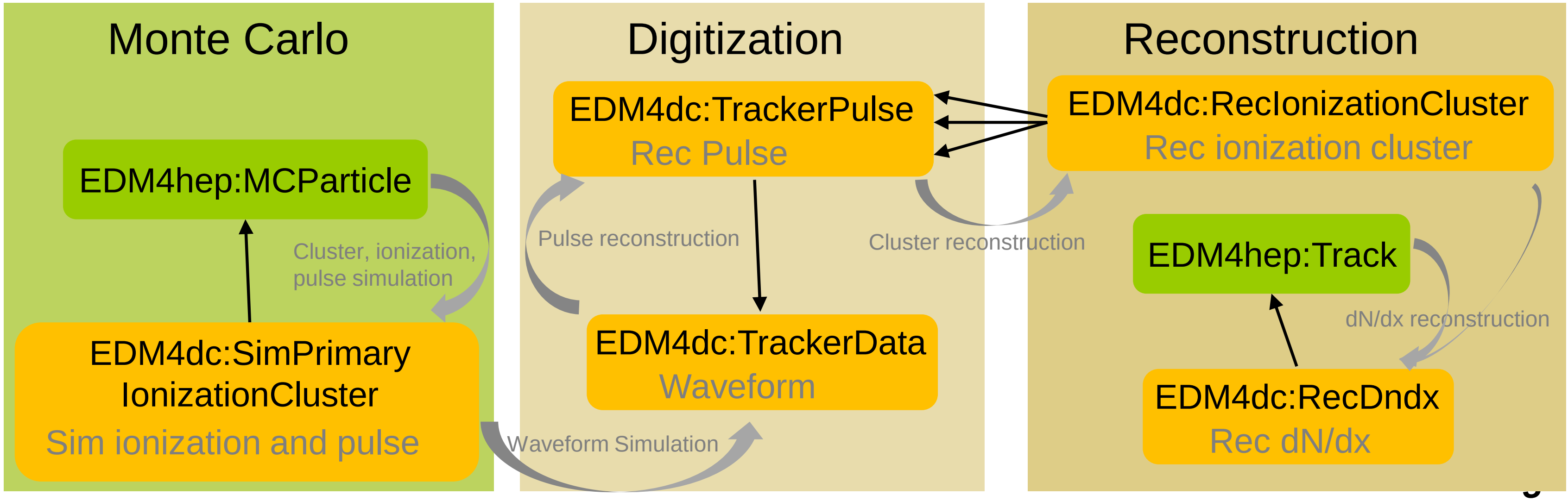
- EDM4hep is the common event data model (EDM) being developed for the future experiments like CEPC, CLIC, FCC, ILC, etc.



- EDM4hep describes event objects created at different data processing stages and also reflects the relationship between them.
- For the drift chamber, MCParticle, SimTrackerHit, TrackHit, Track have been used since the begin of the software project.

Event Data Model (2)

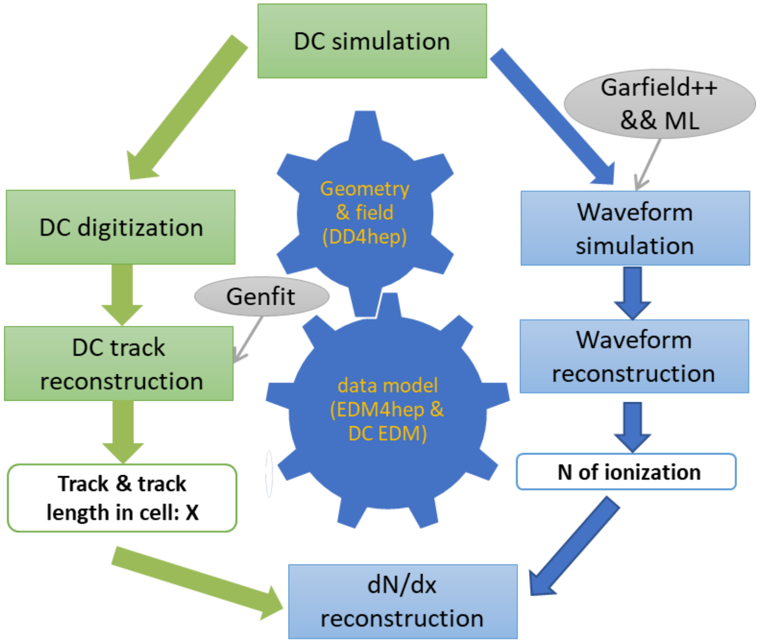
- As the development progressed, the previous versions of EDM appeared not able to fit all the requirements brought by newly added detector like the CEPC' s drift chamber.
- Due to the strong flexibility of EDM4hep, TPCHit was extended to accommodate the new needs:
  - Discussions inside EDM4hep group and also with the IDEA-CEPC drift chamber working group
  - By using the upstream mechanism of PODIO, a common EDM was implemented for both TPC and drift chamber



Data Processing Flow

❖ Detector simulation

- Geant4 is employed to simulate particle's propagation (including particle decay) in the detector, interaction with detector material, etc.
- TrackerHeed (from Garfield++) is used to simulate ionization process of charged particles (e, μ, π, K, p, ...) when they pass through the drift chamber.
- Garfield++ was integrated with the CEPCSW to simulate but its extreme computation intensiveness makes it impossible



Drift chamber simulation and reconstruction flow

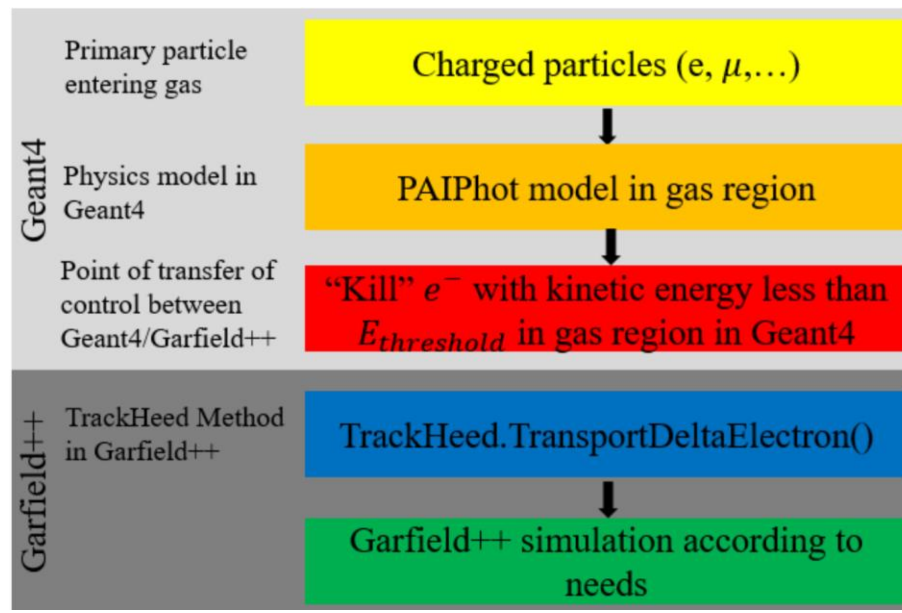
- Machine learning (ML) based simulation: training data is created by Garfield++ and ML model is be executed to replace Garfield++ in the detector simulation.

❖ Reconstruction

- Extrapolating the track segment found in the inner silicon detector to drift chamber, collecting the hits on the path, and applying a Kalman Fit to the found track .
- dN/dx reconstruction: waveform reconstruction + path length calculation

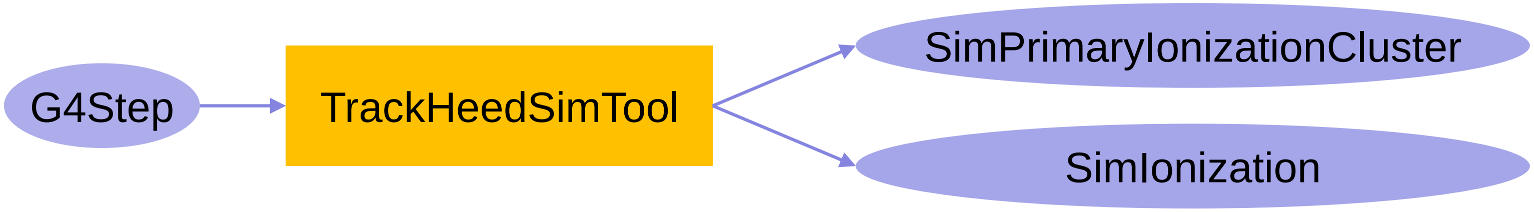
Simulation of Gaseous Detector (1)

- Since Geant4 can not be used to simulate the ionization process properly (arXiv:2105.07064), Garfield++ becomes a common tool for precise ionization simulation.
- “Interfacing Geant4, Garfield++ and Degrad for the Simulation of Gaseous Detectors” studied how to combine Geant4 and Garfield++ to get correct energy deposition or total number of ionized electrons (adopted by COMET experiment)
- Method:
  - Geant4 PAI (Photo Absorption Ionization) model to simulate primary or secondary ionization
  - TrackHeed (from Garfield++) to simulate ionization from residual delta electron



Simulation of Gaseous Detector (2)

- TrackHeedSimTool (Gaudi tool) was implemented for using TrackHeed to do ionization simulation
  - Input: G4Step information (particle type, initial position and momenta, ionization path length)
  - Use TrackHeed (used by Garfield++) to simulate one step length (or multi-step length for speed up) ionization (new API contributed to Garfield++ PR)
  - Output: primary and total ionization information (contains position, time, cell id), saved in EDM
  - The kinetic energy of G4Track will be updated according to the energy loss in the ionization
  - Non-uniform magnetic field can be handled easily

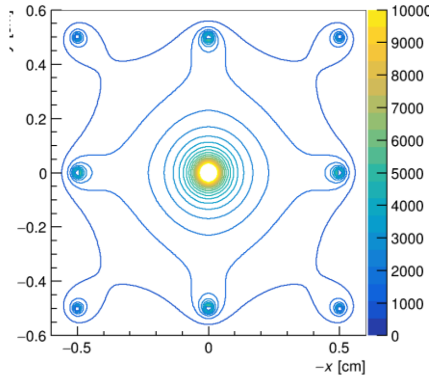


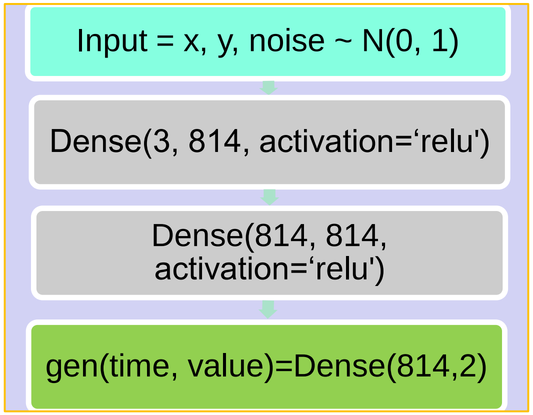
- Garfield++ is precise but extremely time consuming, it takes $\mathcal{O}(1)$ to $\mathcal{O}(10)$ seconds to simulate an electron (a few hours for one track)

ML-based dN/dx Simulation

- Garfield++ simulation events:
  - 350k single electron event with electron position uniformly distributed $1 \times 1\ \mathrm{cm}^2$ drift chamber cell
  - Gas: 50% He + 50 % C_4H_{10}
  - Center signal wire (2000 V), eight field wires (0 V)
- Model: fully connected neural network
  - Consist of input, hidden, and output layers
  - Input: Local x and y positions of ionized electron, N(0,1) distribution noise
  - Output: peak time and value of the single-electron signal
- Loss: a differentiable two-sample test statistics based on smoothed k-nearest neighbor tests (arXiv:1709.01006) between real data and generated data

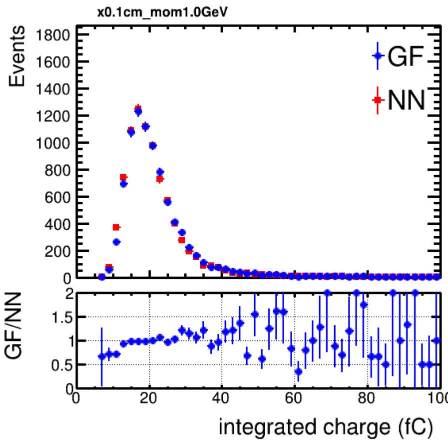
Electric field contour plot

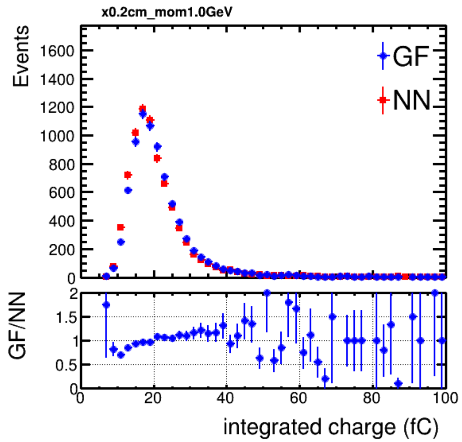


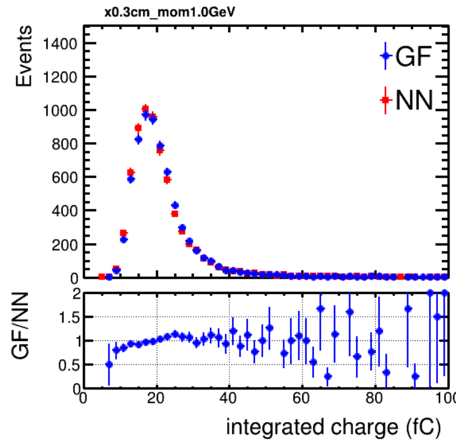


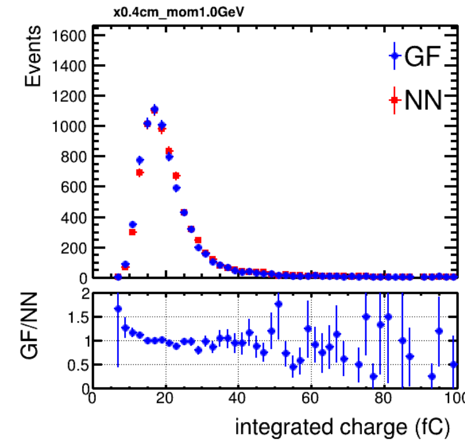
Mini-batch training: using Faiss (fast similar search) to get a batch (1024) of closet data (L2 distance)

dN/dx Simulation Performance (1 GeV e-)







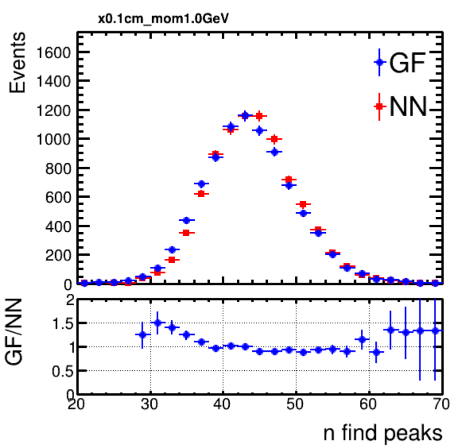


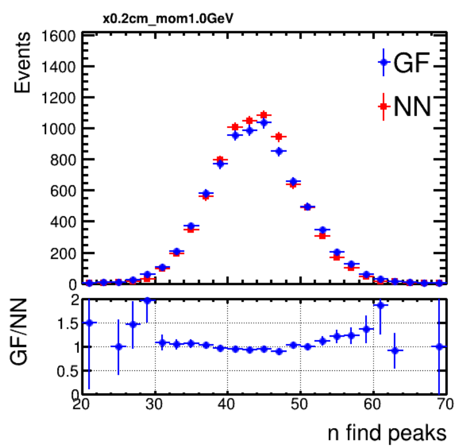
x=0.1 cm

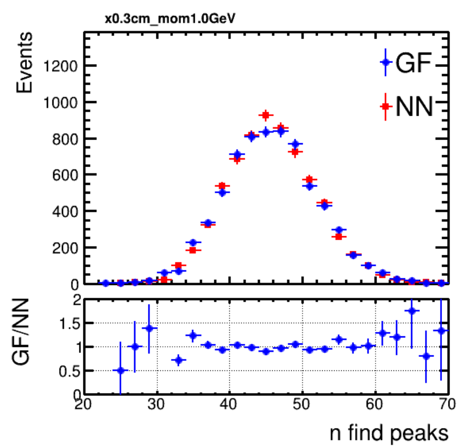
x=0.2 cm

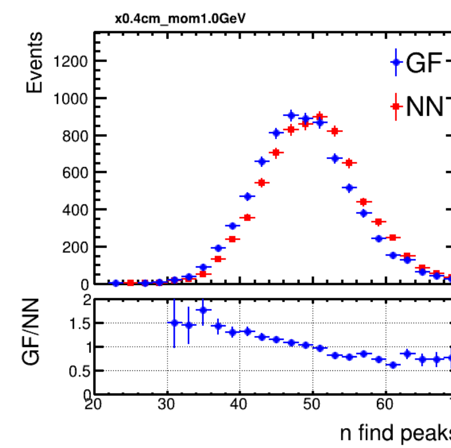
x=0.3 cm

x=0.4 cm



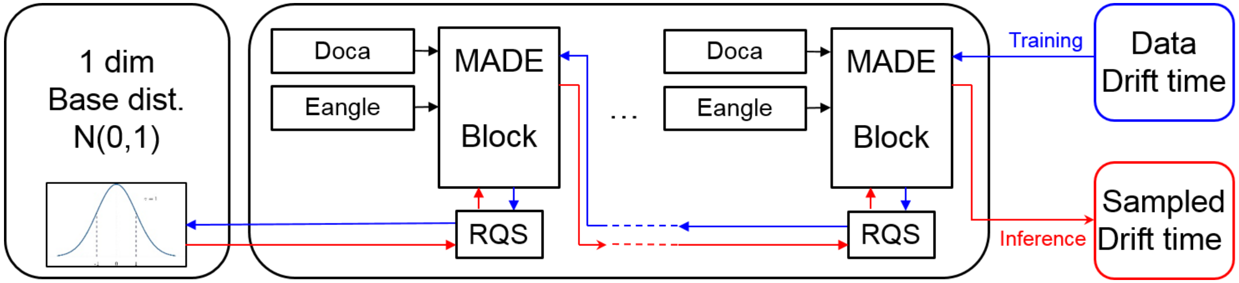




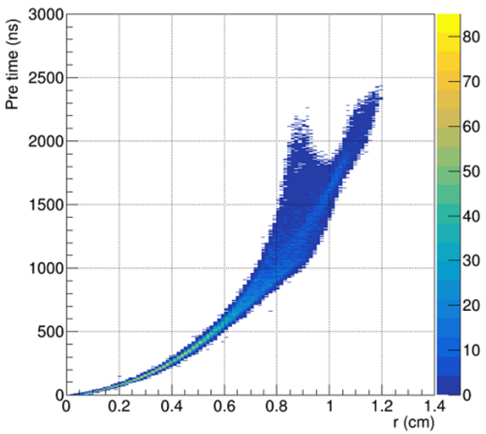


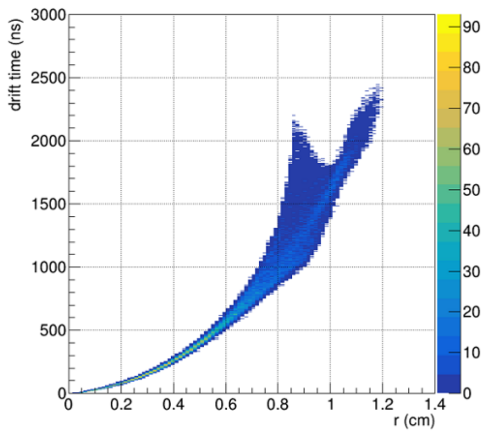
- Checked total integrated charge and number of found peaks(using scipy.signal.find_peaks)
- Good agreement in general (a little bias for the number of found peaks for x=0.4 cm. Could be improved in future), Similar results for other energy points (0.5, 5, and 10 GeV)

Simulation of Drift Time (3)



- Normalizing Flow network was adopted
  - A similar model to CaloFlow is used, RQS (for transformation)+MADE block (for the parameters of RQS)





- By using the NN model, the drift time can be simulated as well:
  - Drift time versus distance of closest approach (Doca) between a track and the signal wire

Performance Validation with BESIII Data (1)

- To investigate the possibility of applying ML to simulation, the real data from the BESIII experiment was used evaluate the performance of the chosen neural network.
- Radiative bhabha events were selected to study the simulation of drift time in the chamber cell
  - X-T relation: the Doca versus drift time

Prepare experimental data for training

↓

Learn drift time distribution: $D_{drift\ time} = f(doca, eangle, layer)$

↓

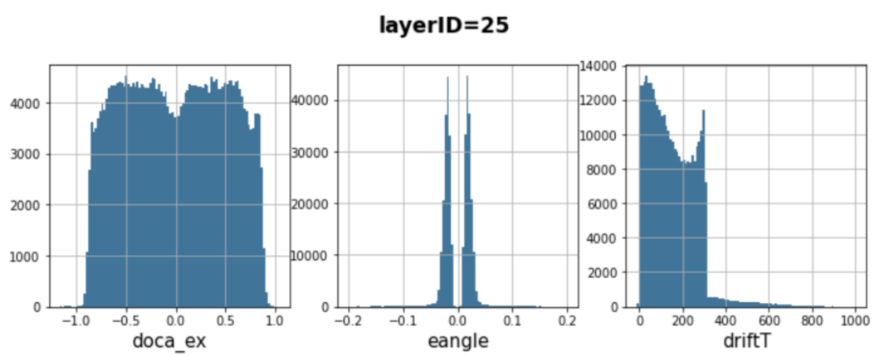
Check $D^{data}_{drift\ time}$, $D^{NN\ sim.}_{drift\ time}$

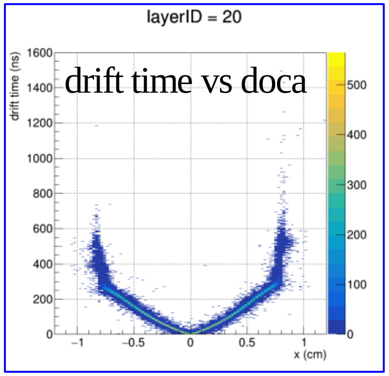
↓

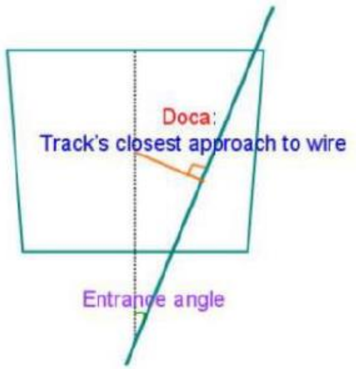
check the X-T cure: C^{data}_{X-T} , $C^{NN\ sim.}_{X-T}$

↓

check spatial resolution: using C^{data}_{X-T} , $C^{NN\ sim.}_{X-T}$

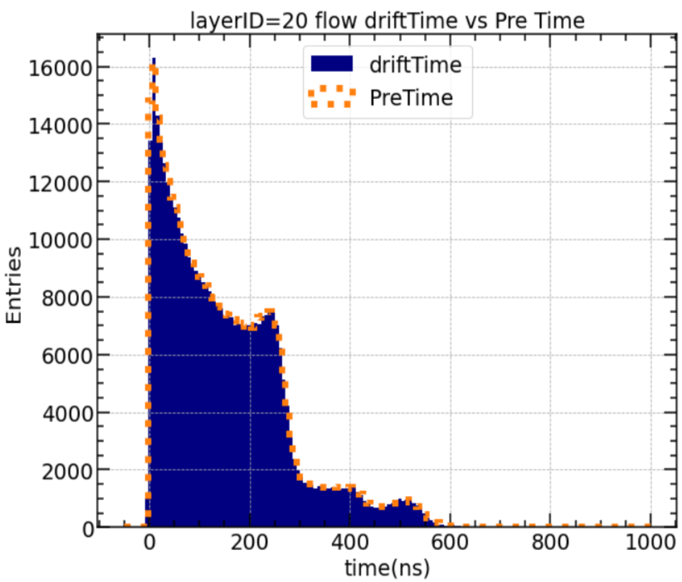


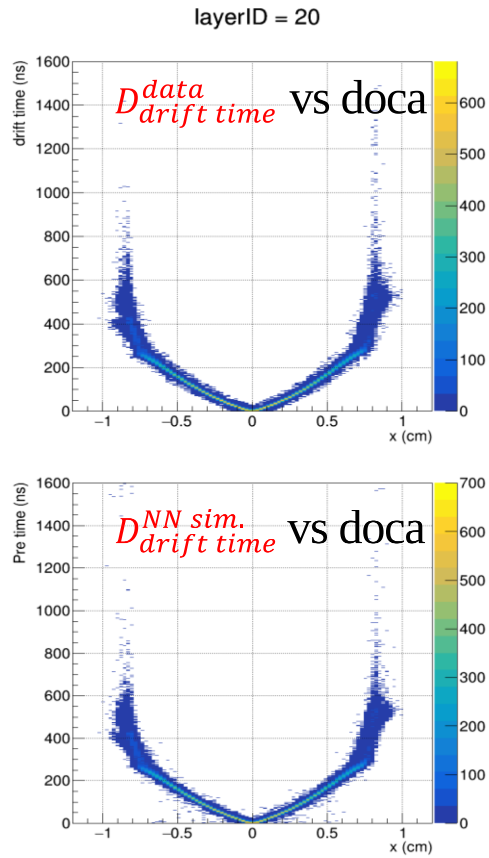




Performance Validation with BESIII Data (2)

- Comparison of drift time distributions between real data and ML-based simulation



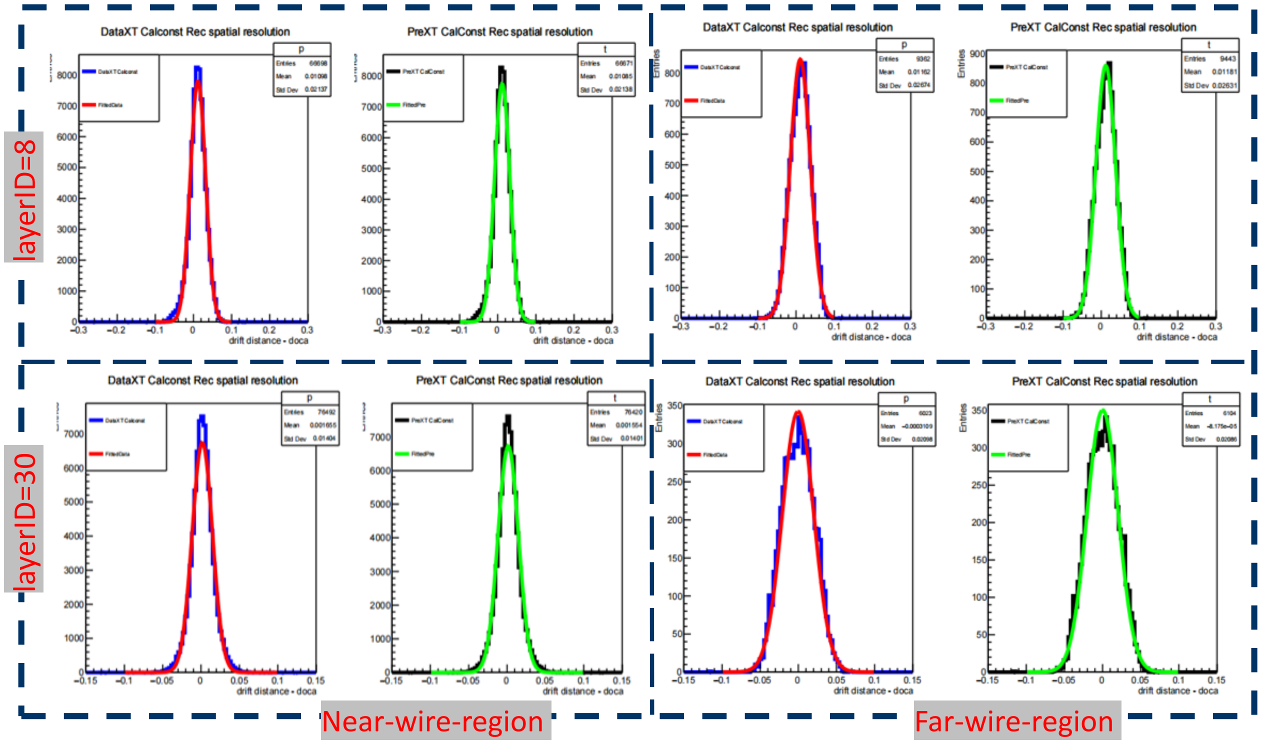


- Prepare experimental data for training
- Learn drift time distribution: $D_{drift\ time} = f(doca, eangle, layer)$
- Check $D^{data}_{drift\ time}$, $D^{NN\ sim.}_{drift\ time}$
- check the X-T cure: C^{data}_{X-T} , $C^{NN\ sim.}_{X-T}$
- check spatial resolution: using C^{data}_{X-T} , $C^{NN\ sim.}_{X-T}$

- Good agreement was found

Performance Validation with BESIII Data (3)

- Checked the spatial resolution of reconstructed tracks, using C_{X-T}^{data} and $C_{X-T}^{NN\,sim.}$ (calibration constants)



layerID	Near-wire-region (C_{X-T}^{data} v.s. $C_{X-T}^{NN\,sim.}$)		Far-wire-region (C_{X-T}^{data} v.s. $C_{X-T}^{NN\,sim.}$)	
	mean	sigma	mean	sigma
8	0.01096/0.01085	0.02137/0.02138	0.01162/0.01181	0.02674/0.02631
30	0.001055/0.001564	0.01404/0.01401	-3.1e-4/-5.1e-5	0.02060/0.02086

Prepare experimental data for training

↓

Learn drift time distribution: $D_{drift\ time} = f(doca, eangle, layer)$

↓

Check $D_{drift\ time}^{data}$, $D_{drift\ time}^{NN\,sim.}$

↓

check the X-T cure: C_{X-T}^{data}, $C_{X-T}^{NN\,sim.}$

↓

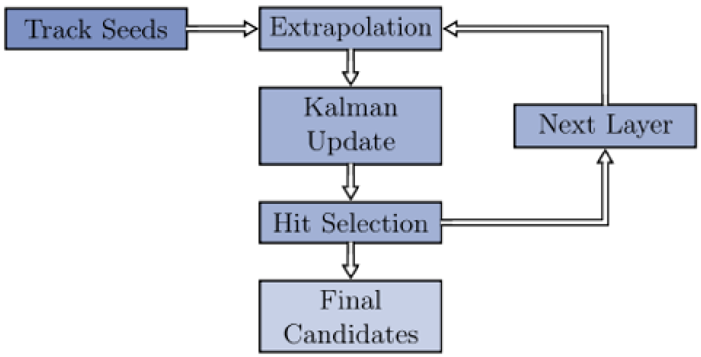
check spatial resolution: using C_{X-T}^{data}, $C_{X-T}^{NN\,sim.}$

- Consistent spatial resolution results obtained for C_{X-T}^{data} and $C_{X-T}^{NN\,sim.}$
- The NN can learn the drift time from the data

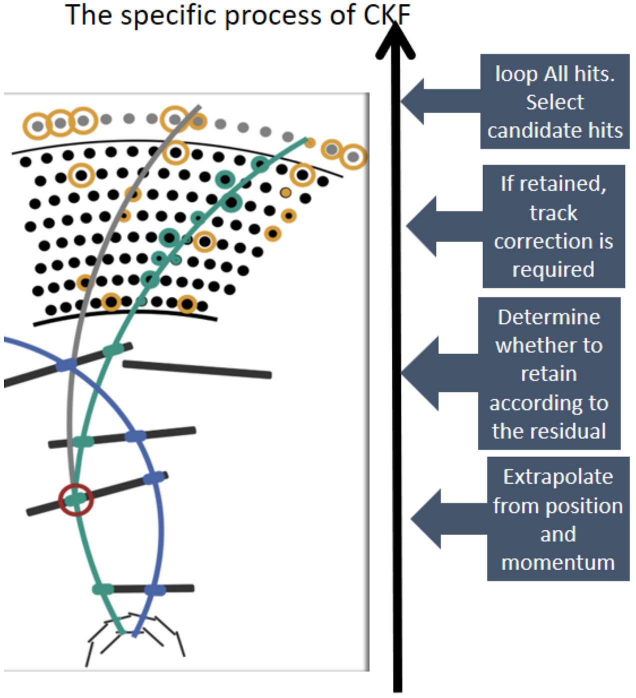
Track Reconstruction

- Tracking with Combinatorial Kalman Filter (CKF) method
  - Combining track recognition and track fitting
  - Used by many high energy physics experiments
- Track finding with CKF in the drift chamber
  - Implementation: track segments reconstructed in the silicon dectctor, called seeds, are extrapolated to the DC and all the DC hits belonging to the track are collected
- Current progress:
  - Managed to port the Belle II code to the CEPCSW
  - Performance validation is ongoing

Basic procedure behind the CKF

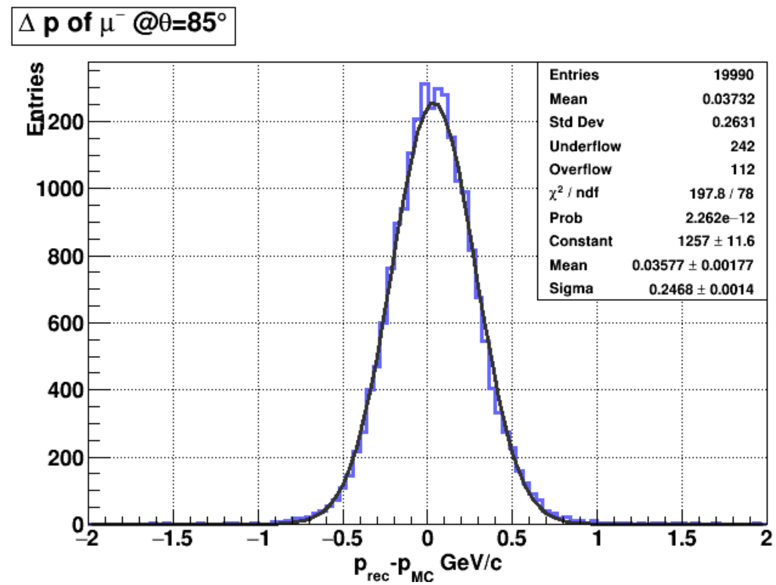


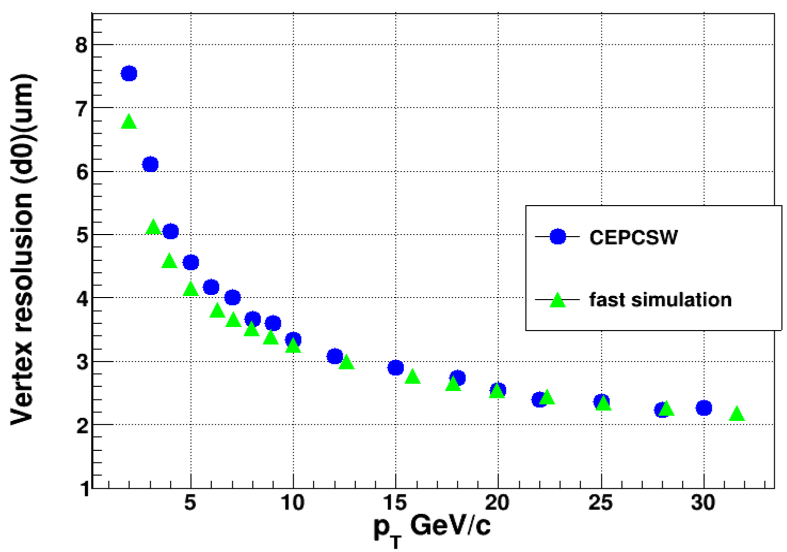
The specific process of CKF

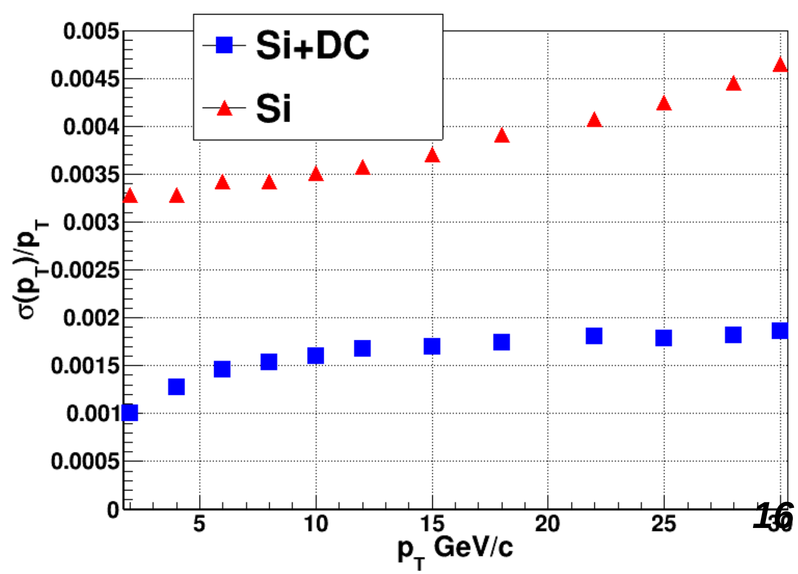


Tracking Performance (1)

- Tack recognition using the MC truth information was also implemented
- Genfit is adopted as the track fitter
  - using kalman filter method
  - non-uniform magnetic field field and material effects are taken into account
- Momentum/vertex resolution
  - Particle gun : mu- , 100 GeV@85°

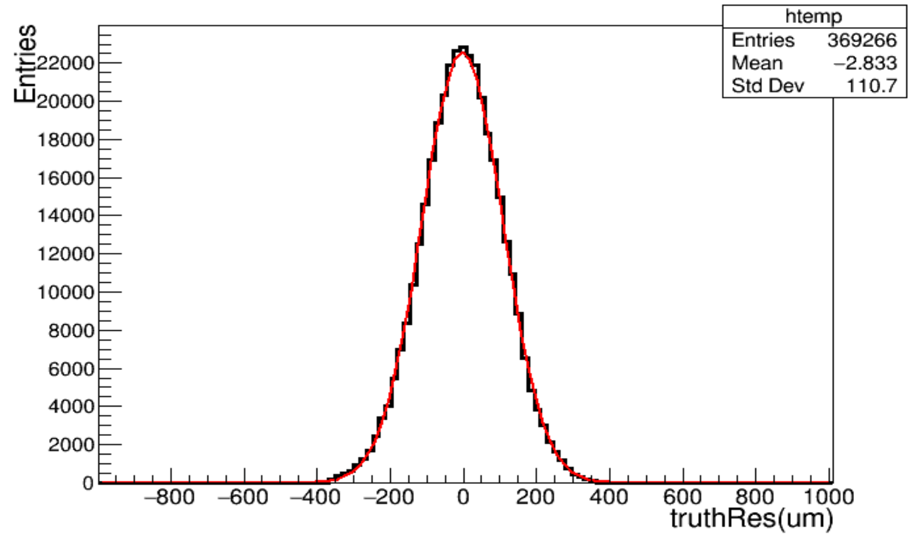


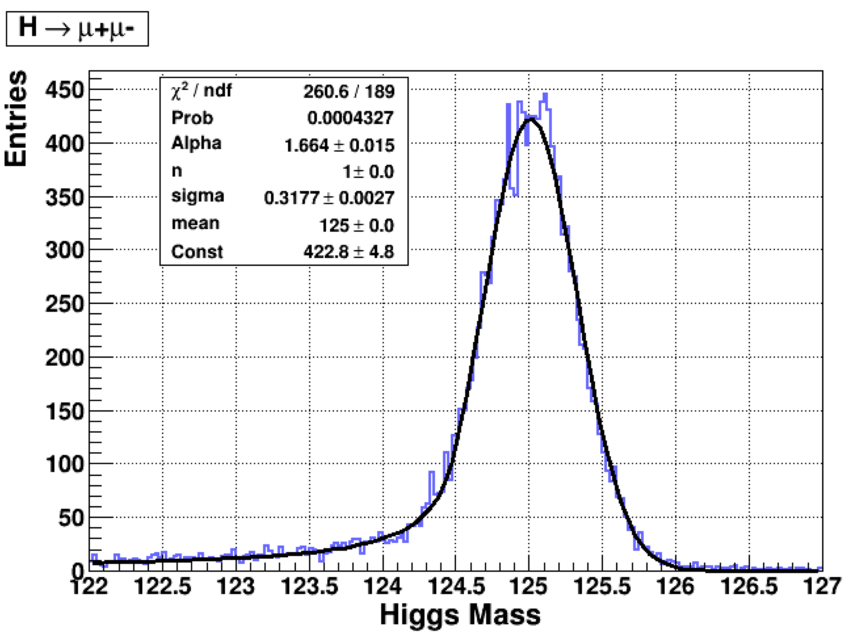


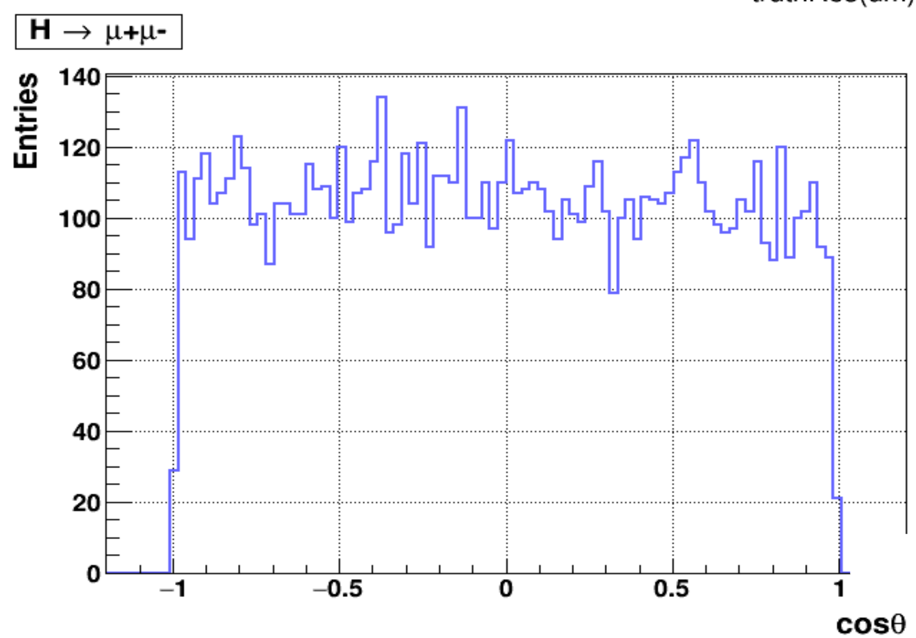


Tracking Performance (2)

- Spatial resolution
  - Spatial resolution: 110 um
  - Consistent with the value set in the simulation
- Events with two muons :
  - $e^+e^- \rightarrow ZH\,, H \rightarrow \mu^+\mu^+$







Summary

- As part of the Key4hep project, event data model of TPC detector was extended to accommodate the requirements from cluster counting studies.
- In CEPCSW, Garfield++ was integrated with the Geant4 to perform a precise simulation for the drift chamber.
- Machine learning based simulation was developed to model the response of the chamber cell.
- Tracking algorithm was implemented by reusing the code of Belle II and performance validation is still in progress.

Thank You！

谢谢