BESIII track reconstruction algorithm based on machine learning

Xiaoqian Jia[1] , Xiaoshuai Qin[1] , Teng Li[1] , Xingtao Huang[1] , Xueyao Zhang[1] , Yao Zhang[2] and Ye Yuan[2]

1. Shandong University, Qingdao

2. Institute of High Energy Physics, Beijing

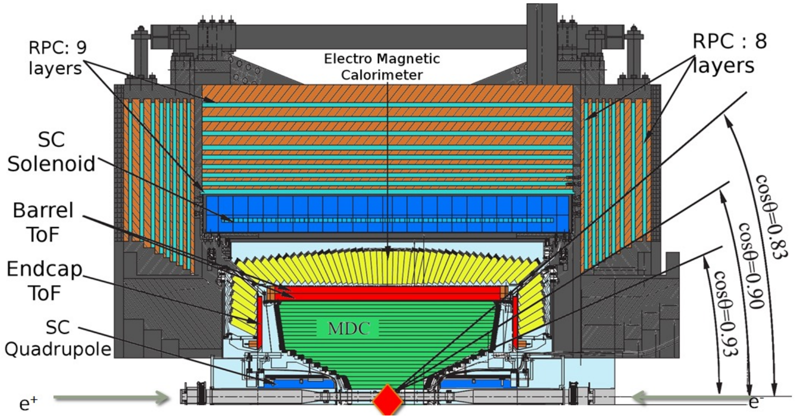
26th International Conference on Computing in High-Energy and Nuclear Physics
May 9, 2023

Outline

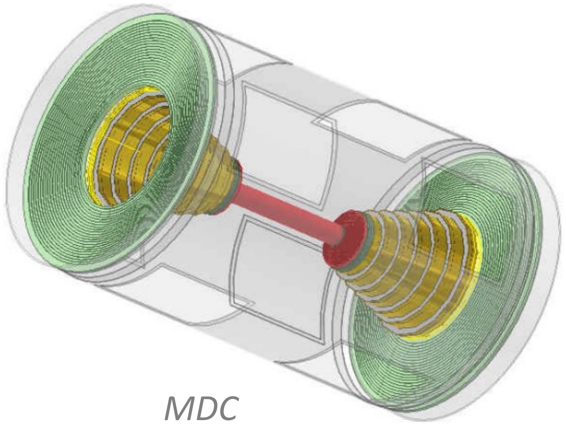
01 BEPCII & BESIII

- Beijing electron-positron collider (BEPCII)
  - Peak luminosity : 10^{33} cm^{-2} s^{-1}
  - CMS: 2.0 - 4.95 GeV, τ -charm region
  - World's largest J/ψ dataset : 10 billion
- Beijing Spectrometer (BESIII)
  - Study the electroweak and strong interactions
  - Search for new physics
- Main Drift Chamber (MDC)
  - 43 sense wire layers
  - dE/dx resolution : 6%
  - Momentum resolution : 0.5%@1GeV/c

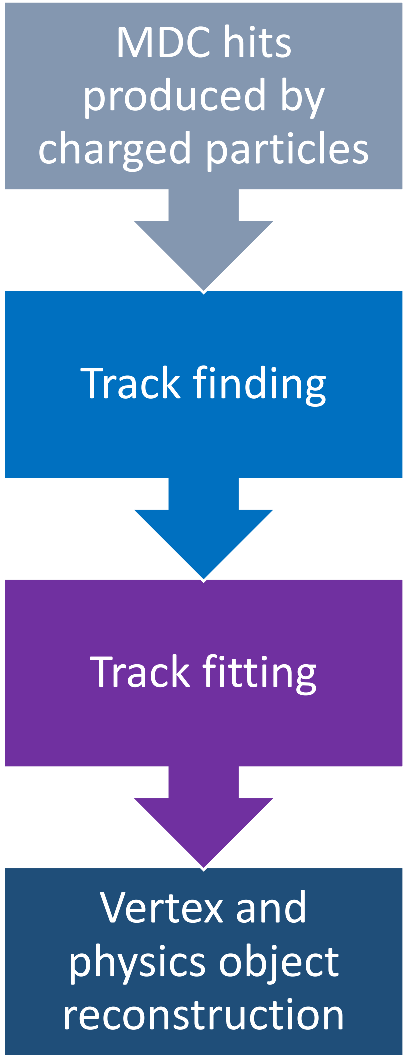
Aerial view of the BEPCII



BESIII detector


MDC

01 Traditional tracking of BESIII drift chamber



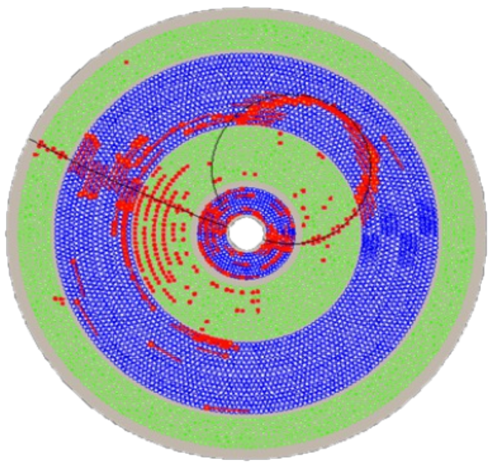
- Identify measurements to individual tracks
  - Global method : Hough transform (HOUGH)

    Affect by energy loss and overlapping track
  - Local method : Template matching for segment (PAT)

    Sensitive to wire inefficiency, layer arrangement and momentum

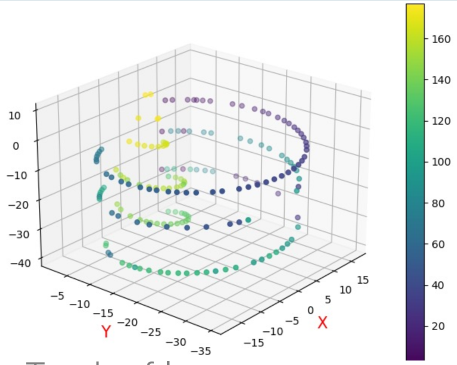
    Seeding and road following (TCurlFinder)

    Affect by noise or background alone the track path
- Estimate the track parameters
  - Kalman filter
- Estimate charged particles properties
  - Momentum and direction
  - Charge

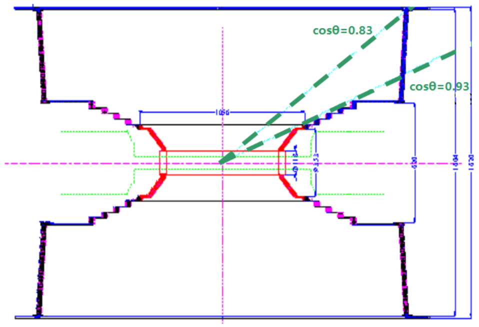


01 Motivation

- Further optimizations: Increase the tracking efficiency and performance for special events
  - Low transverse momentum
  - Large dip angle
  - Secondary vertex
- New Challenge: Higher Background and noise with the upgrade of BEPCII
  - Noise hit resistance
- But the optimization of the traditional tracking algorithm could be very challenging
- Goals of this study
  - Explore the new tracking method with novel technologies
    - GNN, DBSCAN…
  - Develop experiment independent tracking with 2-D measurement (drift chamber) for other experiments (i.e. STCF, CEPC …)

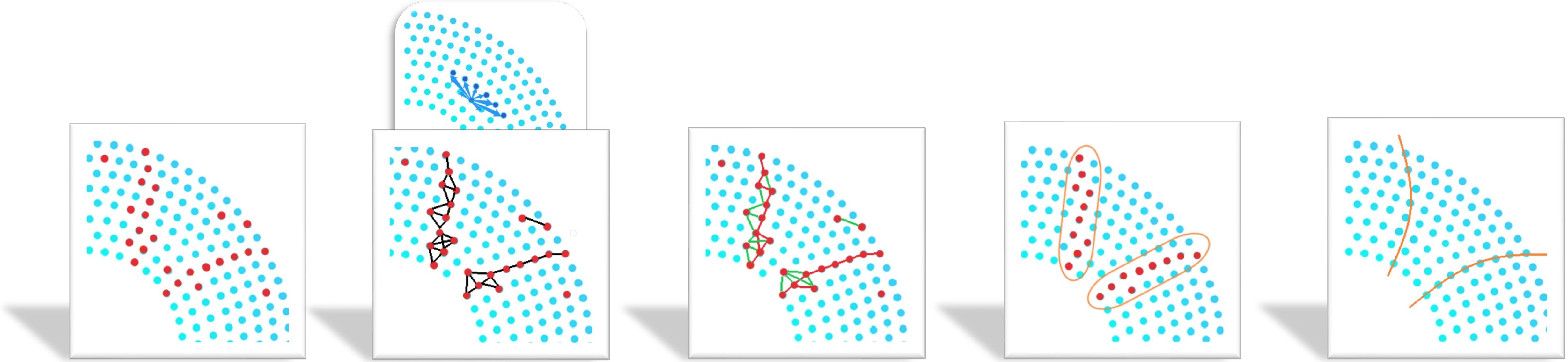


Track of low transverse momentum(51.8 MeV) particle



ZR view of drift chamber

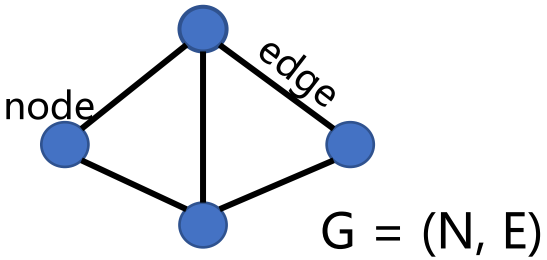
02 Methodology: workflow

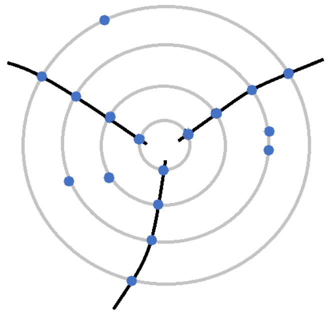


MDC hits produced by charged particles → Construct the graph based on the Pattern Map → Classify the graph edges by GNN → Cluster the selected hits → Track fitting

02 Graph Neural Network

- A type of neural network that are specifically designed to operate on graph-structured data
- Graph: nodes, edges
- Graph → Track
  - Nodes → Hits
  - edges → track segments





- GNN key idea: propagate information across the graph using a set of learnable functions that operate on node and edge features
- Graph Neural Network edge classifier
  - High classification score

    → *the edge belongs to a true particle track*
  - Low classification score

    → *it is a spurious or noise edge*

02 Graph construction

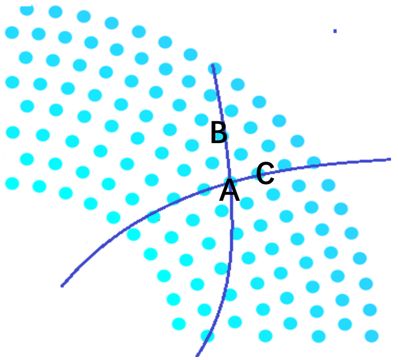
Pattern Map based on MC simulation

To reduce the number of fake edges during graph construction

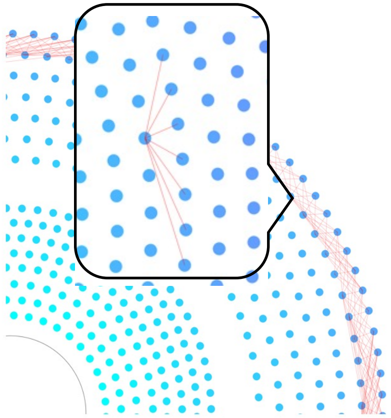
- Definition of valid neighbors
  - Hits on the same layer

    Two adjacent sense wires on the left and right
  - Hits on the next layer

    The collection of sense wires that could potentially represent two successive hits on a track



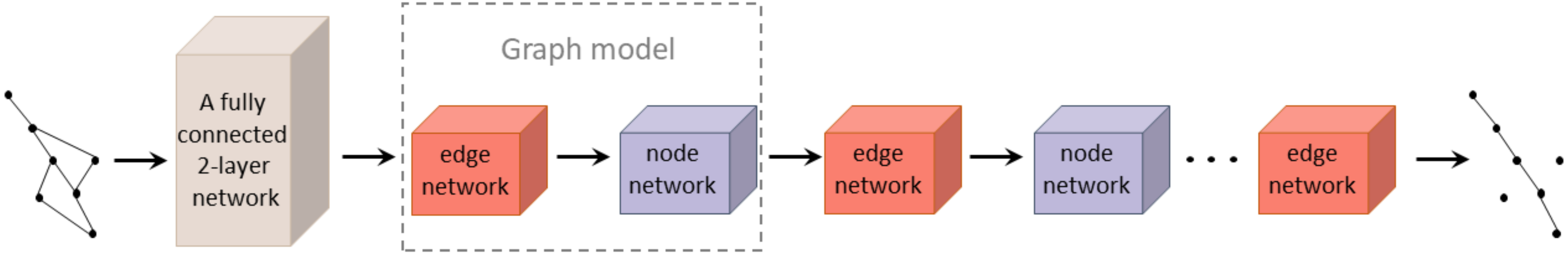
- MC sample used to build pattern map
  - Two million single tracks produced with BESIII offline software (BOSS)
  - 5 types of charged particles ($e^{\pm}$, $K^{\pm}$, $\mu^{\pm}$, $p^{\pm}$, $\pi^{\pm}$)
  - 0.05 GeV/c < P < 3 GeV/c
- Edge assignment based on Pattern Map
  - Hit with its neighbors on the same layer and next layer
  - Hit with its neighbors' neighbors on one layer apart
- To reduce the size of the graphs, the Pattern Map is further reduced based on a probability cut
- Graph representation
  - Node features (raw drift time, position coordinates r, ϕ of the sense wires), adjacency matrices, edge labels



A wire on layer13 and tits neighbors on layer14

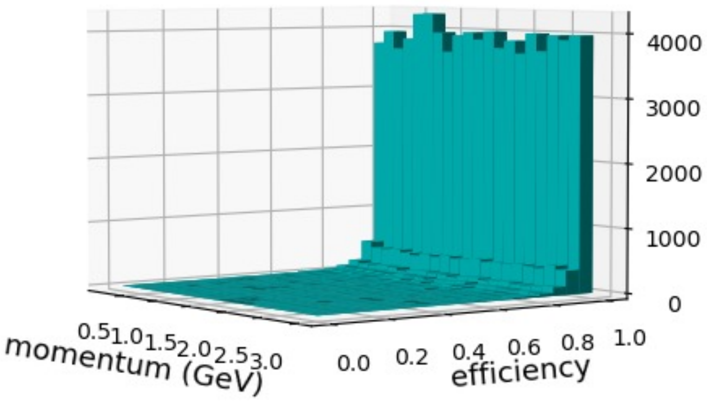
02 GNN edge Classifier based on PyTorch

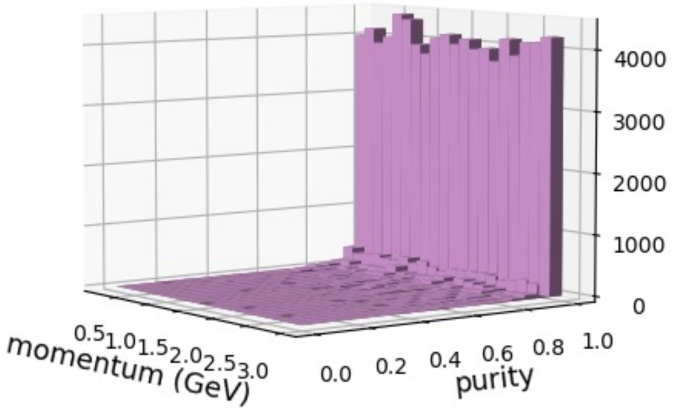
- Input network
  - Node features embedded in latent space
- Graph model
  - Edge network computes weights for edges using the features of the start and end nodes
  - Node network computes new node features using the edge weight aggregated features s of the connected nodes and the nodes’ current features
  - MLPs
  - 8 graph iterations
- Strengthen important connections and weaken useless or spurious ones



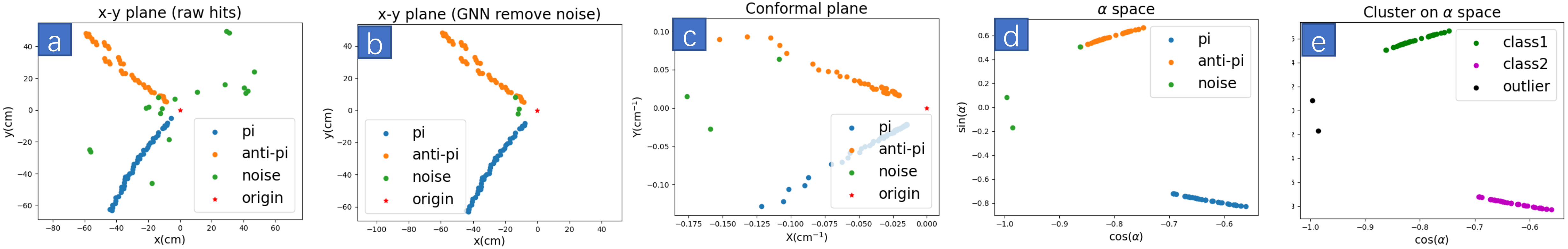
02 Performance of filtering noise

- Dataset
  - Single-particle ($e^{\pm}$, $K^{\pm}$, $\mu^{\pm}$, $p^{\pm}$, $\pi^{\pm}$) MC sample
  - 0.2 GeV/c < P < 3.0 GeV/c
  - Mixed with BESIII random trigger data as background (~45% hits)
  - Train: Validation: Test = 4: 1: 1
- Hit selection performance
  - The preliminary results show that GNN provides high efficiency and purity of hits selection
  - *Hit selection Efficiency* : $\frac{N_{signal}^{\text{predicted}}}{N_{signal}^{real}}$
  - *Hit selection Purity* : $\frac{N_{signal}^{\text{predicted}}}{N_{all}^{predicted}}$





02 Clustering of Tracks Based on DBSCAN



a) Original MC data sample

- $J/\Psi \rightarrow \rho^0 \pi^0 \rightarrow \gamma\, \gamma\, \pi^+ \pi^-$
- π^+, π^- : Pt (0.2GeV - 1.4GeV)

b) Remove noise via GNN

c) Transform to Conformal plane

- $X = \frac{2x}{x^2+y^2}$ $Y = \frac{2y}{X^2+y^2}$
- Circle passing the origin transform into a straight line

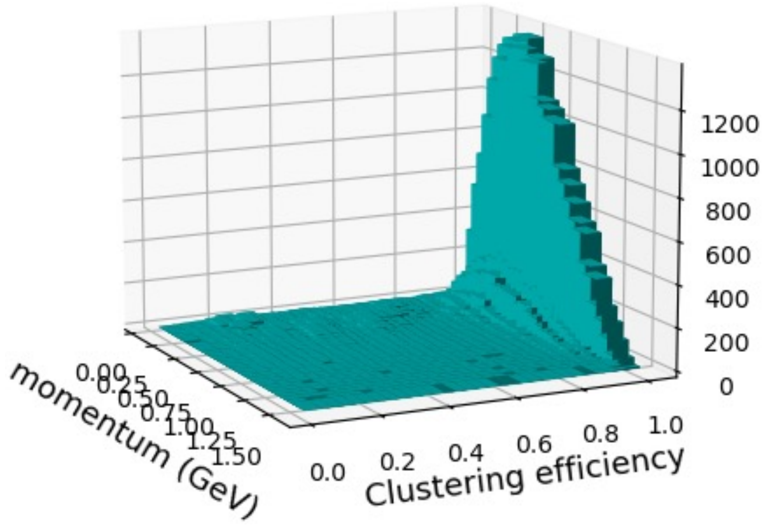
d) Transform to 'α' parameter plane

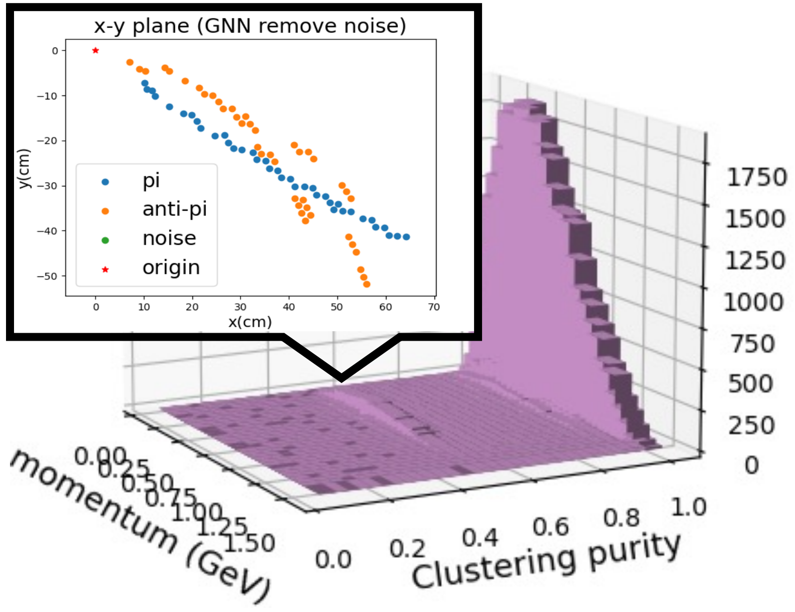
- Hits connected in the X-Y plane in a straight line
- α as the angle between the straight line and X axis
- The parameter space as cosα and sinα

e) DBSCAN clustering in 'α'parameter plane

- Density-Based Spatial Clustering of Application with Noise
- Hits in a cluster are considered to be in the same track

02 DBSCAN Performance

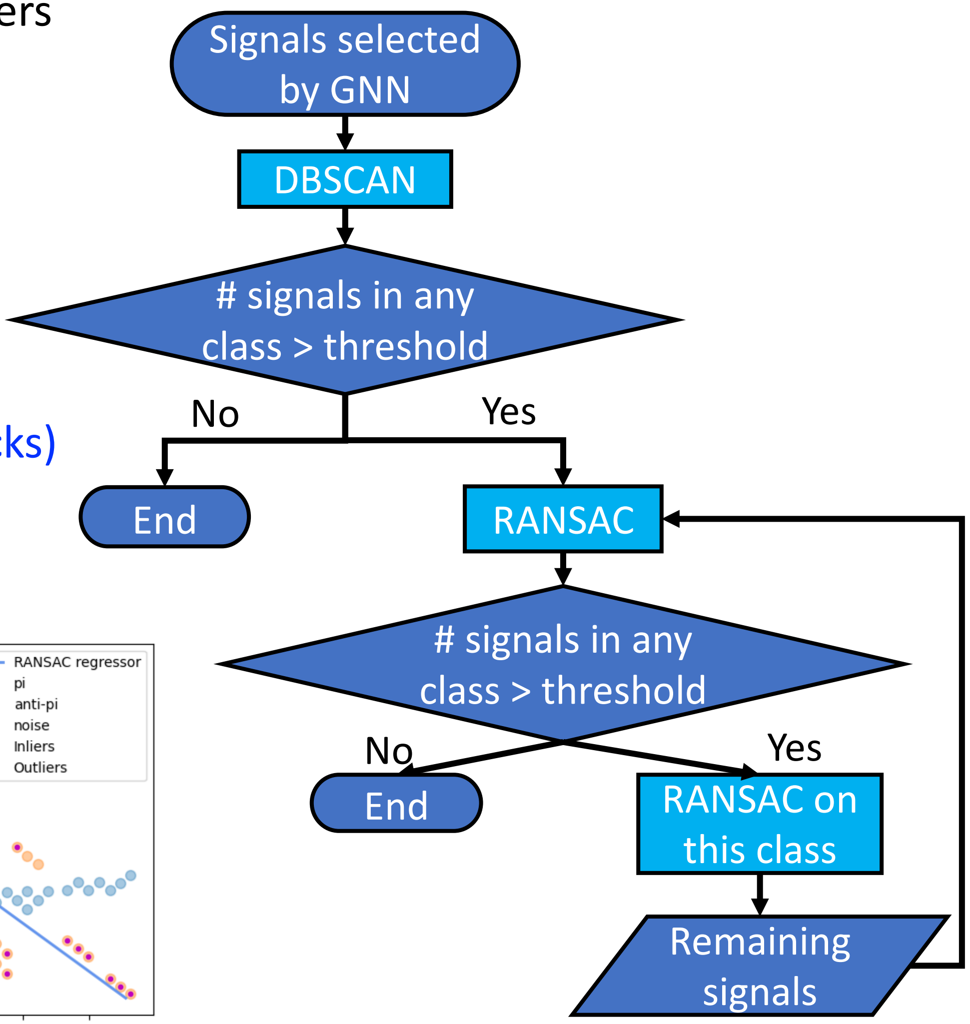


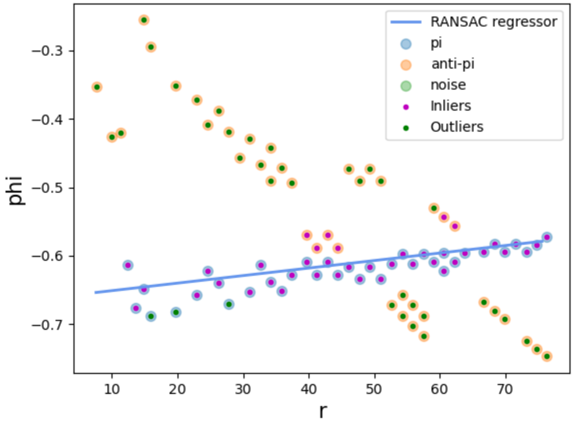


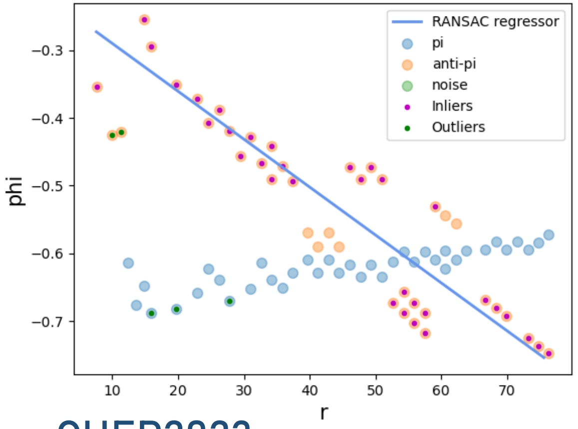
- DBSCAN can achieve high clustering efficiency ($\frac{N_{track}^{\mathrm{predicted}}}{N_{track}^{real}}$)
- An obvious bulge at the purity ($\frac{N_{cluster}^{\mathrm{real}}}{N_{cluster}^{all}}$) of about 0.5
  - Can not separate hits from the two very close tracks
  - It accounts for about 3.5%

02 Optimizations

- Random sample consensus (RANCAS)
  - Estimate a mathematical model from the data that contains outliers
  - Its good robustness to noise and outliers
  - Model can be specified
- RANCAS is triggered by the events that DBSCAN processing fails
  - Polar coordinate space
  - linear model (being optimized to a more suitable model for tracks)
  - Inliers → a track , outliers → other tracks
  - Stop condition: outliers < threshold

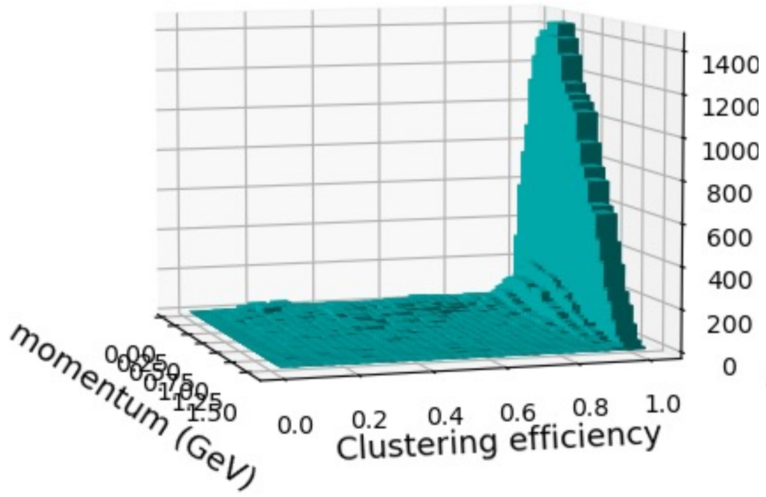


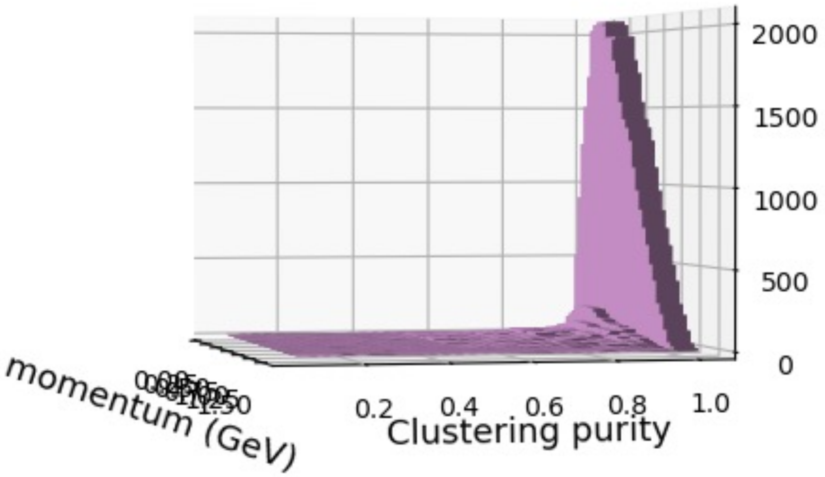


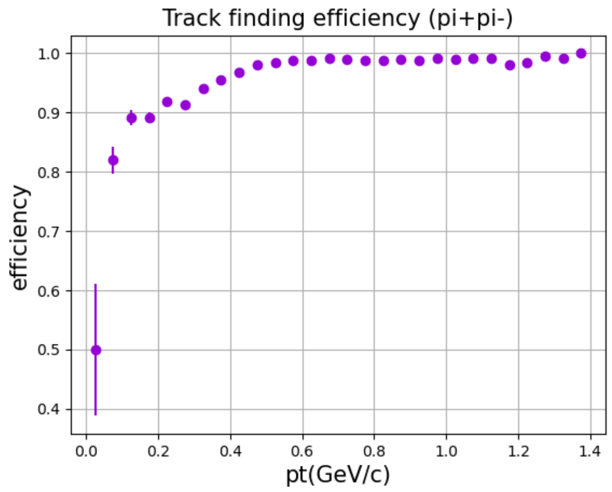


02 Results after Optimizations

- Removed bulges at purity
- Track finding efficiency
  - *track eff* = $\frac{N_{\text{rec tracks}}}{N_{\text{total tracks}}}$
  - Pt > 0.2 GeV/c , track eff > 90%
  - Pt > 0.45 GeV/c , track eff > 98%



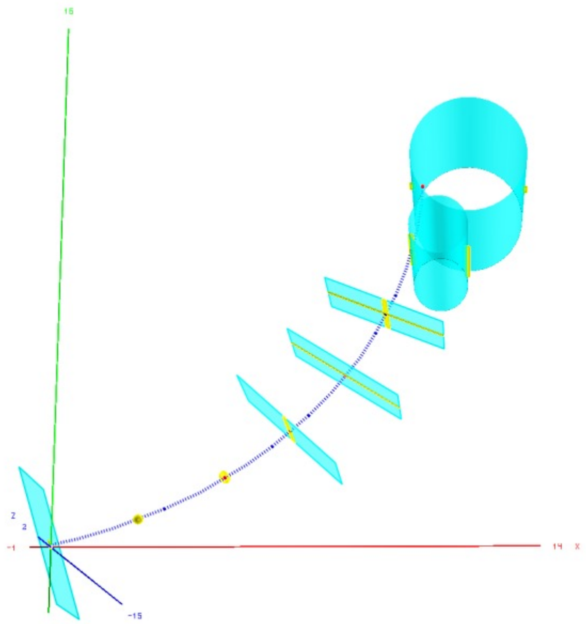




02 Track fitting

◆ Genfit2

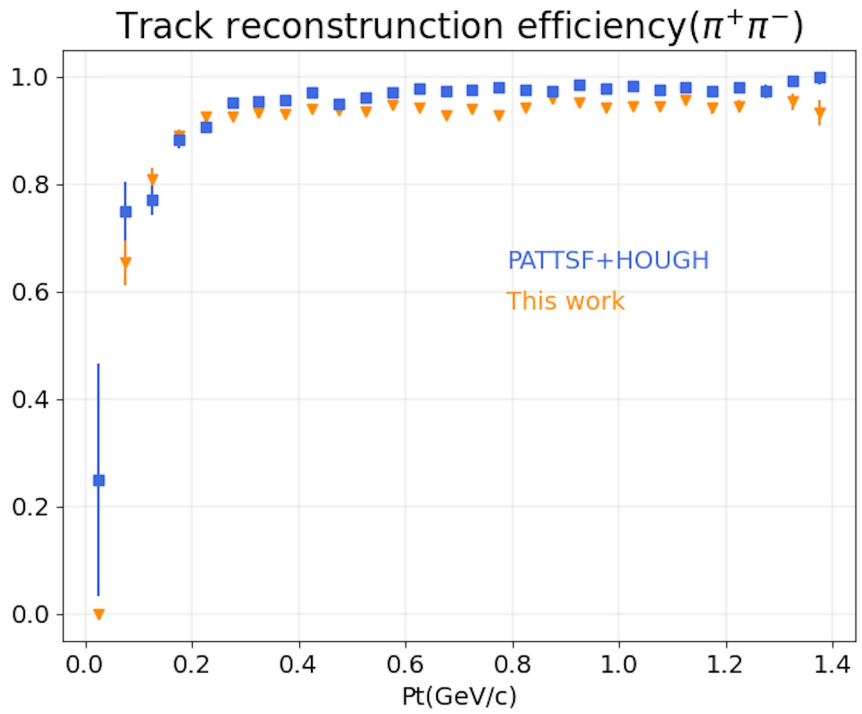
- A Generic Track-Fitting Toolkit
- Experiment-independent framework
- PANDA, Belle II, FOPI and other experiments
- Deterministic annealing filter (DAF) to resolving the left-right ambiguities of wire measurements

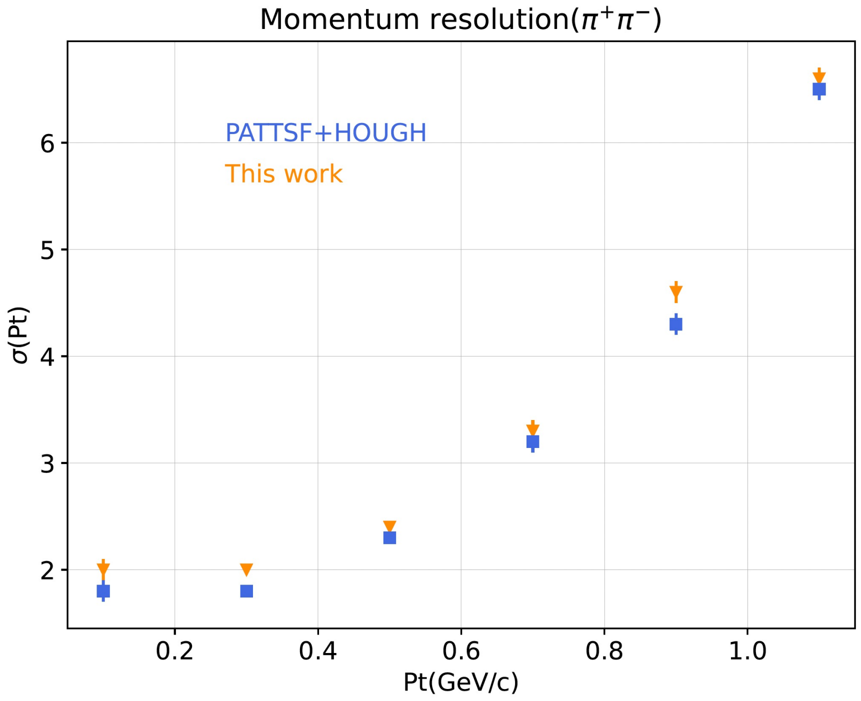


03 Preliminary Results

◆ Particle reconstructed performance

- $J/\Psi \rightarrow \rho^0 \pi^0 \rightarrow \gamma \gamma \pi^+ \pi^-$ from MC simulation
- The preliminary results presents promising performance





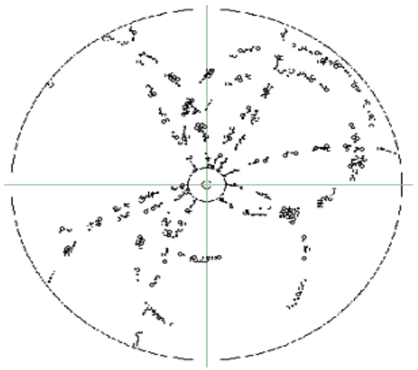
04 Summary

- A novel tracking algorithm prototype based on machine learning method at BESIII is under development
  - GNN to distinguish the hit-on-track from noise hits.
  - Clustering method based on DBSCAN and RANSAC to cluster hits from multiple tracks
- Preliminary results on BESIII MC data shows promising performance

Outlook

- Further optimization of the model is needed

  To improve performance for low PT tracks
- Performance verification concerning events with more tracks

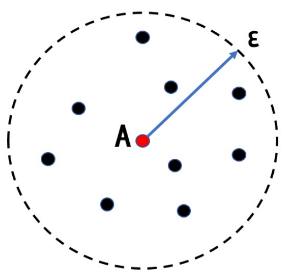


Thank you !

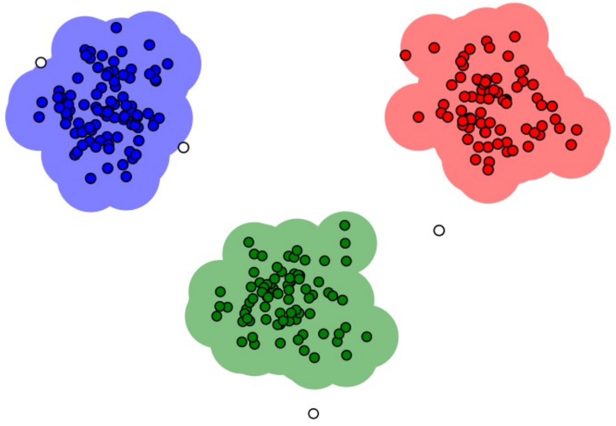
Xiaoqian Jia

Back up

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)



- A density-based clustering algorithm that can automatically discover clusters of arbitrary shapes and identify noise points
- Robust to outliers
- Not require the number of clusters to be told beforehand
- Parameter
  - Epsilon (radius of the circle to be created around each data point)
  - MinPoints (the minimum number of data points required inside that circle for that data point to be classified as a Core point)
  - Choose MinPoints based on the dimensionality (≥dim+1), and epsilon based on the elbow in the k-distance graph



RANSAC (Random Sample Consensus)

- Basic idea: randomly select a subset of data points, fit a model based on these points, and then judge whether the remaining data points belong to the inlier set by calculating their distances to the model
- Accurately estimate model parameters even in the presence of noise and outliers
- The specific steps
  - Randomly select a small subset of data, called the inlier set
  - Fit a model based on the inlier set
  - Calculate the distances between the remaining data points and the model, and classify these points as inliers or outliers based on a certain threshold
  - If the number of inliers reaches a preset threshold, the algorithm exits and the current model is considered good
  - If the number of inliers is not enough, repeat steps 1-4 until the maximum iteration times are reached
- Parameters such as threshold and iteration times need to be preset