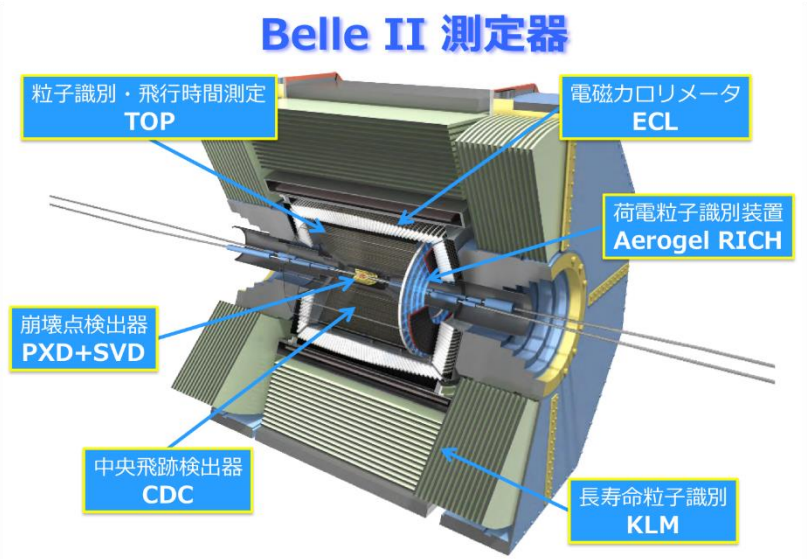
Karlsruher Institut für Technologie

29.08.2016

Optimization of the VXD-CDC Track Merger of Belle II

Institute for Experimental Nuclear Physics



Outline

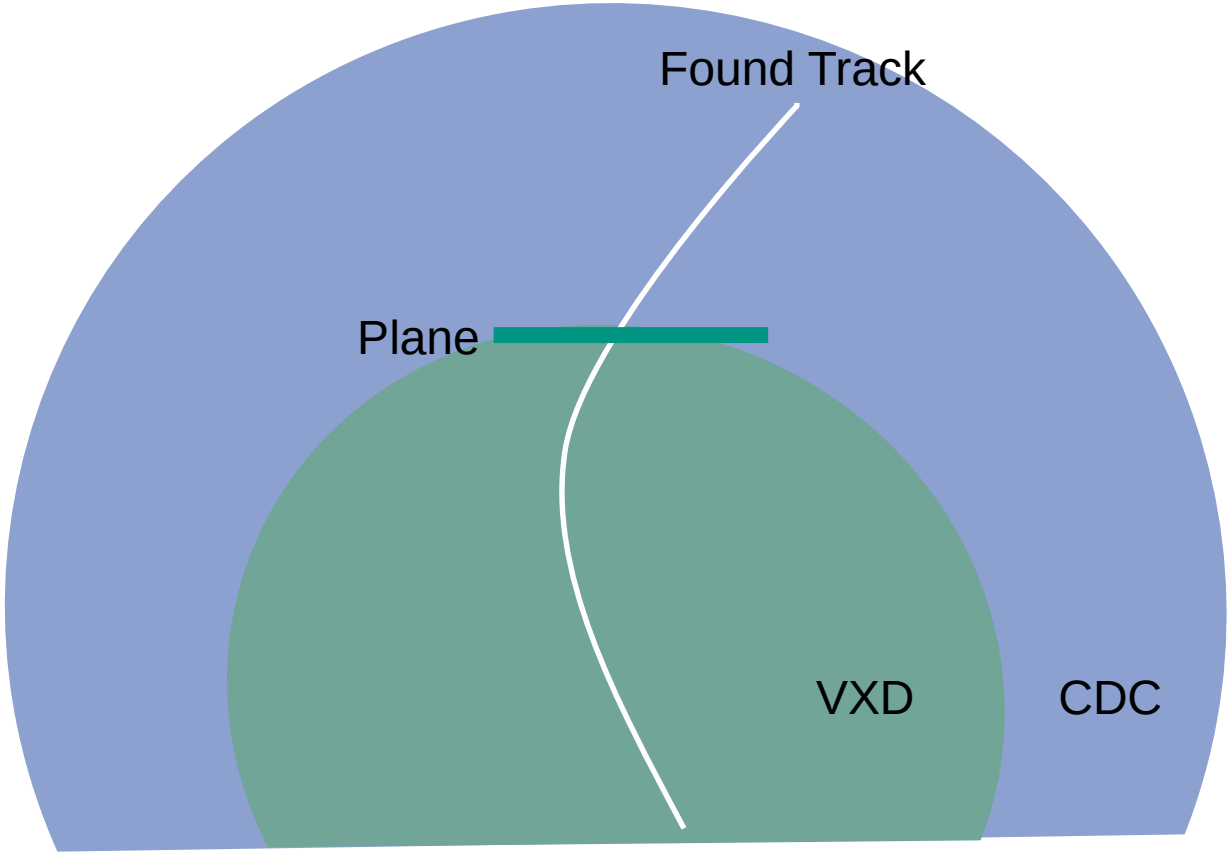
- Problem definition
- Implemented Matching Algorithms
- Results

Problem definition

- Optimization of the VXD-CDC Track Merger
  - Shorten Runtime
  - Increase Merging efficiency
- Implementation within new framework and usage of existing templates
  - Transfer code into new framework to be able to modify and optimize it
- Specs: 1000 events with Background, Truth information obtained from MC Particle Truth

Implemented Matching Algorithms - Extrapolation

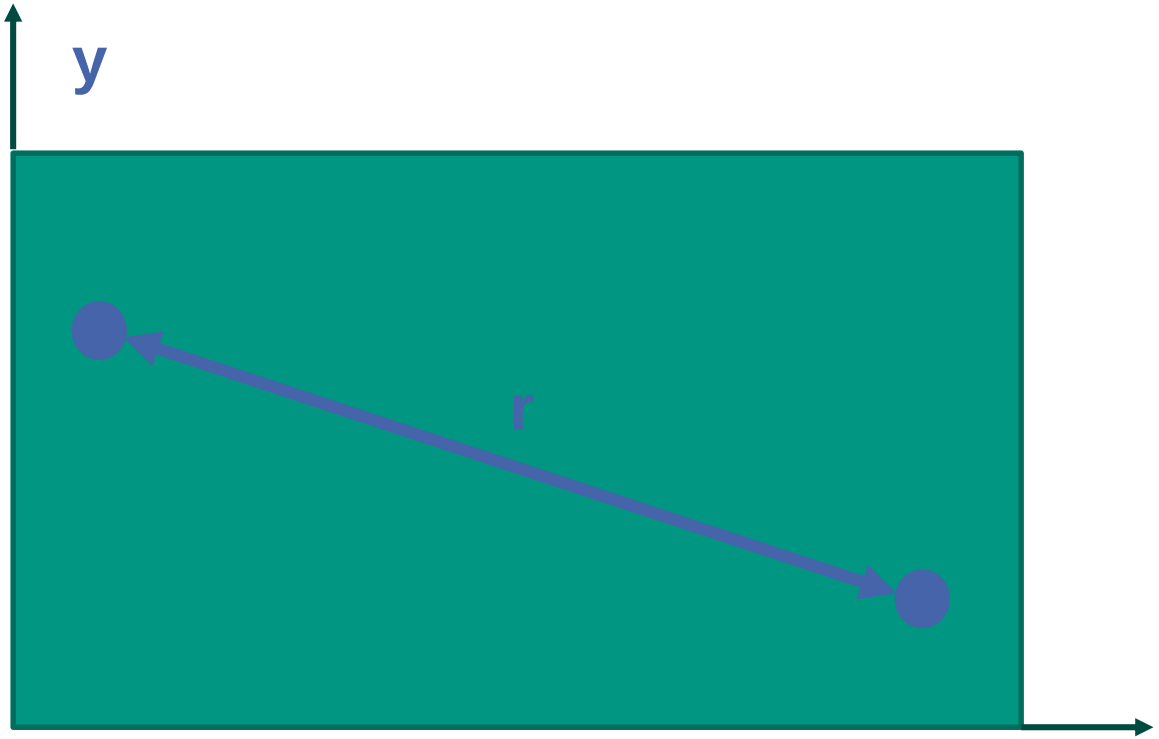
- Tracks from the VXD and CDC are reconstructed by another module.
- These tracks are then extrapolated to a shared plane



Implemented Matching Algorithms - Extrapolation

- On the plane the distance of the two tracks can be determined, called „radius“.
- Radius is a cut value, tracks with lowest radius are merged.
- Used exclusively in old merger, usage significantly lowered in new merger

- Default radius set to 2 cm.
- Used will be 1 cm
  - higher efficiency



Implemented Matching Algorithms - Filters

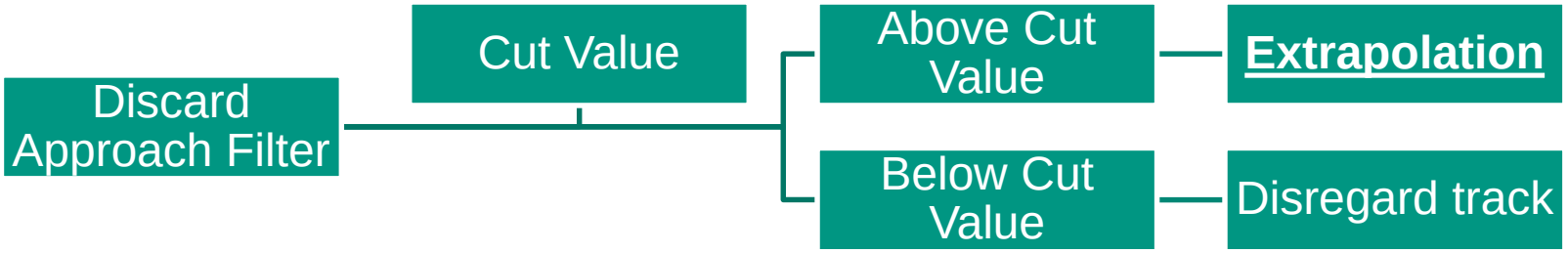
- Problem: Extrapolation of every track takes up lots of ressources and is unnecessary

- Solution: Use a filter based matcher
  - → use particle properties to train a BDT and make decisions before extrapolation

  - Properties used:
    - Charge, angles, momentum, hits, point of origin
    - Relations between VXD and CDC (e.g. angle difference, absolut and relative)

Implemented Matching Algorithms – Discard Approach

- Idea: Discard all matches that are highly unlikely.
- Based on BDT data a cut is implemented at the lower end.
  - Disregard all track matches below the cut value
  - Extrapolate only tracks that are above the cut value
- Benefit:
  - All highly unlikely matches are disregarded saving time during extrapolation (less fitting to be done)

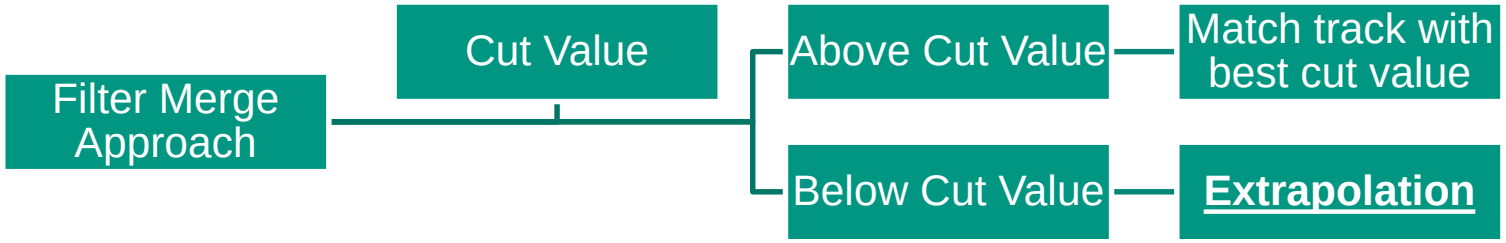


Implemented Matching Algorithms – Filter Merge Approach

- Idea: Merge all tracks that are highly likely to fit together

- Based on BDT data a cut could is implemented at the higher end.
  - Use only tracks above cut value
  - Merge the track with the highest value
  - Extrapolate all remaining tracks that couldn‘t be merged based on the BDT

- Benefit:
  - Highly likely tracks are merged based on the BDT reducing the amount of extrapolation needed (less fitting needed)



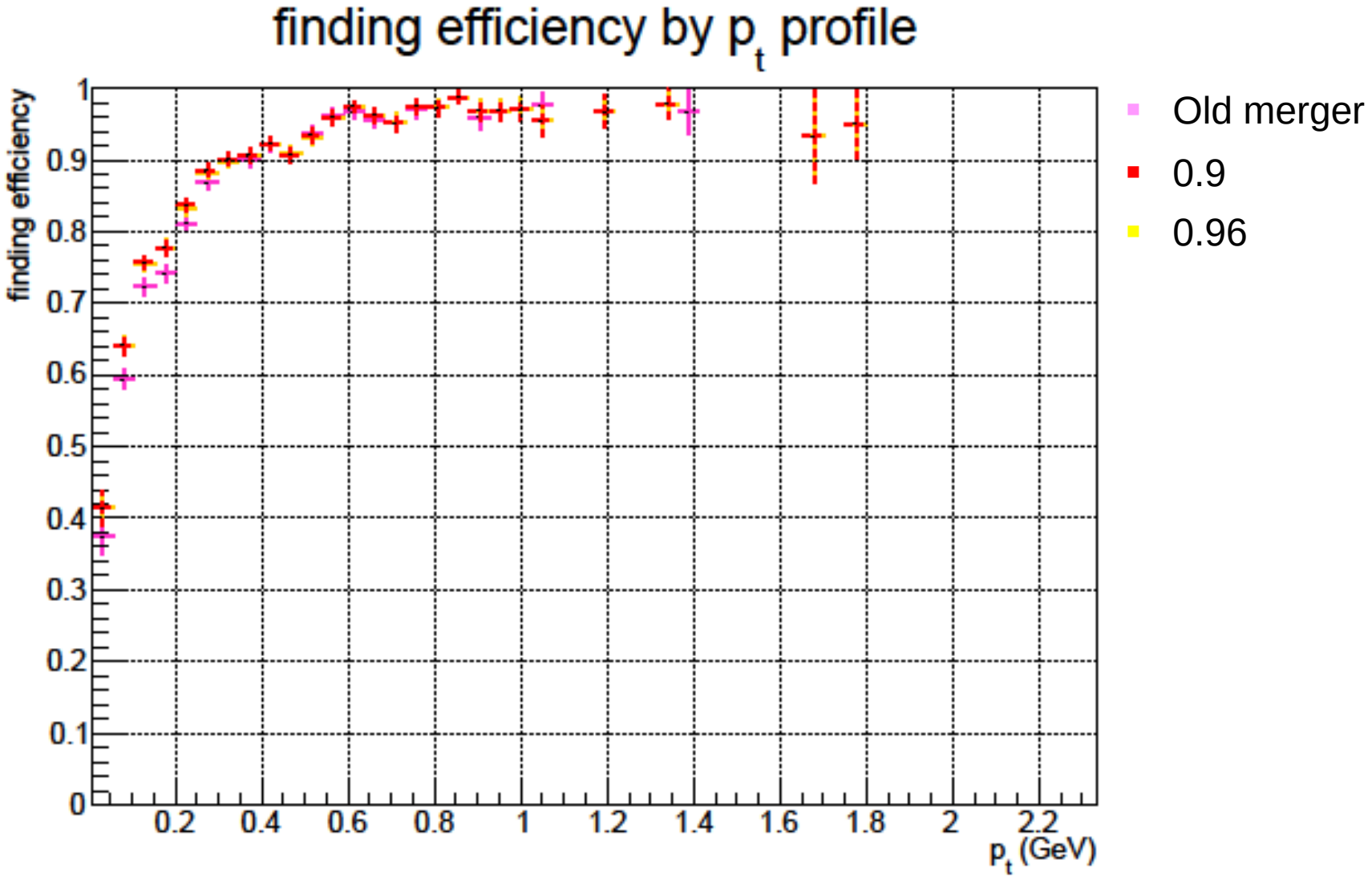
Results

The results displayed are based on a lean validation to improve runtimes of the validation process and do not represent the actual overall validation.

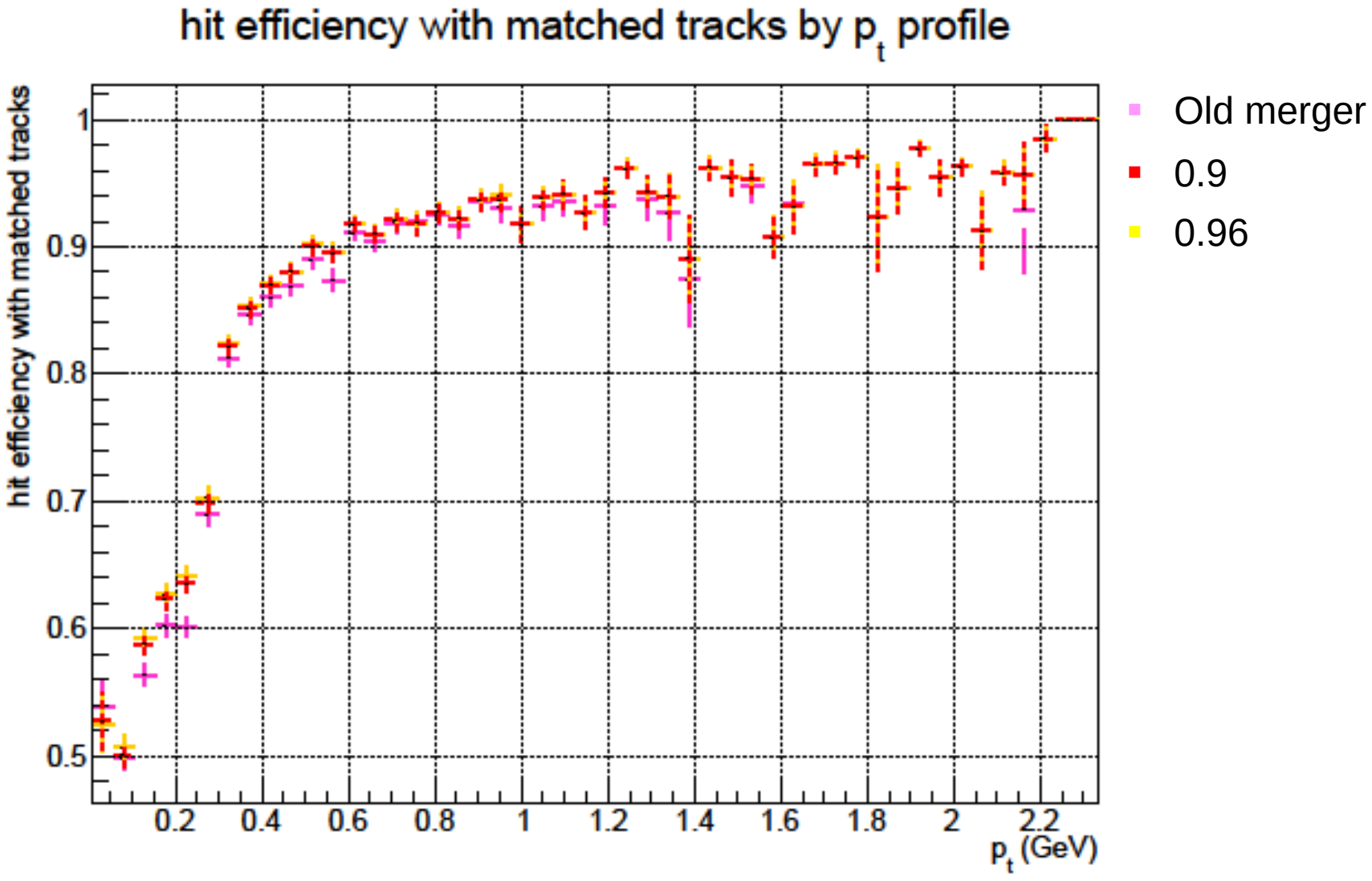
	Tracks correctly merged	Tracks wrongly merged	Not found	Not merged	Runtime
Filter Merge	85.6553998	3.07192910	8.07914262	3.19352844	21.4024828
Discard	83.2086915	2.11204019	8.07914262	6.60012562	56.8531746
Current Implementation	83.4707337	2.57625721	8.07914262	5.87386644	67.6672558

Conclusion:
The Filter Merge Approach is not only more efficient at merging but also provides faster runtimes.
Thus the Filter Merge Approach is to replace the currently implemented one.

Results


finding efficiency by p_t profile
finding efficiency
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
p_t (GeV)
Old merger
0.9
0.96

Results


hit efficiency with matched tracks by p_t profile
hit efficiency with matched tracks
1
0.9
0.8
0.7
0.6
0.5
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
p_t (GeV)
Old merger
0.9
0.96

Results

- Rough comparsion of tracking validation runtimes:
  - Old merger: ~5000 seconds
  - 0.9: ~3000 seconds
  - 0.96: ~3800 seconds

- Conclusion:
  - The Filter Merge Approach provides slightly better results at the lower momentum end and similar results in other areas in comparison to the current one.
  - The runtime improvement, however, saw a significant decrease of ~40%.