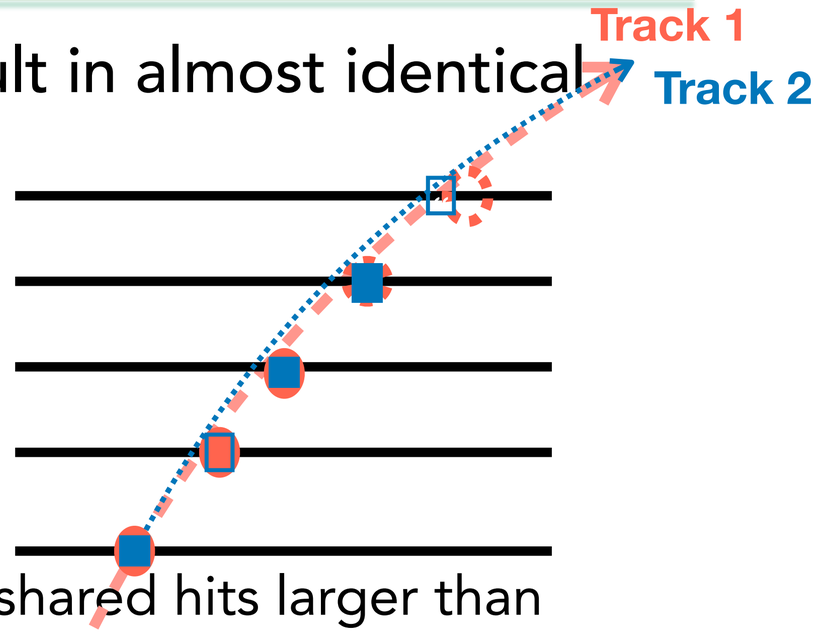
Tracking for ePIC

Berkeley EIC meeting
26. Mar. 2024

Minjung Kim

Greedy ambiguity resolution solver

- After tracking, seeds originated from same particle result in almost identical reconstructed tracks
  - More or less similar reconstructed kinematic variables
  - Almost same sets of associated hits
- **Greedy ambiguity resolution solver**:
  1. Iterate trajectories and find the trajectory having number of shared hits larger than certain threshold
  2. Find the competetors and keep better quality trajectory only
  3. Repeat till you have trajectories having shared hits below certain threshold: <u>number of shared hits required!</u>



➡ **Native support in ACTS,** not implemented in official EICrecon yet

★ ML based methods under development

Ambiguity Resolution in EICrecon

- algorithm/tracking/AmbiguitySolver.cc(h):

  std::vector<const ActsExamples::ConstTrackContainer*> input_container

  - Taking CKFTracking output(s)
  - Call algorithm and process
  - Convert output(s) from the algorithm to standard EICrecon output (compatible with tracking output(s) for seamless transition for the rest of the reco. chain)

std::tuple<
std::unique_ptr<edm4eic::TrajectoryCollection>,
std::unique_ptr<edm4eic::TrackParametersCollection>,
std::unique_ptr<edm4eic::TrackCollection>,
std::vector<ActsExamples::Trajectories*>,
std::vector<ActsExamples::ConstTrackContainer*>
>

- algorithms/tracking/AmbiguitySolverConfig.h
  - Configuration helper for the algorithm:

    max. shared hits, max. iteration and min. # of measurements per track

- global/tracking/AmbiguitySolver_factory.h
  - Factory for the algorithm; to be called after “CKFtracking” inside full tracking chain

Current status

- algorithm/tracking/AmbiguitySolver.cc(h):
  - ~~Taking CKFTracking output(s)~~ **DONE ✅**
  - ~~Call algorithm and process~~ **DONE ✅**
  - **Convert output(s) from the algorithm to standard EICrecon output (compatible with seeding output(s))**
- ~~algorithms/tracking/AmbiguitySolverConfig.h~~
  - ~~Configuration helper for the algorithm:~~

    ~~max. shared hits, max. iteration and min. # of measurements per track~~ **DONE ✅**
- ~~global/tracking/AmbiguitySolver_factory.h~~
  - ~~Factory for the algorithm; to be called after "CKFtracking" inside full tracking chain~~ **DONE ✅**

To do

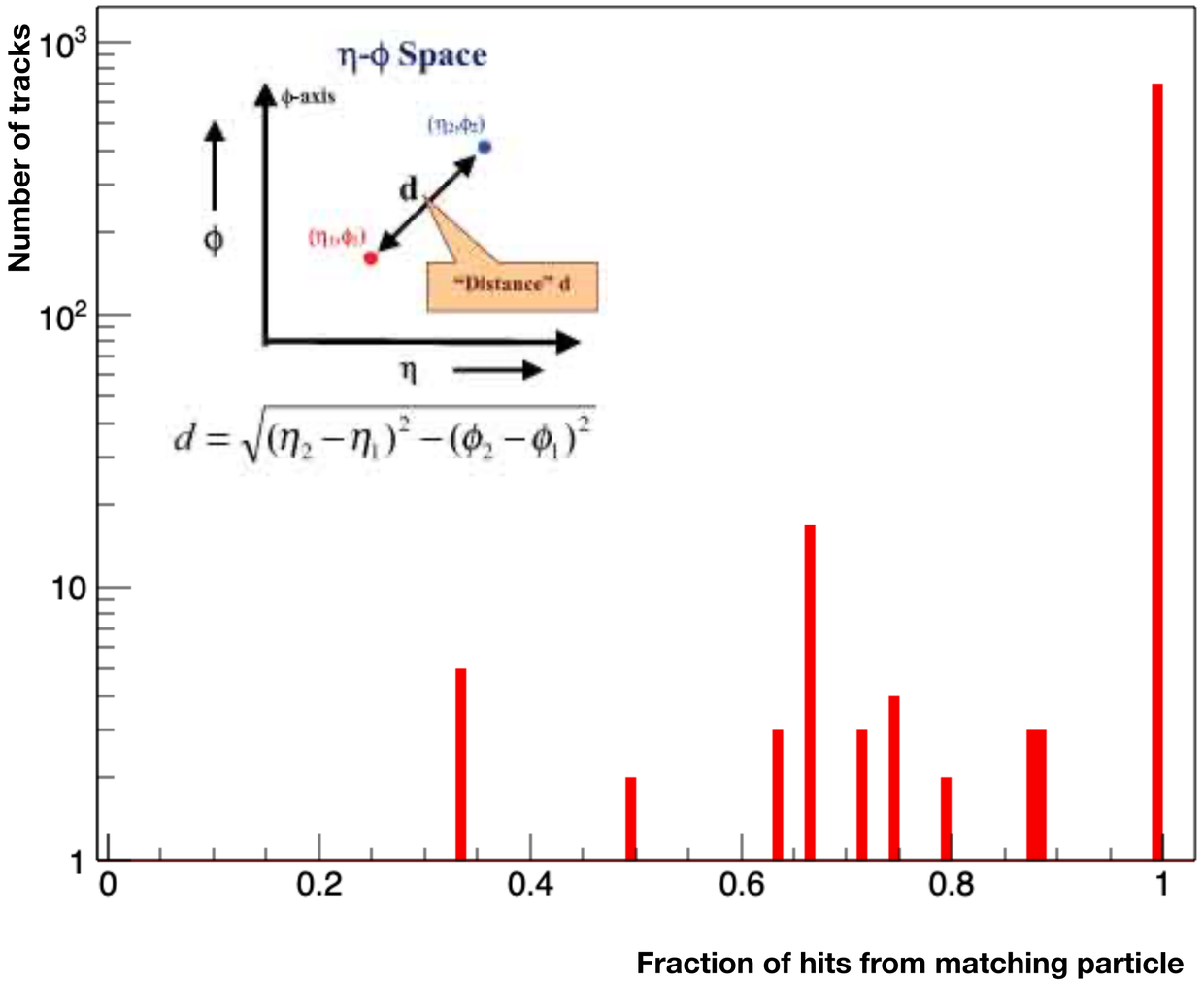
1. Compatibility to current EICrecon main branch (3 commits behind to main since last week) **< 1 Hour**
2. Convert output(s) from the algorithm to standard EICrecon output (compatible with seeding output(s)) **< 1 working day**
3. Check the compatibility to current EICrecon main branch once again **< 1 Hour**
4. Merge request **< 1 Hour**
5. Merge to main branch ! **~ few weeks**

Some idea for the next step

- Validation of Ambiguity Resolution Solver:
  - Standard CKFtracking output vs. ambiguity Resolution Solver for realistic seeding, ideally event by event for different event types (simple exclusive production to high Q2 DIS events)
- Matching between MC particles and reconstructed tracks:
  - Current EICrecon uses geometrical distance
  - 1:1 relation was not fully confirmed in DIS events
  - Limited validation for realistic seedings due to the duplicates
  - Small discrepancy between distance based matching vs. hit based matching
- (Longer term + inputs required) Tracking QA factory (processor)
  - Set of histograms estimating tracking performance (not limited to tracking algorithm performance, but also for geometry/material updates,…)

Backup

Matching between MC particle and reconstructed track



- Two different matching methods were considered:
  - Hit level matching: check the source of hits in the track and matching to the particle giving maximum contribution
  - Angular distance matching: matching reconstructed track with the particle having the closest value of the distance (similar to EICrecon way)
- Angular distance based matching gives similar result with hit level matching, but not identical
  →Can we introduce hit level matching in EICrecon? MC source of generated hits not written in TrackerHit object