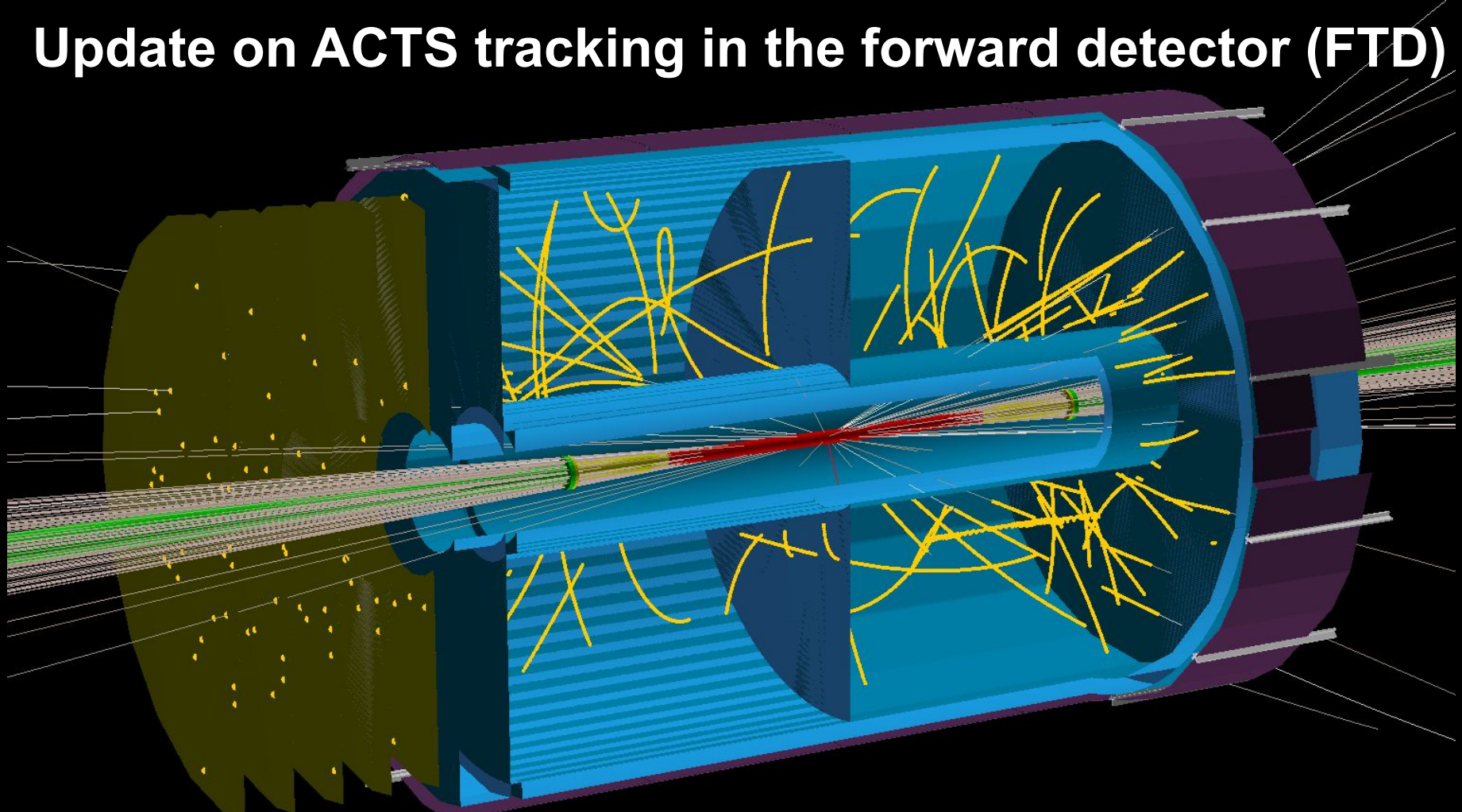
Update on ACTS tracking in the forward detector (FTD)



Evgeny Kryshen, Nazar Burmasov (PNPI, JINR)

MPD cross-PWG meeting, 8 Apr 2025

Reminder

Main challenges:

- Momentum resolution is driven by the radial distance available for the track curvature measurement
  ➜ resolution strongly degrades towards large η
- Large and poorly known material budget of TPC endcaps
- High occupancy in central AA collisions
  ➜ Seeding and ghost hit issues for 1D hits

Considering “ideal” tracker:

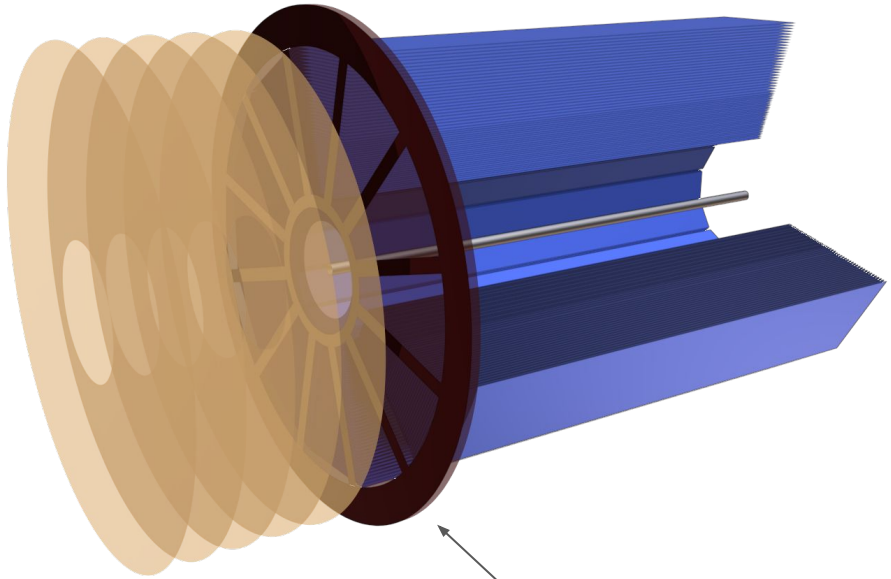
- 5 tracking layers placed between 210 and 300 cm
- Thickness per layer: 0.2% X_0
- Gaussian smearing in x and y with σ = 100 um

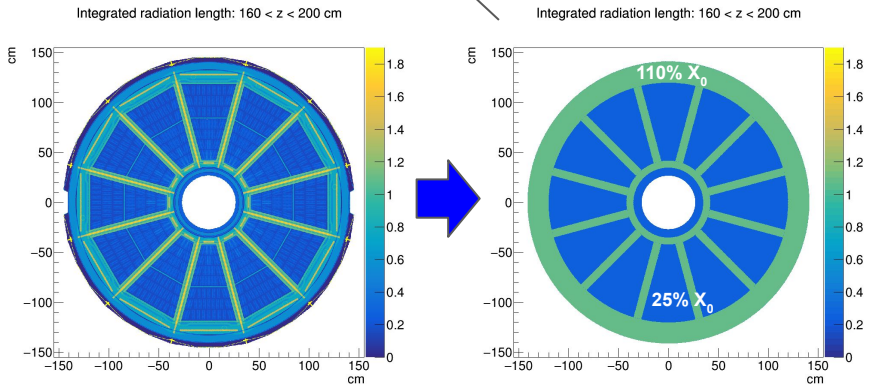
Tracking with ACTS:

- Considering FTD standalone, and FTD + TPC tracking
- Sector-like geometry for TPC (hits @ 1,4,7 padrows for seeding)
- Simplified material description of TPC endcaps

Links:

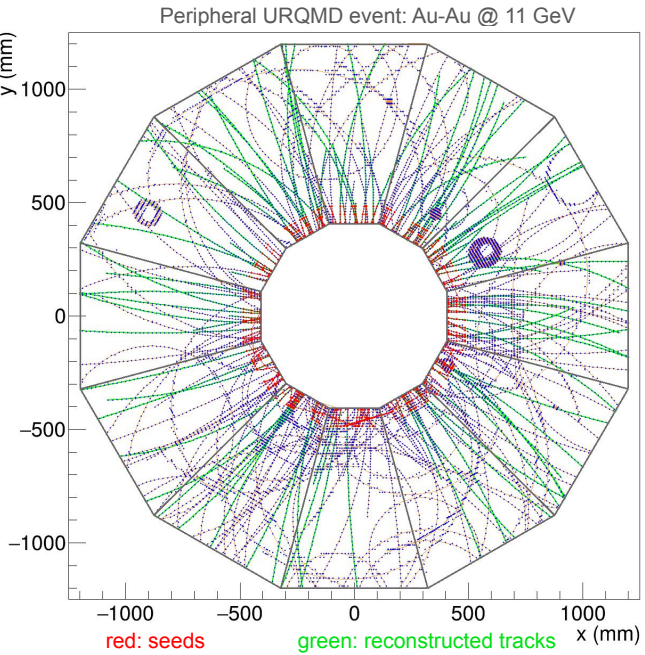
- Forward upgrade for the MPD
- ACTS tracking in TPC with sector geometry
- ACTS tracking in the forward detector using TPC seeds
- Fork in git: https://git.jinr.ru/ekryshen/mpdroot

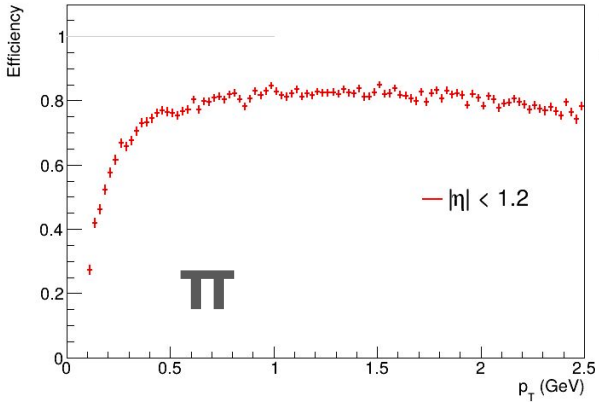


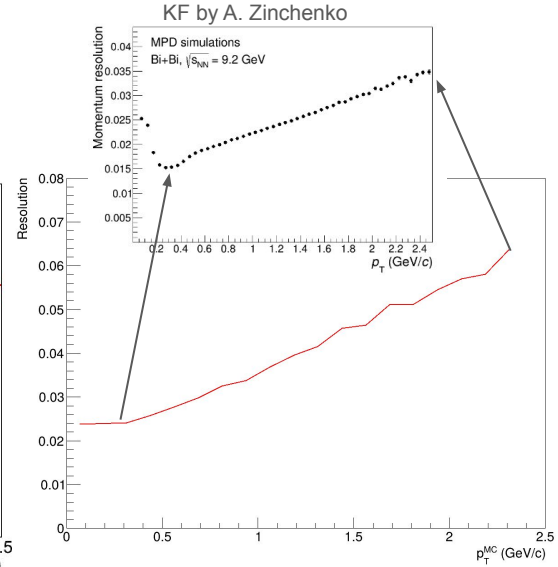


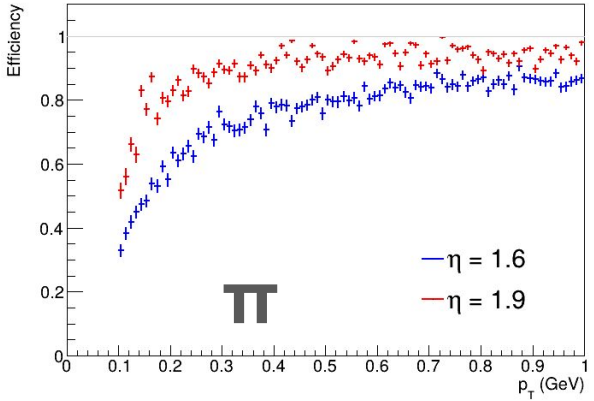
Reminder: problems

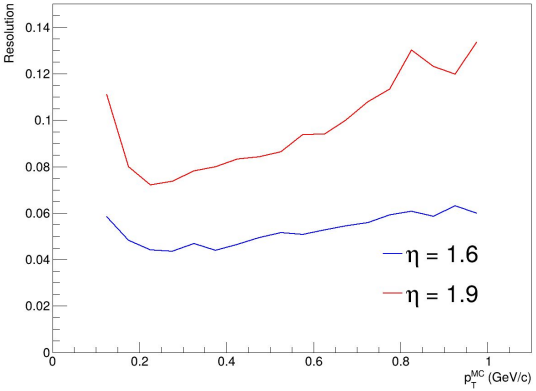
- Tracking efficiency losses due to track propagation errors
- Momentum resolution much worse than expected











Improved TPC sector geometry

Geometry hierarchy: volumes

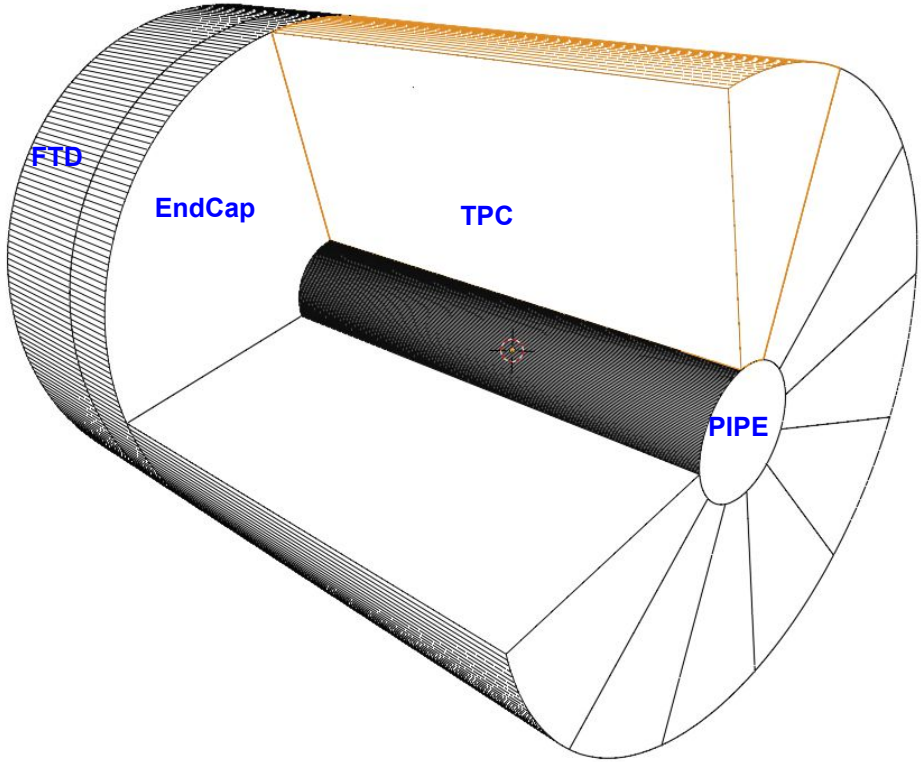
Volumes:

- BARREL
  - PIPE
  - TPC:
    - TPC0
    - TPC1
    - …
    - TPC11
- EndCap
- FTD

r binning

phi binning

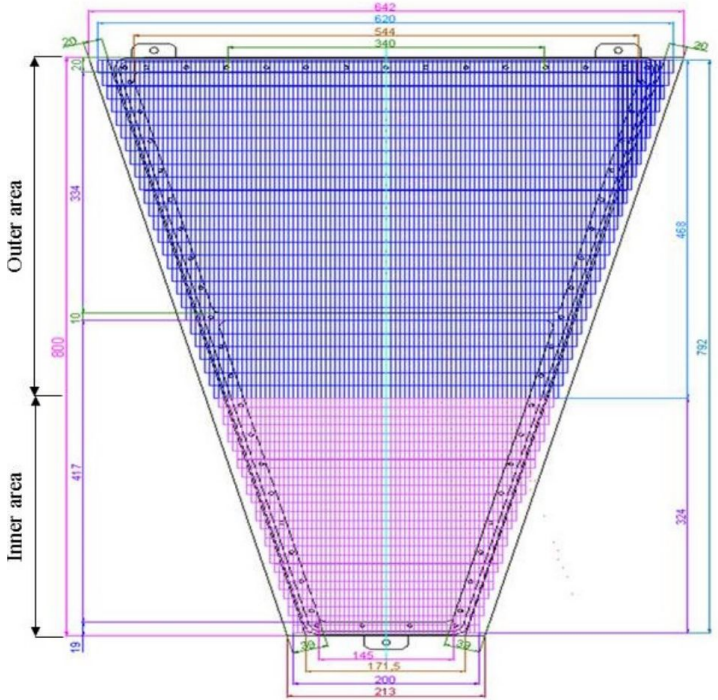
z binning

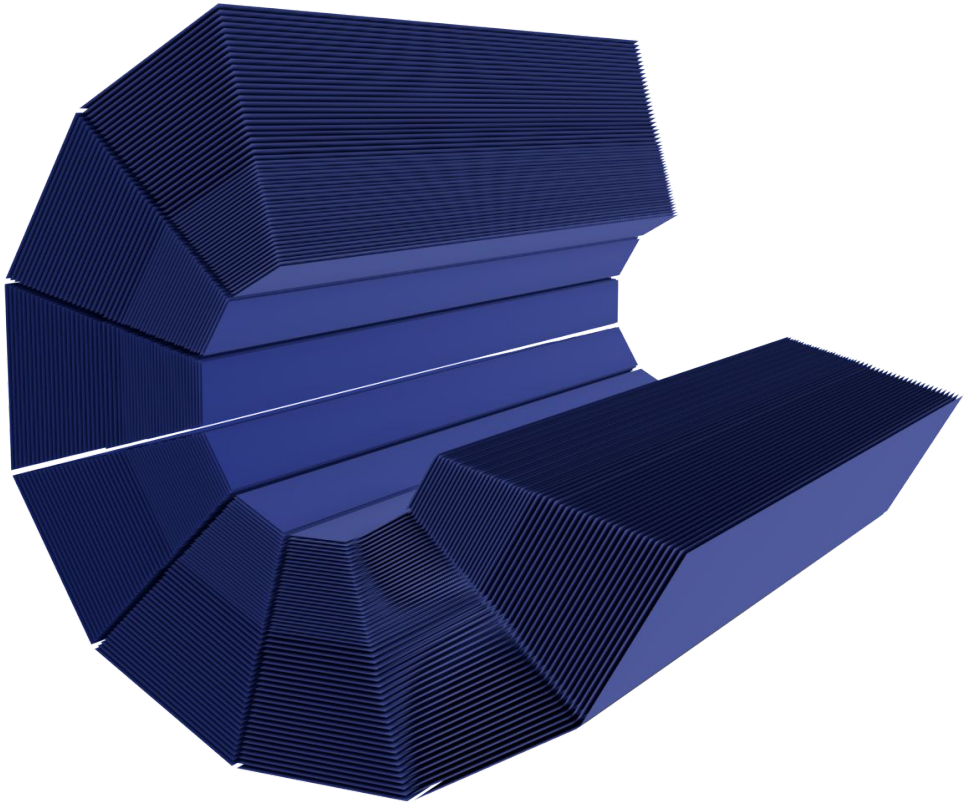


Fully connected geometry:

- Common boundary surfaces are glued (e. FTD and EndCap)
- If boundary is shared by several volumes, volumes must be attached to boundary (e.g. TPC0… TPC11 to pipe boundary)

Virtual sector geometry in ACTS





- Virtual geometry in ACTS:
  - used to account for multiple scattering effects
  - measurements must be attached to surfaces
- Implemented virtual surfaces corresponding to pad rows

Geometry hierarchy: layers and surfaces

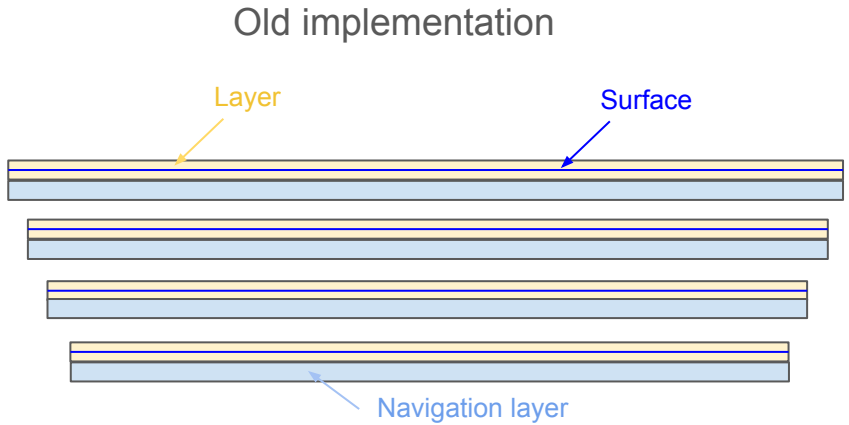
Old implementation:

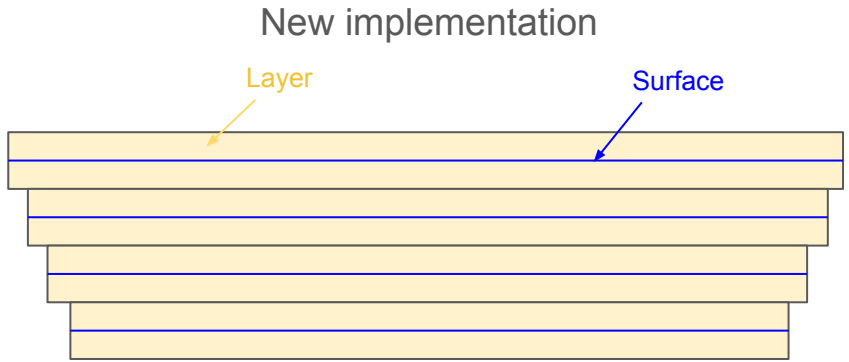
- Layer array (including navigation layers) created and binned automatically using Acts::LayerArrayCreator
- Only X and Y binning available

New implementation:

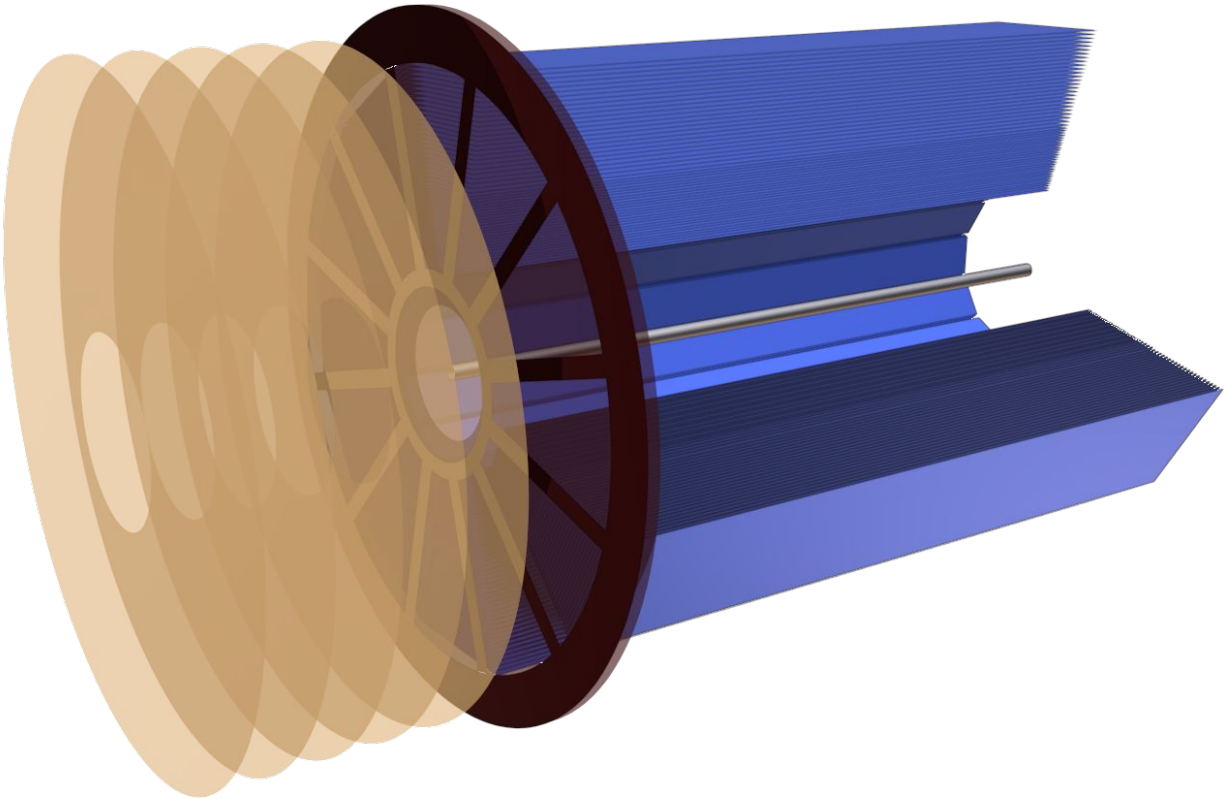
- Layer array created and binned manually
- Get rid of navigation layers
- Layers are fully connected
- Manual binning in R

NB: Also tried single cylindrical TPC volume with single cylindrical layer including 53x12 surfaces: didn't work well due to navigation issues

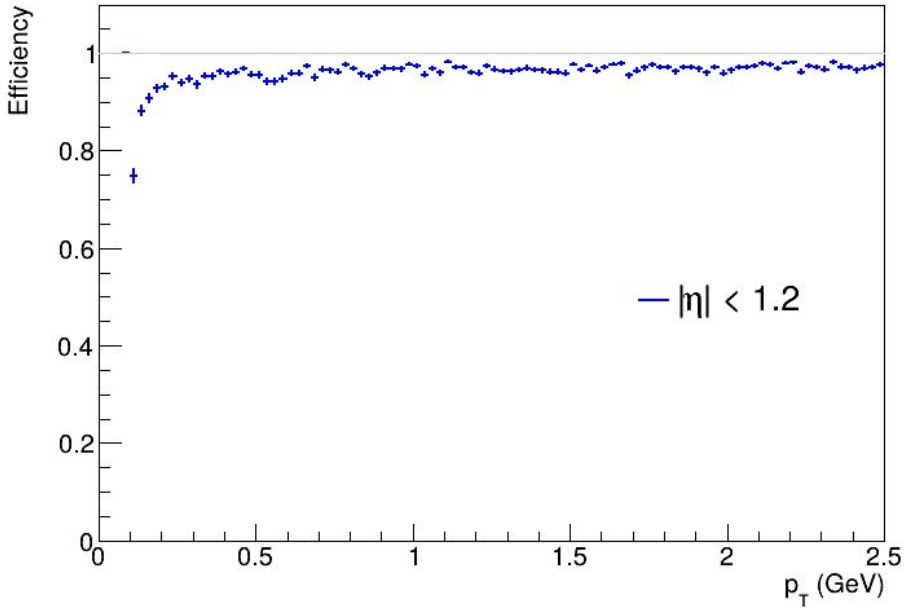


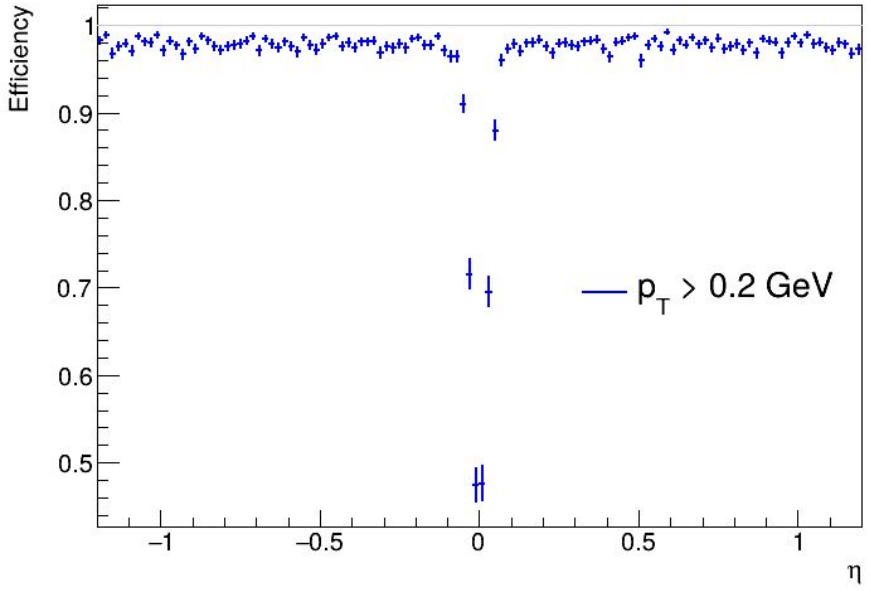


Surfaces in TPC, FTD, Pipe and EndCap (ACTS view)



TPC tracking performance with updated geometry (Boxgen)





- Shown: “trackable” efficiency for single-pion box generator
  Trackable = clusters at 1, 4, 7 padrows (seeding) + at least 20 clusters in total
- Major improvement of TPC tracking efficiency with the updated geometry
- Small inefficiencies mainly due to sector edges (to be checked)

FTD tracking performance: efficiency (Boxgen)

Seeding:

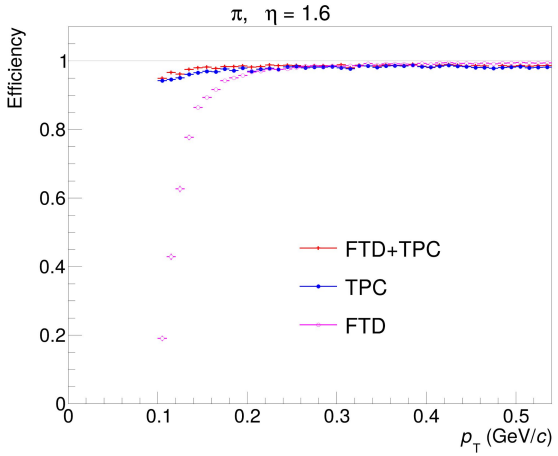
- TPC: for TPC and FTD+TPC
- FTD: for FTD

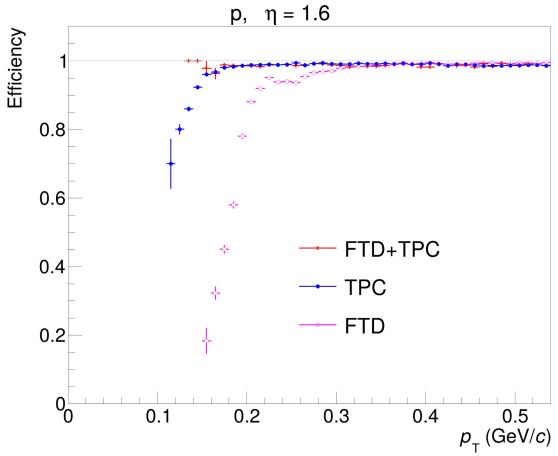
Shown trackable efficiencies.
Minimum requirements:

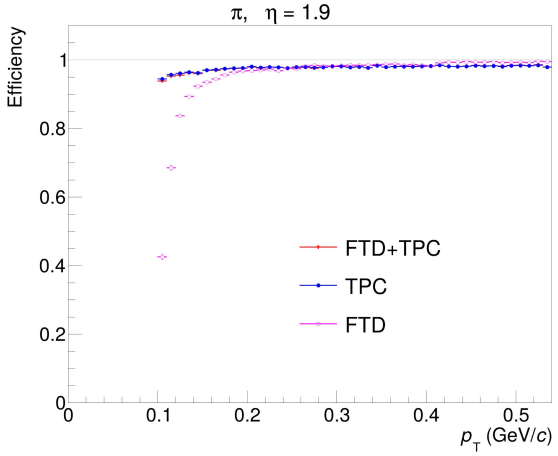
- TPC: hits in 1,4,7 padrows
- TPC+FTD: hits in 1,4,7 padrows, at least 3 hits in FTD
- FTD: 5 hits in FTD
- ALL: stay away from end-cap frame (~110% X_0)

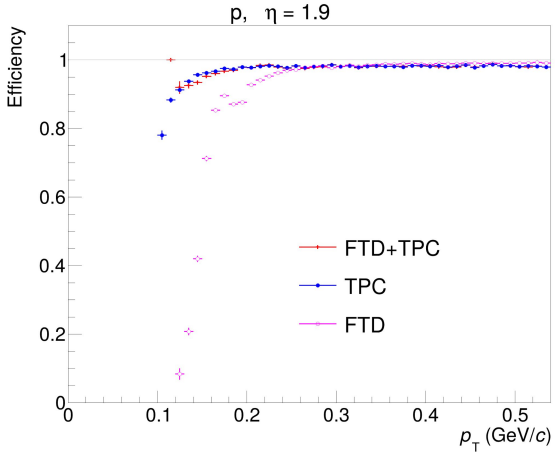
Results:

- Close to perfect efficiencies for single-track boxgen

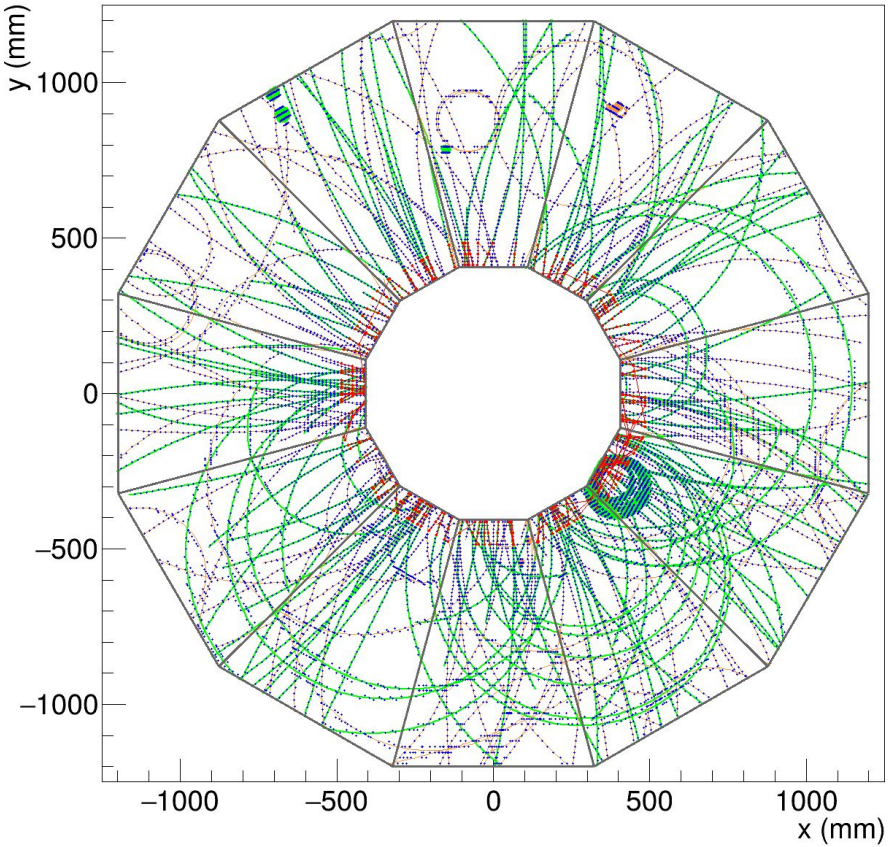


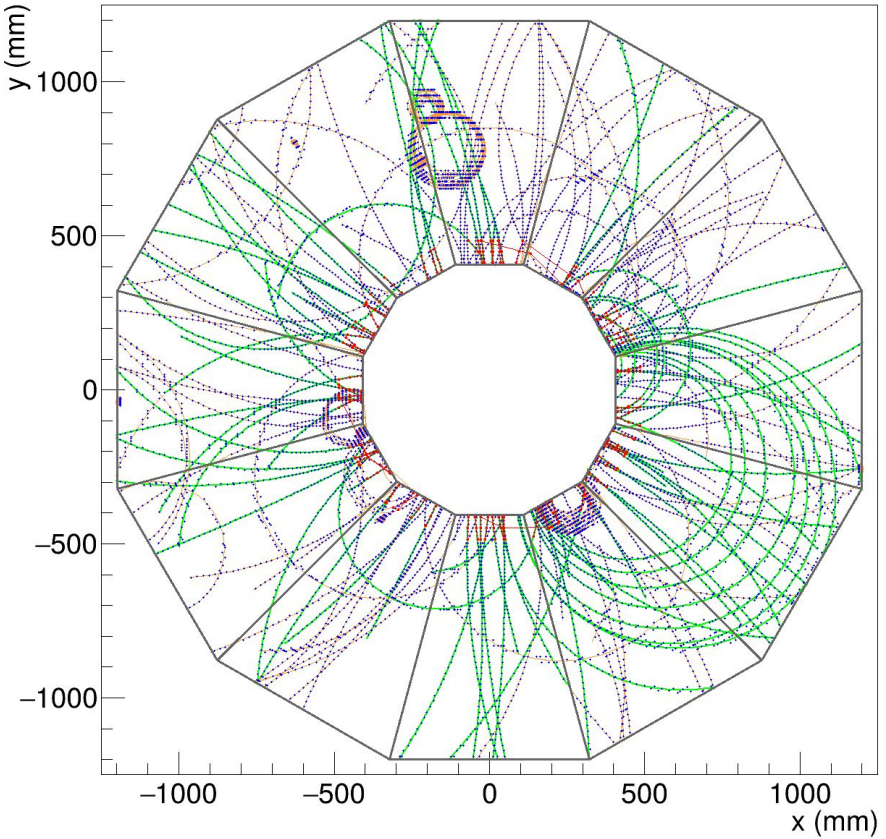






Examples from UrQMD generator (AuAu @ 11 GeV)

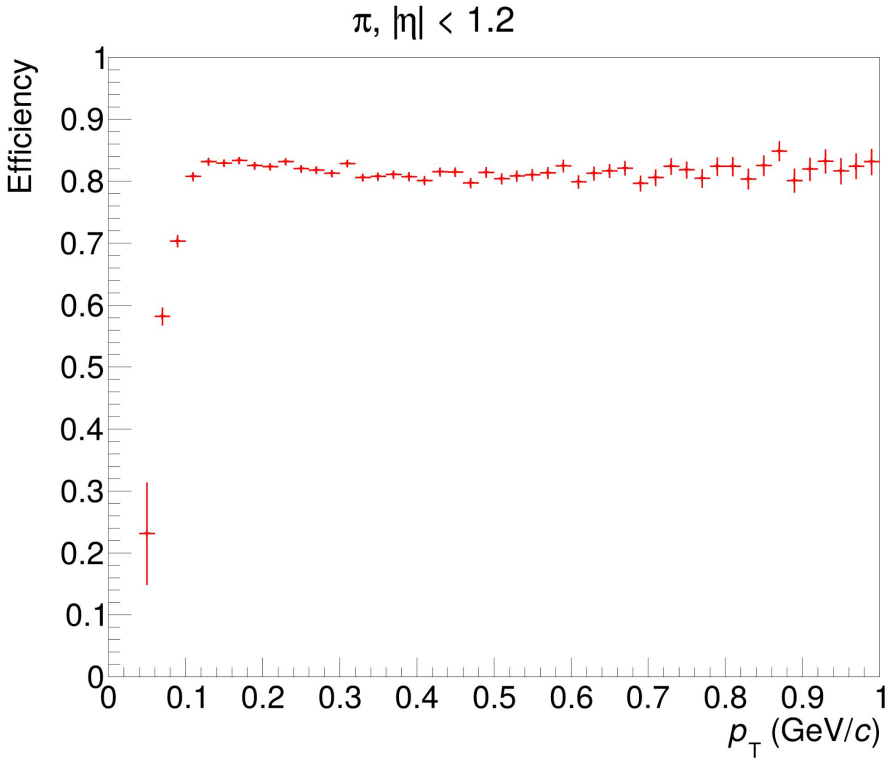


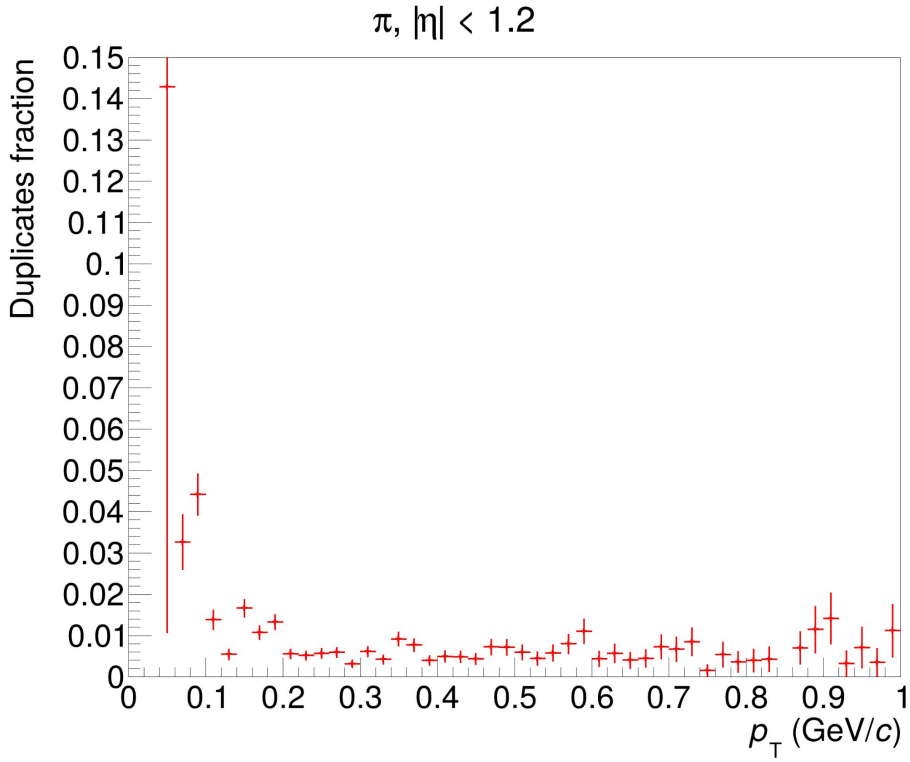


red: seeds green: reconstructed tracks

Seeder needs further tuning

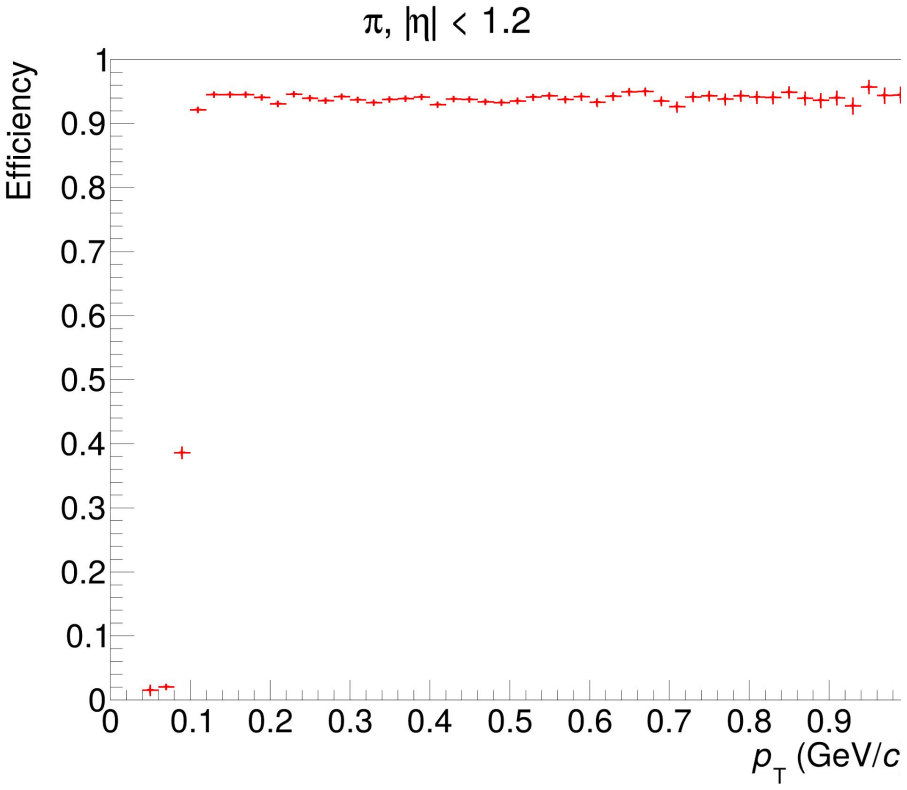
TPC tracking performance with updated geometry (UrQMD)

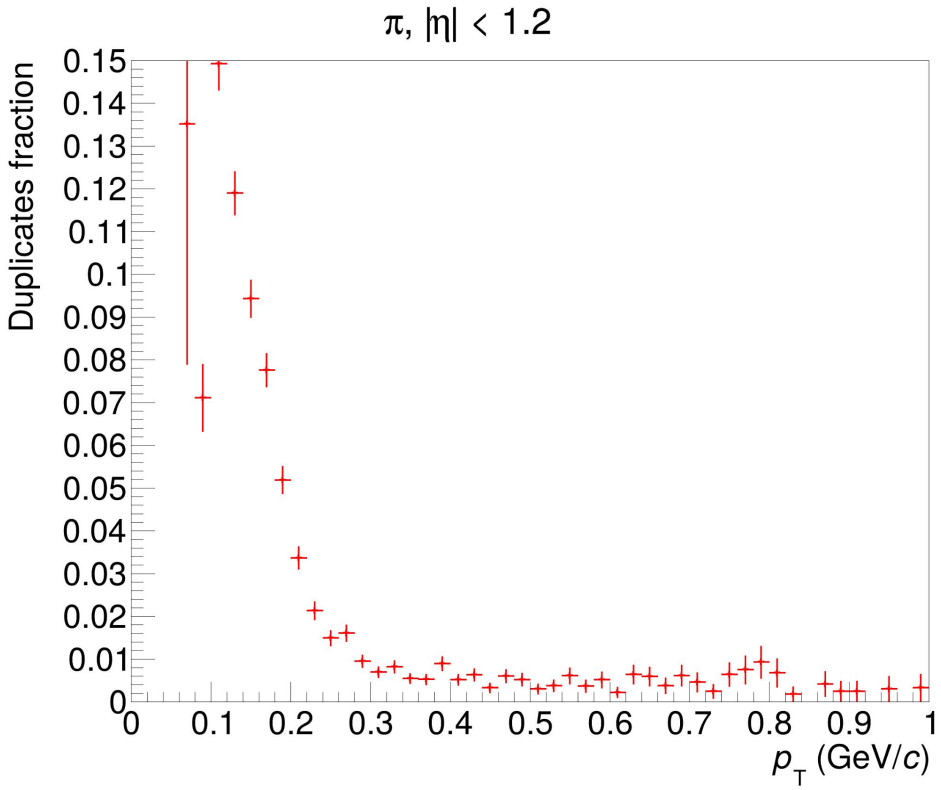




- Efficiency and duplicates with standard 1-4-7 seeding

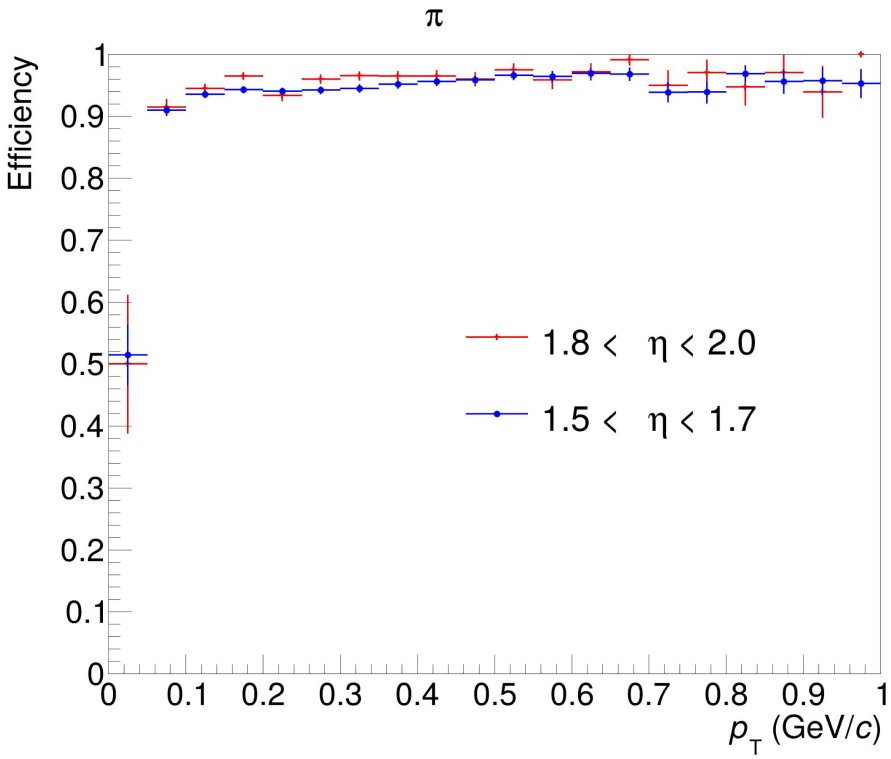
TPC tracking performance with updated geometry (UrQMD)

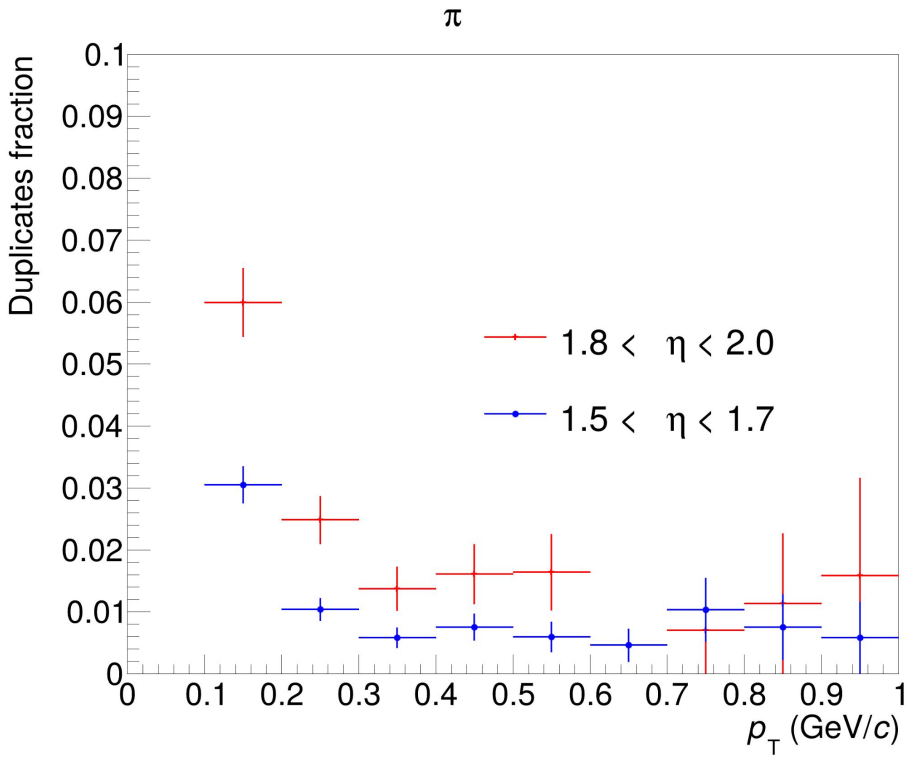




- Efficiency and duplicates with tuned seeding — we have room for improvements

TPC+FTD tracking performance with updated geometry (UrQMD)



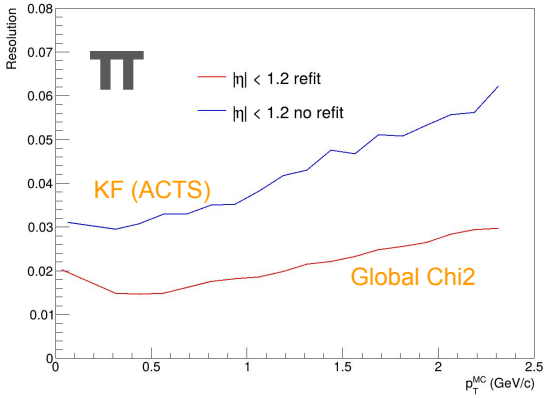


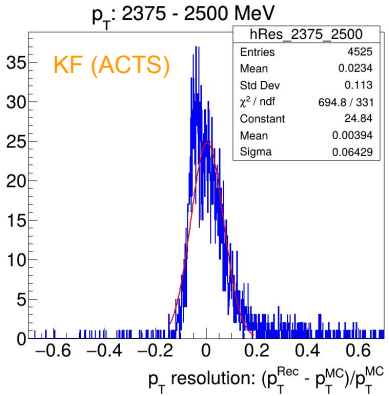
- Efficiency and duplicates with standard 1-4-7 seeding

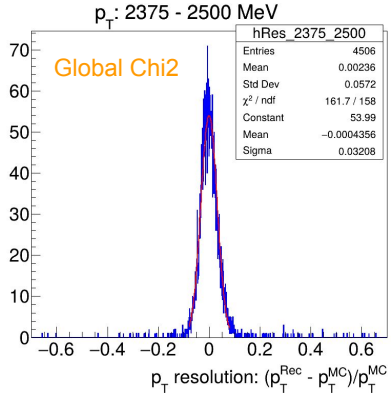
Momentum resolution
and
refitting algorithm

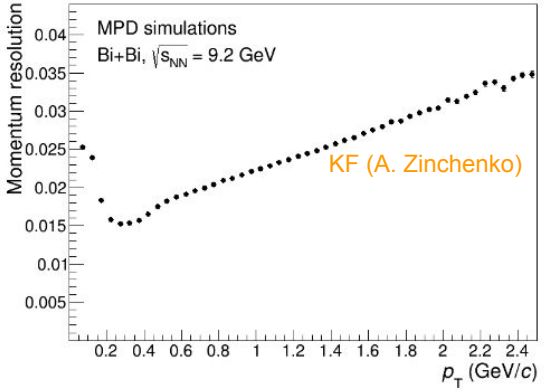
Refitting algorithm

- Standard Combinatorial KF assumes pion hypothesis for multiple scattering and energy loss effects. We can use TOF and dE/dx info to refit the track with updated mass hypothesis
  - Standard KF refitting algorithm doesn't allow one to change mass hypothesis
- We can use different approaches in the refit. Available options*:
  - Kalman Filter
  - Gaussian Sum Filter (correct energy loss effects for electrons)
  - Global Chi2 fitter
- p_T resolution from Global Chi2 fitter appears to be much better compared to KF from ACTS







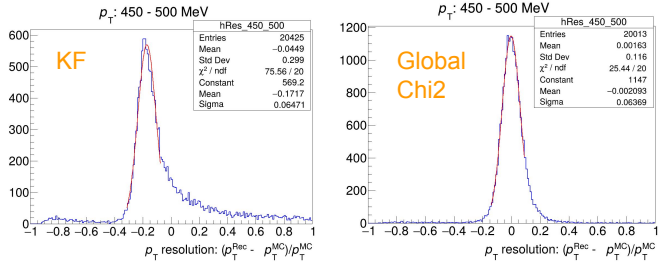


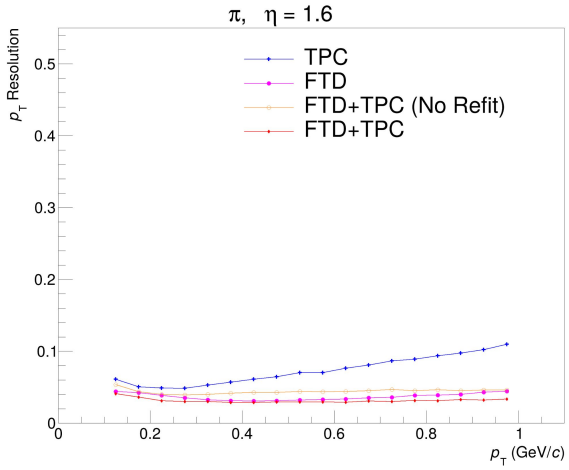
- See ACTS docs

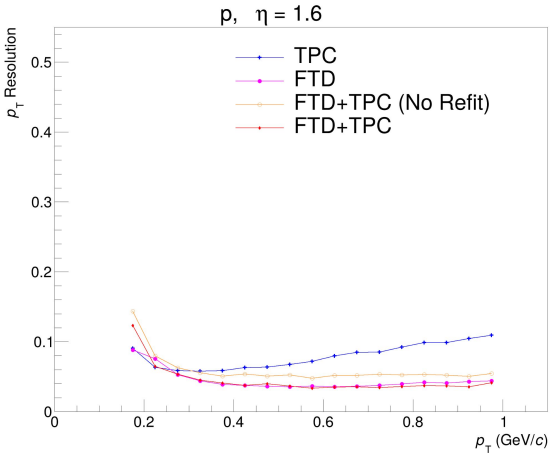
FTD tracking performance: momentum resolution

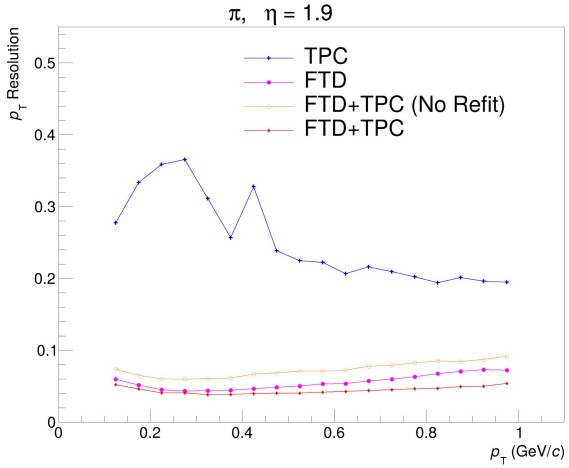
- KF: Biased momentum estimate with long tails
- Global Chi2: Much better Gaussian-like distributions
- FTD significantly improves momentum resolution, especially at large eta
- Combined FTD+TPC fit further improves momentum resolution

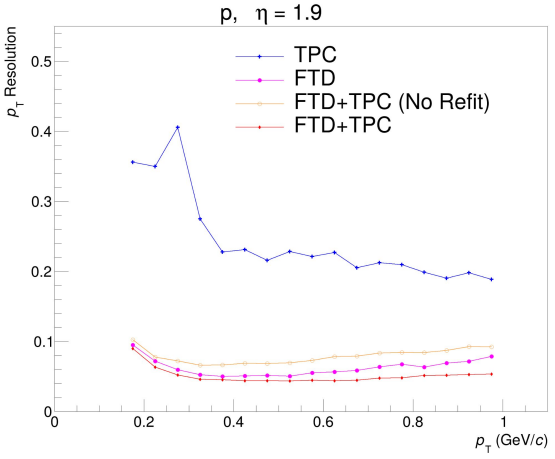
Resolution from TPC-only fit for pions at eta = 1.6





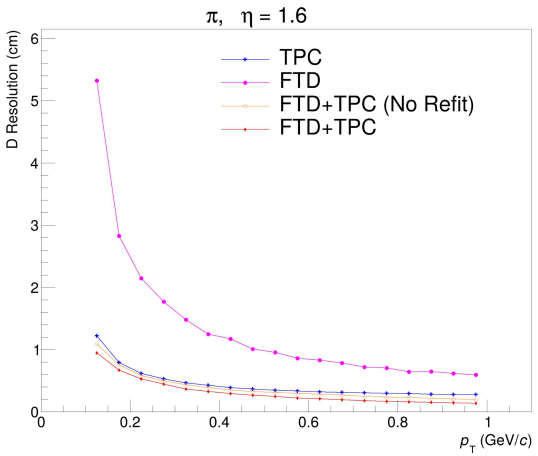


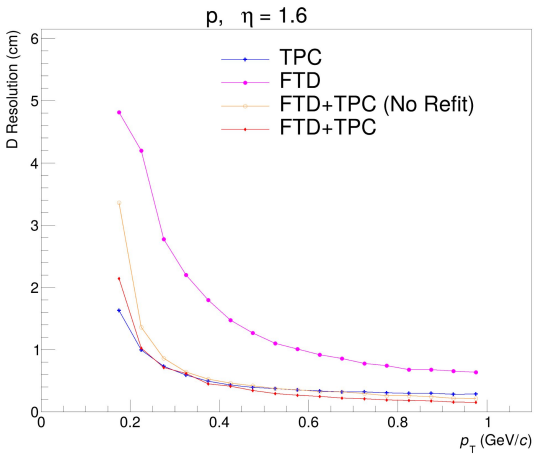


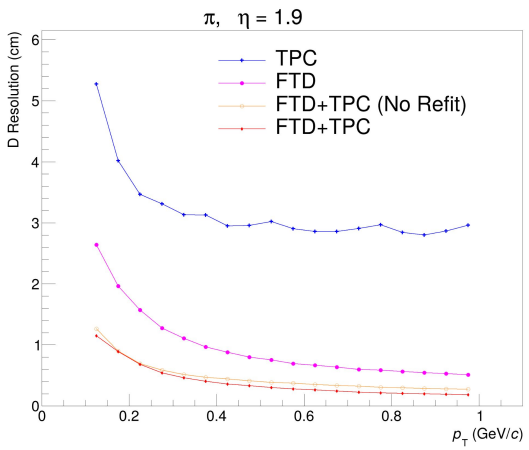


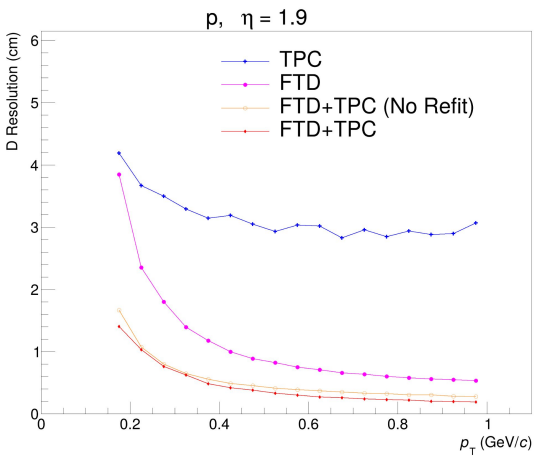
FTD tracking performance: DCA resolution

- TPC-matching helps to improve DCA resolution at moderate eta
- Poor DCA resolution for TPC-only at large eta
- TPC-FTD-matching helps to improve DCA resolution



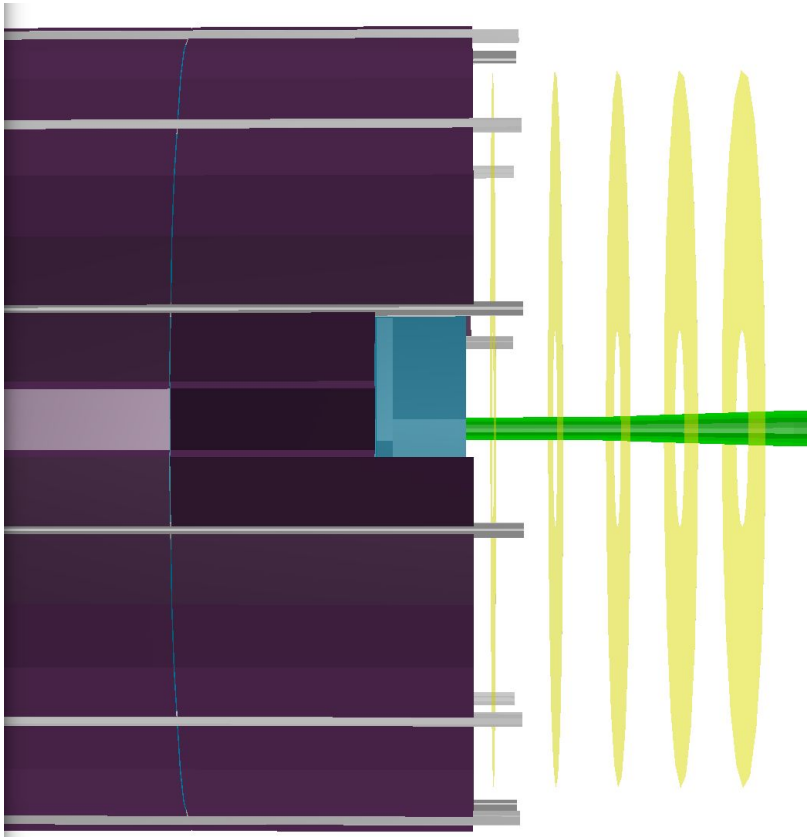




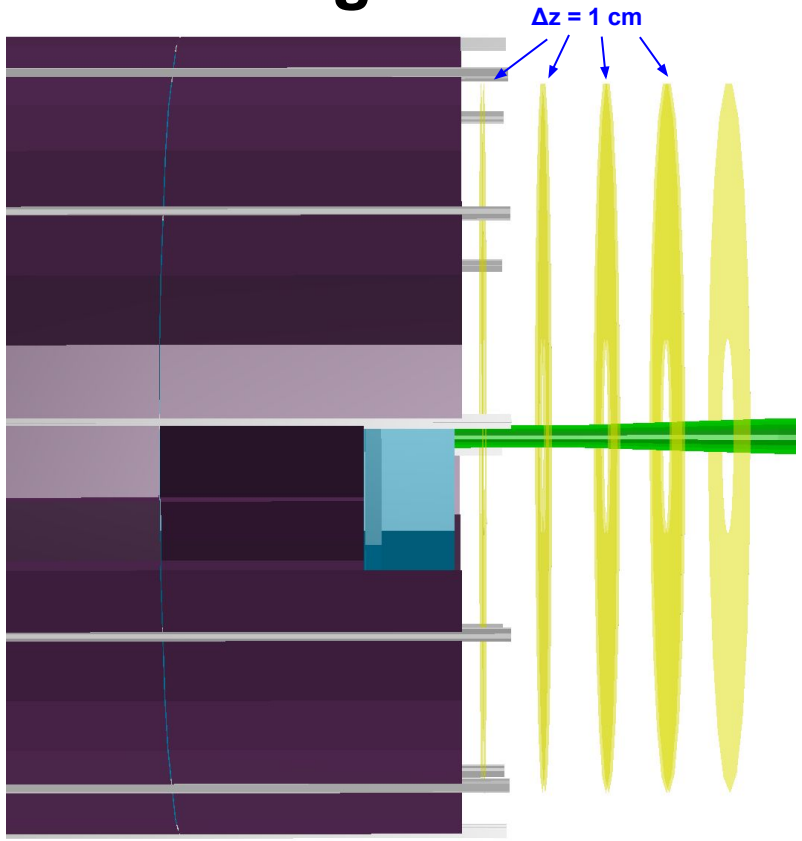


ACTS tracking
in strip-like forward detector
using 1D-hit info

2D tracking vs 1D tracking

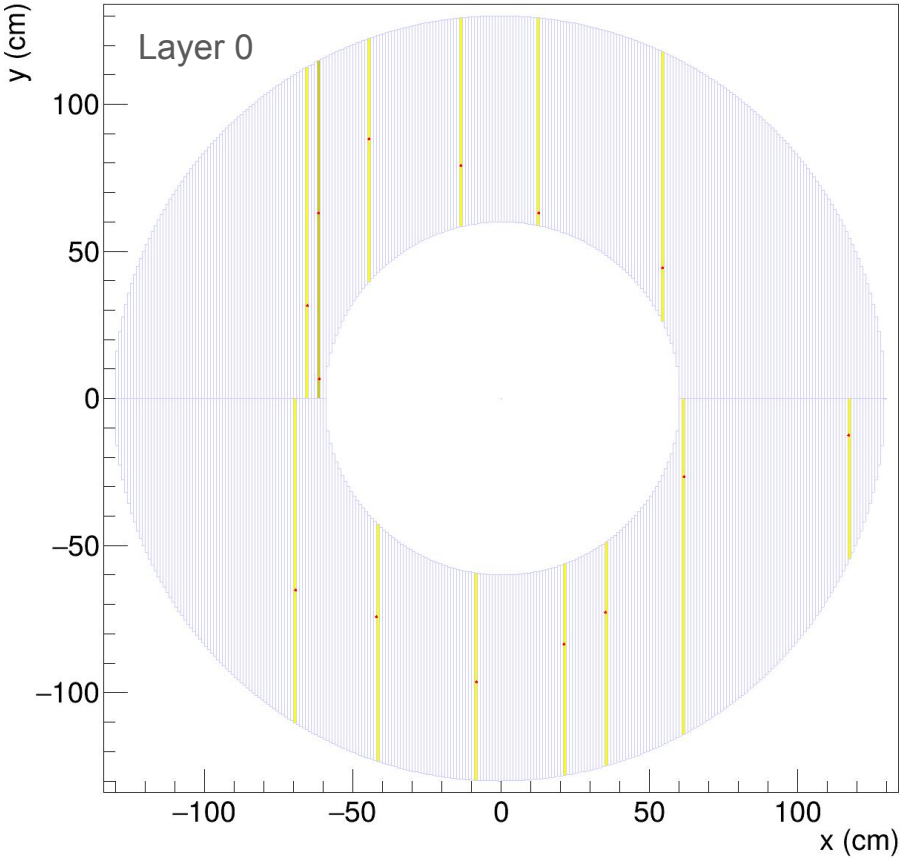


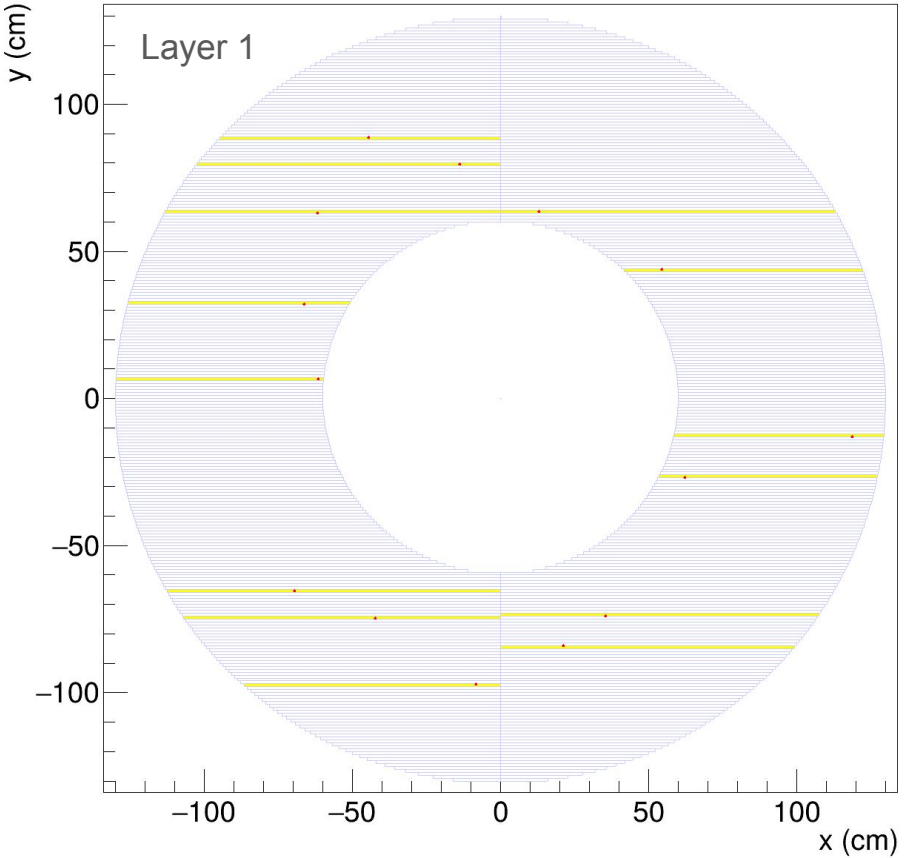
- 5 stations with pixel-like 2D layers



- 4 stations with 2 strip-like 1D layers + 1 pixel-like 2D layer
- In strip-like stations:
  first layer measures x coordinate
  second layer measures y coordinate

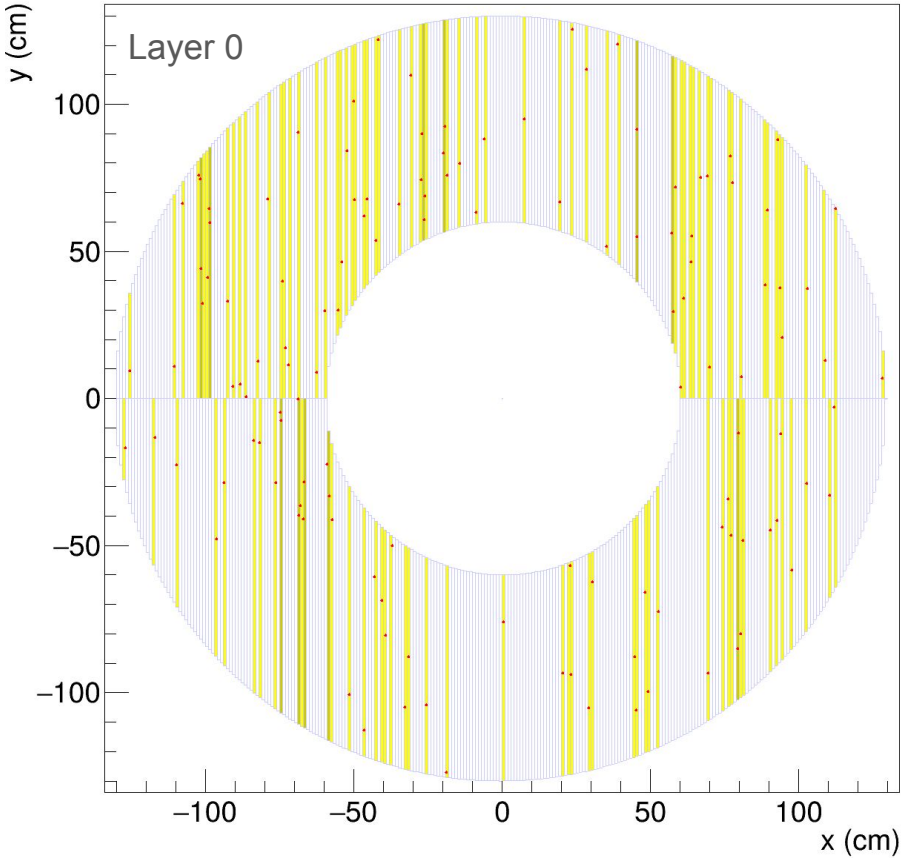
Typical UrQMD event in strip-like forward detector station

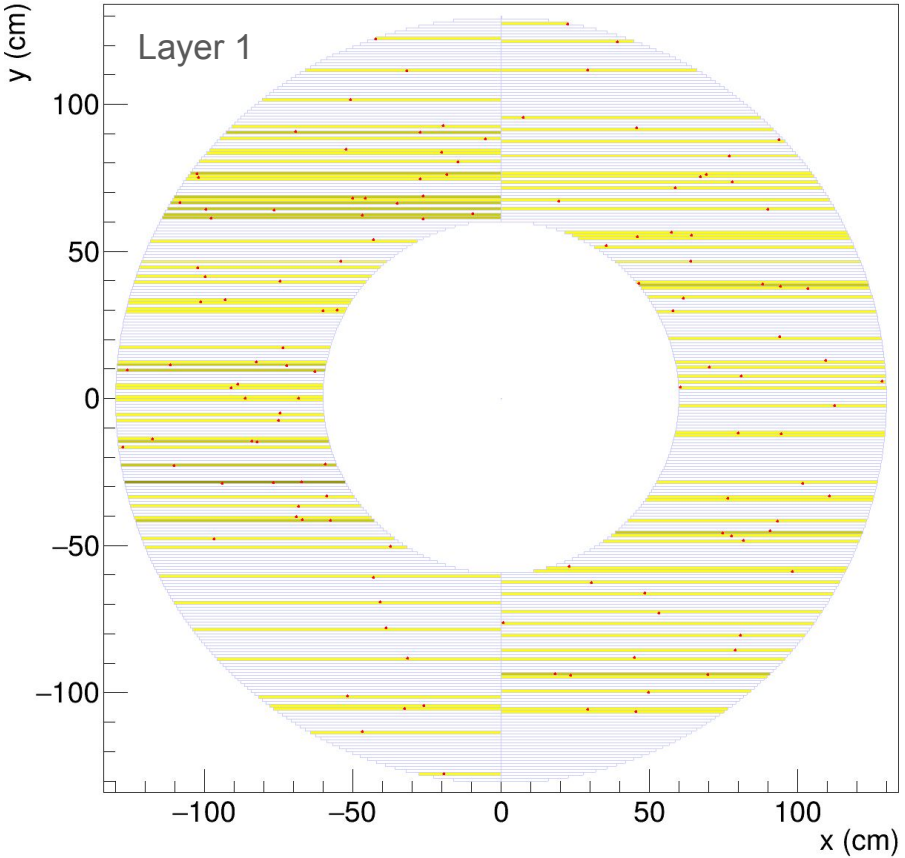




- Considering simple strip-like geometry with 1 cm strips, second layer rotated by 90 degrees

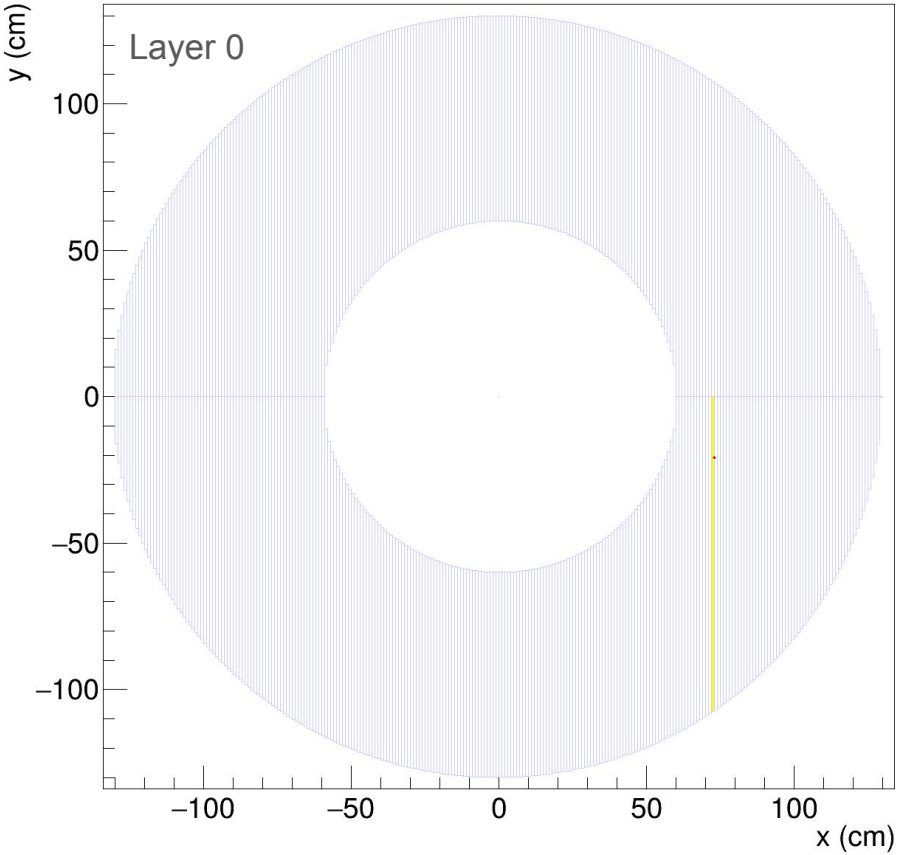
High-multiplicity UrQMD event example

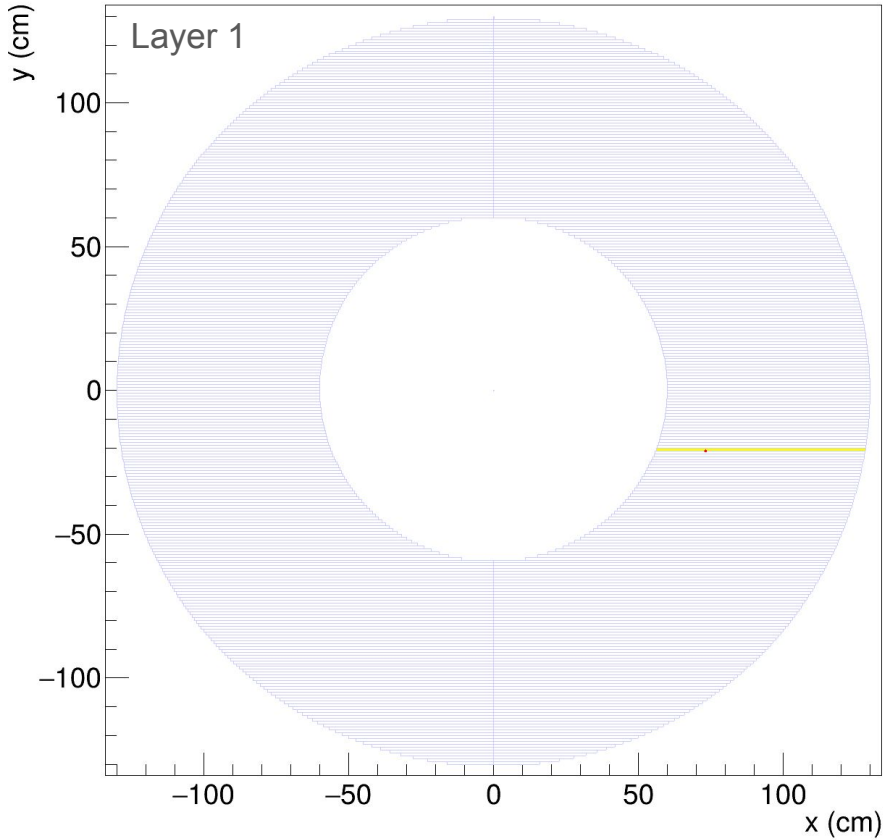




- Too high occupancy… Consider shorter segmented strips?

Single-pion box generator



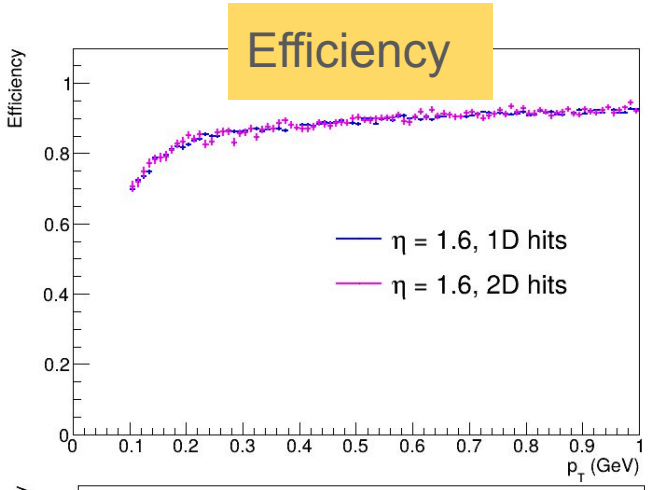


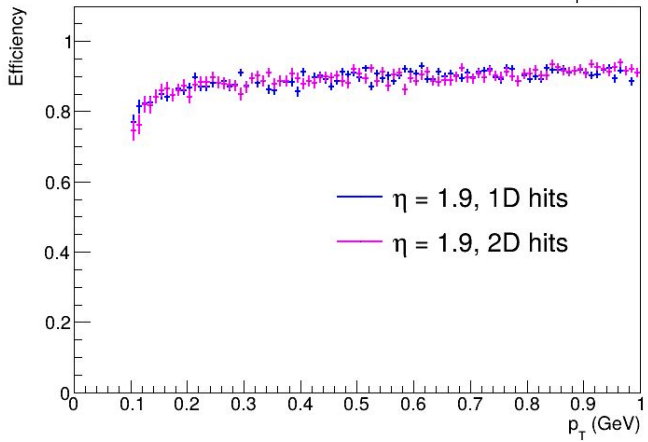
- Let’s first check 1D tracking with single pion box generator

2D-vs-1D tracking performance (single pion boxgen)

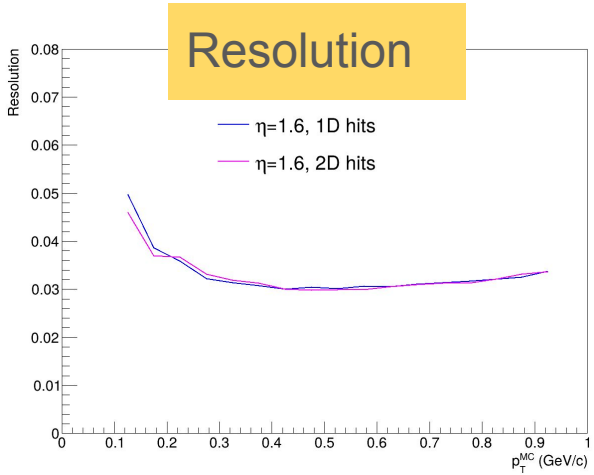
- Single pion boxgen
- TPC+FTD tracking, resolution after refit
- Perfect agreement between 1D and 2D tracking both for track resolution and reconstruction efficiency
- Tools are ready for the forward detector optimization with realistic 1D-tracking algorithms

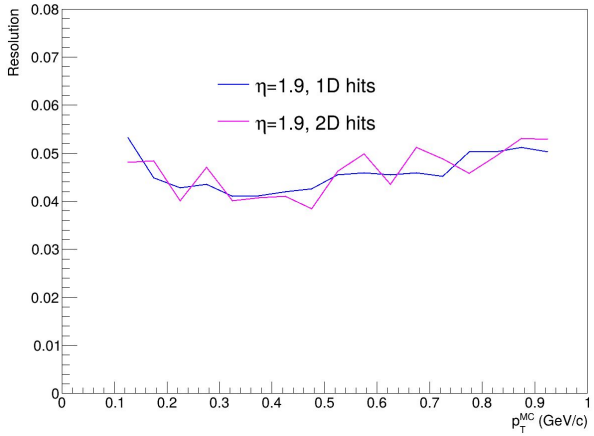
Efficiency



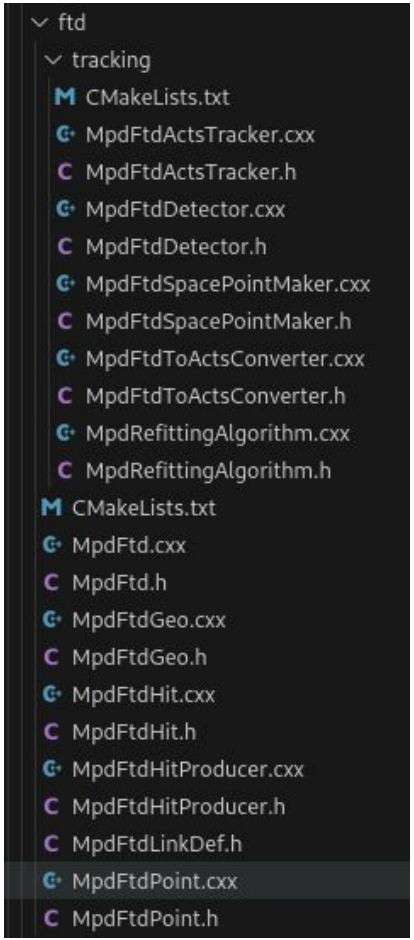


Resolution





Code structure



New detector library (ftd) added under detectors subfolder

Code structure:

- MpdFtd.h/cxx - FTD geometry and MC point processing
- MpdFtdGeo.h/cxx - FTD geometry settings
- MpdFtdPoint.h/cxx - MC point container for FTD
- MpdFtdHit.h/cxx - reco hit container for FTD
- MpdFtdHitProducer.h/cxx - ideal hit producer for FTD
- Tracking subfolder (compiled if ACTS package is found):
  - MpdFtdDetector.h/cxx - ACTS geometry (TPC + FTD)
  - MpdFtdToActsConverter.h/cxx - converts mpdroot info to ACTS format
  - MpdFtdActsTracker.h/cxx - FairTask: ACTS tracking in TPC and/or FTD
  - MpdRefittingAlgorithm.h/cxx - Refitting with GlobalChi2 fitter (+mass hypothesis)
  - MpdFtdSpacePointMaker.h/cxx - Making space points from 1D hits in FTD

Running procedure:

```
FairTask* ftdHitProducer = new MpdFtdHitProducer();
fRun->AddTask(ftdHitProducer);

FairTask* ftdActsTracker = new MpdFtdActsTracker();
fRun->AddTask(ftdActsTracker);
```

Algorithms used in MpdFtdActsTracker

Trying to use native ACTS algorithms when possible

```
// create algorithms
fConverter = new MpdFtdToActsConverter(converterCfg, logLevel);
fSpacePointMaker = new ActsExamples::MpdFtdSpacePointMaker(spCfg, logLevel);
fSeedingAlgorithm = new ActsExamples::SeedingAlgorithm(seedingCfg, logLevel);
fTrackParamsEstimationAlgorithm = new ActsExamples::TrackParamsEstimationAlgorithm(paramsEstimationCfg, logLevel);
fTrackFindingAlgorithm = new ActsExamples::TrackFindingAlgorithm(trackFindingCfg, logLevel);
if (fDoRefit) fTrackRefittingAlgorithm = new ActsExamples::MpdRefittingAlgorithm(trackRefitCfg, logLevel);
fTrackTruthMatcher = new ActsExamples::TrackTruthMatcher(trackTruthMatcherCfg, logLevel);
fRootParticleWriter = new ActsExamples::RootParticleWriter(particleWriterCfg, logLevel);
fRootSimHitWriter = new ActsExamples::RootSimHitWriter(simhitWriterCfg, logLevel);
fRootMeasurementWriter = new ActsExamples::RootMeasurementWriter(measWriterCfg, logLevel);
fRootSpacepointWriter = new ActsExamples::RootSpacepointWriter(spWriterCfg, logLevel);
fRootSeedWriter = new ActsExamples::RootSeedWriter(seedWriterCfg, logLevel);
fRootTrackStatesWriter = new ActsExamples::RootTrackStatesWriter(trackStatesWriterCfg, logLevel);
fRootTrackSummaryWriter = new ActsExamples::RootTrackSummaryWriter(trackSummaryWriterCfg, logLevel);
if (fDoRefit) fRootTrackRefitSummaryWriter = new ActsExamples::RootTrackSummaryWriter(trackRefitSummaryWriterCfg, logLevel);
```

Conclusions and outlook

- Good performance with the improved sector-based TPC geometry
- Seeding still needs further tuning
- Developed custom refitting algorithm supporting KF and Global χ^2:
  - Possibility to set a particle hypothesis → correct accounting for material effects
  - Improved resolution comparable with the standard KF results
- Working towards realistic strip-like FTD geometry with 1D tracking
  - Tools are ready for detector optimization

Backup

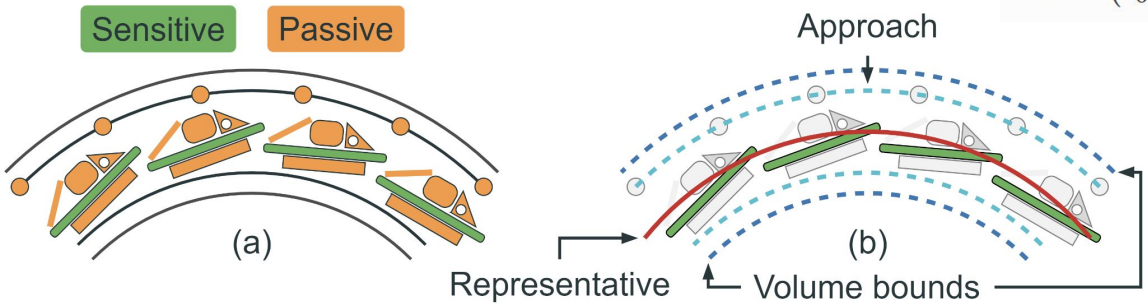
ACTS project

https://acts.readthedocs.io/

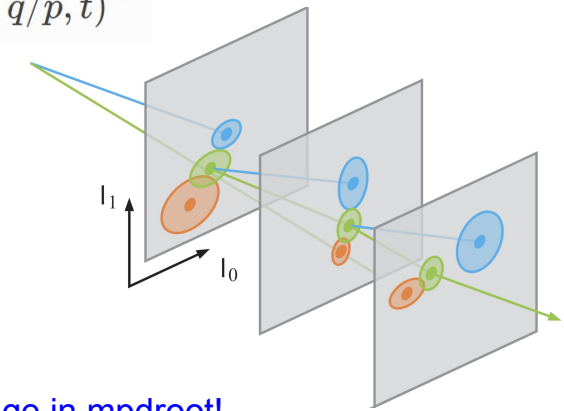
- A Common Tracking Software project
- Contains:
  - Box generator or interface to read external particles
  - Fatras (fast simulation tool) or interface to read hits
  - Digitization algorithm (smearing etc)
  - Seeding (several algorithms, including truth seeding)
  - Track finding/fitting with Combinatorial KF
- Accounting for energy losses, multiple scattering etc.
- Supporting multi-core execution, GPU etc.

```
// Start sequencer
ActsExamples::Sequencer sequencer(sequencerCfg);

if (inputDir.Contains("none")){ // particle gun + fartras simulation
  sequencer.addReader(std::make_shared<ActsExamples::EventGenerator>(evgenCfg, logLevel));
  sequencer.addElement(std::make_shared<ActsExamples::FatrasSimulation>(fatrasCfg, logLevelFatras));
} else { // read particles and hits from input file
  sequencer.addReader(std::make_shared<ActsExamples::RootParticleReader>(particleReaderCfg, logLevel));
  sequencer.addReader(std::make_shared<ActsExamples::RootSimHitReader>(simhitReaderCfg, logLevel));
}
sequencer.addAlgorithm(std::make_shared<ActsExamples::DigitizationAlgorithm>(digiCfg, logLevelDigi));
sequencer.addAlgorithm(std::make_shared<ActsExamples::SpacePointMaker>(spCfg, logLevel));
sequencer.addAlgorithm(std::make_shared<ActsExamples::SeedingAlgorithm>(seedingCfg, logLevelSeed));
sequencer.addAlgorithm(std::make_shared<ActsExamples::TrackParamsEstimationAlgorithm>(paramsEstimationCfg, logLevel));
sequencer.addAlgorithm(std::make_shared<ActsExamples::TrackFindingAlgorithm>(trackFindingCfg, logLevelFinder));
sequencer.addAlgorithm(std::make_shared<ActsExamples::TrackTruthMatcher>(trackTruthMatcherCfg, logLevelMatcher));
sequencer.addWriter(std::make_shared<ActsExamples::RootParticleWriter>(particleWriterCfg, logLevel));
sequencer.addWriter(std::make_shared<ActsExamples::RootSimHitWriter>(simhitWriterCfg, logLevel));
sequencer.addWriter(std::make_shared<ActsExamples::RootMeasurementWriter>(measWriterCfg, logLevelMeasWriter));
sequencer.addWriter(std::make_shared<ActsExamples::RootSpacepointWriter>(spWriterCfg, logLevel));
sequencer.addWriter(std::make_shared<ActsExamples::RootSeedWriter>(seedWriterCfg, logLevel));
sequencer.addWriter(std::make_shared<ActsExamples::RootTrackStatesWriter>(trackStatesWriterCfg, logLevel));
sequencer.addWriter(std::make_shared<ActsExamples::RootTrackSummaryWriter>(trackSummaryWriterCfg, logLevel));
```



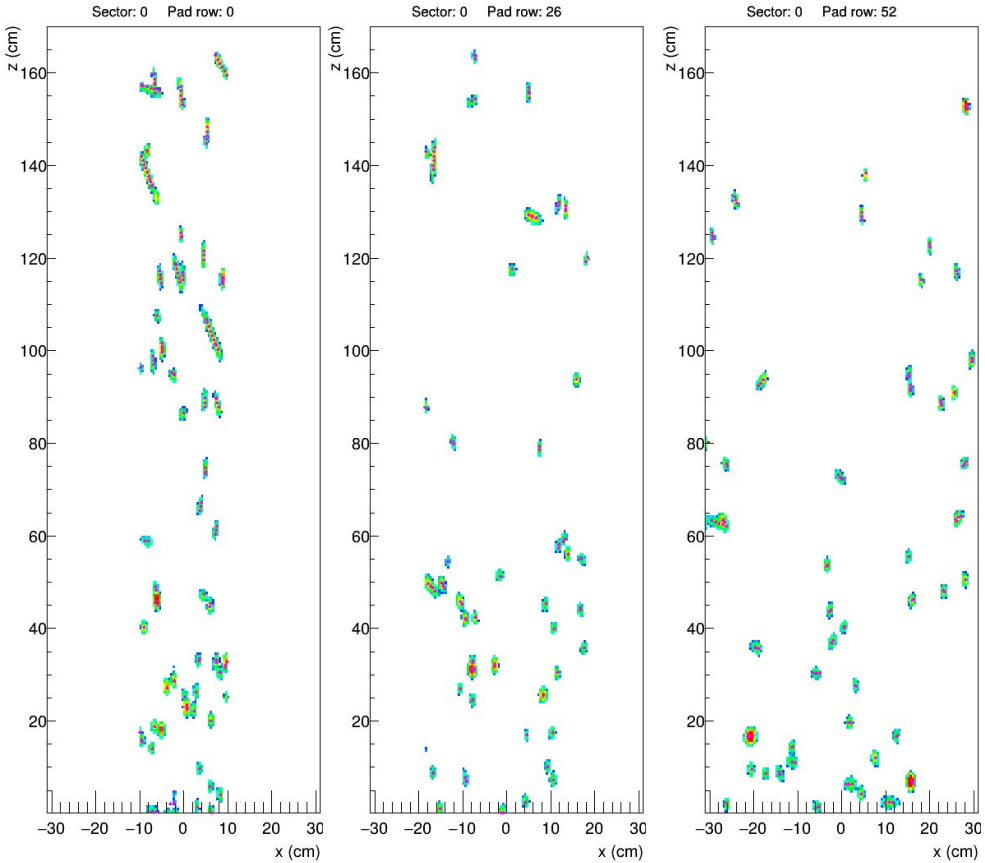
$$\vec{x} = (l_0, l_1, \phi, \theta, q/p, t)^T$$



Using latest v38.1 from nicadist

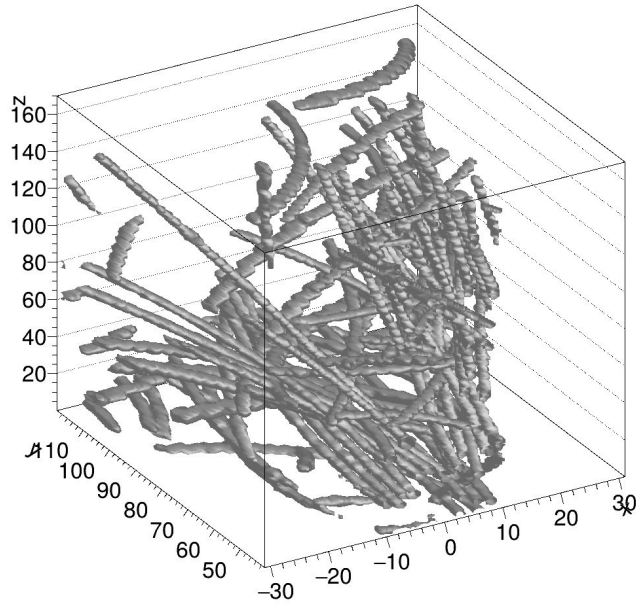
Many thanks to Slavomir Hnatic and Jan Busa for integration of latest ACTS package in mpdroot!

Reminder: digits in TPC



Digits produced by MpdTpcDigitizerAZIt*:

- Pad rows in local y: 12 mm (inner) 18 mm (outer)
- Pad size in local x: 5 mm
- Time bin 100 ns -> z-bin size: 5.5 mm

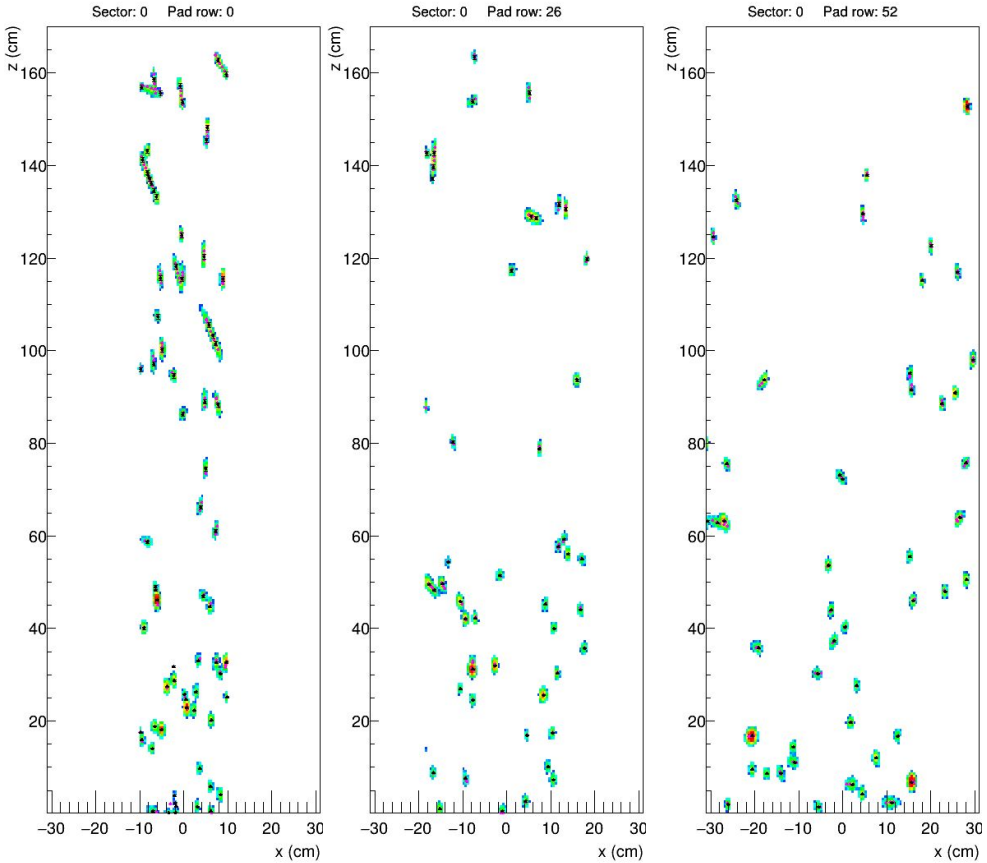


Central URQMD event: Au-Au @ 11 GeV

MC points in magenta

* Core developer: A. Zinchenko

Reminder: hits in TPC



Digits produced by MpdTpcDigitizerAZIt*:

- Pad rows in local y: 12 mm (inner) 18 mm (outer)
- Pad size in local x: 5 mm
- Time bin 100 ns -> z-bin size: 5.5 mm

Clusters and hits produced by TpcClusterHitFinderMlem*:

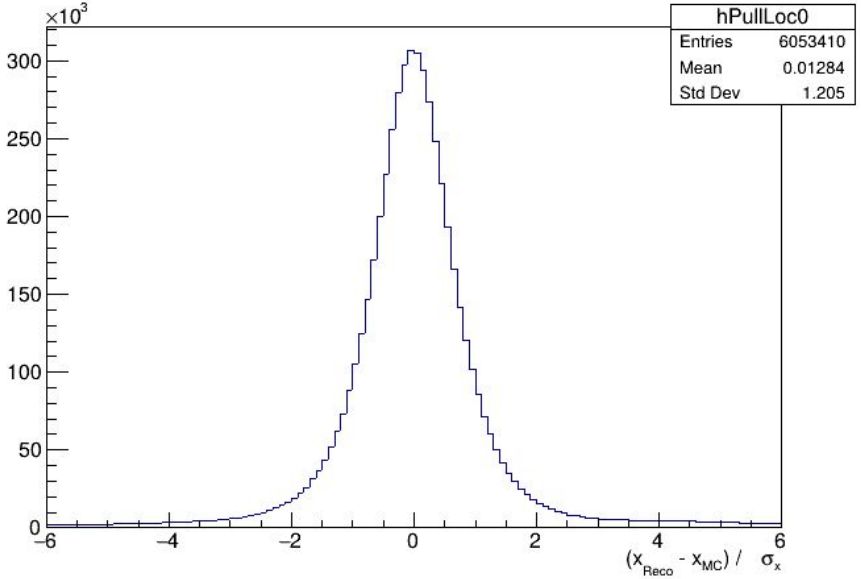
- Several hits produced for complex clusters
- Nice matching between hits and MC points
- Typical assigned uncertainty in local x: 0.25 mm
- Typical assigned uncertainty in local z: 1 mm

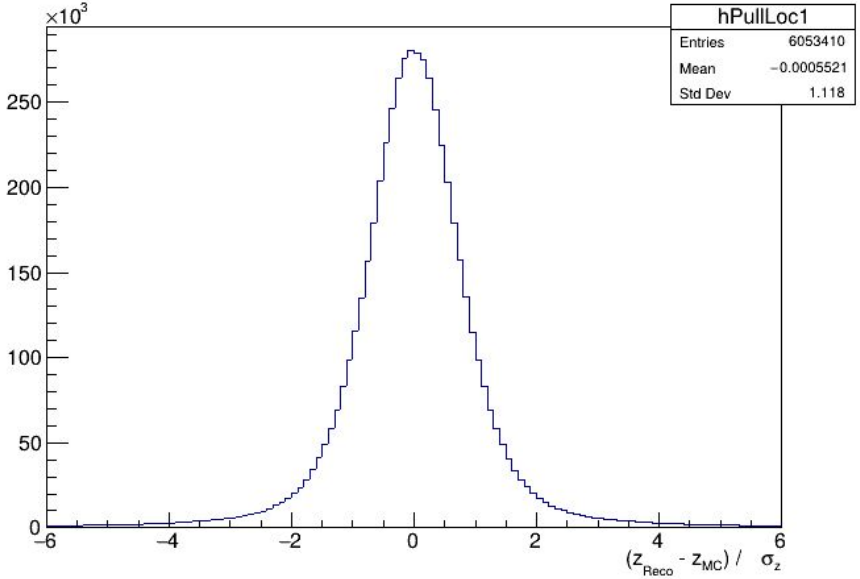
Central URQMD event: Au-Au @ 11 GeV

MC points in magenta, hits in black

* Core developer: A. Zinchenko

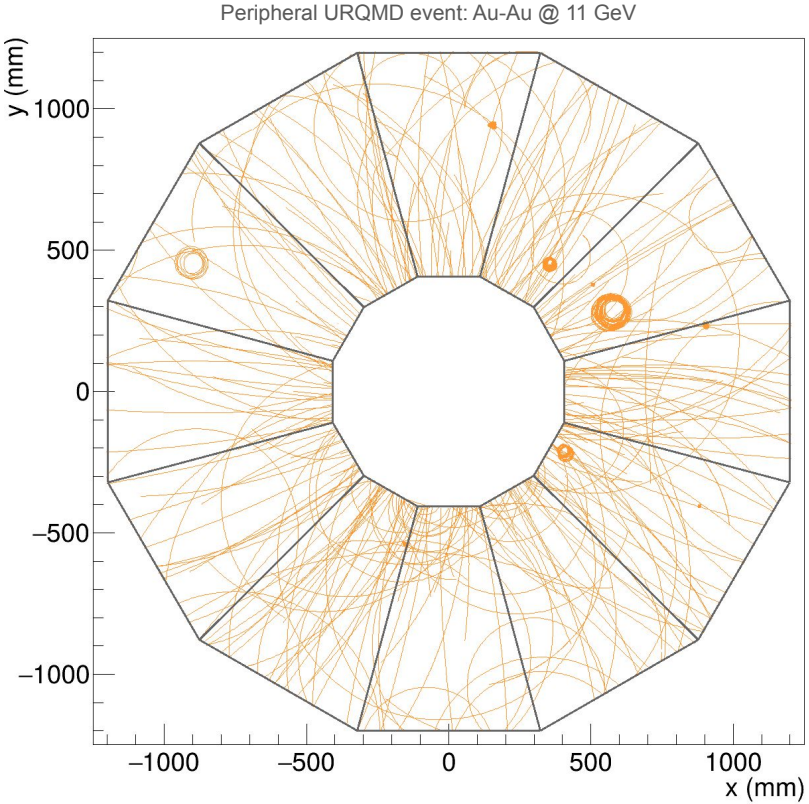
Measurement pulls

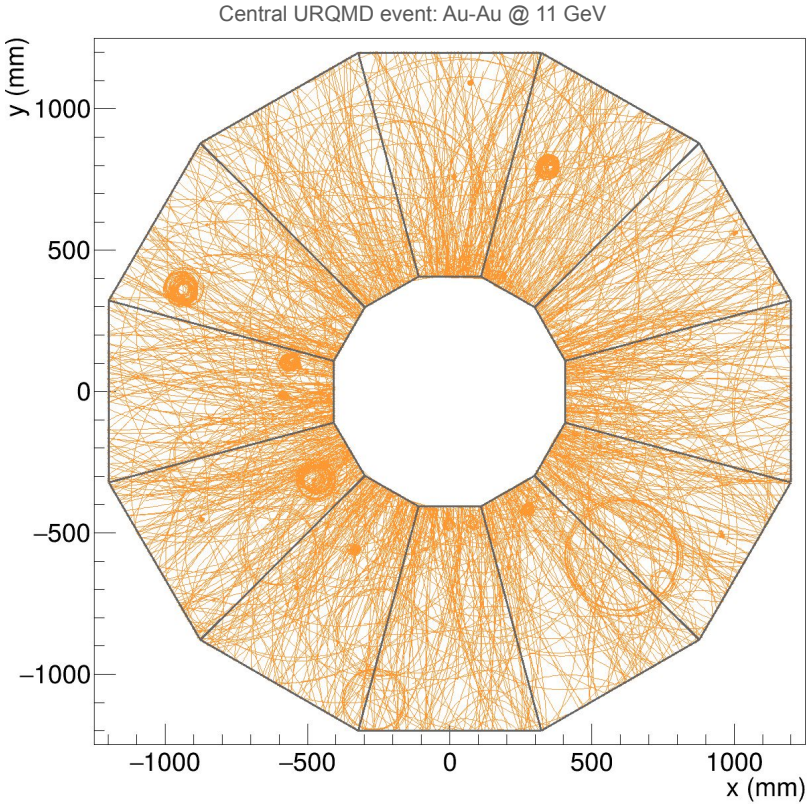




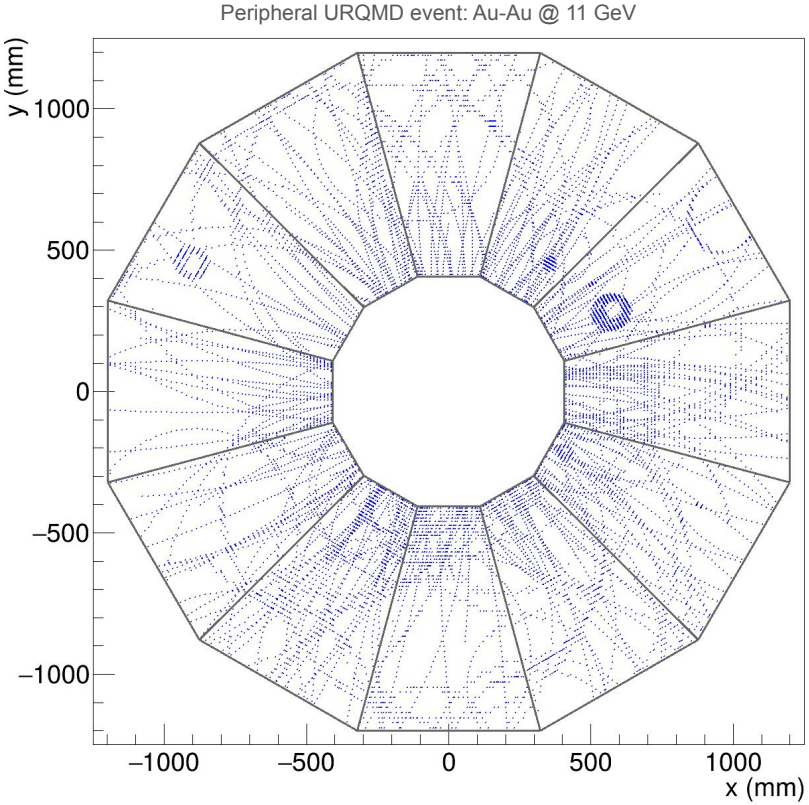
- Pull = difference between reco and MC point / estimated uncertainty
- Allow to control biases and validate uncertainty estimation
- On average: no significant bias neither in x nor in z
- On average: pull widths close to 1
- -> Reasonably good for tracking

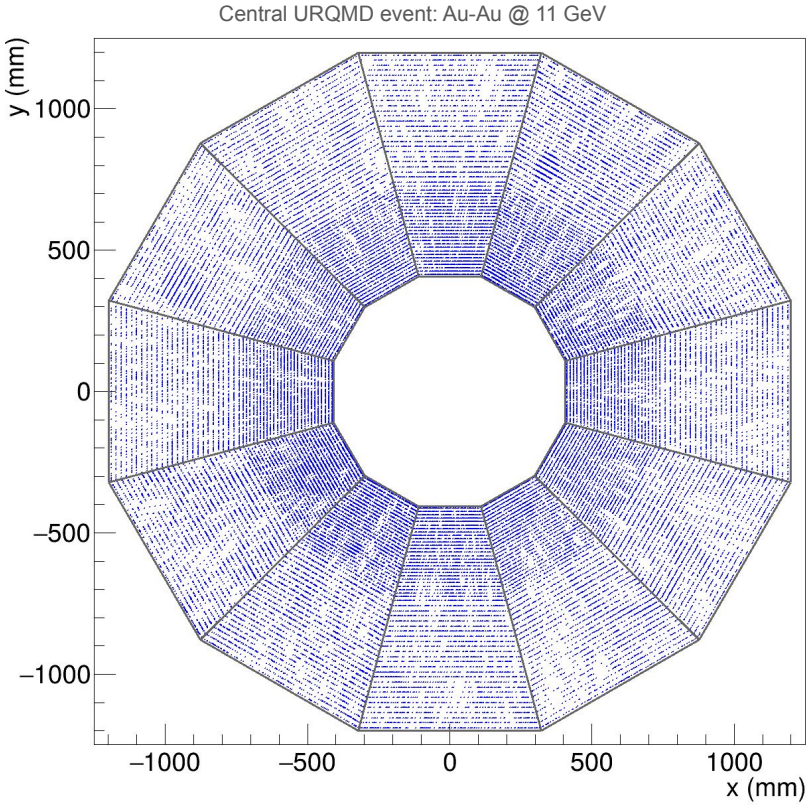
Typical MC point distributions



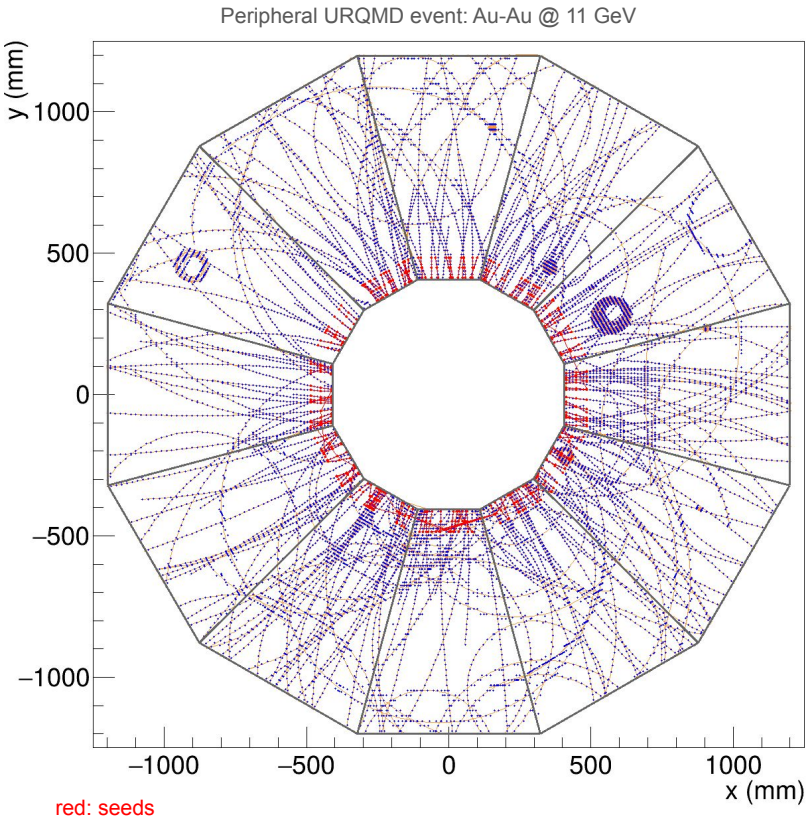


Typical reconstructed hit distributions

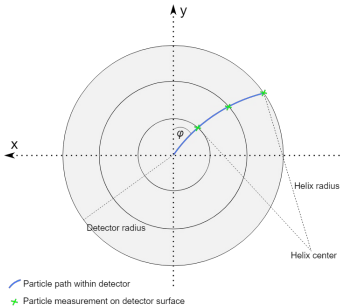


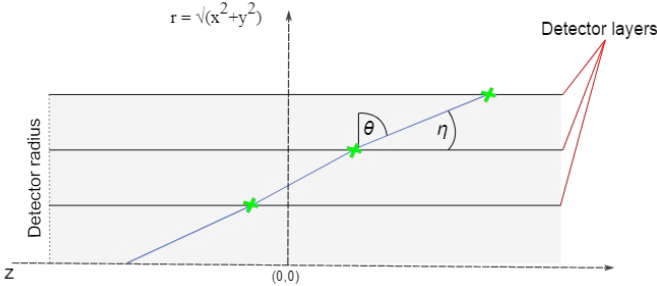


Seeding algorithm



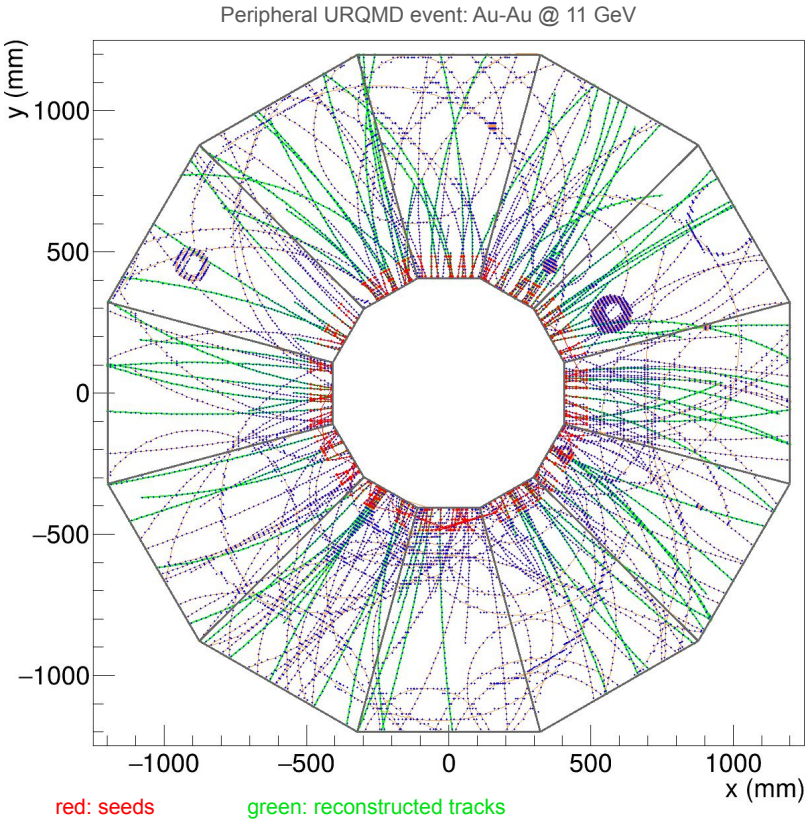
- Selecting triplets of hits from layer 0, 3 and 6
- Seeding algorithm checks:
  - xy plane: helix pointing to (x,y) ~ (0,0).
  - rz plane: angular difference between two doublets consistent with expected mult. scattering
  - selection on impact parameter in r and z directions
- Had to relax most of the selection parameters





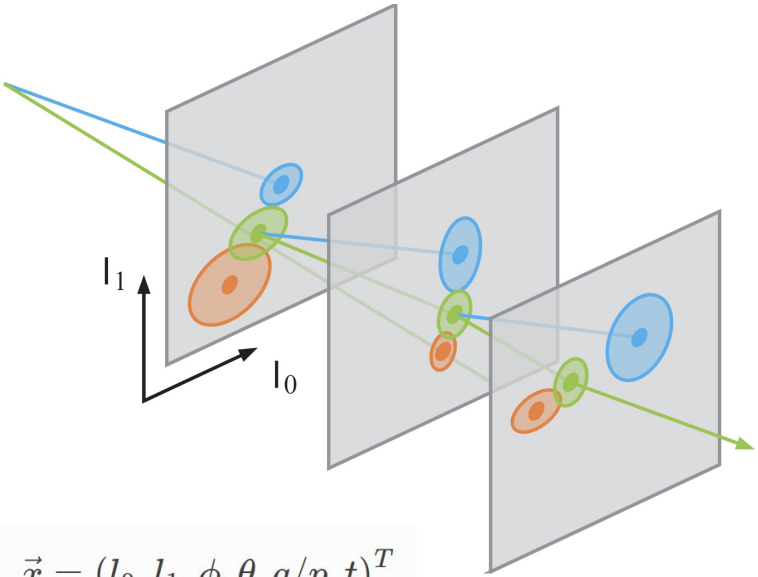
- Still some missing seeds for “good” tracks
- Need further tuning

Track finding algorithm



- Failed track finding for many seeds (propagation errors)
- Need further tuning

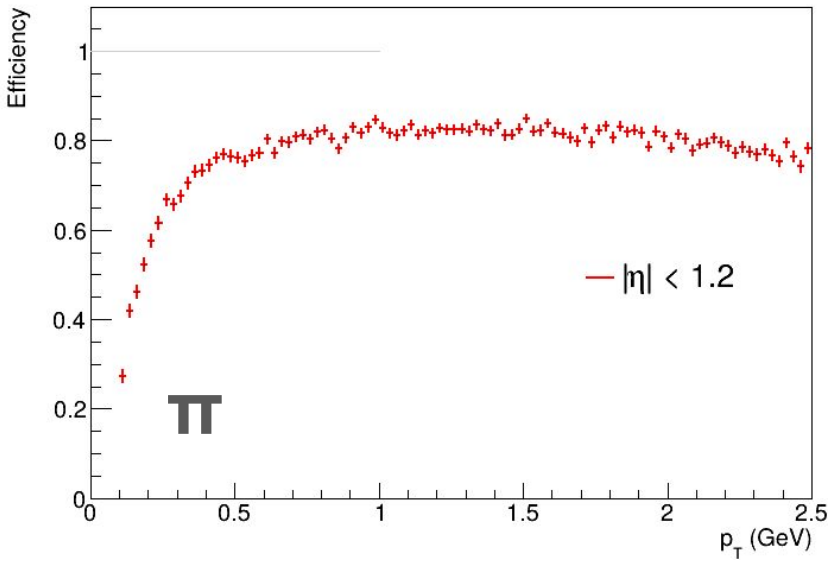
- Combinatorial Kalman Filter using initial track parameters from seeds
- Backward refit + smoothing



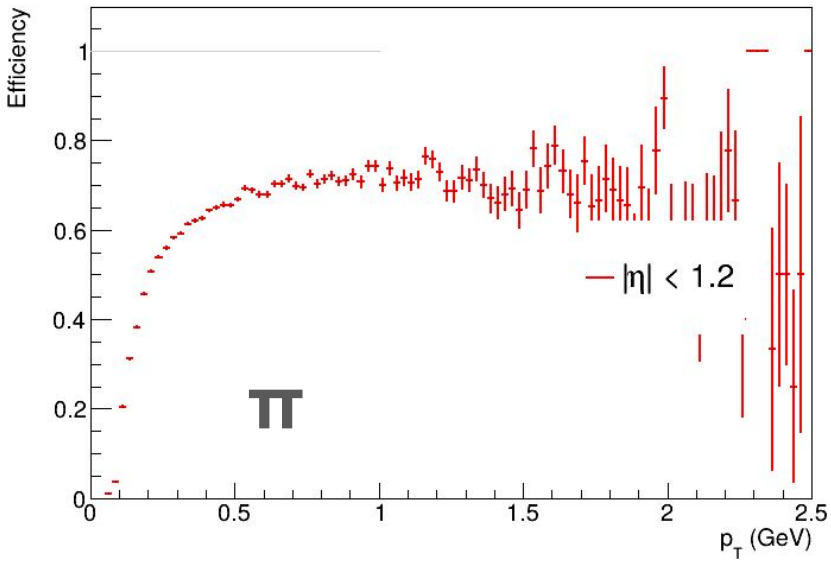
$$\vec{x} = (l_0, l_1, \phi, \theta, q/p, t)^T$$

Tracking efficiency

Single track box generator

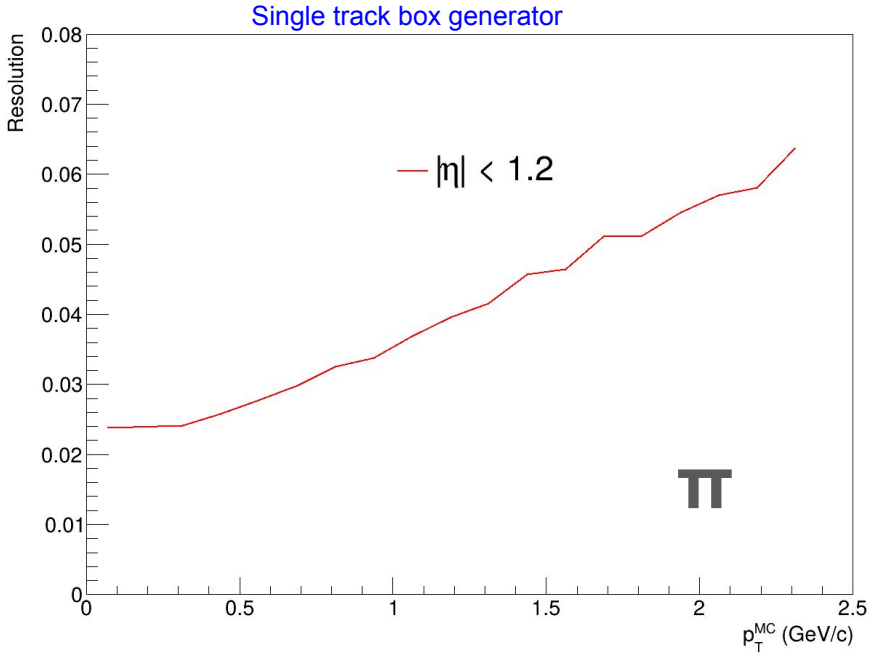


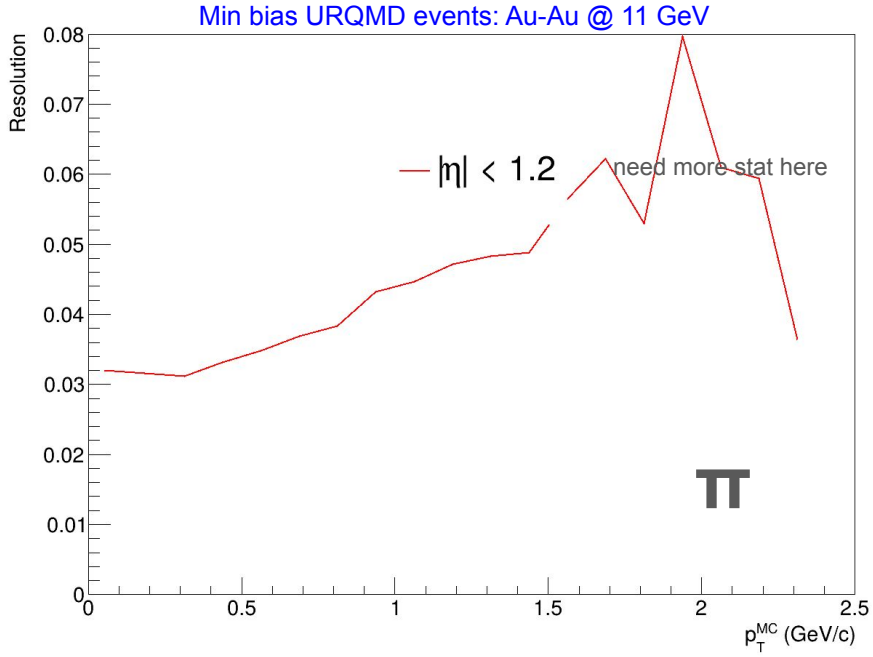
Min bias URQMD events: Au-Au @ 11 GeV



- Efficiency shown for “trackable” charged primary particles:
  - at least 20 hits
  - require hits at layer 0, 3 and 6
- Significant efficiency losses, especially at low p_T:
  - Due to seeding inefficiencies
  - Due to failed extrapolation errors
- Visible degradation of efficiency with UrQMD compared to clean single track events

Momentum resolution





- Visible degradation of momentum resolution with UrQMD
- Resolution a bit worse compared to the standard KF performance
  - was the plot produced with realistic digitizer/clusterizer?

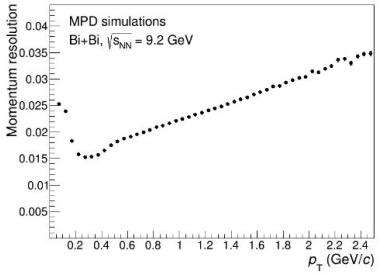


Illustration of TPC-only fit for pions at eta = 1.9