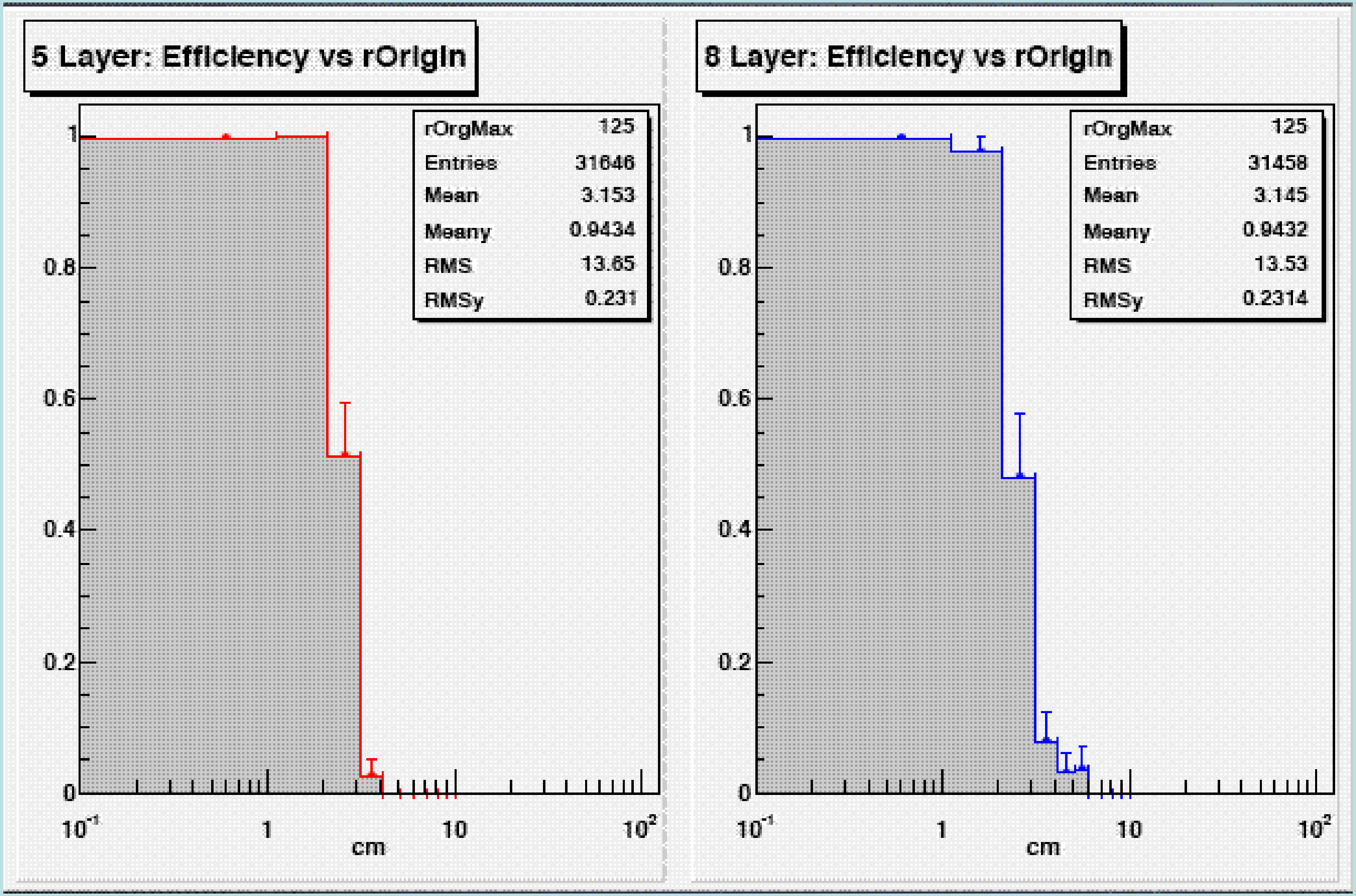
Reconstruction of Non-Prompt Tracks Using a Standalone Barrel Tracking Algorithm

Overview

- Examine efficiency of reconstruction of non-prompt tracks with outside-in tracker barrel reconstruction algorithm
- Outside-in reconstruction algorithm written by Tim Nelson at SLAC
- Optimized by us for use as “cleanup” code with inside-out algorithm

Efficiency vs. rOrigin with VXDBasedReco



Motivation

- Vertex-based reconstruction (such as VXDBasedReco) misses tracks that originate outside innermost layers of vertex detector, ~5%
- An algorithm that can pick up a significant portion of these missed tracks will be an important part of a final reconstruction package

Cheater

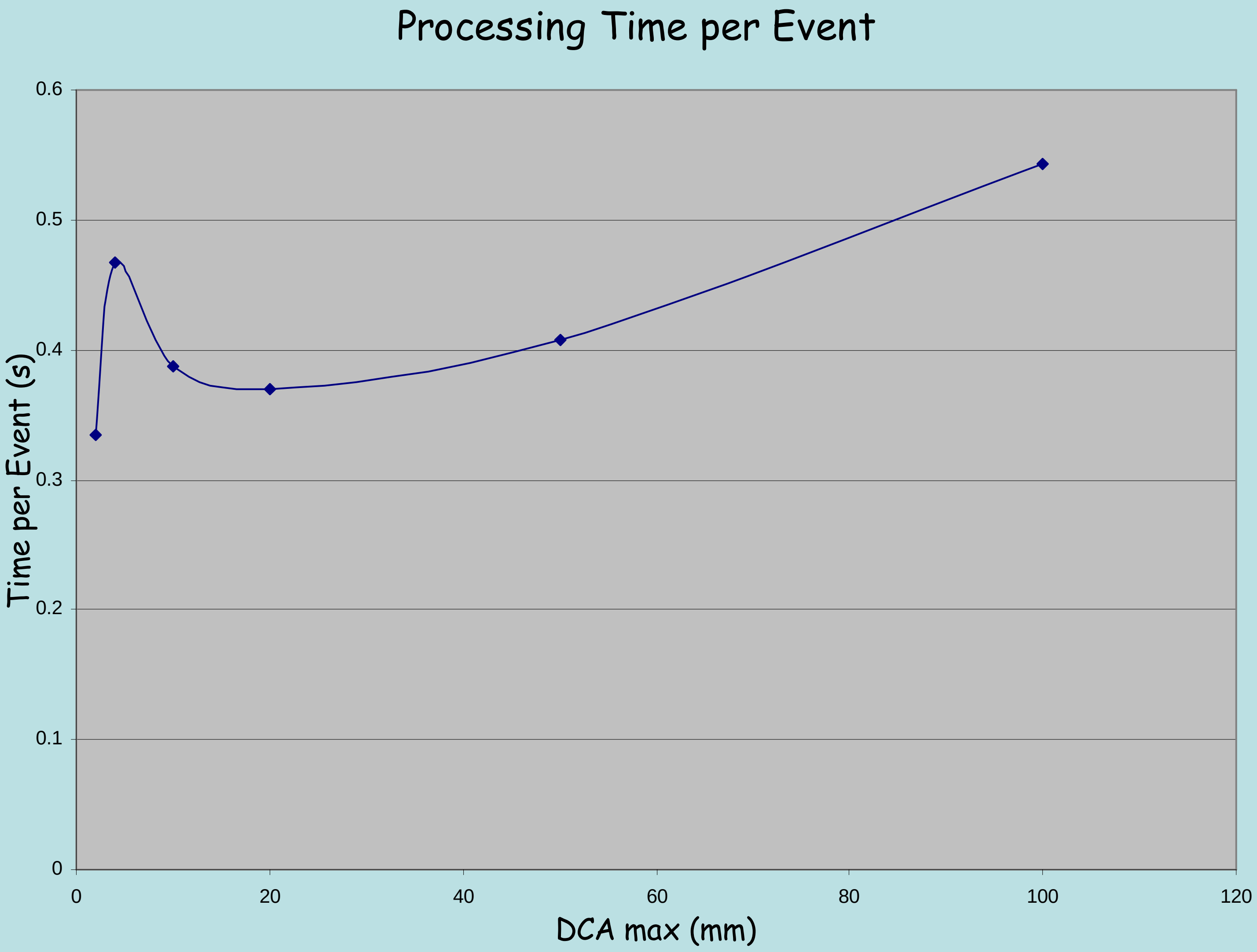
- VXDBasedReco has not yet been ported to org.lcsim framework, so…
- Wrote “cheater” to emulate perfectly efficient VXDBasedReco; assume anything that can be found by VXDBasedReco is found and the hits flagged as used
- Loops over TkrBarrHits and MCParticles, finds particles with rOrigin < 20mm and hits from those particles, removes them from collections
- rOrigin defined as sqrt(particle.getOriginX()^2 + particle.getOriginY()^2)

AxialBarrelTrackFinder

- Loops over all hits in each layer, from the outside in, and finds 3 “seed” hits, one per layer
- Performs CircleFit to seed hits
- If successful, looks for hits on the remaining layers that can be added to seed fit, refitting after each hit added.
- If at least 4 hits on track, and Chi^2 of fit reasonable, creates track object and adds to collection

DCA cut

- Original code placed tight limitation on dca of seed fit to IP, in order to make combinatorics manageable. Limiting the number of hits that need to be considered allows for easing this restriction without drastically increasing computation time.
- Easing the max dca from 2mm to 100mm almost doubled efficiency, with computation time staying under 1 sec per event. However, also produces significant increase in reconstruction of “fake tracks”.


Processing Time per Event
Time per Event (s)
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20
40
60
80
100
120
DCA max (mm)

MC Track Association

- Added a constructor to StandaloneAxialBarrelTrack to include association with majority MC particle and track purity.
- MCParticle associated with a given track is just majority particle - particle associated with largest number of hits used to create that track
- Not always meaningful; if a track has four hits from four different particles, majority particle will just be the one associated with the first hit in the list
- In order for efficiency analysis code to count a track as found, it must have a purity (# hits from majority particle/#hits in track) > .74
- Only counts one found track per MCParticle. If there are several tracks associated with the same particle, only counts the first one.
- Tracks with purity <.74 and successful fit counted as fake tracks

Z segmentation

- Z segmentation logic as written requires each new hit to be within half a module length of a line from the origin to the first hit used.
- Requires a certain stiffness of tracks, excludes a low momentum regime that we wanted to include in our analysis, so commented out segmentation logic.
- Did maintain requirement of same sign in z

Other Modifications

- Code as written was associating same set of hits with many tracks. Turns out this was mainly superficial (wasn't affecting the properties of the tracks themselves), problem seems to be with scope of variable storing set of hits. Fixed.
- Algorithm creates multiple tracks with same first 1 or 2 seed hits; once it picks 2 hits, it will make a track with every workable combination of those 2 with a third. Not sure if this is intentional or not; could add a few lines to adjudicate between multiple tracks using same hits, e.g. pick one with best chi^2. Added temporary fix to only allow single use of each hit.

Fiducial Volume

For found tracks/MC denominator

- 20mm < Rorg < 400mm (>3 hit layers)
- Pt > .75 GeV (>3 hit layers)
- Cos(theta) < 0.5
- Final state, not backscatter

For Fake Tracks

- Dca < 100mm
- Cos(theta) < 0.5 (oops!)

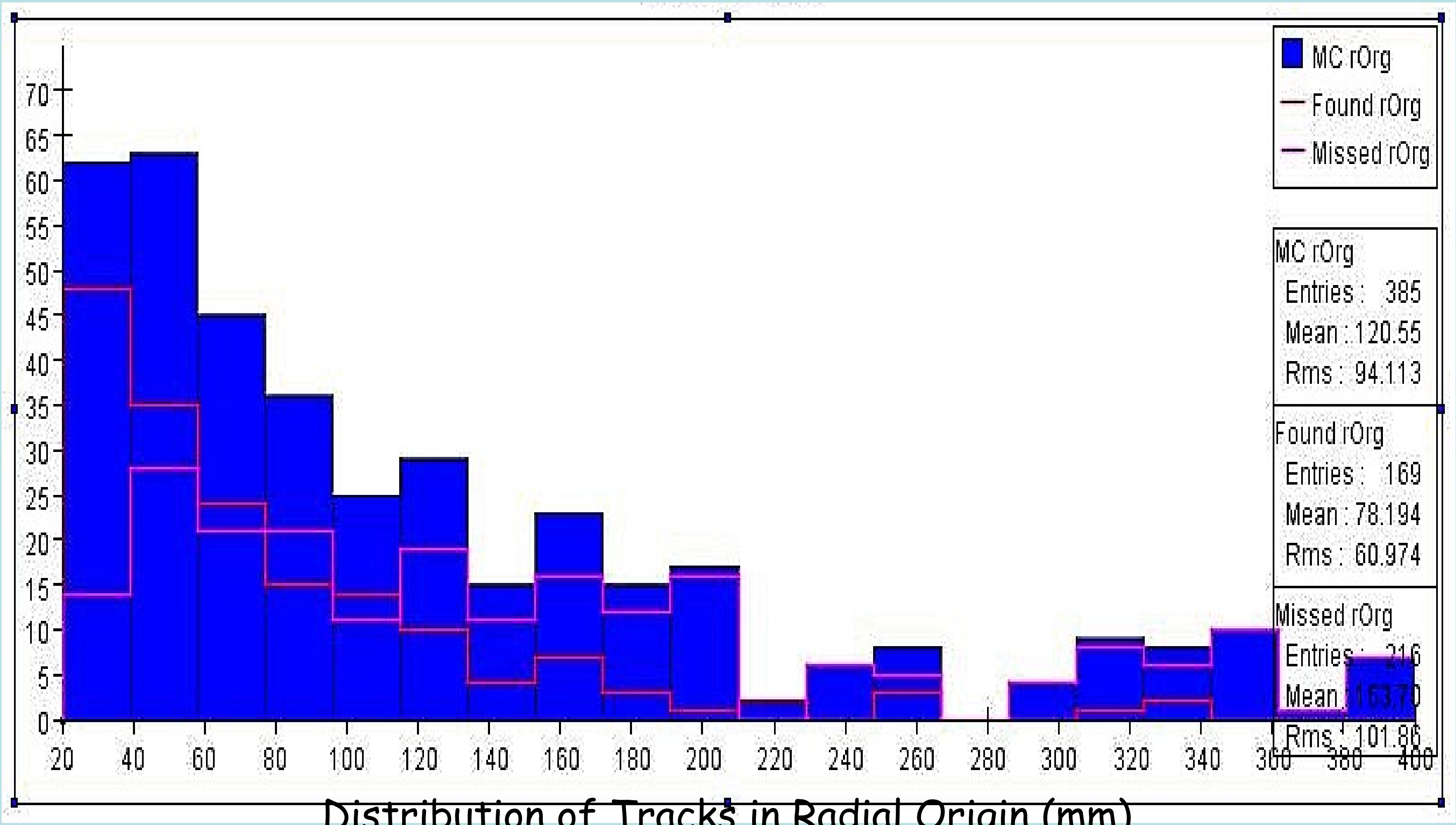
Fake Tracks

- Large number of fake tracks compared with found tracks is problematic
- Still trying to find reason for so many fake tracks; they don’t all seem to be coming from loopers.
- Working on analysis of distribution in r and z of hits contributing to fake tracks, and distribution in rOrigin and momentum of MC tracks that contribute hits to fake tracks.

Conclusions…

- Efficiency overall looks good; using AxialBarrelTrackFinder to clean up non-prompt tracks looks promising.
- Need to figure out where fake tracks are coming from
- Next steps:
- Replace cheater with VXDBasedReco, or similar algorithm
- Implement z segmentation. Is there available code for performing helical fits?
- Try running barrel tracking code in two steps, once with restricted dca cut, again with it opened up
- Modify standalone barrel tracking to add leftover VXD hits to tracks
- Examine potential use of calorimeter-based stubs
- Compare efficiency and fake rates with 5 layers and 8 layers

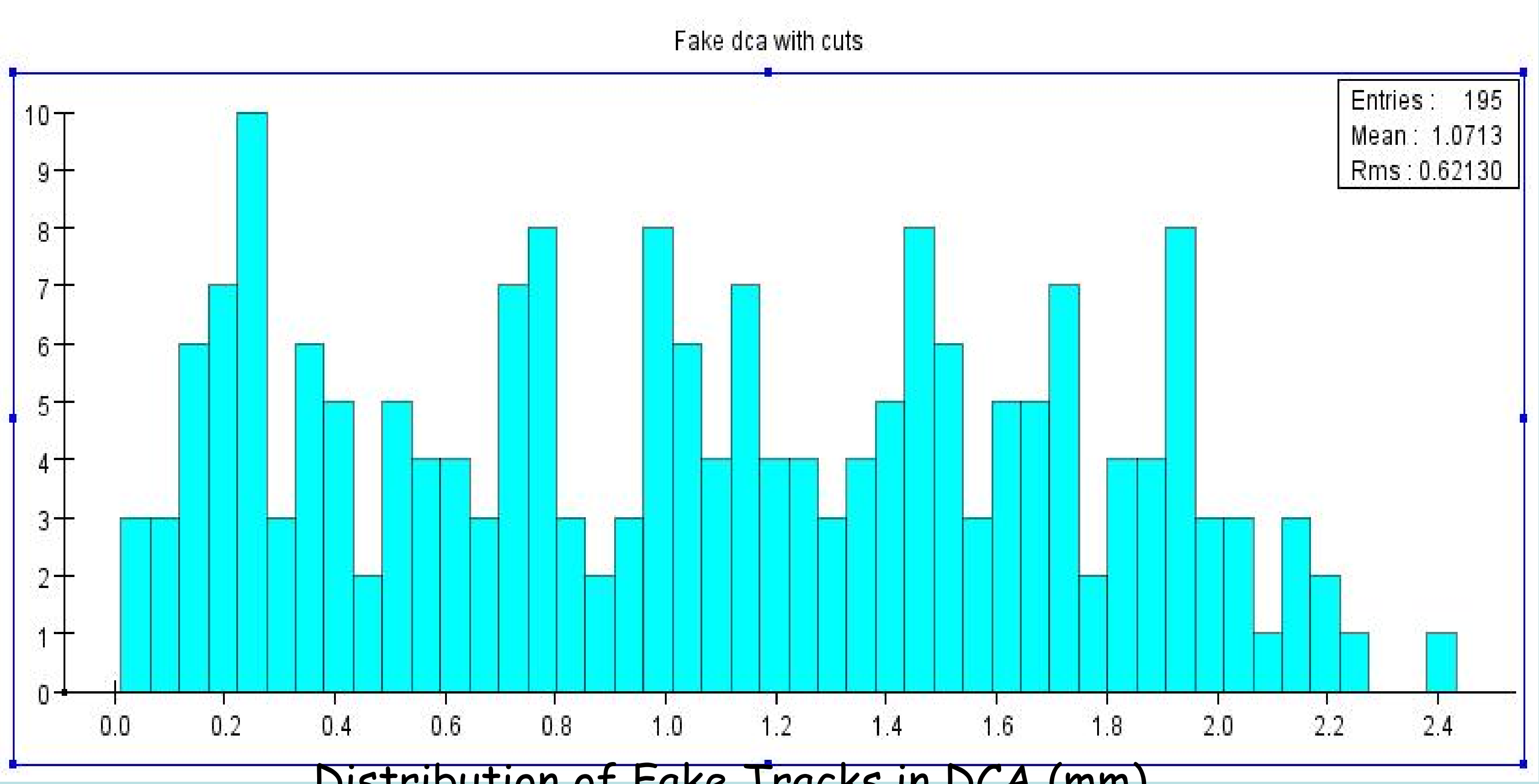
Distance of Closest Approach < 2.0 mm



Distribution of Tracks in Radial Origin (mm)

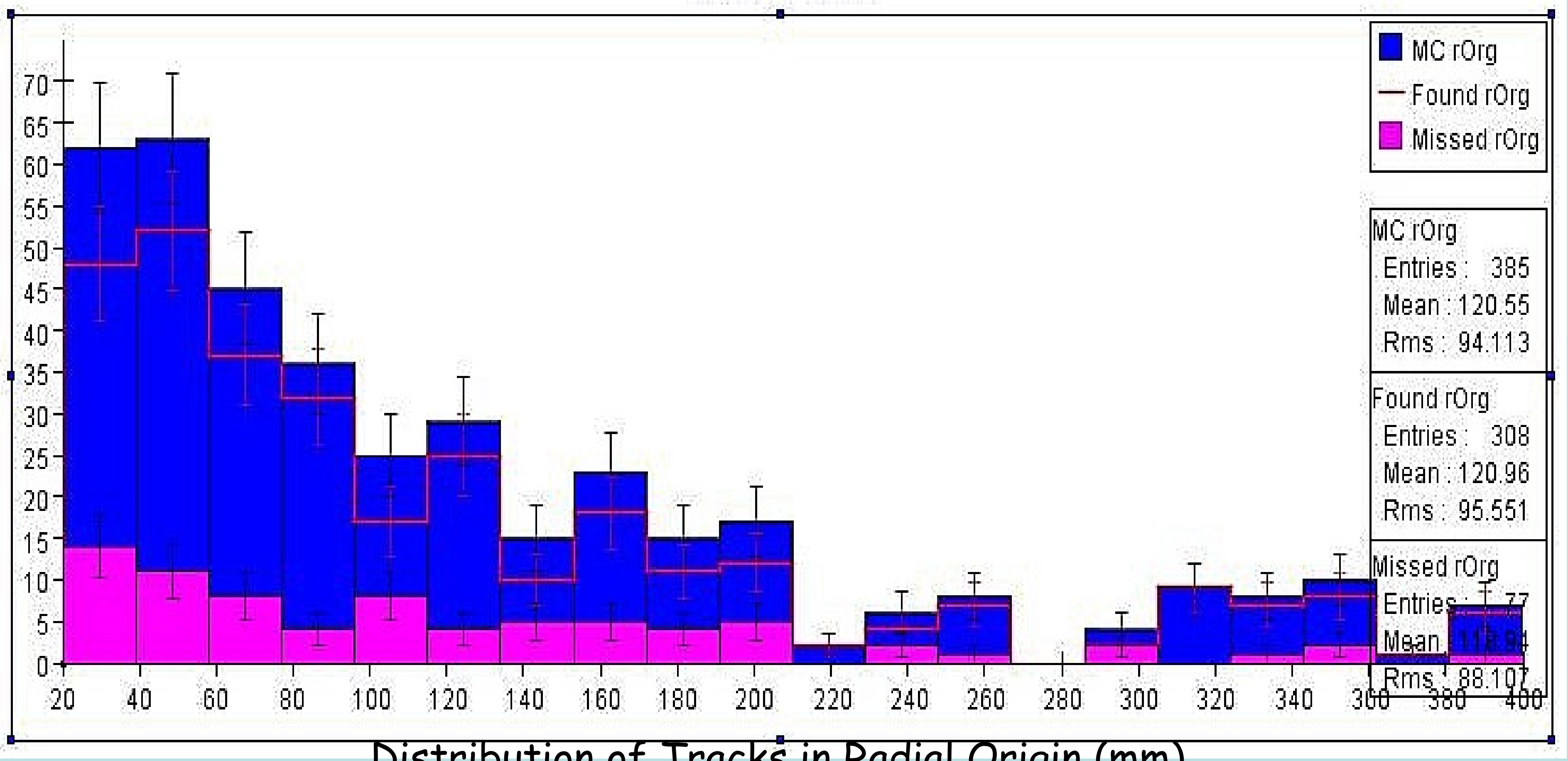
Efficiency ~ 45 %

Distance of Closest Approach < 2.0 mm



Distribution of Fake Tracks in DCA (mm)

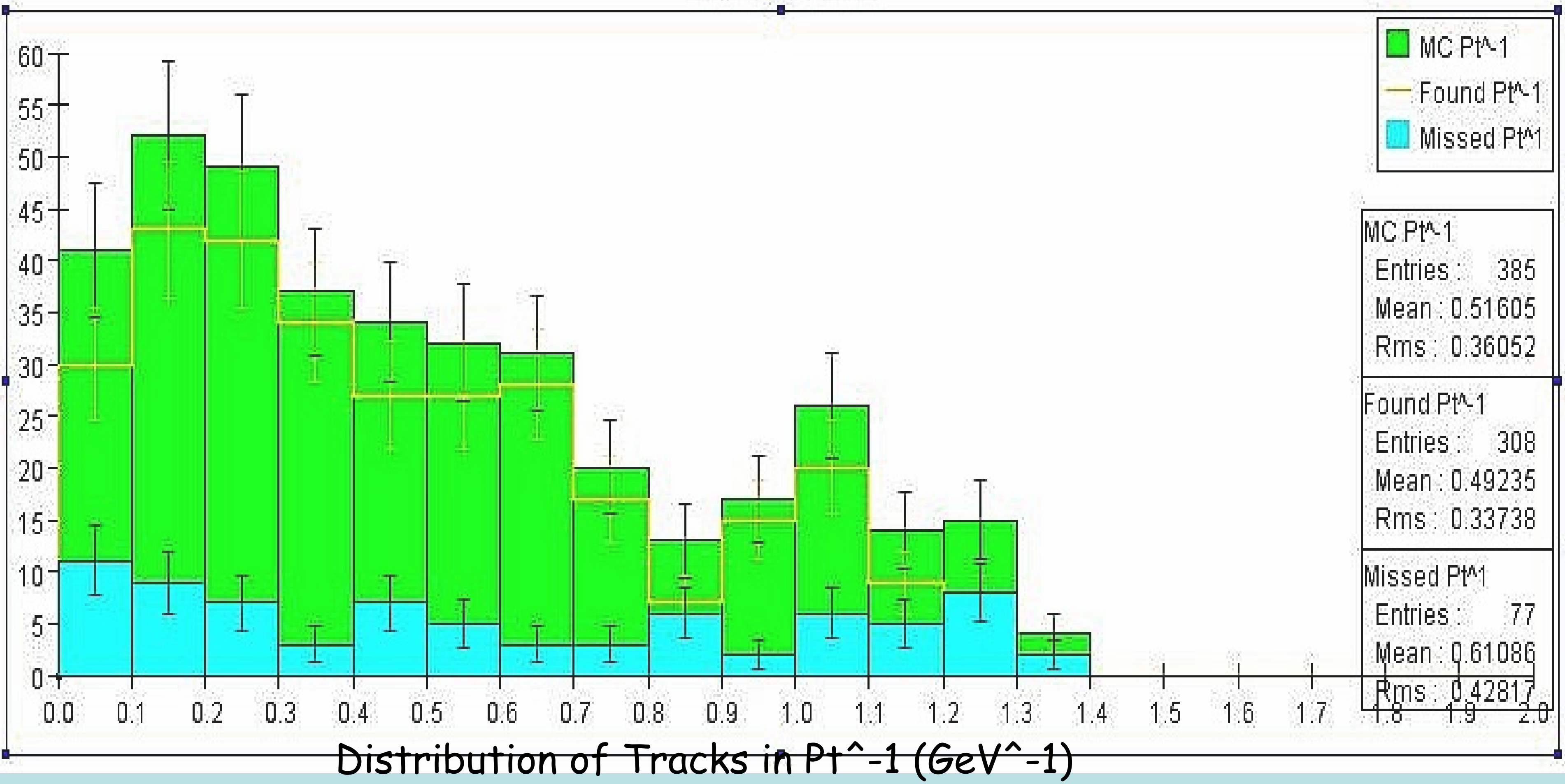
Distance of Closest Approach < 100mm



Distribution of Tracks in Radial Origin (mm)

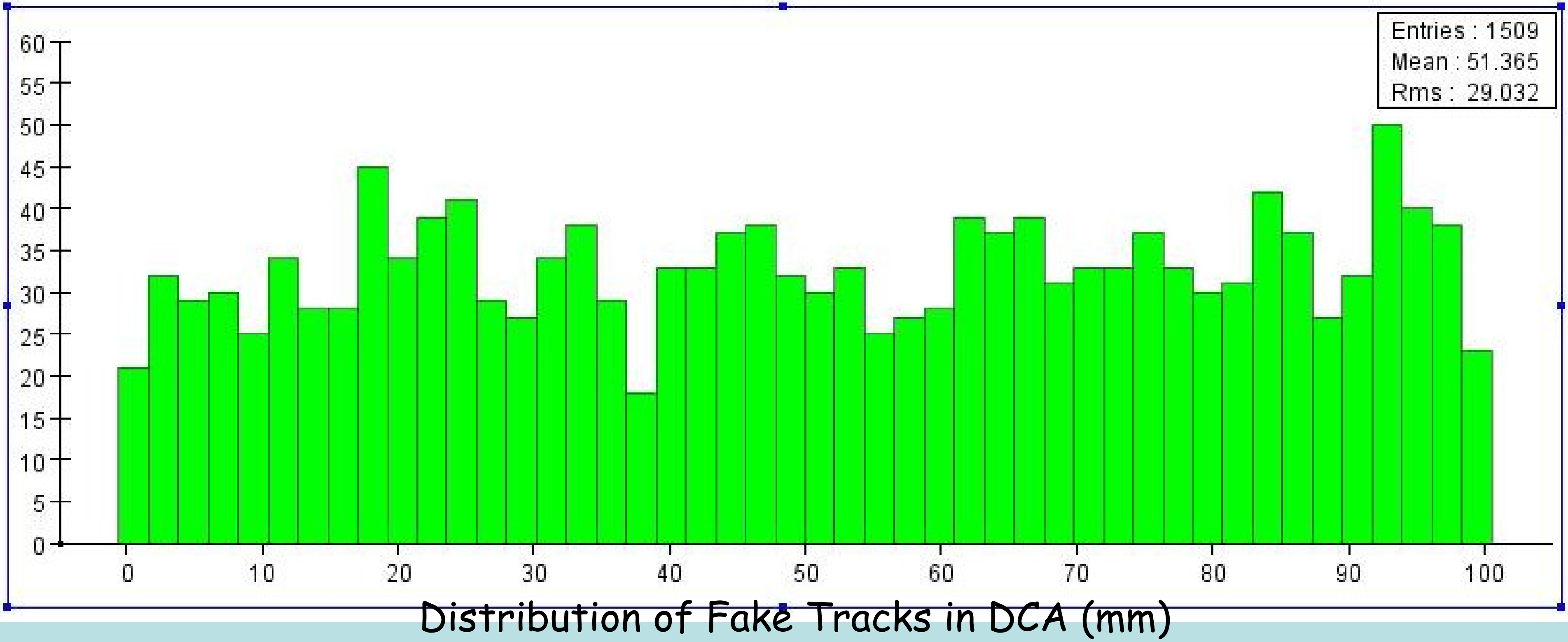
Much better efficiency, ~80%

Distance of Closest Approach < 100mm



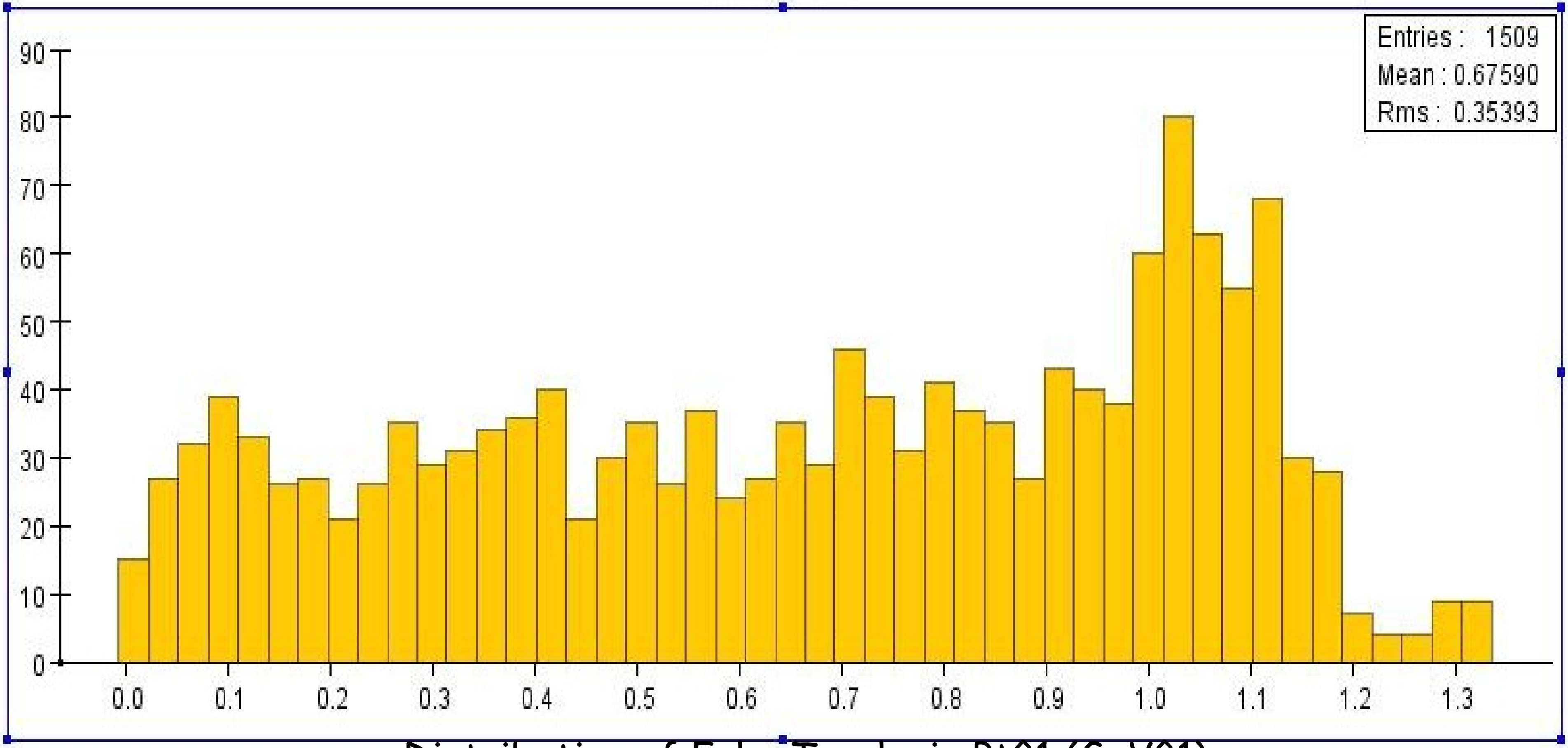
Distribution of Tracks in Pt^-1 (GeV^-1)

Distance of Closest Approach < 100mm

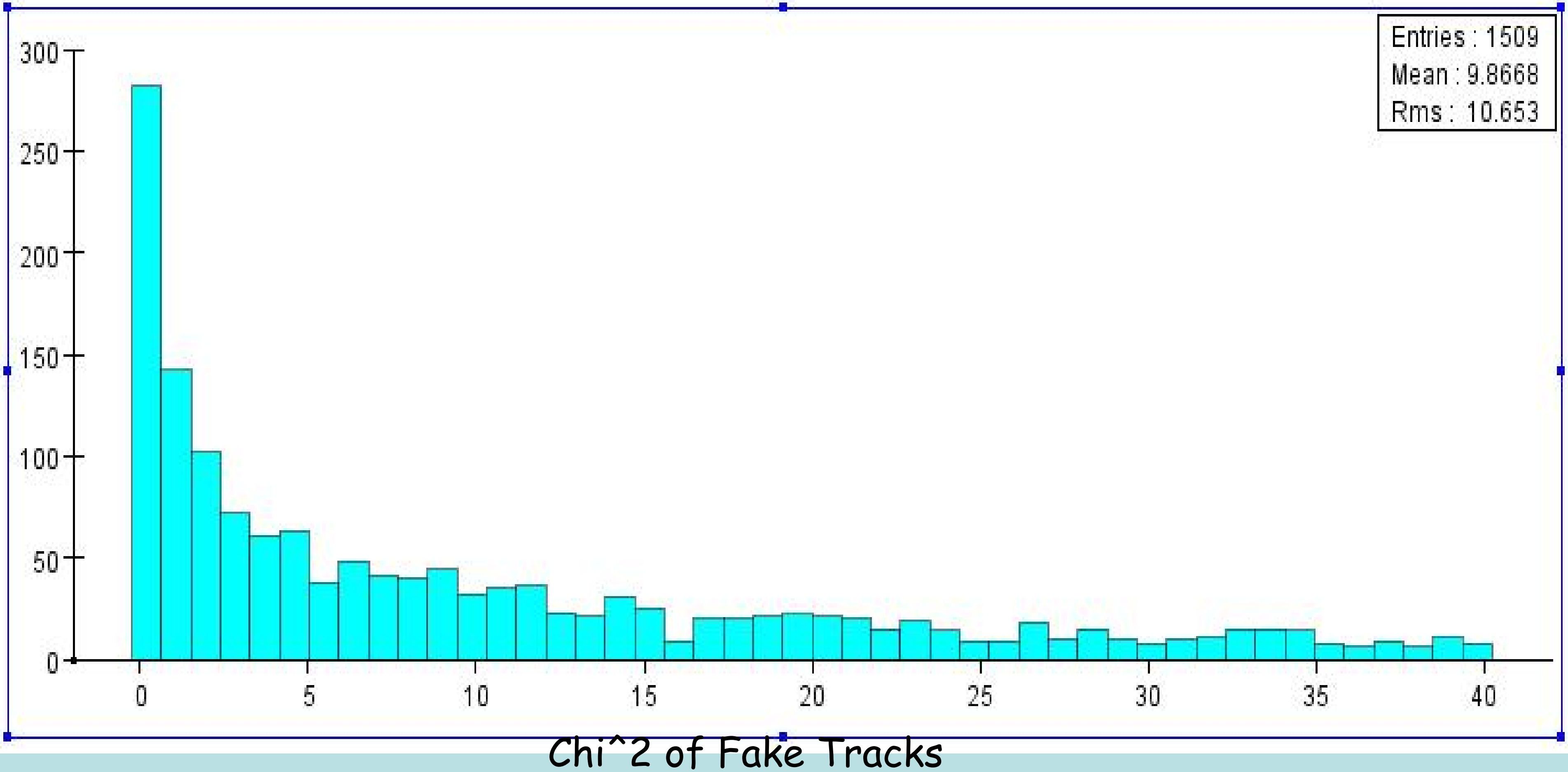


Distribution of Fake Tracks in DCA (mm)

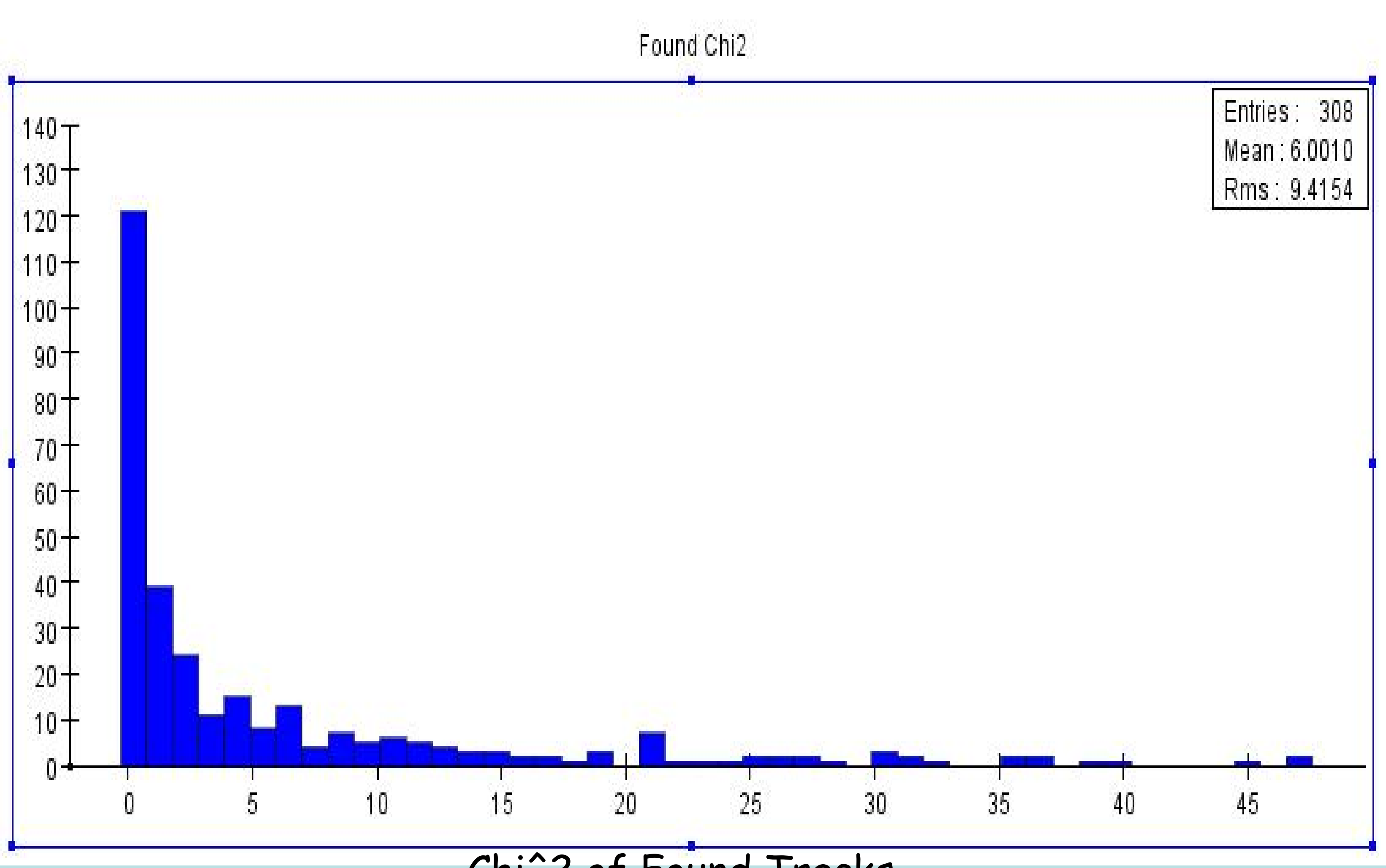
Distance of Closest Approach < 100 mm



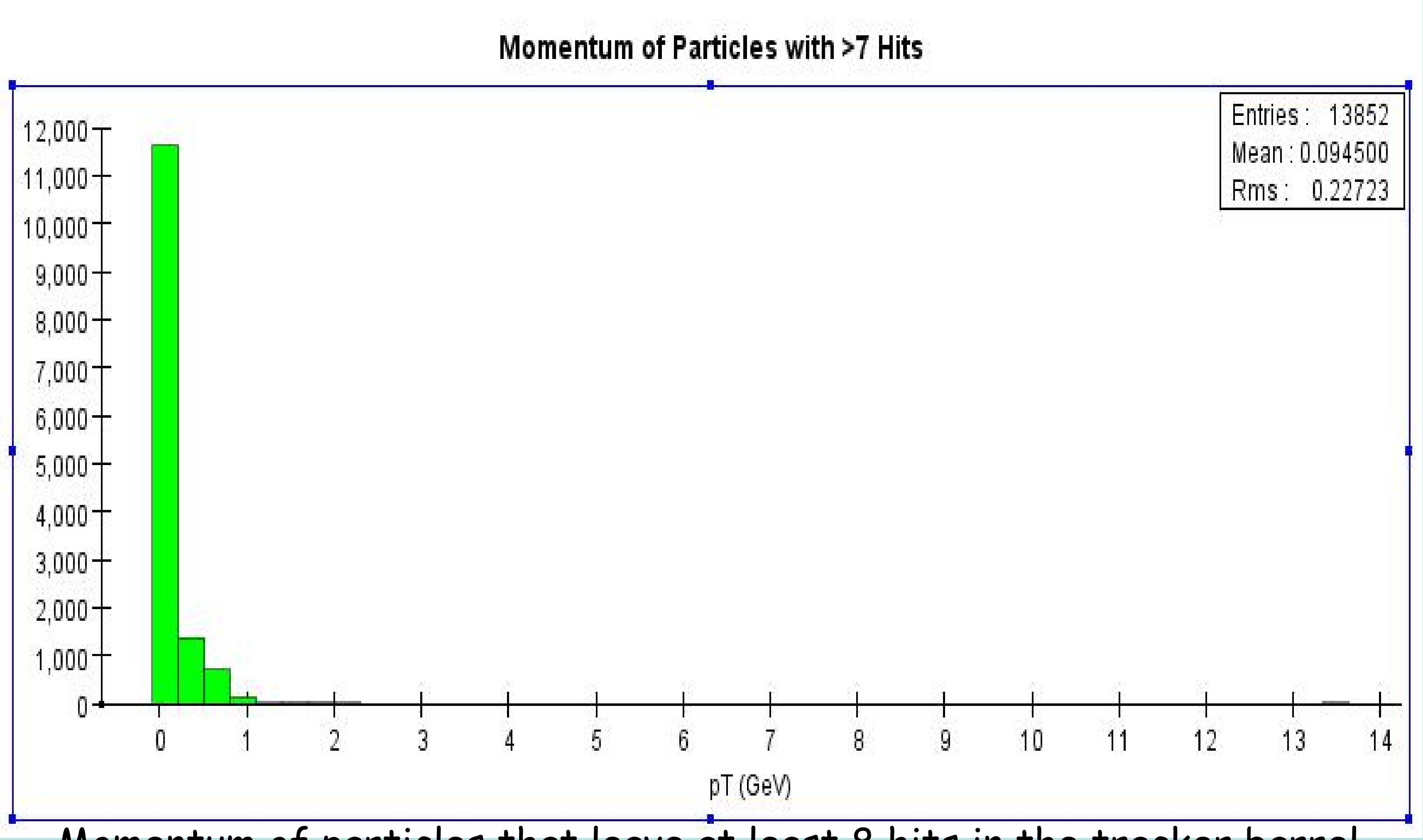
Distribution of Fake Tracks in Pt^1 (GeV^1)



Chi^2 of Fake Tracks



Chi^2 of Found Tracks



Momentum of particles that leave at least 8 hits in the tracker barrel