Preliminary Tracks in pySiDR or, Conformal Tracking: A Red Herring

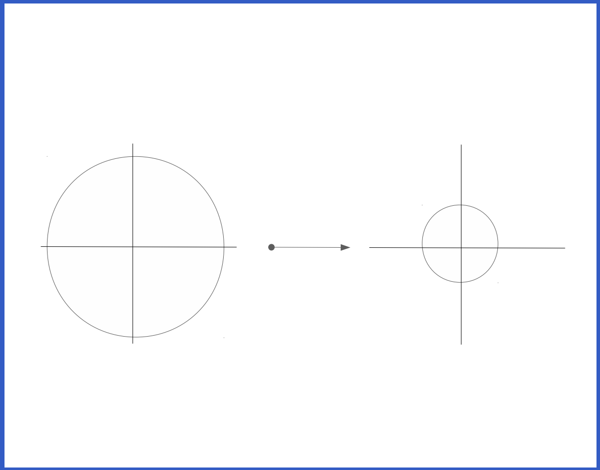
Chris Potter

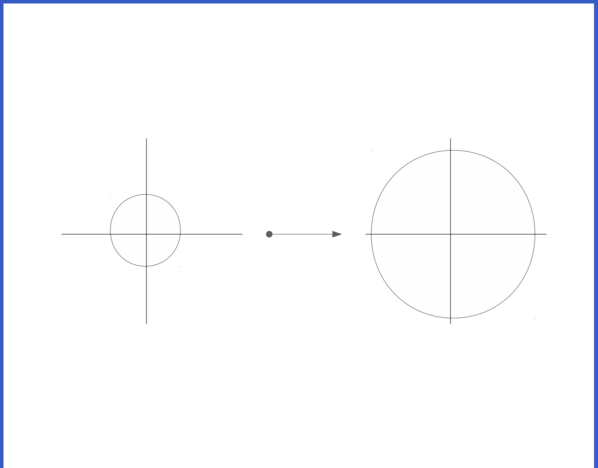
University of Oregon

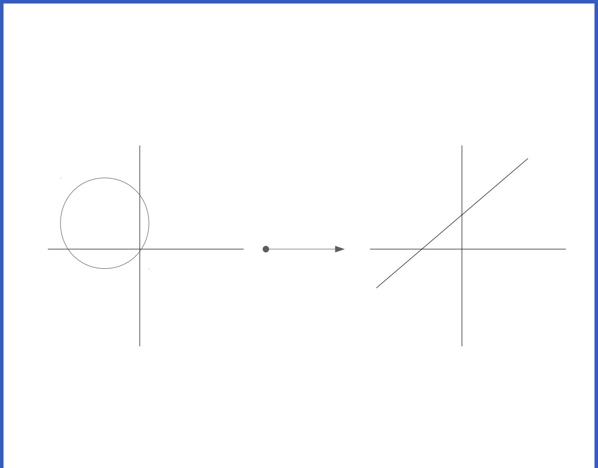
Recent Developments

- CLICdp have released "Conformal Tracking for all-silicon trackers at future electron-positron colliders", CLICdp-Pub-2019-003 (arXiv:1908.00256).
- This is the first comprehensive public account of the algorithm and its performance.
- I have finished the main implementation of conformal tracking for the vertex detector and tracker barrel in pySiDR, the Python package for SiD event reconstruction.
- My implementation of conformal trackfinding runs as follows:
  - Find seeds starting with all hits in the innermost layer of the vertex detector.
  - Linearly extrapolate each hit in (u, v) space to the origin, equivalent to assuming $p_T = \infty$, and associate the closest hit in layer 2 to the seed hit.
  - Feed these these two seed hits to subsequent layers, performing a new (u, v) fit with each new layer after picking up the closest hit.
  - Use vertex detector tracks as seeds for the tracker, using the same technique as in the vertex detector.
- My implementation of the trackfitting is conventional. For any hit collection:
  - Find the least squares circular fit to hit (x, y) for parameters x_0, y_0, R. From these Ω, d_0 are determined.
  - Find the least squares linear fit to hit (s, z) (arclength s) for parameters $z_0, \tan\lambda$.

Properties of Map $(x,y) \mapsto (u,v) = (x/x^2+y^2, y/x^2+y^2)$



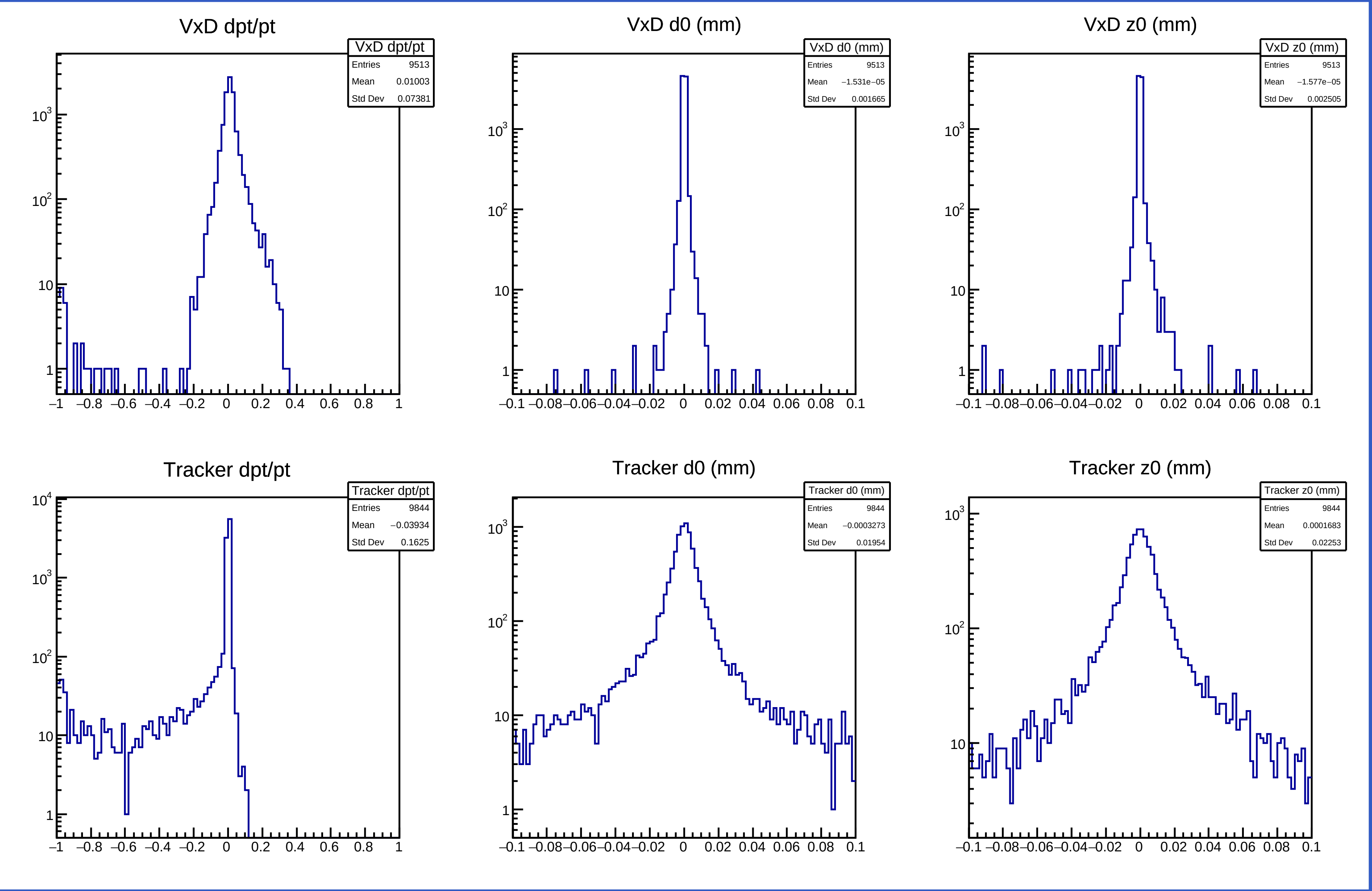




- Project charged particle helical trajectory $(x,y,z) \mapsto (x,y)$ where $B = \hat{k}B_z$.
- Centered Circles to Centered Circles: $x^2+y^2=r^2 \mapsto u^2+v^2=1/r^2$
  Barrels: circles centered on $(0,0)$ with radius r map to circles with radius r^{-1}.
- Uncentered Circles to Lines: $(x-x_0)^2+(y-y_0)^2=r^2 \mapsto v=au+b$
  Tracks: circles centered on $(x_0,y_0) \neq (0,0)$ which pass through $(0,0)$ map to lines.
  - slope $a=-x_0/y_0$
  - y-intercept $b=1/2y_0$
  - radius $r=\sqrt{a^2+1}/2b$
- Because $p_T = qB_z r$, $p_T \propto b^{-1}$. In particular, $b=0 \Leftrightarrow p_T=\infty$ and $b=\infty \Leftrightarrow p_T=0$.

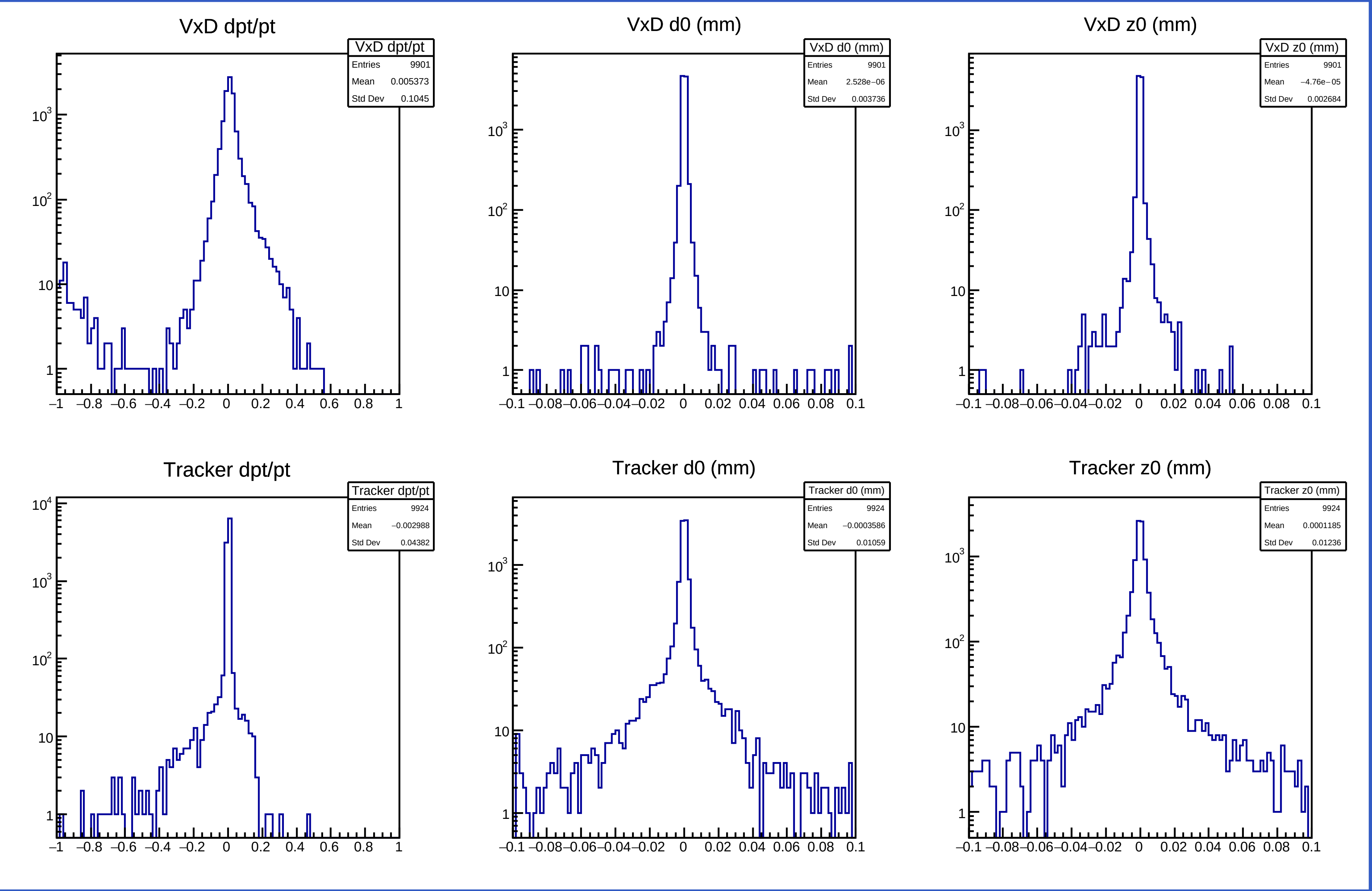
Single Muons, No Trackfinding: p_T, d_0, z_0 Resolution

Single muons originating from $(0, 0, 0)$ with $1 < p < 125$ GeV, $\pi/4 < \theta < 3\pi/4$ and $0 < \phi < 2\pi$.
No conformal trackfinding is applied, all hits are given directly to trackfitting.



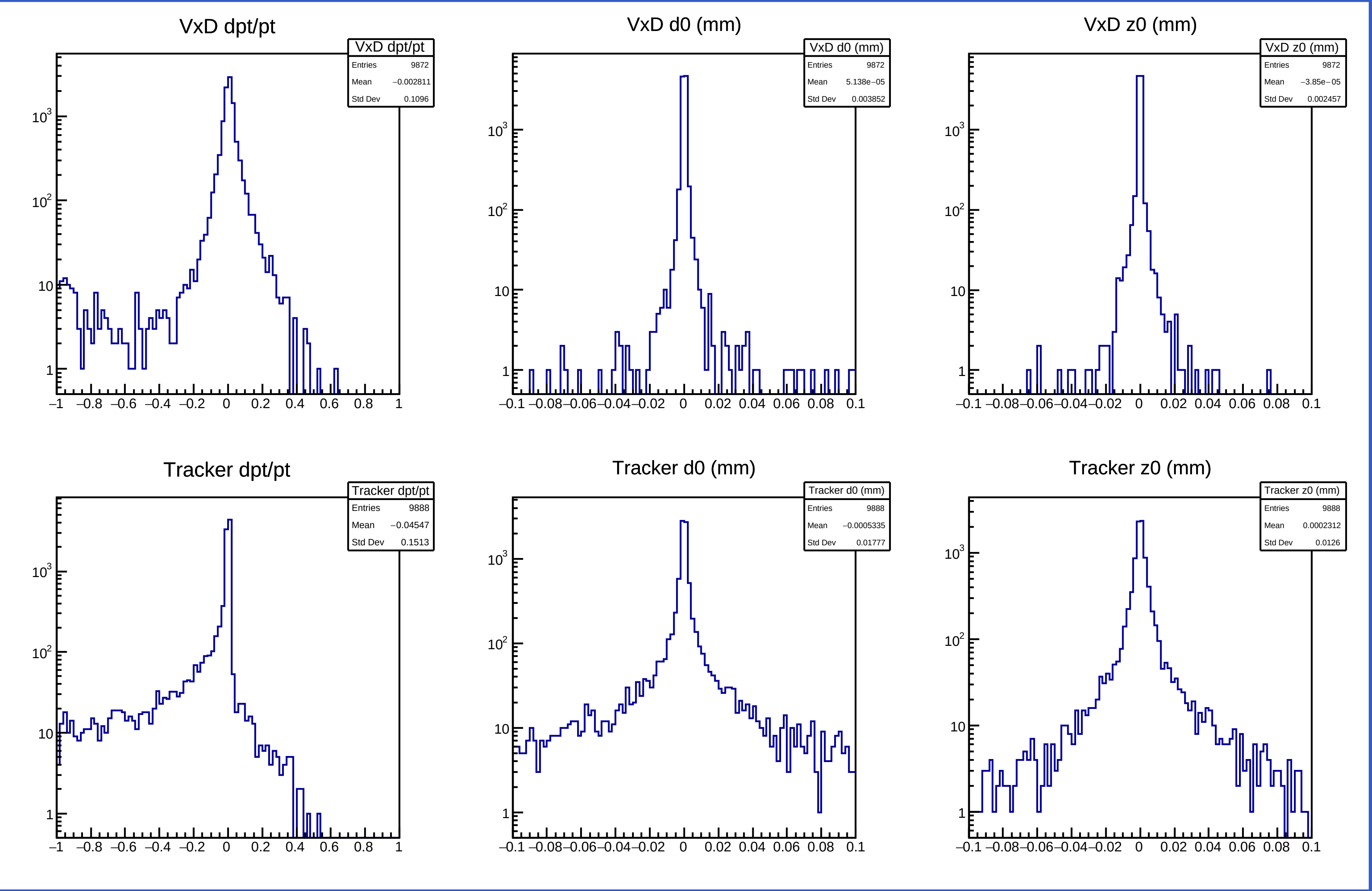
Single Muons, Conformal Trackfinding: p_T, d_0, z_0 Resolution

Single muons originating from $(0, 0, 0)$ with $1 < p < 125$ GeV, $\pi/4 < \theta < 3\pi/4$ and $0 < \phi < 2\pi$.
Conformal trackfinding is applied, only tracks are given directly to trackfitting.

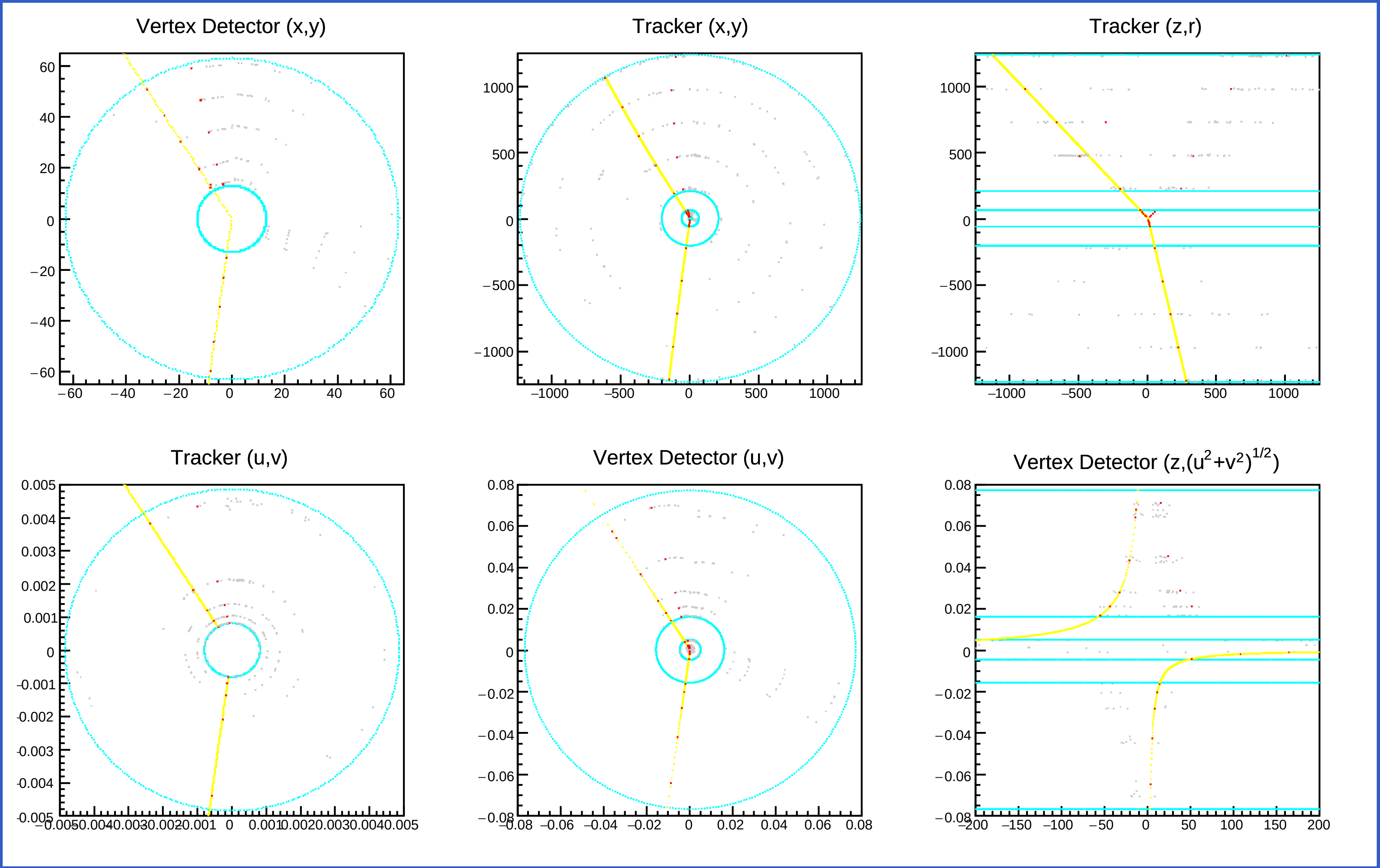


Single Electrons, Conformal Trackfinding: p_T, d_0, z_0 Res.

Single electrons originating from $(0, 0, 0)$ with $1 < p < 125$ GeV, $\pi/4 < \theta < 3\pi/4$ and $0 < \phi < 2\pi$.
Conformal trackfinding is applied, only tracks are given directly to trackfitting.

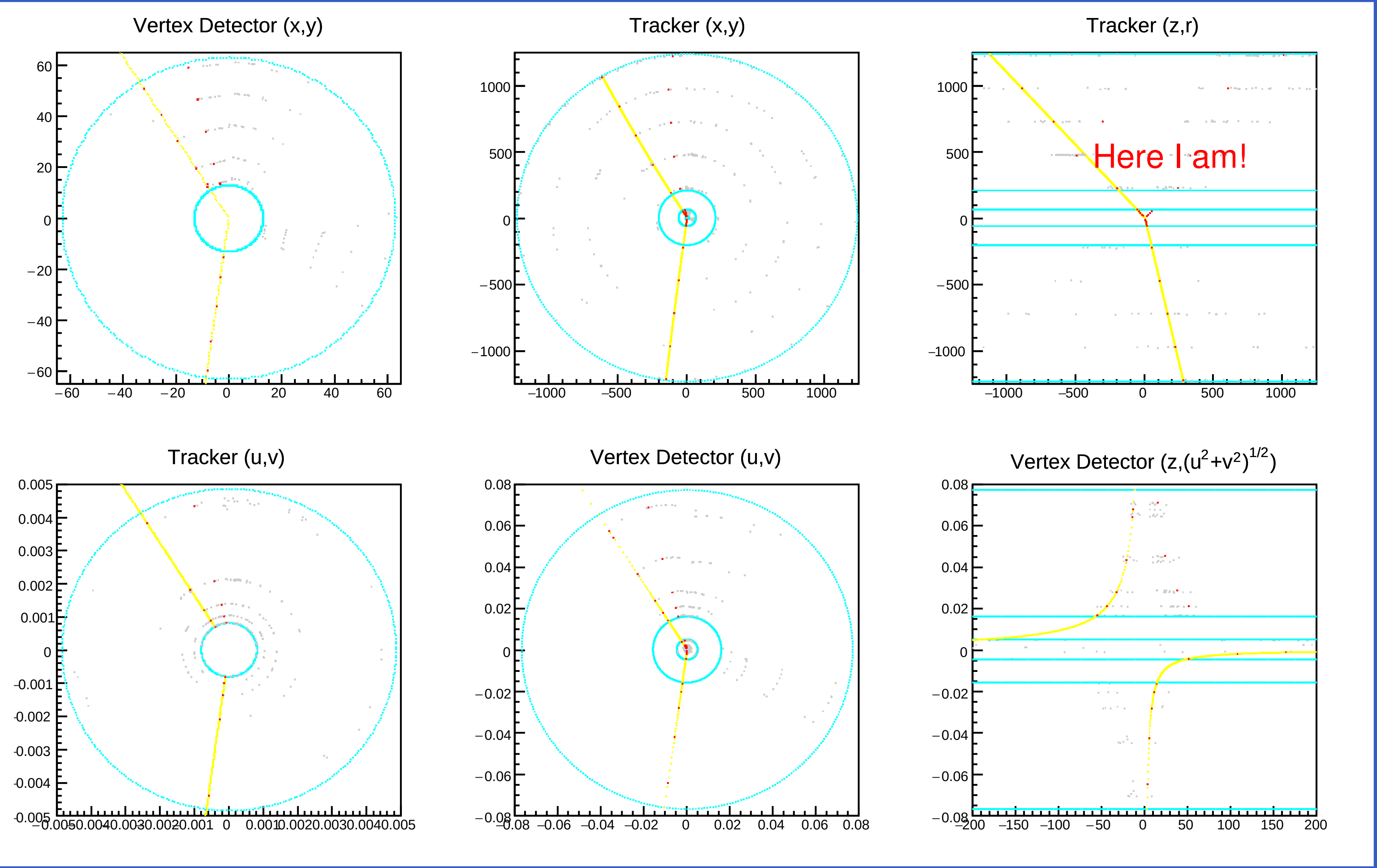


A Failed Track Reconstruction in $(x, y), (u, v), (z, r)$



Color code: MC truth trajectory, detector barrels, hits, hits in fitted tracks with p_T>10 GeV.

A Failed Track Reconstruction in $(x, y), (u, v), (z, r)$



Color code: MC truth trajectory, detector barrels, hits, hits in fitted tracks with p_T>10 GeV.

Conclusions

- Conformal trackfinding omits key information about tracks:
  - nonzero track impact parameters: $d_o = z_0 = 0$ in the map to conformal space
  - hit z information: chance good (u, v) matches can have (very) bad z matches
- Conformal tracking can (and does in the ILCSoft implementation) fix both problems, but *only by reverting to* (x, y, z) *space*.
- This begs the question: why go to (u, v) space to begin with? All of the important track information, e.g. xy and z resolution and z information is in (x, y, z) space.
- Perhaps computer timing performance? Loss of critical information may speed the algorithm, but I find this highly dubious.
- I do not see where any actual leverage is gained in tracking performance by going to (u, v) space, and CLICdp-Pub-2019-003 does not explain it. [a]
- Consequently I am reverting to (x, y, z) space in pySiDR and will go head to head with CLICdp conformal tracking in performance benchmarks.

[a] Please explain it to me in 25 words or less.