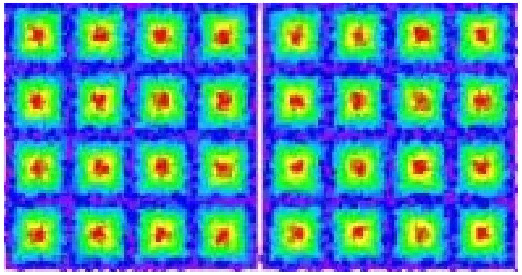
Alignment in Gleam



Leon Rochester
Tracy Usher
Hiro Tajima
SLAC

Instrument Analysis Workshop 2
SLAC, September 27, 2004

Goals

- Our original goal was to align each wafer.
  - Hierarchy of volumes: tower, tray, face, ladder, wafer
  - Transformations from higher to lower levels
- Currently, we plan to perform alignment only at the tower level.
  - Obviously needed
  - Indications are that after being characterized, pointing of towers will not move by more than 7 arc-seconds due to temperature variations, so this calibration will be effective.
  - Expect ±50μm (max) deviations for ladders in trays, perhaps ±100μm for trays in tower.
    - If necessary, ladder alignment data exist, and tray alignment can be measured independently in each tower using cosmic rays.
- But we will need to monitor the intra-tower residuals anyway
  - Full characterization will still be needed to generate simulated data with realistic internal misalignments, to help us write and test monitoring programs.

Current Scheme for Constants

- Misalignments of each element are characterized by six constants: three translations ($\Delta x, \Delta y, \Delta z$) and rotations ($\alpha, \beta, \gamma$) around the three axes.
  - Δx, Δy, and γ are "first-order," that is they produce displacements that don't depend on the track slopes.
  - Δz generates displacements proportional to the track slopes. But since the slopes can be large, these displacements are not necessarily small.
  - α and β generate displacements of both kinds.
- There are two sets of constants, one for simulation and one for reconstruction.

Constants File

- The elements subject to alignment are: Tower, Tray, Face, Ladder, Wafer. Each element in the tracker can be displaced with respect to the one above it in the hierarchy. So:
  - Trays can be displaced with respect to the tower
  - Faces (bottom or top) can be displaced with respect to the tray
  - Ladders can be displaced with respect to the face
  - Wafers can be displaced with respect to the ladder
- As you will see in the example, this scheme allows the description to be as simple or as complex as required by the given task.

Example Input File

```
// example alignment constant file for simulation
 Tower 3
 //         delta in microns      rot in mrad
 Tray    1   45.  17.  -30      1.5  -0.7   0.3
 Face    1   21  -13    43      0.0   0.5  -1.6
 Face    2  -14    7   -26      0.0  -0.5   1.6

 Tower   4    0   15    -6      0.0   0.0   0.0
 Face    0
 Ladder  1
 Wafer   2   12   -7    24      0.3  -0.5   1.1
 Wafer   1   14  -14    18      0.2  -0.4   1.3
```

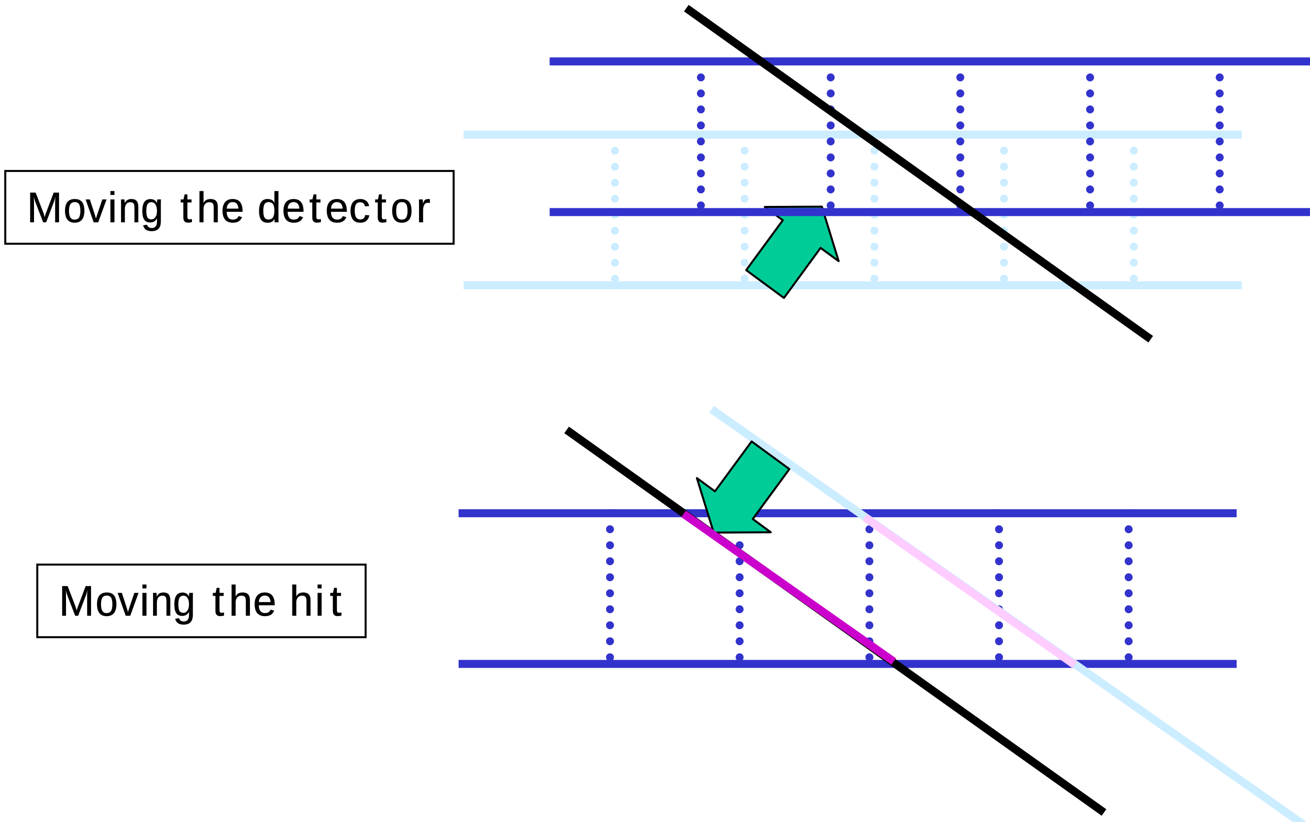
(If no constants are given, zeros are assumed.)

Setting up Internal Arrays

- This division reflects the way alignment information will be collected in real life. The code reflects this hierarchy.
- At each level
  - Alignment constants at that level, if any, are read in
  - Constants are merged with those from the level above
    - …including nulls for any not specified
  - The merged constants are passed down to the next level
- At the lowest level (wafers) the constants are inserted into an array containing one entry for each wafer in the detector (…9216 in all for the flight instrument. Treatment is general; two towers is a special case.)

A Fundamental Choice

Alignment can be introduced by moving the detector or by moving the hit/cluster.



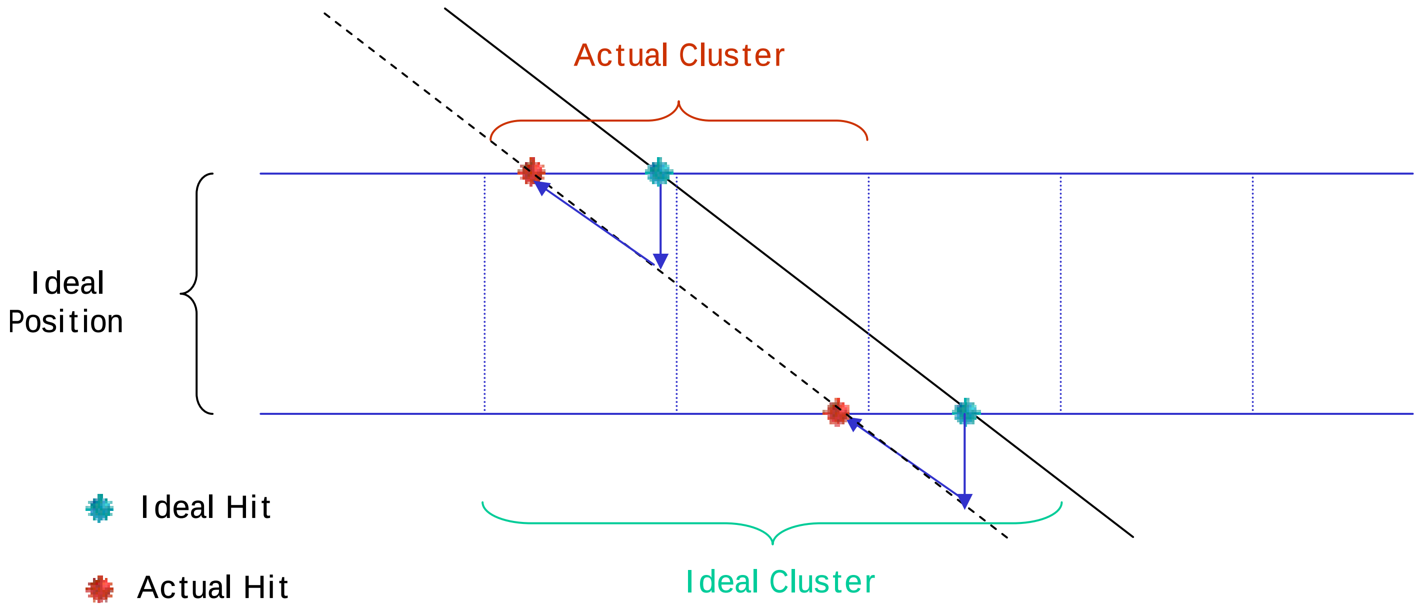
We move the hits.

- **Benefits**
  - The geometry (Geant, propagator) can be kept simple. Every element is replicated uniformly.
  - Geometry is nominal, so events don't have to be re-simulated every time the alignment is updated.
- **Drawbacks**
  - Geometry is not quite "correct"
  - Certain pathologies arise during simulation
  - Clusters may not lie on the MC and recon tracks in the display. They show up where they would be if the element were actually displaced.

Procedure

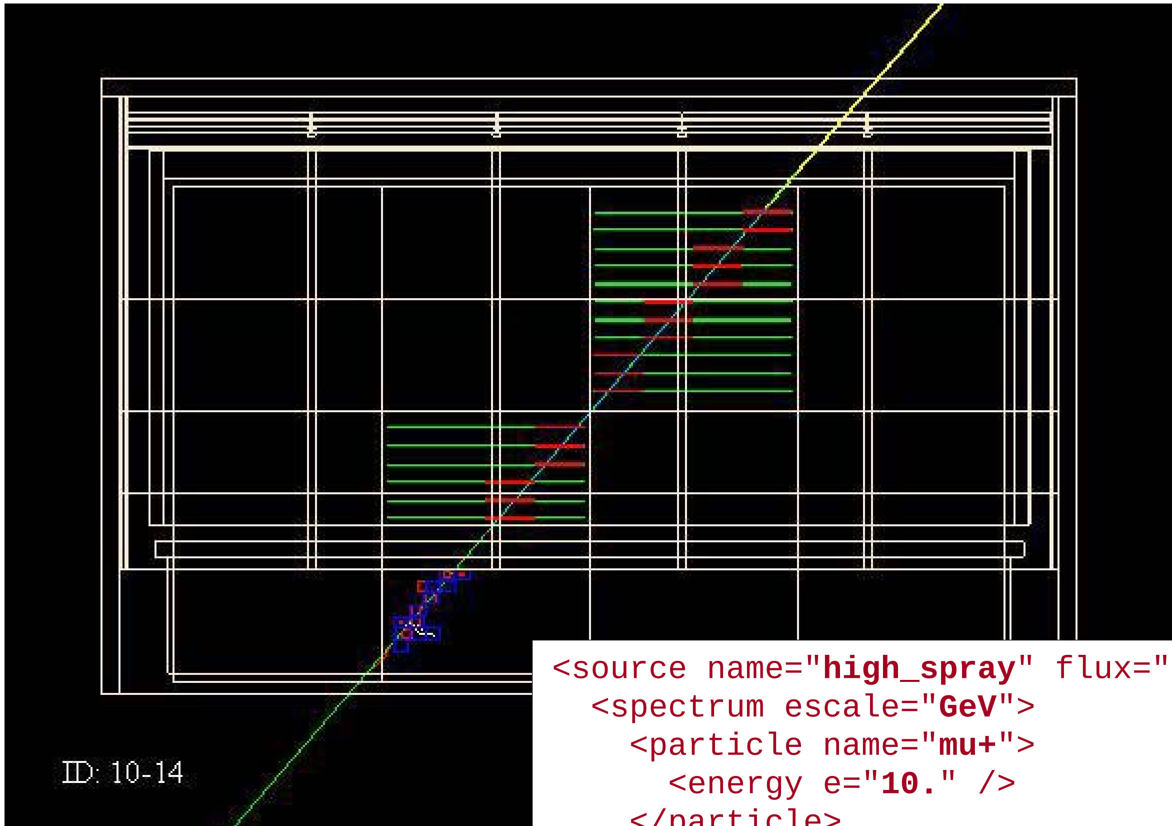
- **Simulation**
  - For each hit the track is moved according to the constants, and then the resulting track is re-intersected with the active element.
- **Reconstruction**
  - For each cluster on a track, the nominal position is modified according to the constants and the slopes of the track.
  - Currently, the correction is applied just before fitting.
  - In principle, the first-order corrections could be made when the clusters are constructed, and the remaining corrections could be applied at patrec time. In practice, this would probably be more complicated and confusing.

Example of Moving a Hit



Here we show the result of moving the silicon plane up, so that the apparent track moves down. Then we re-intersect the track with the silicon plane and calculate new entry and exit points.

Some Details

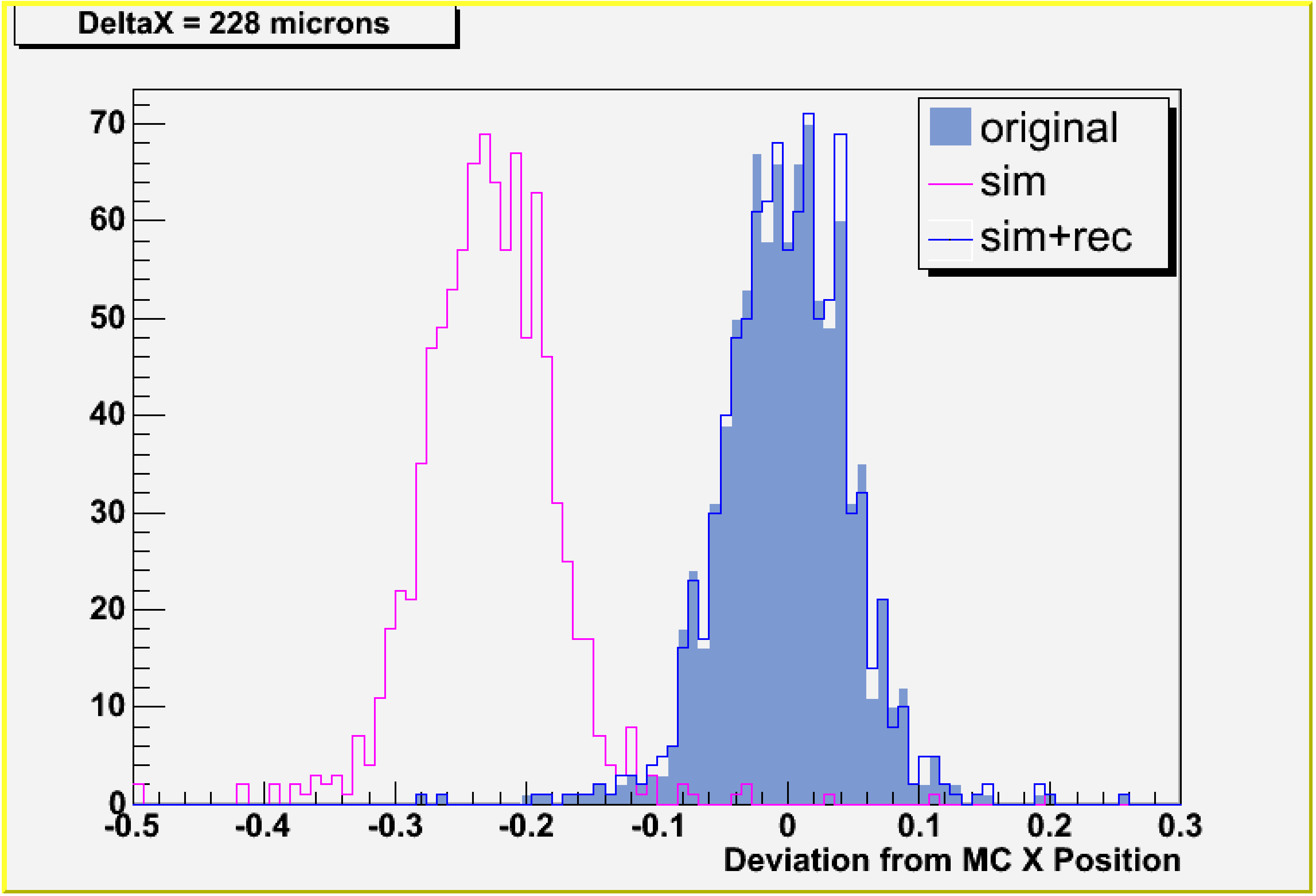



```
<source name="high_spray" flux="1.0">
  <spectrum escale="GeV">
    <particle name="mu+">
      <energy e="10." />
    </particle>
    <solid_angle mincos="0.700" maxcos="1.0" />
    <patch xmin="-500" xmax="500"
      ymin="-500" ymax="500"
      min="655" zmax="665" />
  </spectrum>
</source>
```

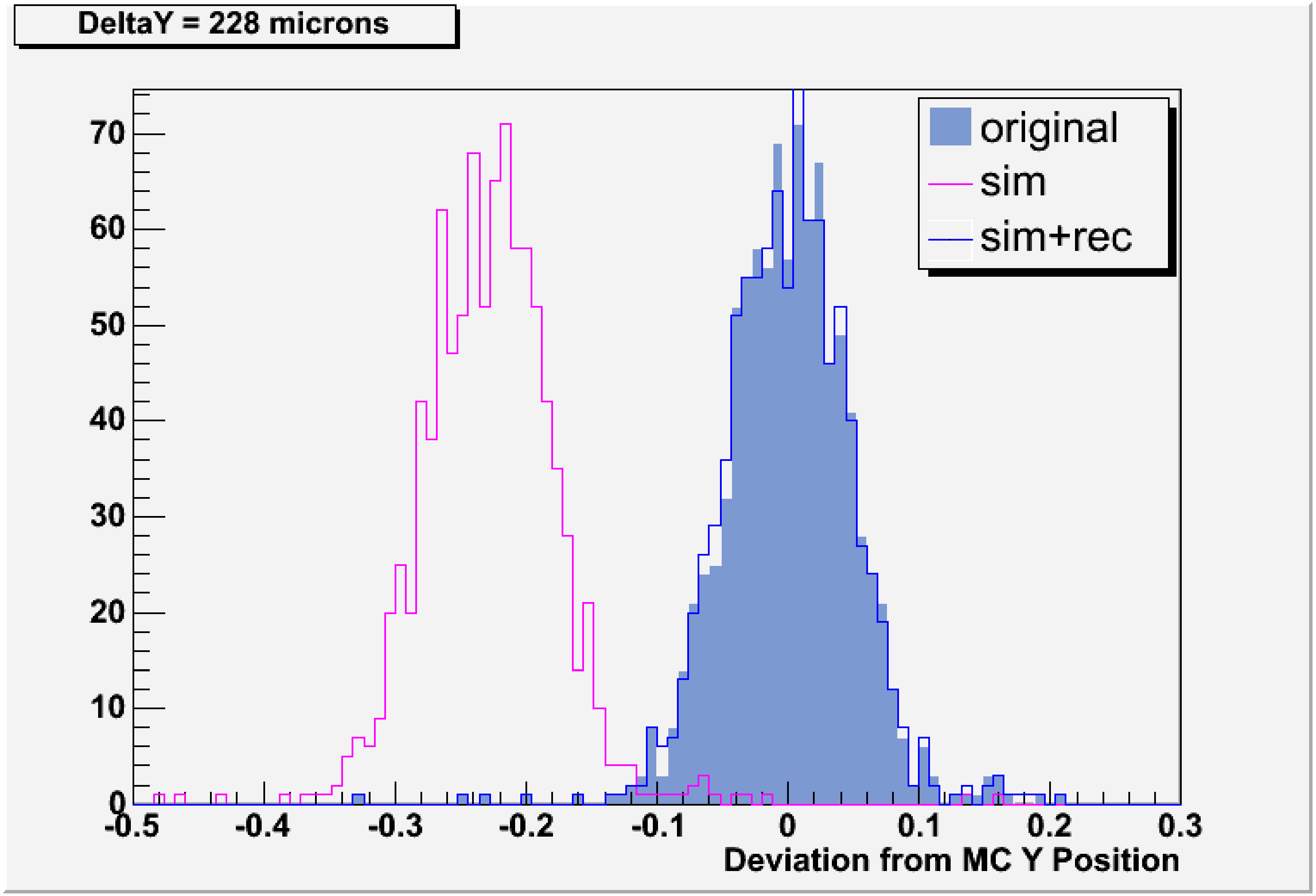
Does it Work?

- In the next six slides, you can see the results of applying the alignment corrections. On each slide, three distributions are overlaid:
  - Vanilla, no alignment applied
  - Alignment corrections applied during simulation
  - Alignment corrections applied during simulation and reconstruction.
- In each slide, the variables plotted are the ones most sensitive to the correction in question.
- The goal is for the first and last distributions to be essentially the same.

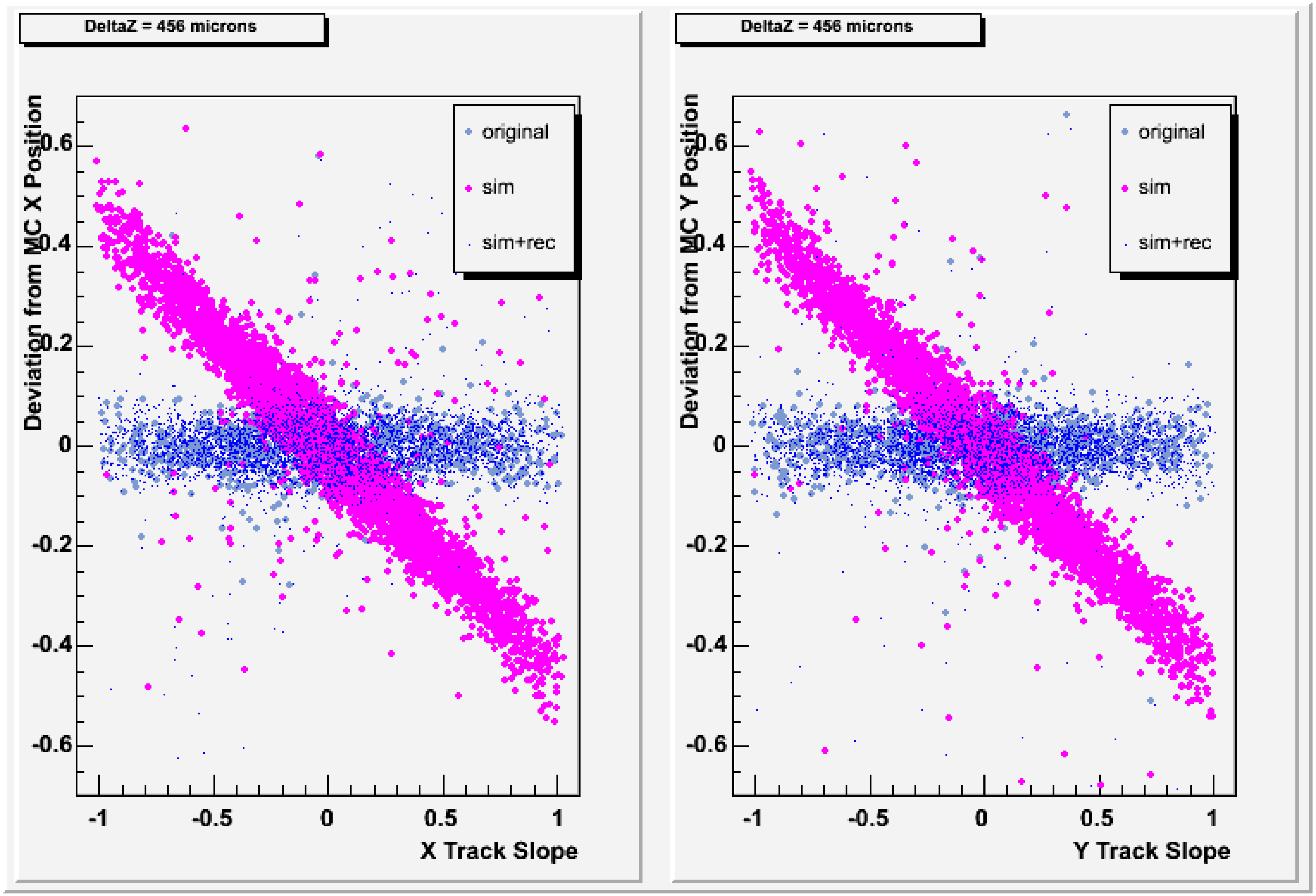
Translation in X



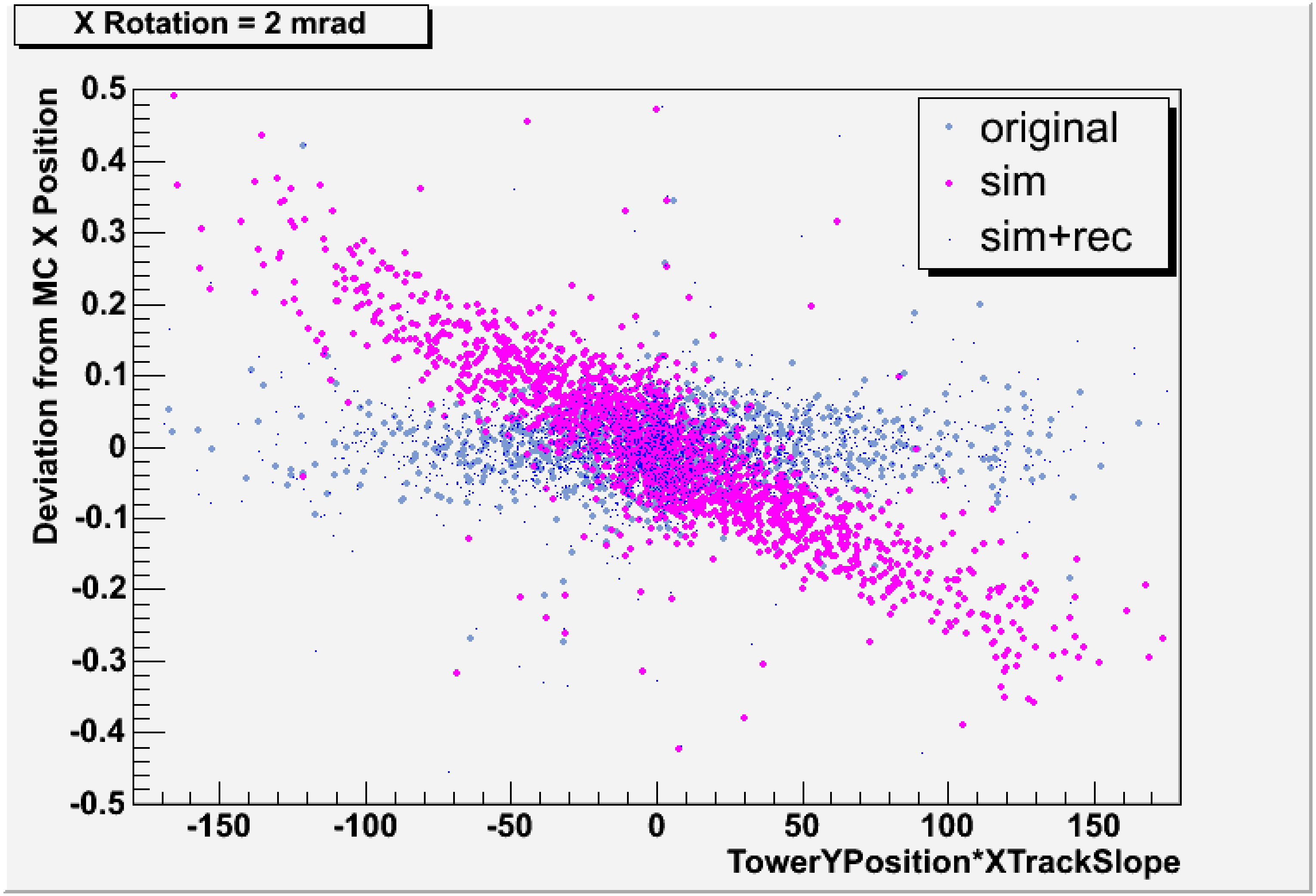
Translation in Y



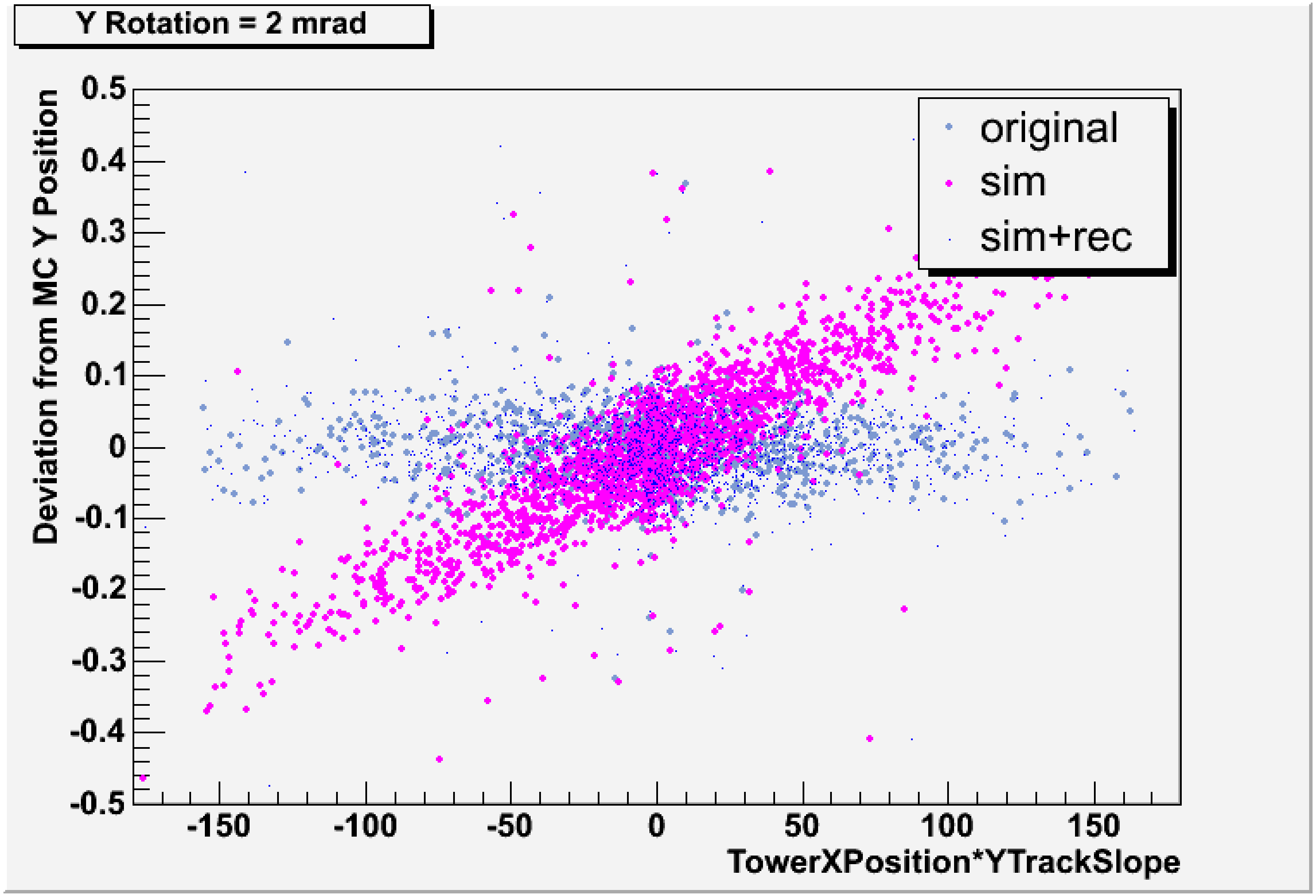
Translation in Z



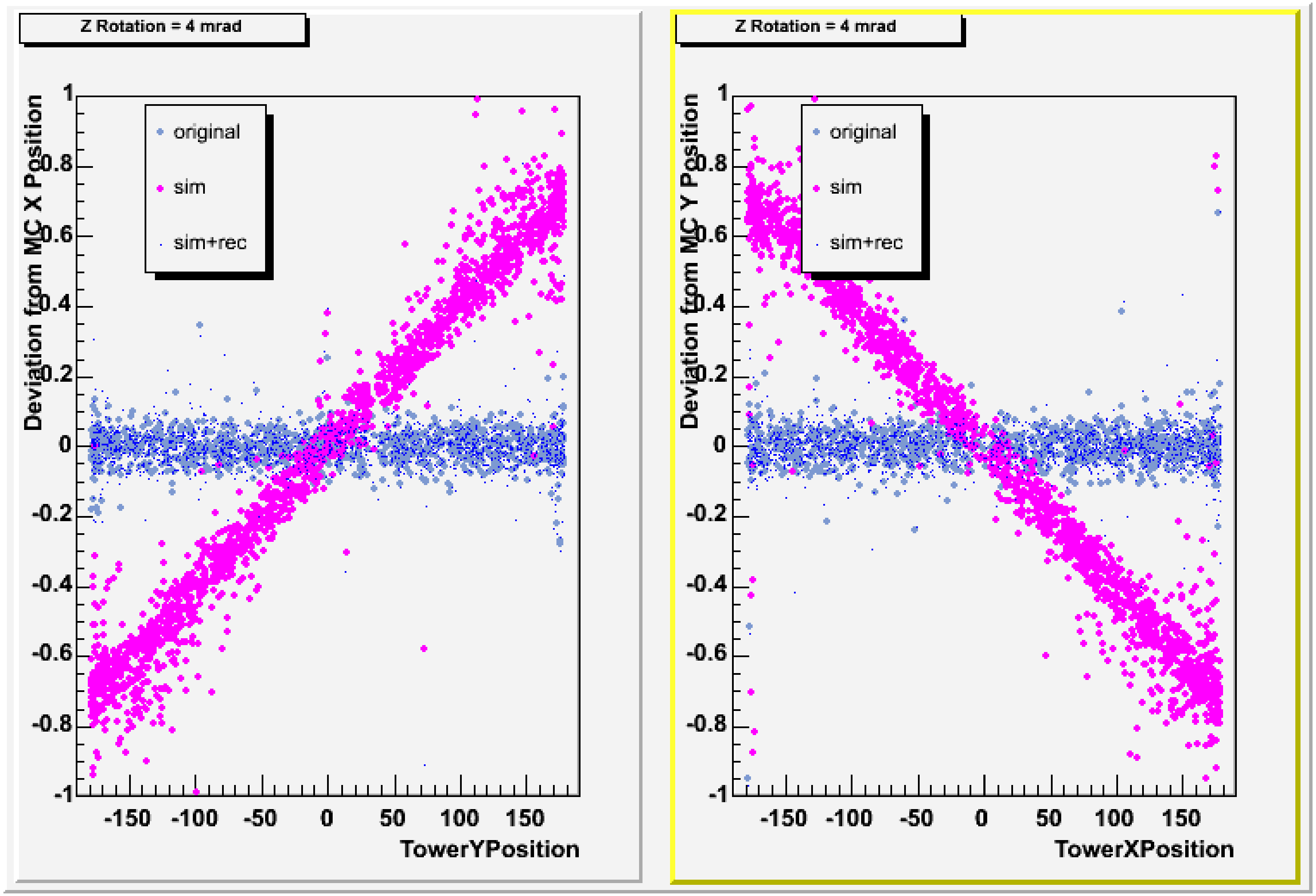
Rotation around X Axis



Rotation around Y Axis



Rotation around Z Axis

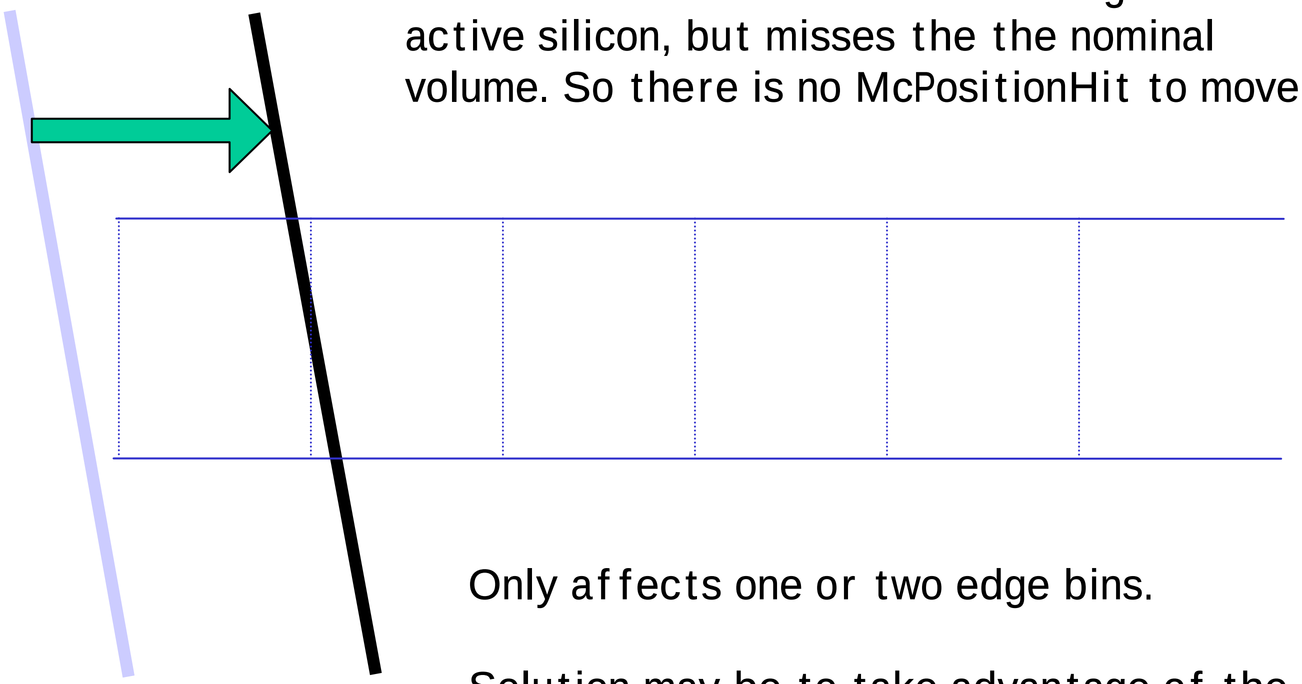


Areas for improvement

- Reconstruction corrections should probably be applied earlier in the process.
  - Patrec may fail to find a hit on a high-energy track in a misaligned detector.
  - Since the new plan is to make a TkrTrack during patrec, it would seem logical to do the alignment when adding hits.
  - Will slightly increase patrec time… probably not significant
- Simulation: works well in spite of the items below:
  - Hits at the edges of the active areas
  - Interactions in the silicon
  - Nearly horizontal tracks

Edge Hits

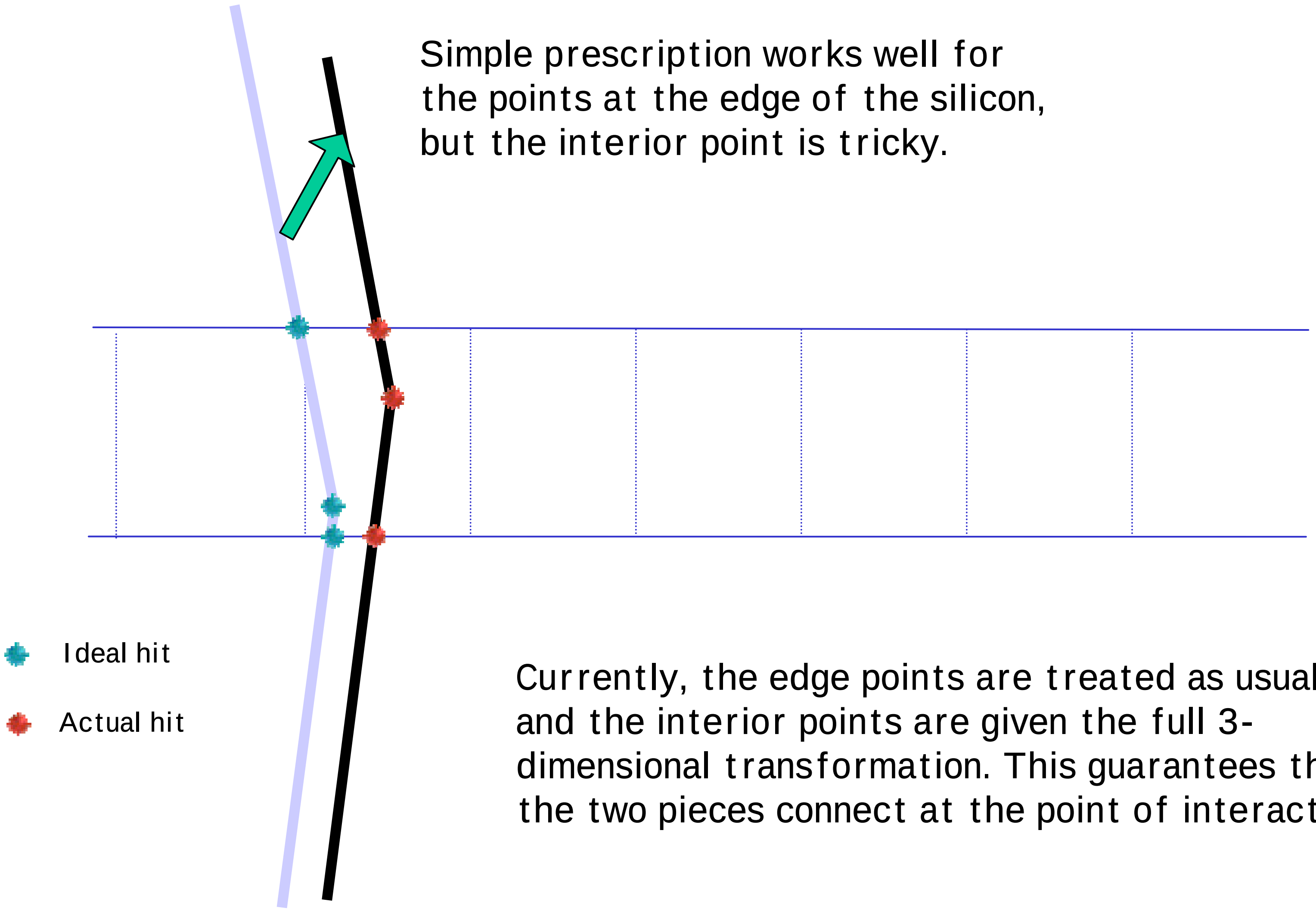
The track would have hit the misaligned active silicon, but misses the the nominal volume. So there is no McPositionHit to move.



Only affects one or two edge bins.

Solution may be to take advantage of the ~1 mm dead zone framing the active silicon. By making this active, hits could be recorded there and then handled correctly during digitization.

Interactions in the Silicon

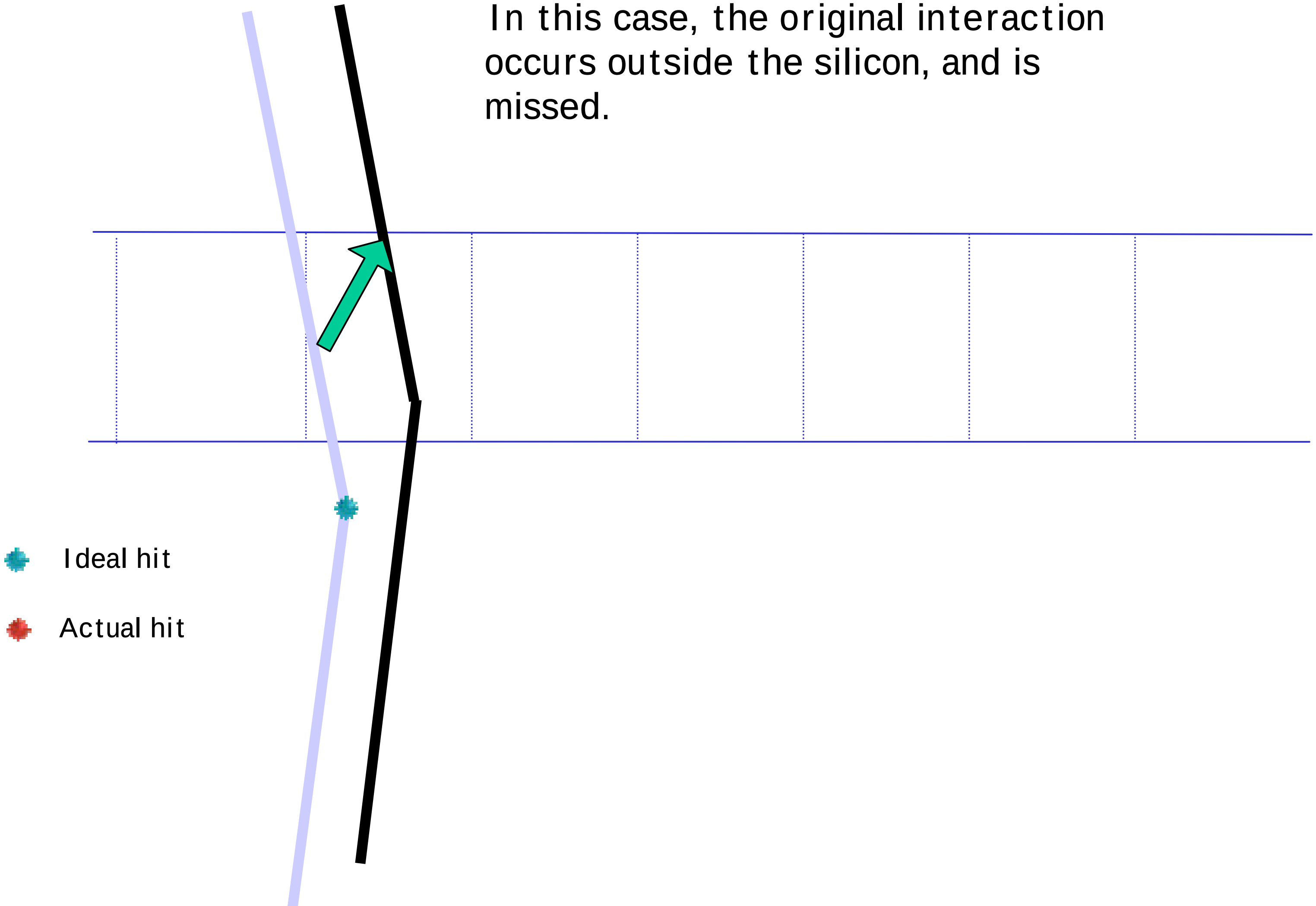


Simple prescription works well for the points at the edge of the silicon, but the interior point is tricky.

Currently, the edge points are treated as usual, and the interior points are given the full 3-dimensional transformation. This guarantees that the two pieces connect at the point of interaction.

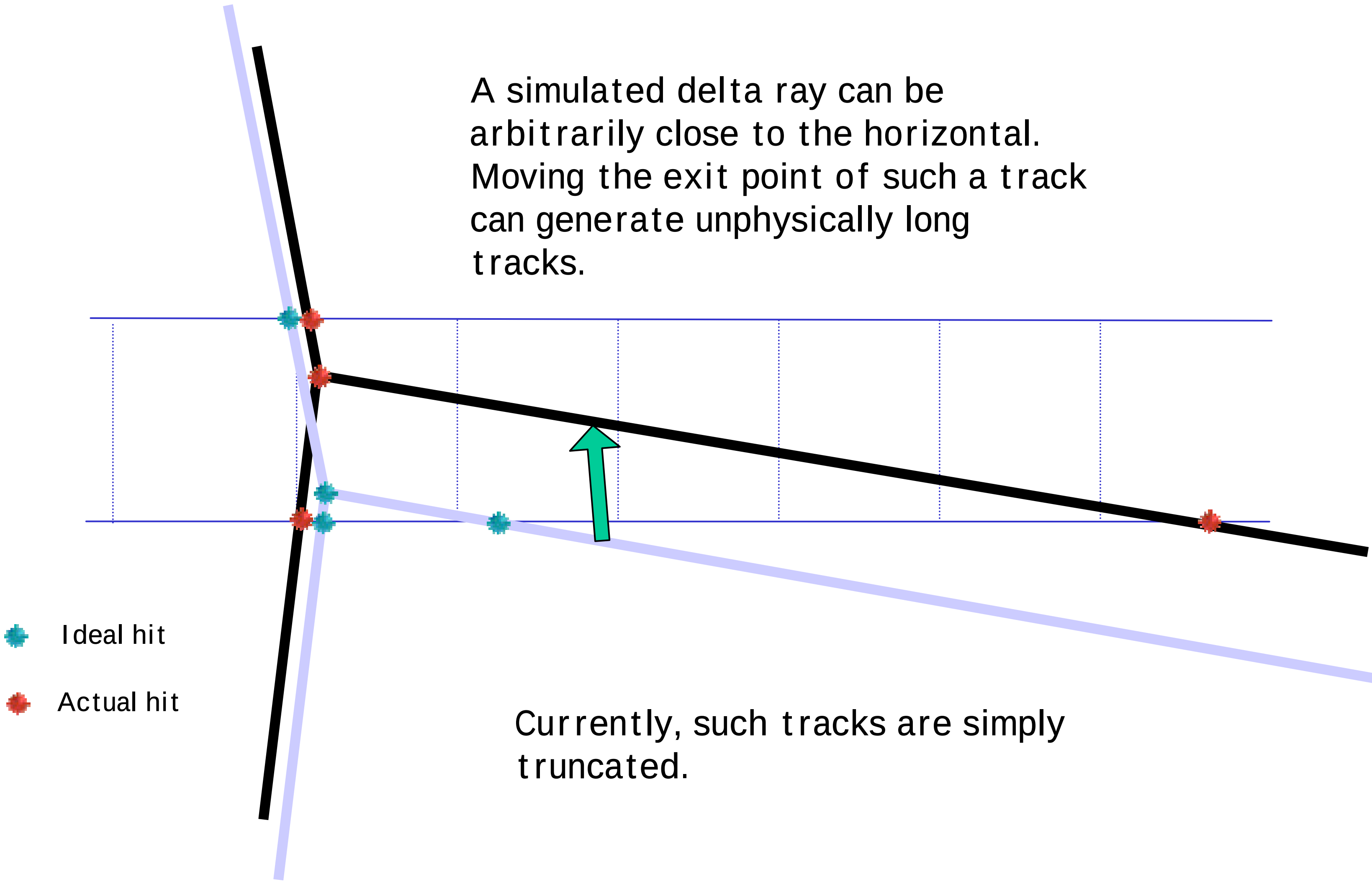
Lost Interactions

In this case, the original interaction occurs outside the silicon, and is missed.



Nearly Horizontal Tracks

A simulated delta ray can be arbitrarily close to the horizontal. Moving the exit point of such a track can generate unphysically long tracks.



Currently, such tracks are simply truncated.

Constants Finding

(Apologies to Tracy!)

Some History

- Based on the work of Hiro Tajima
  - Standalone ROOT macro
  - Did its own patrec, fitting
  - Geometry put in by hand (now outdated!)
  - Lots of copying of files, hand modifications
- But it seems to have worked!
  - (But couldn't really check, because Leon's stuff wasn't ready…)

Tracy's Goal

- Integrate with Gleam
- Use as many existing tools as possible
- Break down to functional components to allow exploration ofalternative schemes
- Be ready for real data!

How It's Done

- Generate events in misaligned detector, using standard code
  - For this exercise, diagonal high-energy muons
  - To understand what we're doing, we cheat and use MC patrec.
- Perform standard pattern recognition and fitting
- Pick tracks
  - Minimum number of clusters in reference and target tower
- Separate tracks into two parts
  - Reference tower
    - Refit, using only the clusters in that tower
  - Target tower
    - Store measured position and covariance matrix for each hit plane.
    - Replace fit position with extrapolation of reference track.

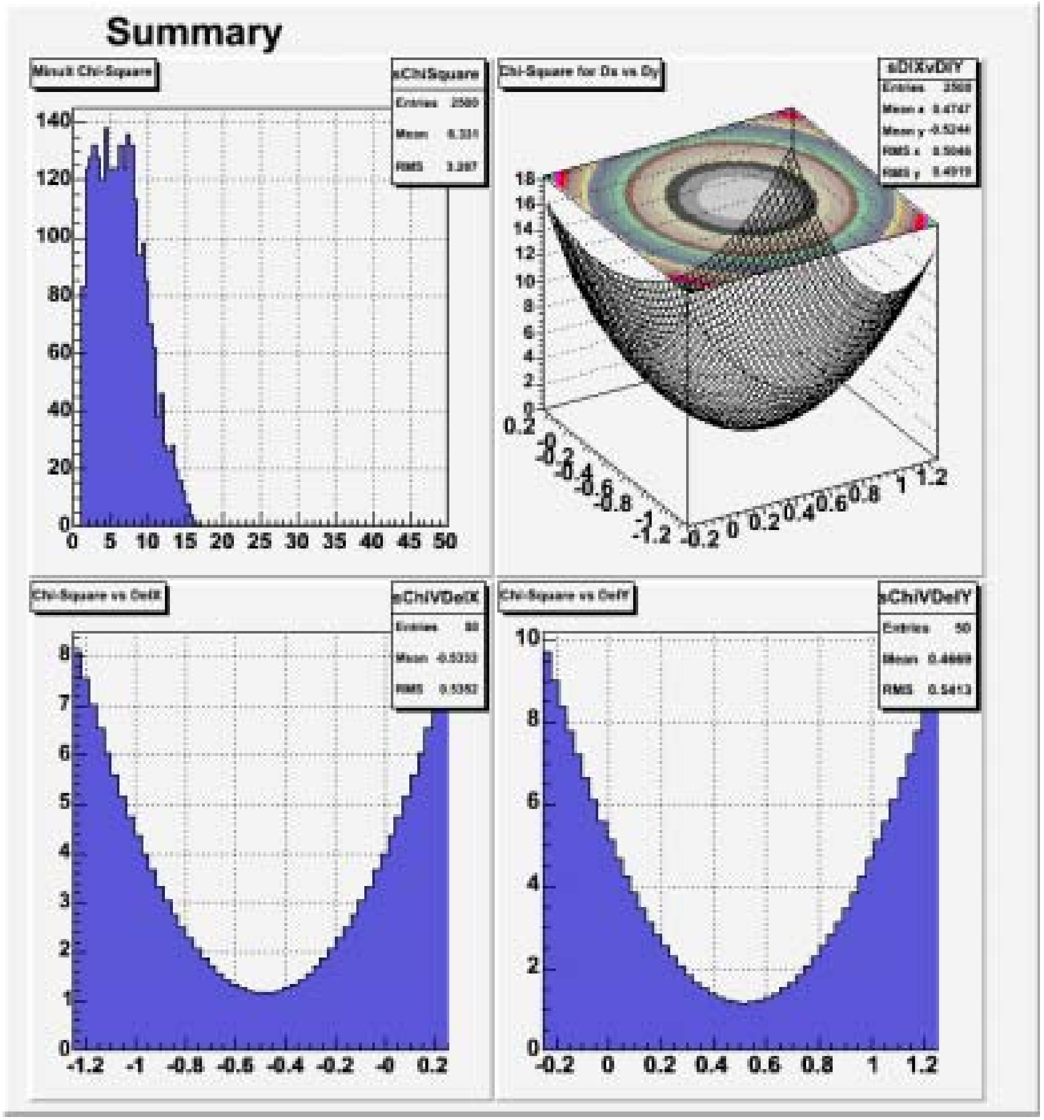
How It's Done (2)

- Accumulate events
- Perform minimization (Minuit)
  - Vary parameters in n-dimensional space (n<=6)
  - For each set of parameters, transform measured positions using existing tools
  - Calculate residuals and chi-squared, using weights derived from covariance matrix and measured errors
- Compare results with inputs

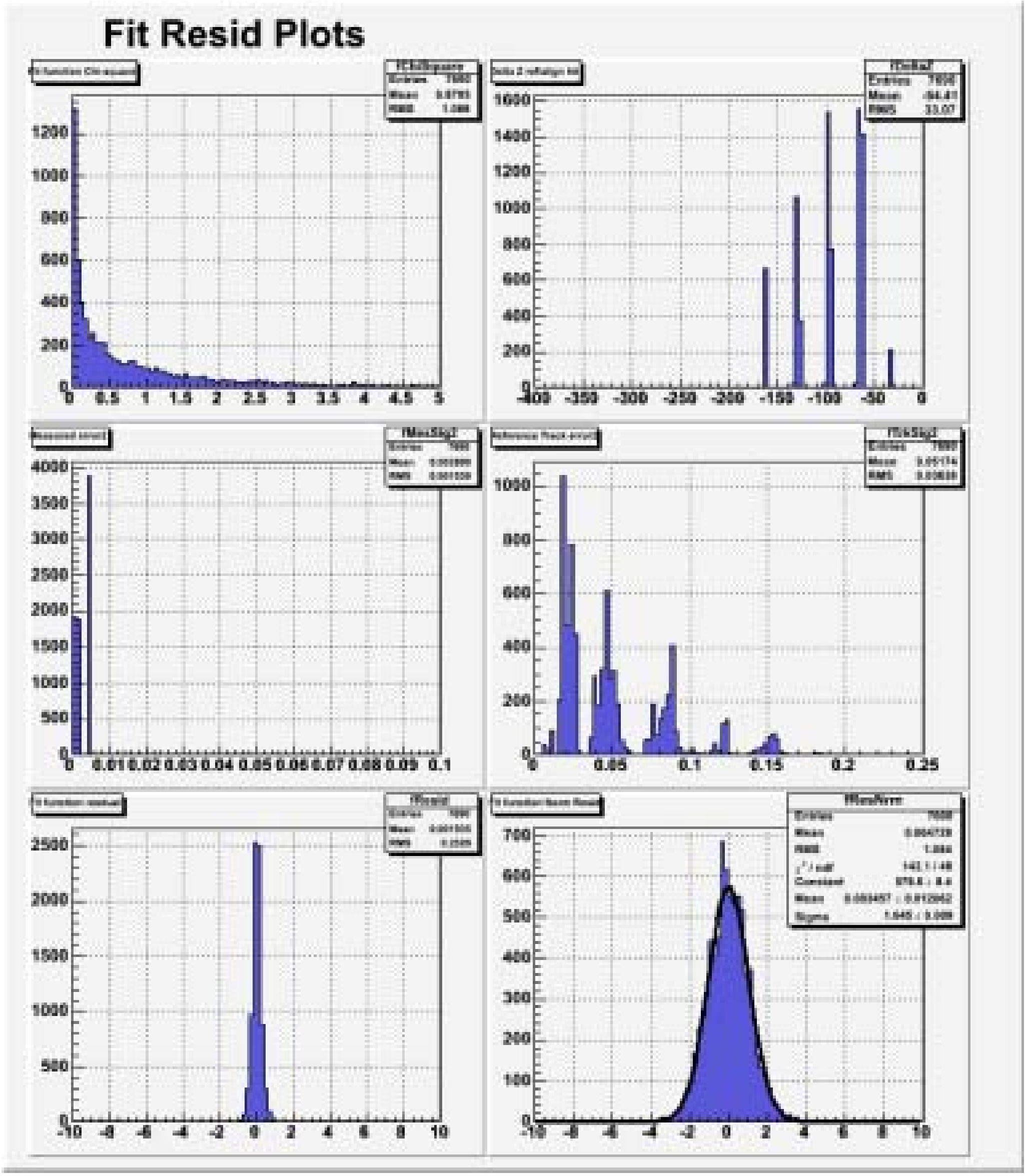
How Are We Doing?

- We have done end-to-end checks of procedure
  - Translation in X
  - Translation in Y
  - Simultaneous translation in X and Y
  - Rotation around X axis
- Original offsets are reproduced!

Some Diagnostic plots



More Diagnostic Plots



What Next?

- Clean up code
- Understand how to interpret errors
- Realistic source spectrum
- Study performance in the presence of internal misalignments
- Exploration of alternate schemes
  - Tracy has one or two in mind

Where We Stand

- We have checked two-tower alignment end-to-end:
  - Generated events using misaligned tracker
  - Used those events to measure the misalignment
  - Verified that the constants found agree with the input constants
  - Use the constants to correct the fitted tracks.
- We have identified a modest program of improvements to take care of remaining details.
- We await real two-tower data to demonstrate that we are not only consistent, but also correct!