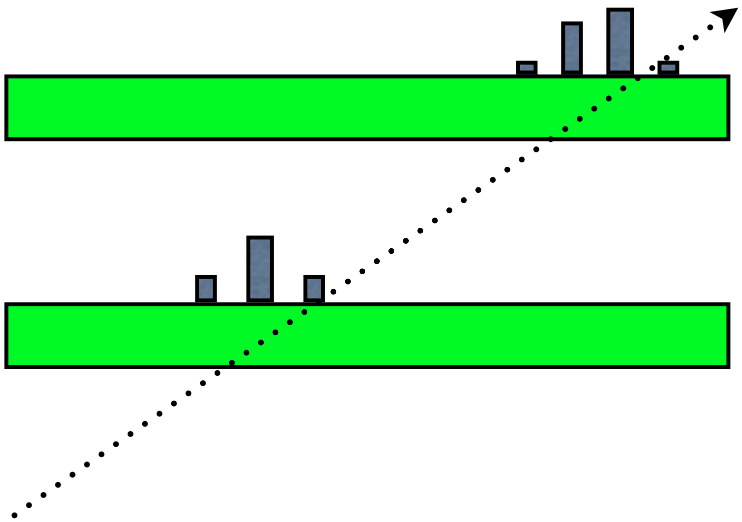
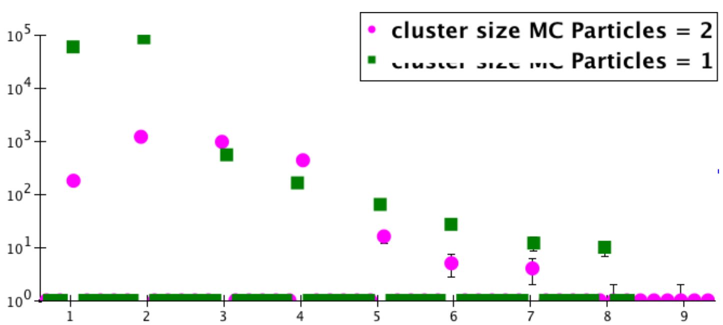
HPS Tracker Hit Reconstruction, Tracking & Vertexing in lcsim

Matt Graham
SLAC

Hit Reco: clustering





# of strips/cluster

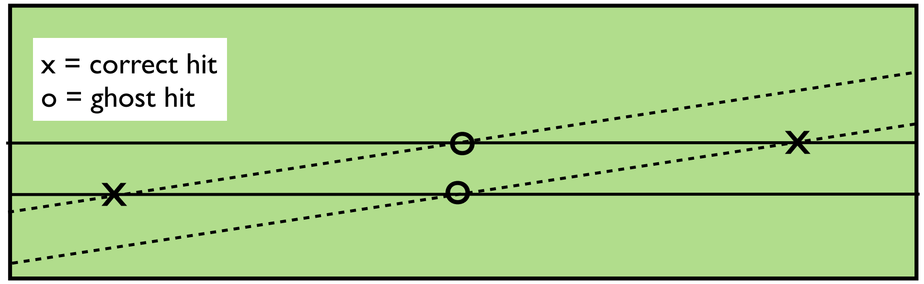
The clustering algorithm we use (NearestNeighborRMS) is pretty simple...

- •set thresholds for seed, neighbors, and total cluster
- •take hit over seed threshold and add neighboring hits until we find a hit below (neighbor) threshold
- •repeat until no hits above seed threshold
- •remove clusters>MAX_STRIPS
- •calculate the position via pulse-height weighted mean
- •currently thresholds are set to: seed=$4\times\sigma_N$; neigh=$3\times\sigma_N$; clust=$4\times\sigma_N$; MAX_STRIPS=10 (σ_N = noise RMS)

➡incorporate cluster shape information

➡incorporate timing information

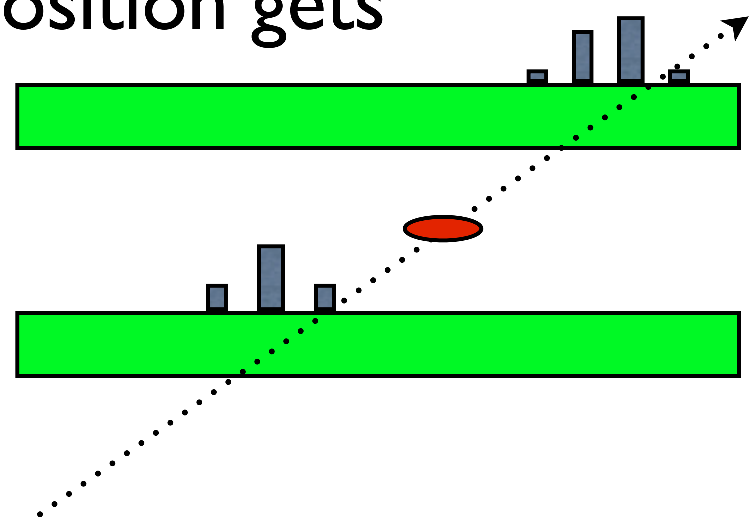
Hit Reco: stereo hits



•take all cluster pairs in adjacent stereo layers and create a 3d spacepoint (HelicalTrackHit)

•position between layers is taken as the midpoint

•for hits on tracks, the hit position gets corrected for the track direction



lcsim tracking conventions

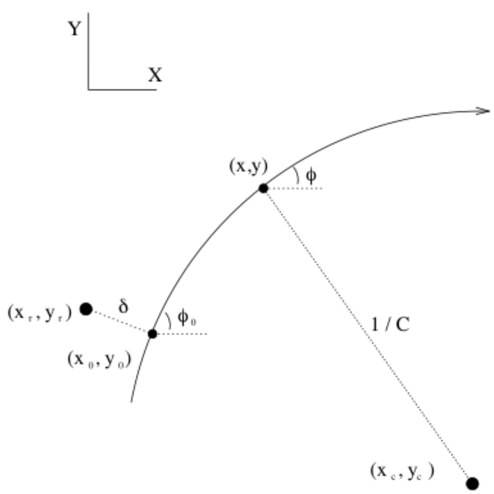
remember! In lcsim, the B-field is in the z-direction!
The beam is in x and the bend is y...

- tracks use a "perigee" parameterization similar to what was introduced by Billoir & Qian (NIM A311, 1992)

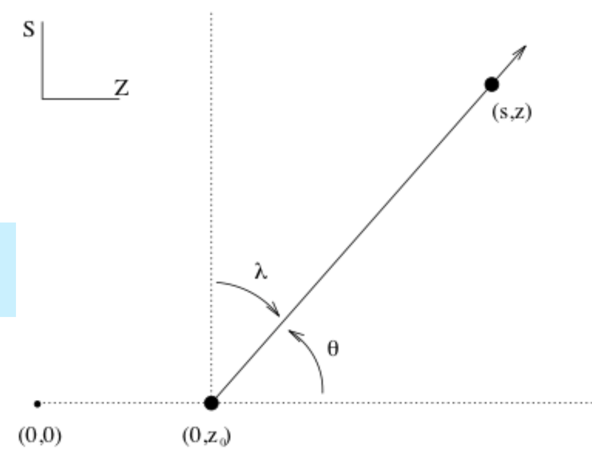
$$(\varepsilon, z_0, \theta, \phi_0, \rho) \Rightarrow (\delta, z_0, \tan\lambda, \phi_0, \rho)$$
$$\varepsilon = -\delta;\ \theta = \pi/2 - \lambda$$

} All quantities measured wrt point of closest approach to z-axis

- this isn't the most natural coordinate system for us...better to have the beam in z



Figures from "Helicoidal Tracks", J. Alcaraz, L3 Internal Note 1666



Track Finding: Strategies

```xml
<?xml version="1.0" encoding="UTF-8"?>
<StrategyList xmlns:xs="http://www.w3.org/2001/XMLSchema-instance" xs:noNamespaceSchemaLocation="http
    <TargetDetector>DarkPhoton-Thin</TargetDetector>
    <Strategy name="HelicalTrackHit Strategy">
    <!--Cutoffs-->
        <MinPT>0.200</MinPT>
        <MinHits>5</MinHits>
        <MinConfirm>1</MinConfirm>

        <MaxDCA>4.0</MaxDCA>
        <MaxZ0>4.0</MaxZ0>
        <MaxChisq>25.0</MaxChisq>
        <BadHitChisq>10.0</BadHitChisq>
    <!--Layers-->

     <Layers>
            <Layer type="Seed" layer_number="5" detector_name="Tracker" be_flag="BARREL" />
            <Layer type="Seed" layer_number="3" detector_name="Tracker" be_flag="BARREL" />
            <Layer type="Seed" layer_number="1" detector_name="Tracker" be_flag="BARREL" />
            <Layer type="Confirm" layer_number="7" detector_name="Tracker" be_flag="BARREL" />
            <Layer type="Extend" layer_number="9" detector_name="Tracker" be_flag="BARREL" />
        </Layers>
    </Strategy>

</StrategyList>
```

$p > 200$MeV
5 hits
1 confirm

$\delta < 4$mm
$z_0 < 4$mm
$\chi^2_{tot} < 25$
$\chi^2_{hit} < 10$

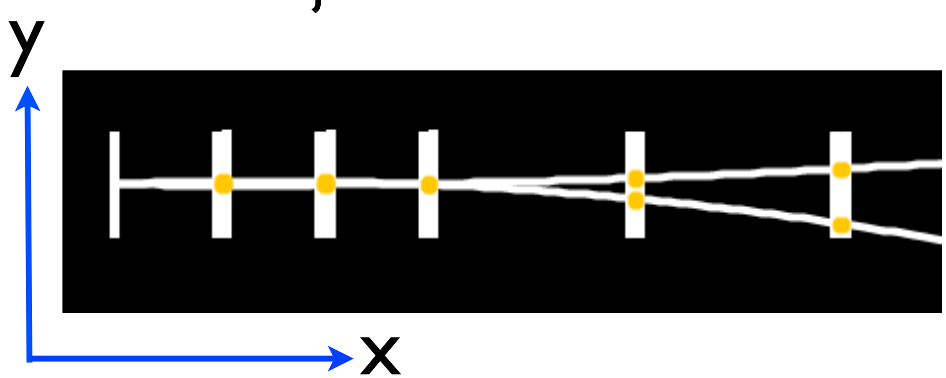
- seed-confirm-extend: seed track with inner hits, confirm with next layer, extend with outer layers
- tracking code uses HelicalTrackHits, so for the test run detector there are 5 layers

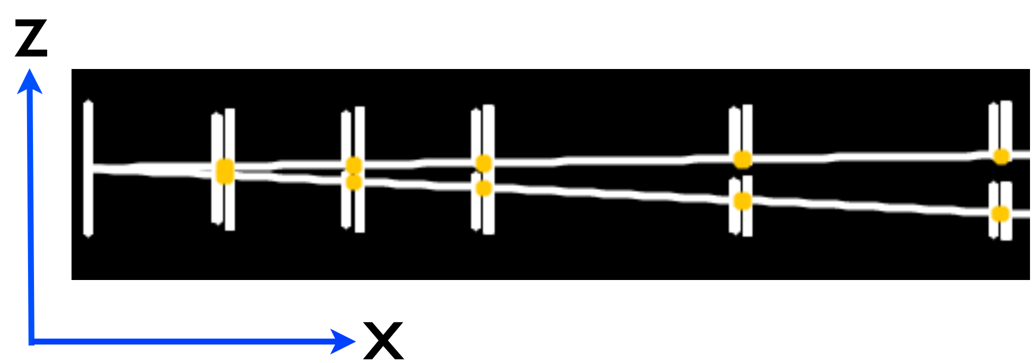
Track Finding & Fitting

- Track finding and fitting is done stepwise using "SeedTracker" and following the given strategy
- Loop over all HelicalTrackHits in the seed layers to create a 3-hit seed track...check if it passes strategy cuts
  - the DCA and z0 cuts are implemented as constraints...eg $\chi^2 = \chi^2 + (z_0 - z_{0max})^2/\sigma^2(z_0)$ if $z_0 > z_{0max}$
- For each seed, add in confirm layer and associate best hit...check chi2 again
- add in extend layers...reject if the added χ^2 exceeds χ^2_{hit}
- require track has required number of hits and chi^2
- after all tracks found, make sure no tracks with more than single shared hit

Helix Fitting

- The actual fitting of the track is done in "HelicalTrackFitter"
  - Circle fitter: P(x,y) ➪ C(δ,ϕ_0,ρ)
    - org.lcsim.fit.circle.CircleFitter
  - Z-segment fitter: P(s,z) ➪ L($\tan\lambda$,z_0)
    - s is the path length from the POCA to z-axis to the hit
    - org.lcsim.fit.zsegment.ZSegmentFitter
  - Each calculated the best fit parameters and covariance matrices
  - Both of these routines are non-iterative ➪ very fast
  - The results of the two fits are pasted together to form "HelicalTrackFit" object...which then gets put into a "SeedTrack" object which also includes the hits belonging to the track.





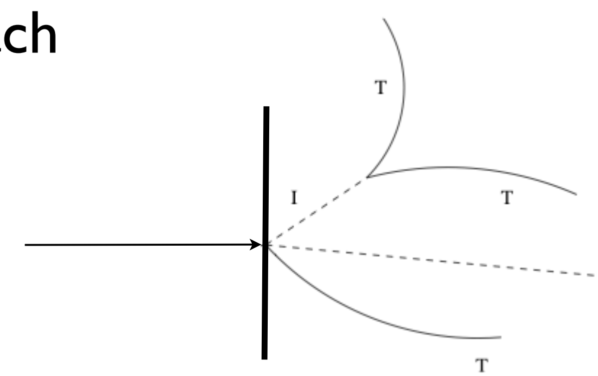
Multiple Scattering and Track Fitting

- The effects of multiple scattering are accounted for in the fit by adding to the uncertainty of the hit positions
  - calculates the amount of material transversed
    - org.lcsim.recon.tracking.seedtracker.MaterialManager
  - ...and the corresponding MS angular deviation based on path of track through material
    - org.lcsim.recon.tracking.seedtracker.MultipleScattering
  - MS error for hit in layer N is the angular deviations for layers 1⇨N−1 added in quadrature
    - e.g. the hit in layer 4 is given an MS error of sqrt($\sigma_{MS1}{}^2$+$\sigma_{MS2}{}^2\sigma_{MS3}{}^2$)
    - ...actually based on "scattering layers" != tracking layers
  - the MS error for the hit is added in quadrature to the intrinsic measured hit error

Vertexing

- 2-track vertexing is based on the Billoir et al. method
  - Billoir, Fruhwirth, Regler NIM A241, 1985
  - Billoir and Qian NIM A311, 1992
- Uses Kalman filter techniques and the perigee helix parameterization to calculate the vertex position and fitted track parameters
- Assumes no curvature near the vertex...probably need to iterate for long-lived decays
- Adding constraints is straightforward...currently we implement a target/beamspot constraint for prompt decays.

➡ Want to add in functionality to fit a third track originating at target...”TreeFitter” approach



To-do (and wish) list...

- At some point we should isolate HPS reconstruction from LCsim proper so things don't change out from under us...first we need to decide on the software framework
- Coordinate system isn't natural for a fixed target experiment...change it?
- Helix parameters...is perigee the best if we change coordinate system?
- Hit & cluster reconstruction (also needs simulation work):
  - incorporate timing information
  - use timing/cluster shape information in clustering
  - use timing in stereo hit making
- Track finding
  - speed is the issue here, and most of the time is taken looping over combinations of hits
  - we can be smarter about choosing stereo hits to include in fits
    - outer regions of inner layers cannot make a track...remove these hits
    - use calorimeter to sweep out a range of hits
    - generally, do sectoring of hits (there is infrastructure for this)
- Track fitting
  - need Kalman routine for track fitting...this has been started but needs more work
  - alignment algorithm