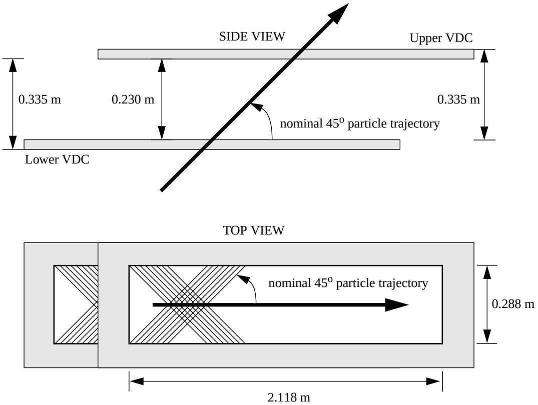
HRS Tracking

Ole Hansen

Jefferson Lab

Hall A DVCS Collaboration Meeting
Old Dominion University
December 19, 2013

HRS Tracking System: VDCs

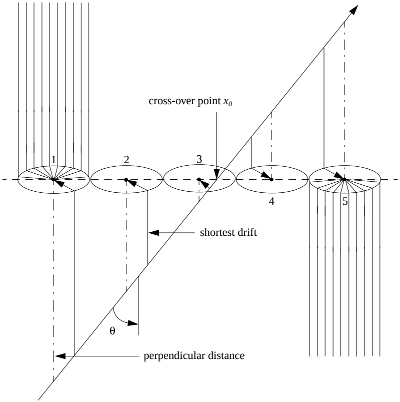


- *Vertical* Drift Chambers. (Ions drift vertically, see next slide.)
- Optimized for precision measurement of single tracks
- Two chambers, each with two wire planes (u/v) at $\pm 45°$
- 368 wires per plane, 4.24 mm wire spacing
- Standard tracking system for both HRSs. In use since 1996

VDC Operation: Clusters

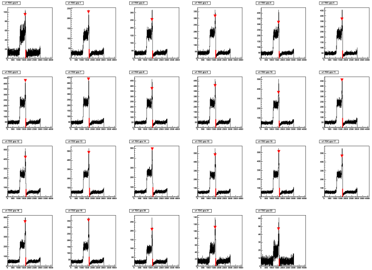
- Nominal track typically activates 4–6 wires → cluster
- Hit times w.r.t. trigger → drift times
- Must convert drift times → drift distances. Non-linear function
- Advantage of VDCs: Cross-over coordinate x_0 to first order independent of errors in the drift time-to-distance conversion
- Fit yields an x_0 position resolution of $\approx 225\ \mu$m FWHM

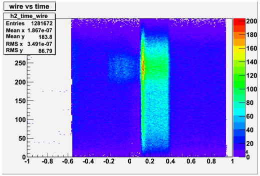
View along wires



VDC Calibrations

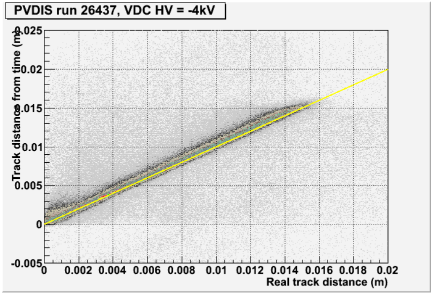
VDC time offsets





- Search for edge of timing spectrum peak in white spectrum calibration runs

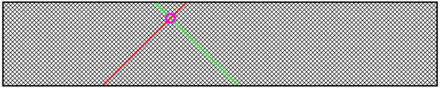
VDC time-to-distance conversion



- Fit analytic expression approximating time-to-distance relation
- Two linear sections with dependence on 1/tan(track angle)
- Resulting drift distance distribution should be flat
- Can use the same calibration runs as time offset calibration

Current (Traditional) Tracking Algorithm I

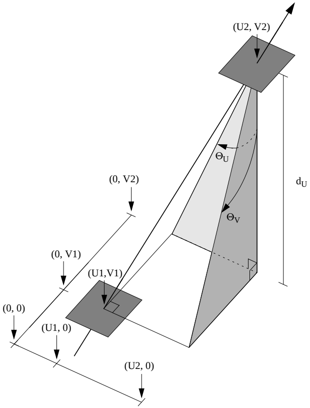
- Find clusters in all 4 planes
  - Allow up to 1 missing hit (gap size 1)
  - If multiple hits per wire, use the one with the shortest drift
  - If any plane has no cluster at all, no track is reconstructed for this event
- Fit cluster hits (drift distance vs. wire position) → cross-over coordinate, cluster slope
- Match u and v clusters in each chamber



  - Obvious if only one cluster per plane
  - If multiple clusters in any plane, see later
- Calculate "local track" (UV track, "stub") and its detector coordinates (x, x, x', y') from the matched u and v cross-over positions and slopes. Positions will be accurate, but angles will not.

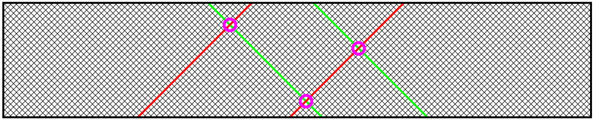
Current (Traditional) Tracking Algorithm II

- Combine UV tracks from lower and upper chamber
- Re-calculate u and v cluster slopes from upper and lower cross-over positions → "global" angles. These angles have good accuracy now, directly related to the position resolution of the cross-over point.
- Recalculate detector coordinates based on the updated cluster slopes



- The lower plane's UV track coordinates (x, x, x', y'), are used as the detector coordinates of the reconstructed focal plane track
- Focal plane tracks are reconstructed to the target by multiplication with the reverse transport matrix

Current Tracking Algorithm With Multiple Clusters I



This is where trouble starts. With only two readout coordinates, ambiguities from multiple clusters cannot be resolved.

The code attempts this:

- "UV matching": Find pairs of u and v clusters in each chamber
  - Determine if u or v have more clusters $\rightarrow$ p, q, with $n_p \geq n_q$
  - Pair each p-cluster with the one in q whose *pivot wire drift time* is closest to the p-cluster's pivot wire drift time
  - Yields exactly n_p UV pairs
  - Pairs are not rejected if outside of the physical chamber area
  - This is obviously wrong (see later)
- For each UV pair, calculate "local track" coordinates, as before

(over)

Current Tracking Algorithm With Multiple Clusters II

- "BT matching": Consider all combinations of the n_B pairs in the lower (B) chamber to the n_T pairs in the upper chamber (T) ("BT pairs")
  - Project the local track of each B-cluster onto the upper plane T and calculate the distance d_{BT} from the projected point to the T-cluster's cross-over point
  - Repeat, this time projecting the T-cluster onto B, yielding d_{TB}
  - Assign the "error value" $E = d_{BT}^2 + d_{TB}^2$ to this BT pair
  - Sort the BT pairs by error value
  - Pick the BT pair with the smallest error as the best reconstructed track
  - Mark the two UV pairs (matched UV clusters) of the picked BT-combination as "used"
  - Continue selecting tracks from the BT pairs in order of increasing error value, *skipping pairs with any already-used UV pairs*
  - There is currently no upper limit on the allowable error
  - Yields exactly $\min(n_B, n_T)$ final tracks
  - This is better, but still wrong (see later)
- Calculate overall χ^2 for each track, based on differences of track crossing positions to drift distances.
- Reconstruct each final track to the target

Current Algorithm With Multiple Clusters: Discussion

What is wrong with these algorithms?

- In the UV matching step: Pivot wire drift times of matching *U* and *V* clusters are not correlated. At best, a cluster with a large time offset (accidental) will fail to match any in-time cluster, but matching between in-time clusters by pivot drift time is essentially random
- In the BT matching step: Marking UV pairs as "used" does not prevent two different tracks from containing the same *cluster*. However, multiple use of same clusters is what should be prevented. Clusters are almost never shared by two different tracks, and if so, will likely be corrupted (bad cluster fits).

Additional problems:

- No rejection of UV pairs outside of the active chamber area
- No error value cutoff
- χ^2 calculation probably rather poor since perpendicular track crossing points are compared to shortest drift coordinates

Effects On Tracking Performance

My preliminary analysis:

- For (2,1;1,1), (3,1;1,1) cluster occupancies and similar (only one plane has multiple clusters), the correct track is most likely found
- (2,2;1,1) and similar give one track, but there is a $\approx 50\%$ probability of picking the wrong cluster, hence getting bad reconstruction
- For (2,1;2,1) and similar, there will always be two tracks, one good, the other most likely bogus (ghost track)
- For (2,2;2,1) and similar, two tracks will be found, one bogus, the other also bogus with $\approx 50\%$ probability
- For (2,2;2,2) and higher, ghost tracks continue to appear in higher numbers and the probability that the correct track is found continues dropping
- $\rightarrow$ track multiplicities too high, tracking efficiency reduced
- $\rightarrow$ must reject all events with multiple clusters in *more than one plane*

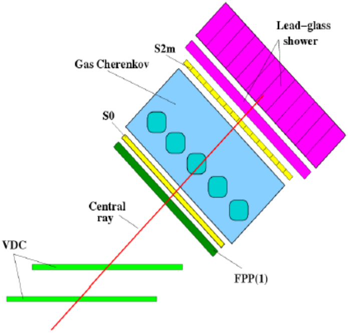
Immediate Fix To The Tracking Algorithm

- Keep all UV cluster combinations, except those outside of the chamber area
- When picking BT pairs in order of increasing error, ensure that each underlying *clusters*, not the UV pairs, are only used exactly once
- Apply a cutoff to the allowable BT matching error, estimated from the measured angular resolution of the local cluster track slopes
- Improve the χ^2 calculation
- This is straightforward. Estimate 1 week of programming, 2 weeks for testing.

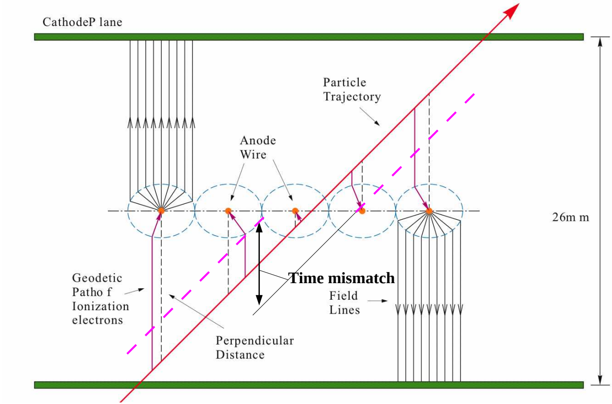
Further Improvements

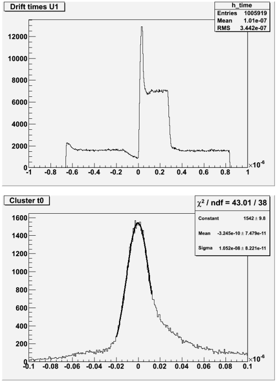
The problem boils down to the question how to resolve UV matching ambiguities without a 3rd readout coordinate

- Rely on the BT matching error value described previously
  - May actually work fairly well — to be tested, ideally quantify with simulation
- Add an additional readout plane → planned for the upcoming G_M^p run
  - Can only help, although with an u/v-only FPP plane, maybe not as much as hoped
- Do a 3-parameter cluster fit to extract the cluster time offset
  - Definitely useful to reject accidentals occurring at high rates, probably won't help with low rate data
  - → see next page



New Algorithms: 3-Parameter Cluster Fit





- Non-linear 3-parameter fit to extract track time offset t_0
- Computationally expensive: ca. $\times$20 slower than 2-parameter fit
- $\approx$ 20 ns FWHM time resolution $\rightarrow$ background rejection factor $\approx$ 10-20
- Required for APEX: expect $\approx$ 2 accidental tracks per trigger
- Code written, still needs testing/debugging and integration

What Did ESPACE Do?

To the best of my recollection, single-cluster events were handled exactly as described here. Multi-cluster events prompted ESPACE to

- perform a 3-parameter fit to all clusters
- consider all possible 4-tuples of clusters and calculate an "error parameter" for each tuple, similar to χ^2, considering all the wire hits from all the clusters, but also including each cluster's fitted time offset
- reconstruct exactly one track, *viz.* the one corresponding to the 4-tuple of clusters with the smallest error parameter, subject to certain cutoffs

Comments

- No obvious incorrectness
- There is a discontinuity between clean one-cluster-per-plane events and events with any additional clusters, no matter how spurious
- One might be concerned that the poor resolution of the fitted t_0 could lead to accidental misassignments
- The fitted time offset is not statistically independent of the drift distances

Conclusions

- The current HRS tracking algorithm is definitely broken for events with multiple clusters in more than one plane. Such events should be rejected in any analysis with the present code.
- It appears that the errors in the algorithm are fairly easily correctable
- Additional improvements are possible with more work, both in software only (3-parameter fit) and by using additional tracker planes (*e.g.* FPP)
- Unfortunately, the HRS tracking will always have poor noise resistance due to construction of the VDCs with only two readout coordinates. This is an inherent design limitation of the VDCs.