SvtTrackFinder

Gerry Lynch

November 19, 1997

- At present there is only one Track Finder for the SVT.
- It finds tracks in the SVT without using information from other detectors.
- In the future we will have other track finders that work with SVT data, especially one that adds SVT hits onto tracks that are found in the DCH.
- But for finding low momentum tracks a stand-alone SVT track finder is essential.
- SvtTrackFinder is designed to find all tracks in the SVT that
  - have points in at least 4 layers,
  - have a radius of curvature greater than 8 cm, and
  - come within some distance from the origin in z (now 10 cm).
- When this track finder was written, it was expected that others would be written. But, since we are now relying on SvtTrackFinder, it needs to be described.

SvtTrackFinder Tasks

- The function of SvtTrackFinder is to read in SvtHits and to produce SvtTracks.
- The hits are described geometrically as a trajectory in space, the z-hit trajectories are normal to the z-axis and most of the phi-hits are parallel to the z-axis.
- The main tasks in SvtTrackFinder are
  - choose acceptable hits
    - ⇒ there is a pulse height cut here
  - form space points from all pairs of phi-hits and z-hits in the same wafer.
    - ⇒ reject a few of these if the z-hit pulse height and the phi-hit pulse height disagree greatly.
  - Give these space points to BBPat2, which finds the tracks.
  - Put these tracks in the SvtTrackAList and make fits to them.
- The rest of this report concerns what is done in BBPat2.

General description of BBPat2

- BBPat2 is a track finder that is derived from a track finders called Pat1 and Pat2 that were written about 1980 by Henri Videau for the TPC.
- It is written in Fortran and has in it many hard-wired constants. If we decide to keep this method, we should rewrite the code.
- One reason that this has not been done is that it is not at all clear if it will operate satisfactorily in the presence of expected backgrounds.
- The approach of BBPat2 is to try a large number of combinations of points as track candidates and try to make the calculations for each combination relatively fast.
- Although the code is not organized in a way that makes this clear, one can think of BBPat2 as having two regimes:
    - One is the code that finds and accepts track canditates from points in three layers.
    - The other is the strategy that is employed in selecting which points to consider and which tracks to keep.

The Track Finding Method of BBPat2

- The track finder is given three space points, each one on a different layers, either three measured points or two measured points and the origin.
- It calcuates the helix that corresponds to these points, ignoring the z position of one of the points.
  - The point for which the z is ignored is either the origin or the middle measured point.
- This set of three points is accepted as a track candidate if:
  - The z of the point not used to calculate the orbit is within tolerance. The tolerance for the origin is large (10 cm).
  - The arc of the track that is formed by the three points spans less than 180 degrees and does not cross either the inner wall or outer wall of the SVT.
- A loop is made over all space points, and those points that are within some tolerance are put on the track.
- If this track has points on 4 or 5 layers, it is accepted, though it may be rejected at a later stage.
- All of the preceding calculations are done with rather simple and fast calculations. No fit is done. Only one trig-function call is made per track candidate.

General Strategy Considerations

- There are two extreme strategies that might be followed.
  - ♦ The early bird strategy.
    - ⇒ With this strategy whenever a track is found, it is kept, and the points on the track are not allowed to be used on other tracks.
    - ⇒ This method is the one that was used in the original Pat2.
    - ⇒ It has the advantage that it is simple and fast.
    - ⇒ For the SVT, it is terrible (more on this later)
    - ⇒ The success of this method is greatly influenced by the strategy for selecting triplets, that is, what track gets to be the early bird.
  - ♦ The complete strategy
    - ⇒ With this strategy, one does not mark points as used All possible tracks are found.
    - ⇒ Often this method finds tracks that share points with many other tracks.
    - ⇒ Then this method requires the use of a selection procedure that finds the "best" set of tracks.
    - ⇒ This method is relatively slow and is better than the early bird strategy only if the selection procedure is effective.
- BBPat2 uses a strategy that is not at either of these extremes.

SVT Ambiguities

- There are ambiguites for the SVT that are not present for a truly three-dimensional detector.
- When there are N hits in a half-module, we calculate N^2 space points.
- If two tracks go through the same half-module for 4 or 5 layers, we expect to have at least N^2 tracks accepted. Almost always the z-hits for one track will form a good looking track with the phi-hits of the other track.
- In this case the early bird strategy is a poor one.
- Some cases are much more ugly. Recently I have seen two cases (one an electron and the other a muon) where a track at zero dip angle and a momentum of about 100 MeV/c makes 5 turns, which puts two pairs of 5 closely lying tracks. for one of these more than 100 passing track candidates with at least 5 points were found.

Momentum Strategy

- For the SVT the dominant position errors are Coulomb errors.
- Therefore the spatial tolerance (the "road") gets larger as the minimum radius gets smaller. The road also has an angle dependence.
- SvtTrackFinder is set up to call BBPat2 a number of times, first for a relatively large radius of curvature, and last by the minimum radius. The idea is to find the high momentum tracks first and save them out before looking with a larger road.
- Up until quite recently it did four calls, at radii of 80 cm, 30 cm, 15cm and 8 cm.
- One problem with this approach is that the program sometimes finds the points for a relatively low momentum track in layers 1, 2, and 3, and adds a wrong point on layers 4 or 5 to these to get an incorrect track that passes the tests.
- Recently it was found that with an improved selection procedure the number of mistakes is reduced by doing only the last two radius calls.

Triplet Strategy

- The program can do the search using a number of triplets of layers.
- At one time the program used quite a few layer triplets, but as the search in any one triplet has become more complete, the number of triplets has decreased. Now, in the standard case, three triplets are used in the order (1,3,5), (2,4,5), and (1,3,4).
- The present strategy temporarily keeps all the track candidates found for one triplet and then does a track selection that eliminates ambiguities from this set before going to the next triplet.
- So it uses the complete strategy for the track candidates that are found in one triplet, and the early bird strategy for tracks from different triplets.

Selection Procedure

- O When there points that are shared between two tracks, there is a selection procedure that eliminates this problem.
- O The program may go through the selection code many times to reject bad tracks.
  - ♦ At each pass it accepts all tracks that have no shared points and rejects the “worst” track.
- O The worst track is the one with the smallest Quality number. The Quality number has a number of components
  - ♦ The number of points on the track is added to the quality.
  - ♦ The number of points on other tracks that are also on this track (times some factor) is subtracted from the Quality.
  - ♦ A kinematic chi-squared per degree of freedom is subtracted. Since no fit is done, this chi-squared has only how well the points agree with the three point orbit.
  - ♦ A dE/dx chisquared is subtracted also.

Performance without Track Background

- In the most of running that has been done we have had electronics background, but no track background.
- We can test test the performance of the track finder with SvtTestTracks, written by Paola Grosso and Diego Zanin. It tries to find the efficneincy for finding good GTracks and the purity of the tracks that are found. It's main limitation is that the Monte Carlo truth information that it has access to is not as good as we would like.
- Here is a table of the results from a large sample of multi-hadron events before and after the improvements that were made in October.

For good GTracks:	Before	After
one missing point	5.4%	3.5%
two missing points	1.7	0.8
track missed	2.2	2.0
For found tracks:		
one wrong point	2.8%	1.7%
> one wrong point	2.5	1.9

Running with Background

- We can now add background from lost beam particles to our generated events.
- With Ed Frank's help we are set up to use the Mixer modles with SvtReco to add the background.
- In the releases of this month one can still do the usual link of SvtReco with "gmake SvtReco.bin". In addion one can do "gmake SvtReco.bkg", which produces SvtRecoBApp, which can be run with SvttestBkg.tcl.
- The only place where I have run this is on design at LBL.
- This operation has not been ironed out and there are many aspects of it that are as yet unclear to me.
- It is clear that the time increases rapidly as more background is added, and so does number of fake tracks.

Very Preliminary Results with Background Added

- These results are from a small sample of 30 events. The 2a column uses the same program as before. The 2b and 4b colums reject tracks that have a chi-squared that is more than 10 times the expected.

times nominal	0	2a	2b	4b
numbr of Digis	193	454	454	715
time per event (sec)	1.0	9.3	9.3	27.
for good GTracks:				
one missed pt	2.4 %	3.0 %	3.0 %	9.5 %
two mised pts	0.0	0.0	0.0	0.0
missed track	1.2	2.4	3.0	2.4
for found tracks:				
one bad point	1.1	2.2	2.2	1.6
>1 bad point	0.6	8.1	4.5	9.7

- Much of the time is taken by the mixers and SvtMakeDigis.
- Almost all of the bad tracks have radius < 15 cm.