SpinQuest tracker algorithm

E. Fuchey

October 5, 2022

1 Summary

1.1 Sources

https://github.com/E1039-Collaboration/e1039-core/blob/master/packages/reco/ktracker/KalmanFas
https://github.com/E1039-Collaboration/e1039-core/blob/master/packages/reco/ktracker/KalmanFas
https://github.com/E1039-Collaboration/e1039-core/blob/master/packages/reco/interface/FastTrackl
https://github.com/E1039-Collaboration/e1039-core/blob/master/packages/reco/interface/FastTrackl

1.2 Main functions

buildTrackletsInStation(stationID, listID, pos, window) Makes pairs of hits in xx', uu', vv', in selected station; if a view doesn't have hits, stop here.
Combination of hits to form tracklets:

- loop on x hits: combine x with u: to each x can only corresponds a range in u_min<u_pos<u_max;
- inside loop x, loop on u hits: reject all u hits which do not meet u_min<u_pos<u_max: for those who do, calculate v_window, v_min, v_max;
- inside loop u, loop on v hits: reject all v hits which do not meet v_min<v_pos<v_max: for those who do, add a tracklet with the combination of hits, and fit it;
- if tracklet is "valid" (see below) it is kept, otherwise it isn't;

Once the combinations have been made, tracklets are added intot the tracklet list

buildBackPartialTracks() Combination of tracklets from station 2 and 3 to form backtracks.

- loop on station 3 tracklets; if not coarse mode, loop on the tracklet 3 hits to extract only the X hits;
- inside loop 3, loop on station 2 tracklets; if not coarse mode, loop on the tracklet 2 hits to extract only the X hits;
- fit the backtrack in X; then check the proportional tubes: we want at least one hit there;
- otherwise, add the two tracklets together to obtain tracklet 23 (aka backtrack), and fit it. If the fit χ^2 is too high, reject tracklet; if not coarse mode, resolve left right for backtrack.
- Then keep only the best backtrack (i.e. with best χ^2 or best proba).

buildGlobalTracks() Combinations of backtracks and hits in station 1 to form global tracks:

- Loop on backtracks: evaluation of windows with Sagitta method if KMag ON, with extrapolation otherwise; then build tracklets in station 1 (using the windows obtained with the search window method);
- inside loop on backtracks, loop on station plane (2 stations);
- inside loop on station plane, loop on station hits; multiply (?) tracklet 1 and backtrack, and fit; reject if no hodo hits; if not coarse mode, resolve left right for backtrack, then remove bad hits (on what criteria?); then keep the global track with the best fit; If Kalmann filter, reconstruct vertex, keep the track with the best vertex χ^2;
- after the loop on station1 hits, keep the very best track: The selection logic is, prefer the tracks with best p-value, as long as it's not low-pz; otherwise select the one with best vertex χ^2; then fall back to the default only choice.
- After the loop on backtracks, if best track from each station have momentum less than a defined value of ?, merge tracks; if the merged track is is better than the separate ones, keep it, otherwise, keep the best one of the two (better = with best χ^2 or best proba).

1.3 Other useful functions

”valid” tracklet: in station 1 to ”nStations” (nstations being the number of stations involved in the tracklet); slope, interesection within the limits assigned for the station; error for these parameters positive; probability has to be better that ”loose” probability defined for the station if the station is the last one if station ID < nStations-1 (?) the tracklet has to have at least one hit in each station, and 4 hits total, χ^2 has to be lower than 40 for station 2,3 the tracklet has to have at least one good hit in each station, and 4good hits total; for a full track, station 1 tracklet has to have at least one good hit in each station, 4good hits total, and the track prob has to be better than defined ”tight” proba+ inverse momentum has to be between defined limits; tracklet prob: prob (χ^2, ndf) with ndf = number of hits -5 for full tracks, number of hits-4 for partial tracks

”hodomask” for tracklets: Returns true if hodoscope hits can be found on the path of the track; loops on stations, then on hits: evaluate the track position on the hodoscope plane, then check that hodoscope hits correspond to that position (within some errors).

ResolveLeftRight: 4 possibilities: ++, +-, -+, –; Loop on pair of consecutive hits; then loop on all 4 possibilities: calculate local intersection and slope for each hit, considering each possibility; compare it with the global slope and interesection with the ”pull” (the square root of the sum of delta slope squared /err_slope square and delta inter squared / err inter); when the ”pull” is below the user defined threshold, the hit sign of each hit is affected with the possibility being considered.