Speed up approaches in Cellular Automaton (CA) track finder

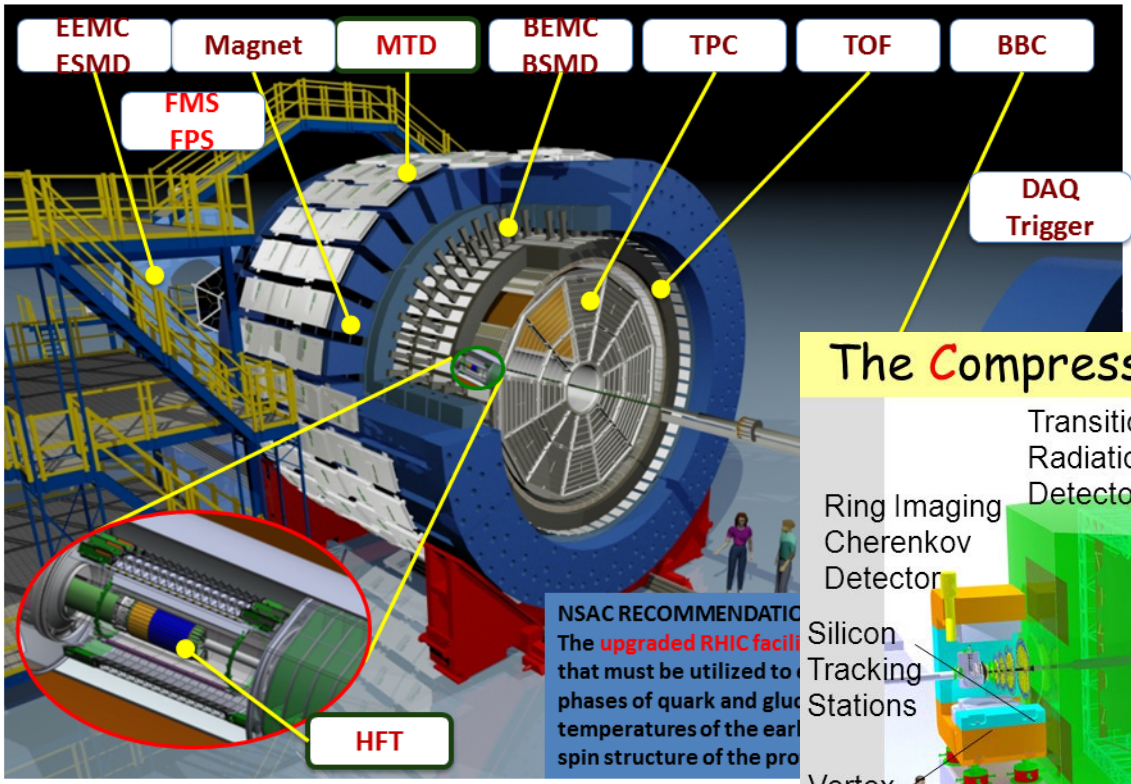
G.Kozlov[1,2,3], I.Kisel[1,2]

[1]Goethe-Universitat, Frankfurt, Germany
[2]FIAS, Frankfurt, Germany
[3]JINR LIT, Dubna, Russia

[FAIR phase 0]

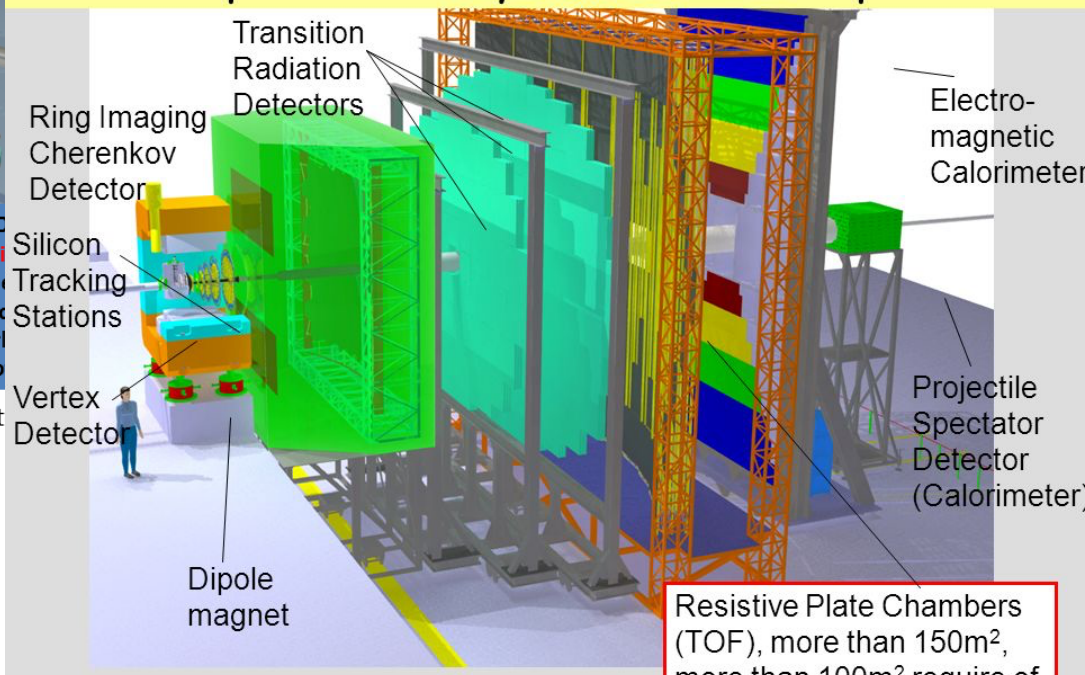
STAR Detector System

15 fully functioning detector systems



X10^3 increases in DAQ rate since 2000, most precise Silicon Detect

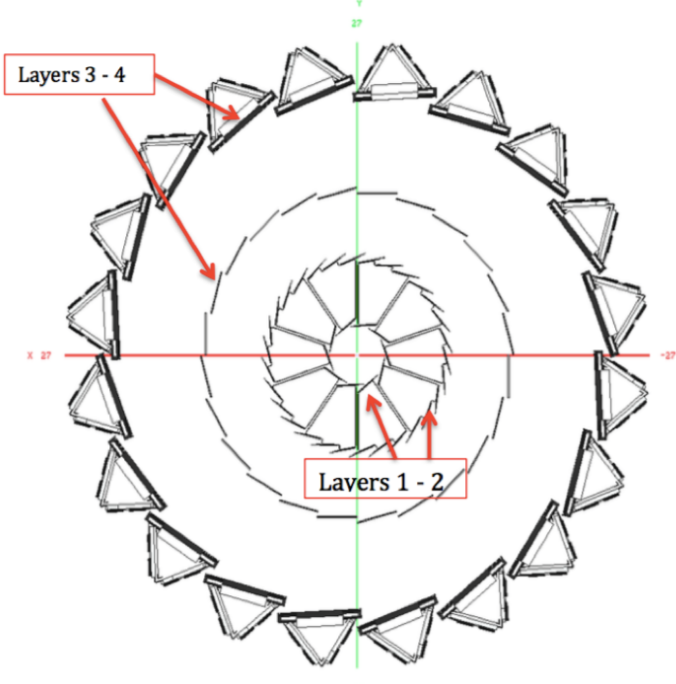
The Compressed Baryonic Matter Experiment

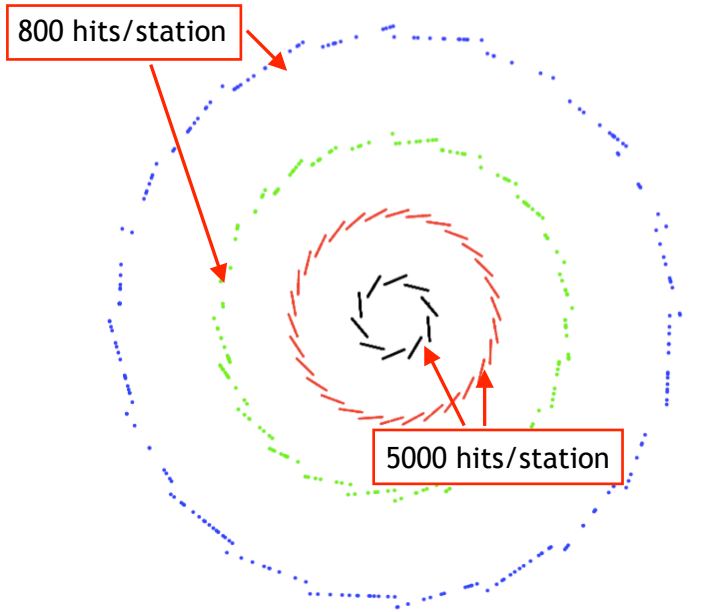


Resistive Plate Chambers (TOF), more than 150m^2, more than 100m^2 require of strip-based coverage

HFT detector at STAR

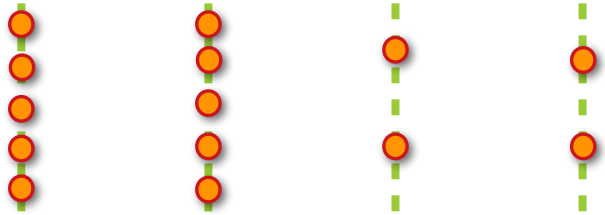
- 4-layers:
  - Layer 1 - 2: PIXEL (2.5 cm, 8 cm)
  - Layer 3: IST (14 cm)
  - Layer 4: SST (22.3 cm)
- Up to 800 tracks per event
- Pileup on each PIXEL layer is about 5000 hits
- CA tracking procedure is similar to CBM L1 track finder





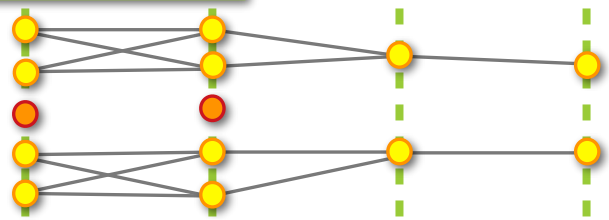
CA track finding procedure

1. Hits



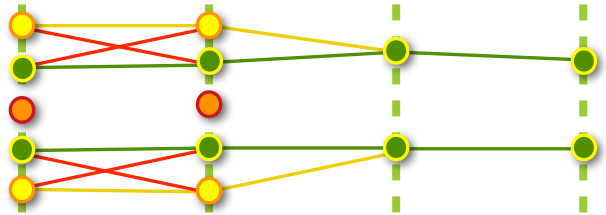
Read input hits, sort them in memory taking into account grid structure and associate them with grid.

2. Track segments



Find all possible track segments - triplets using grid structure, select those that meet the criteria, mark neighbours.

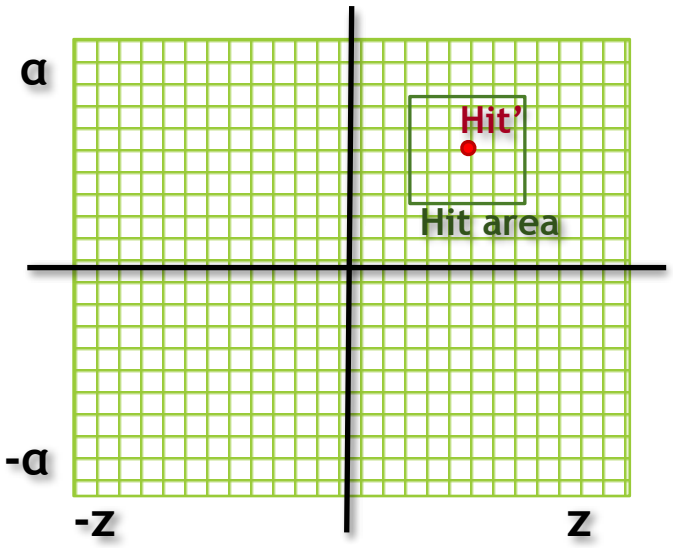
3. Tracks



Collect track segments into track candidates, fit them, select the best candidates and save output tracks.

Grid structure in HFT

Grid structure allows to establish compliance between hit coordinates and bins of this structure.



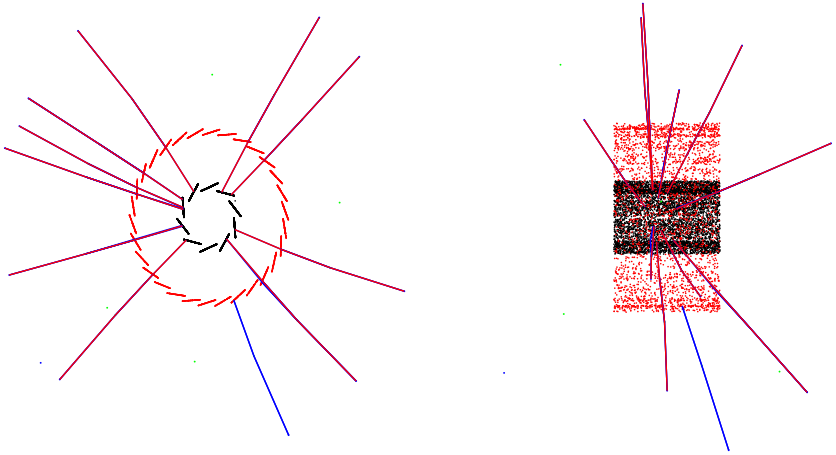
- ***Grid*** is based on *Z-coordinate* and *angle*.
- Track finding direction – *from outer station*.
- Main steps of grid usage:
  - Extrapolate hit -> hit' to the previous station in direction of primary vertex (*PV*);
  - Create *Hit area* around the hit' using ***dz*** and ***da***;
  - Search for the next hit inside of the *Hit area*.
- Number of bins in grid depends on the number of hits on station.

Track finding examples

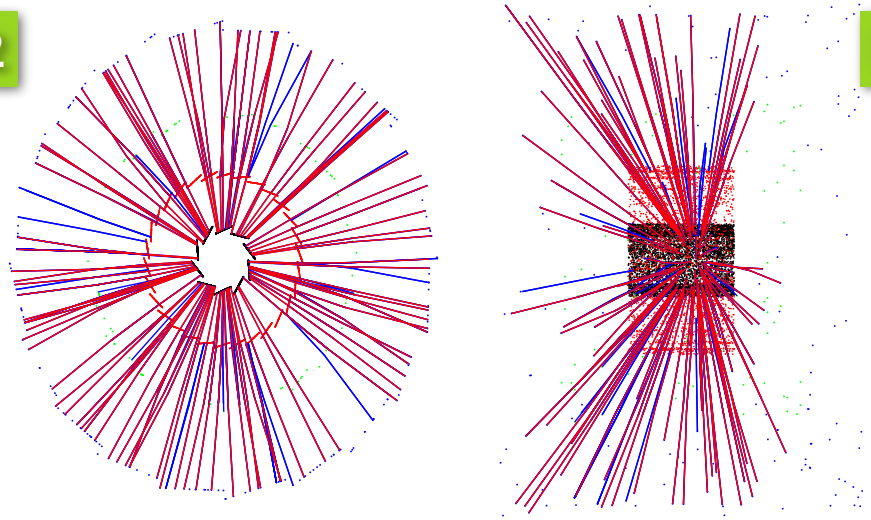
Triplets

Tracks

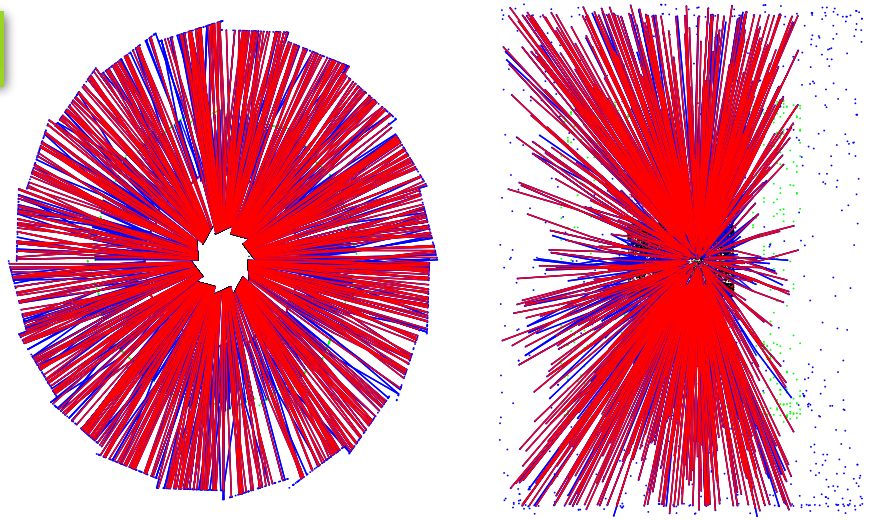
1



2

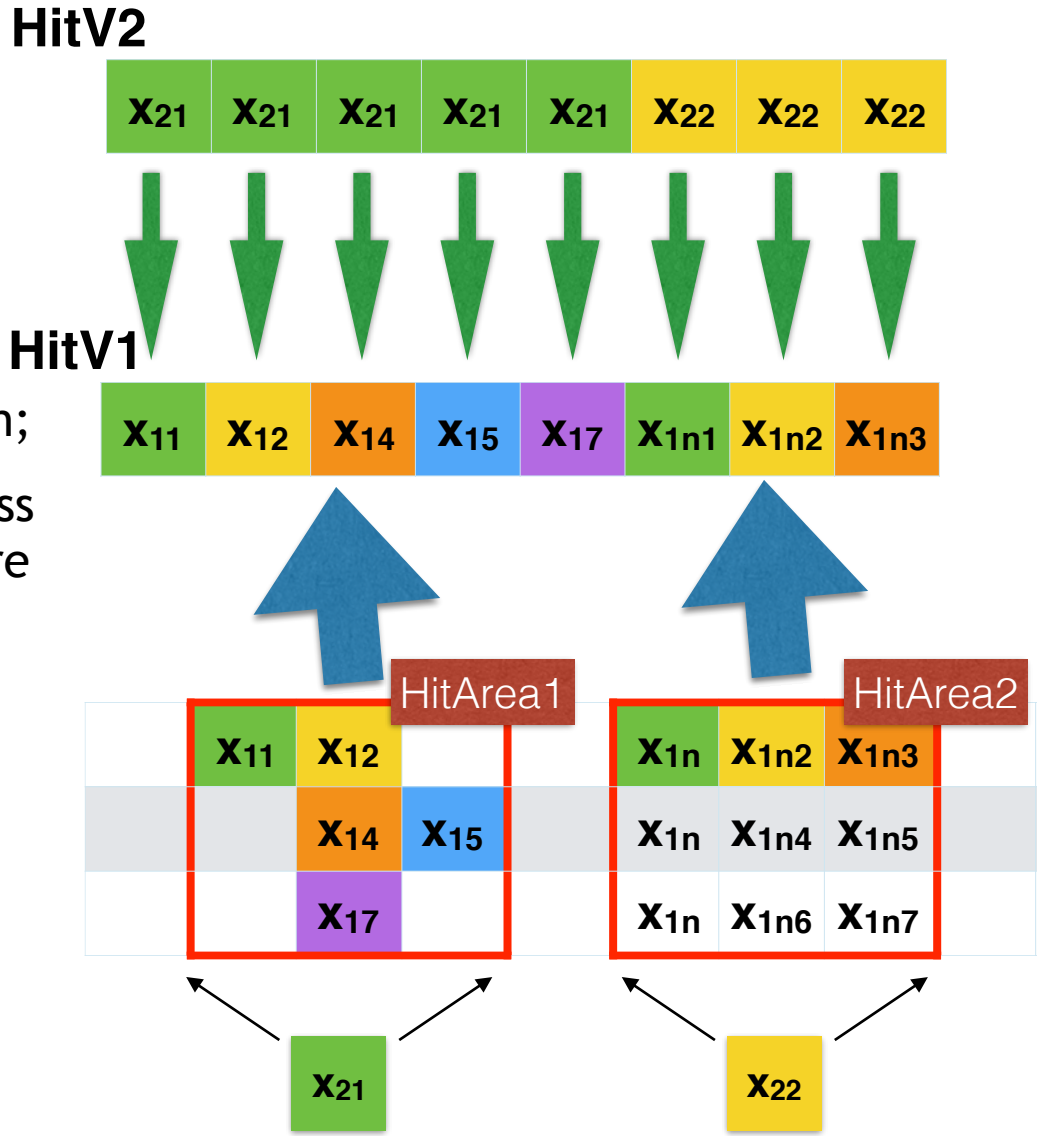


3



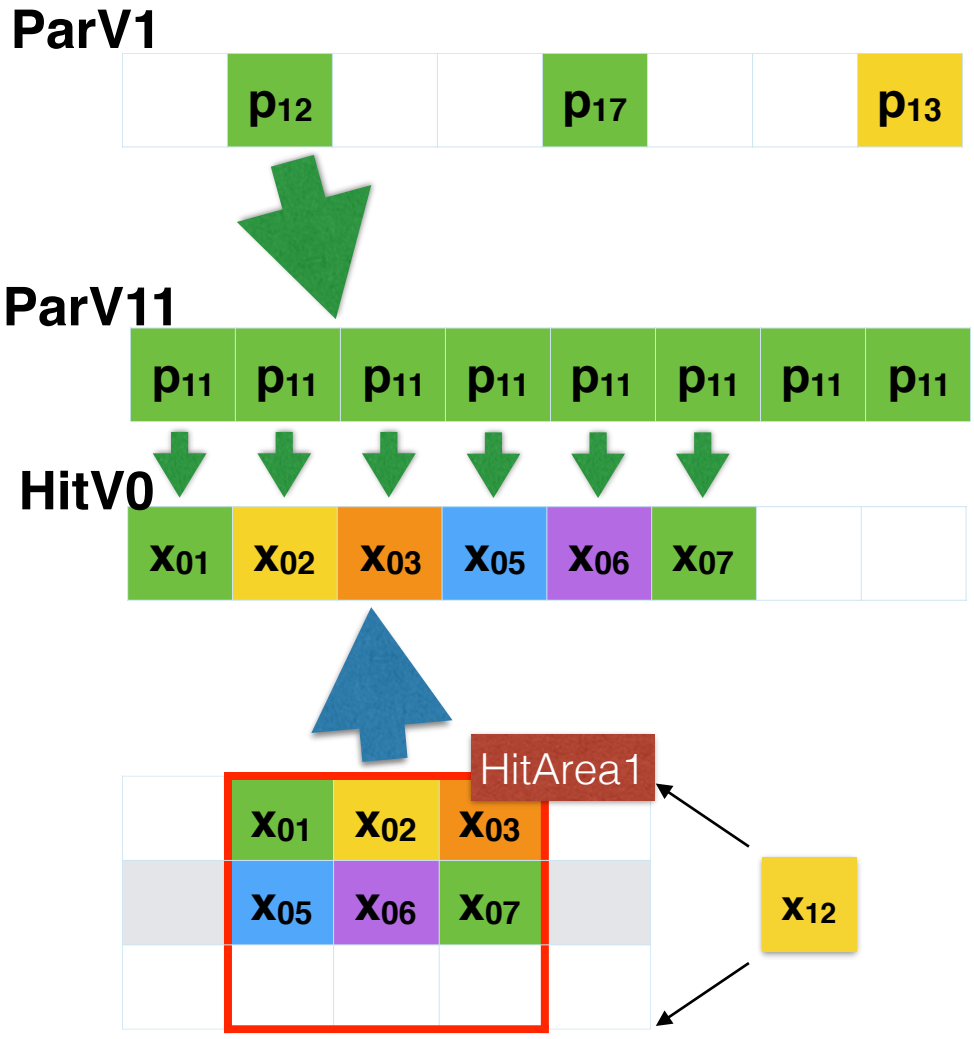
Triplet finding vectorization

- ***First segment calculation:***
  - Take hit from outer station;
  - Extrapolate it to PV, create HitArea and look over the hit candidates from middle station;
  - If the number of candidates less than vector size, take one more outer hit;
  - Create maximal filled hit vectors;
  - Initialise vector of parameters for the outer hit;
  - Transport it to the middle station in vector mode, check cuts.



Triplet finding vectorization

- ***Second segment calculation:***
  - Take active element from parameters vector and fill new parameters and hits vectors by this element;
  - Extrapolate corresponding middle hit to PV, create HitArea and look over the hit candidates from inner station;
  - Create vector of inner hits;
  - Transport to the inner station in vector mode, check cuts;
  - Save triplets and go to the next active element from parameters vector.



Triplet vectorisation speed up

Efficiency: 90.3% (75% correct hits)

Efficiency: 81.8% (100% correct hits)

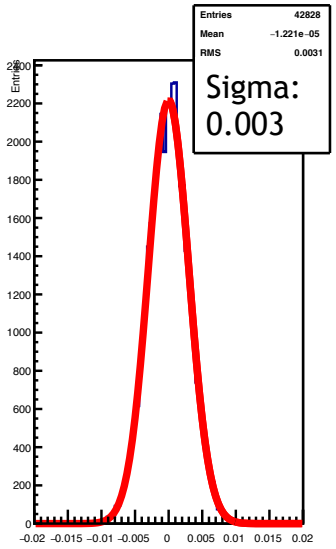
Calculation time in ms/event

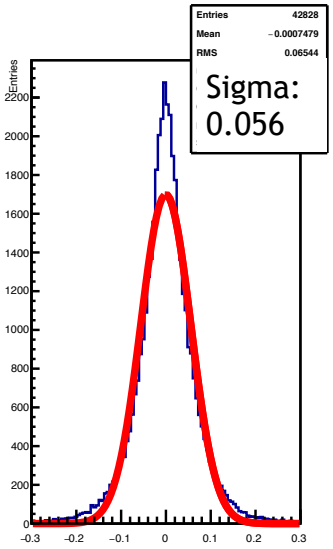
	Scalar	Vector SSE(gcc)	Vector SSE(icc)	Vector AVX-1(icc)
-Input hits	*0.6*	*0.6*	*0.6*	*0.6*
Sort and fill grid	4.7	1.7	2.3	2.3
Triplets	27.1	6.6	5.5	4.6
Tracks	4.7	1.5	1.4	1.2
-Output tracks	*0.8*	*0.8*	*0.8*	*0.8*
Total	37.0	10.2	9.3	8.2

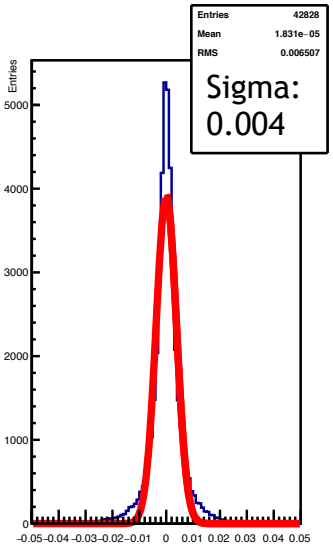
Efficiency

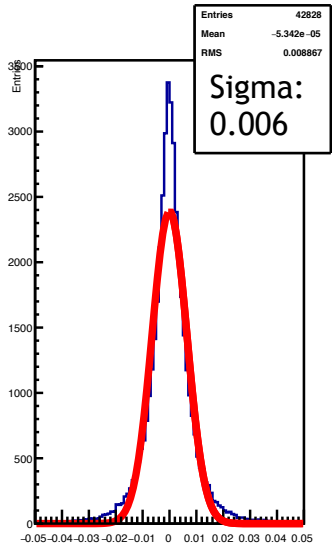
	75% correct hits	100% correct hits	MC Tracks / event
All tracks	90.3%	81.8%	237.2
Primary high-p	98.1%	93.7%	27.4
Primary low-p	92.2%	83.3%	198.1
Secondary high-p	83.6%	76.0%	0.7
Secondary low-p	36.5%	26.1%	10.9
Ghost level	17.5%	36.9%	
Correct tracks	215	194	
Time / event	8.2 ms		

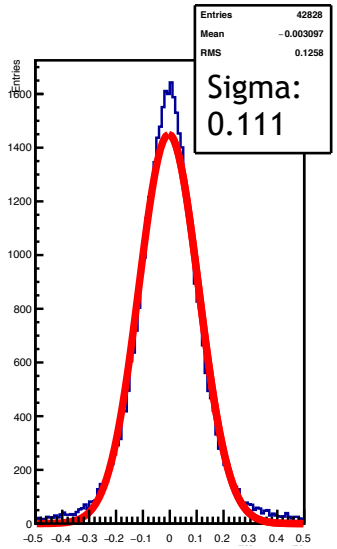
Residuals and Pulls in last hit

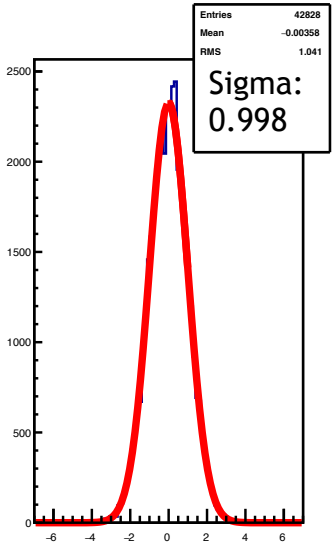


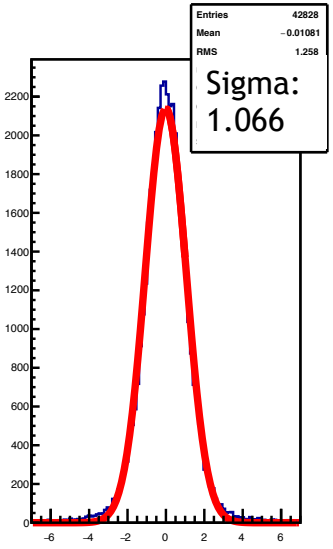


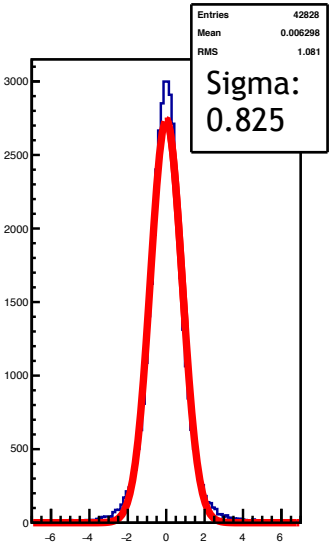


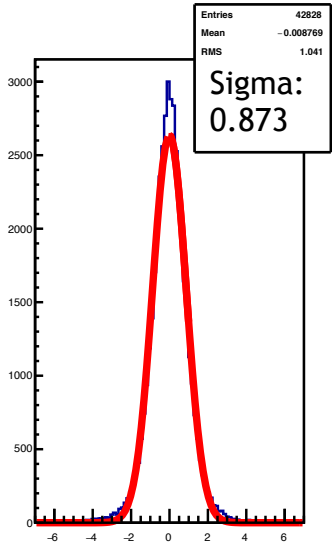


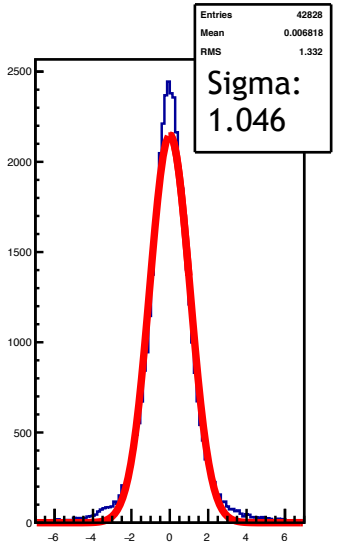












Summary

- Usability of data structures appropriate for vector calculations was tested with the HFT CA track finder. Such approach could be implemented in the other CA tracking algorithms, including CBM L1 CA track finder.
- Memory usage was strongly decreased by excluding of needless calculation steps.
- Vectorization of triplets and tracks calculation gives us speed up with factor up to 3.9 on SSE and 4.5 on AXV1.

Plans

- Run CA track finder on Intel Xeon Phi card.
- Vectorize hit sorting.
- Parallelize calculations between logical cores.
- Maximally reduce the amount of used memory.