Iterative Retina for high track multiplicity in a barrel-shape tracker and high magnetic field

Wendi Deng
Université libre de Bruxelles (ULB) & Central China Normal University (CCNU)

Real Time conference - 2020

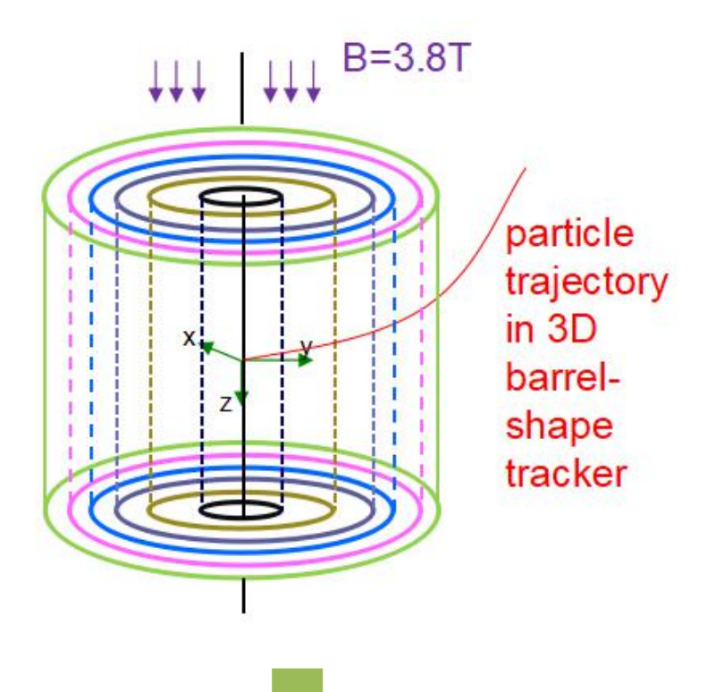
Introduction

Real-time track reconstruction in high energy physics experiments at colliders running at high luminosity is very challenging for trigger systems. To perform pattern-recognition and track fitting, artificial Retina or Hough transformation algorithms have been introduced in the field which have usually to be implemented in the state of the art FPGA devices.

Our study is try to use the Retina which runs in an iterative way to identify track for barrel-shape tracker in case of high magnetic field and high track multiplicity. Our work on a concrete case, we are studying the performance of our hardware design on fully simulated t-tbar events within the LHC CMS experiment. This sample serve as a benchmark for our developments and does not represent the most up-to-date CMS track trigger performance. With such events, the efficiency and the purity of Retina are both higher than 90%. We have also added a Kalman filter after the Retina fit to improve the resolution on the track parameters. Our preliminary results show that the Kalman filter can work well together with the Retina algorithm to find track through TTbar event and gives very high resolutions of the parameters reconstructed in the end.

Retina and Iterative Retian algorithm for curved track with strong magnetic field

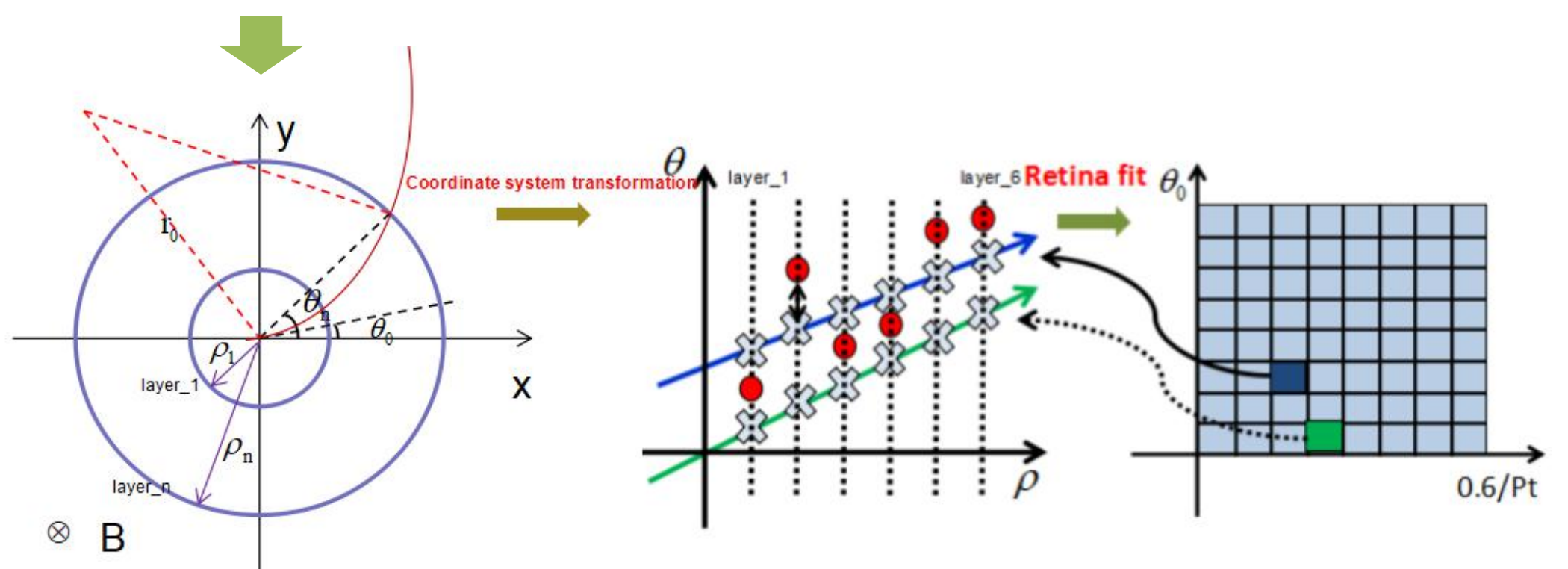
Principle of Retina for tack fitting :



Retina algorithm is inspired from the processing of visual images by the brain where each neuron is sensitive to a small region of the retina. The strength of each neuron is proportional to how close the actual image projected on the retina region is to the particular shape that particular neuron is tuned to [1][2].

The left plots show the tracker geometry and the Retina algorithm steps to fit curved tracks in a ideal barrel-shape tracker embedded in a uniform high magnetic field. In the magnetic field … bend). This ideal tracker is composed of n=6 concentric sensitive layers equally spaced. The layer radius ranges from r1=0,2 m to r6=1,15m. All the simulated particles start from the center of the circles, with a given initial angle θ$_0$, and cross the 6 layers from the center outwards. Retina inputs are the hit coordinates on each layer in the transverse x-y plane. Retina outputs possible track candidates in the (0,6/P_t, θ$_0$) parameter space.

Some highlights of Retina's use in this particular geometry: first we do an approximation by linearizing the particle trajectory equation in the transverse plane: Eq. 1 -> Eq. 2. Secondly we divide the parameter space (0.6/Pt, θ$_0$) into M*K cells(bins) and scan the full space to find the possible track cell location who's output value is over the threshold we set according to formula ⑥.



$\sin(\theta_n-\theta_0)=\frac{\rho_n}{2r_0}$ ① **assume** $r_0 >> \rho_n$:

$\sin(\theta_n-\theta_0)\approx\theta_n-\theta_0$

$\sin(\theta_n-\theta_0)=\frac{\rho_n}{2r_0}\Rightarrow\theta_n=\theta_0+\frac{\rho_n}{2r_0}$

r_0:radius of Trajectory

$P_T=0.3BQr_0\rightarrow r_0=\frac{P_T}{0.3BQ}$

Set B=4T, Q=+1 then:

$\theta_n=\frac{0.6}{P_T}\cdot\rho_n+\theta_0$ ②

approximation

Divide the entire parameter space into MxK parts

$$Sum(m,k)=\sum_{i=1}^{n}e^{-[\frac{(\theta\_scan_i(m,k)-\theta_i)^2}{2\sigma^2}]} \quad ③$$

$$\theta\_scan_i(m,k)=\frac{0.6}{P_T\_scan_k}\cdot\rho_i+\theta_0\_scan_n \quad (m=1....M,k=1....K) \quad ④$$

$$P_T\_scan_m=P_T\_\min+(P_T\_\max-P_T\_\min)\frac{m}{M},\ \theta_0\_scan_k=-\pi+2\pi\frac{k}{K} \quad ⑤$$

$$Retina\_output=Max[Sum(m,k)] \quad ⑥$$

Scanning & Fitting

Fig.1 3D detector model without M.F（magnetic field）

Iterative Retina:

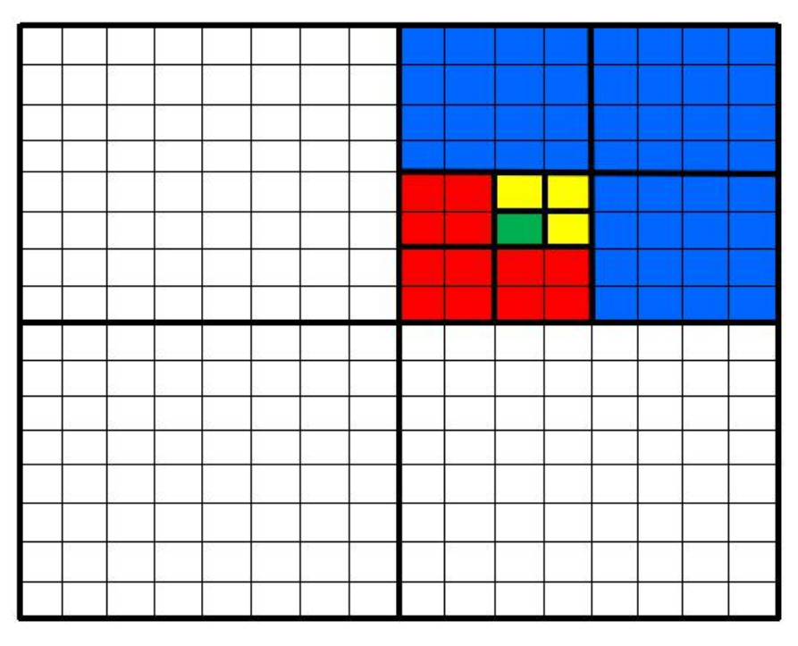
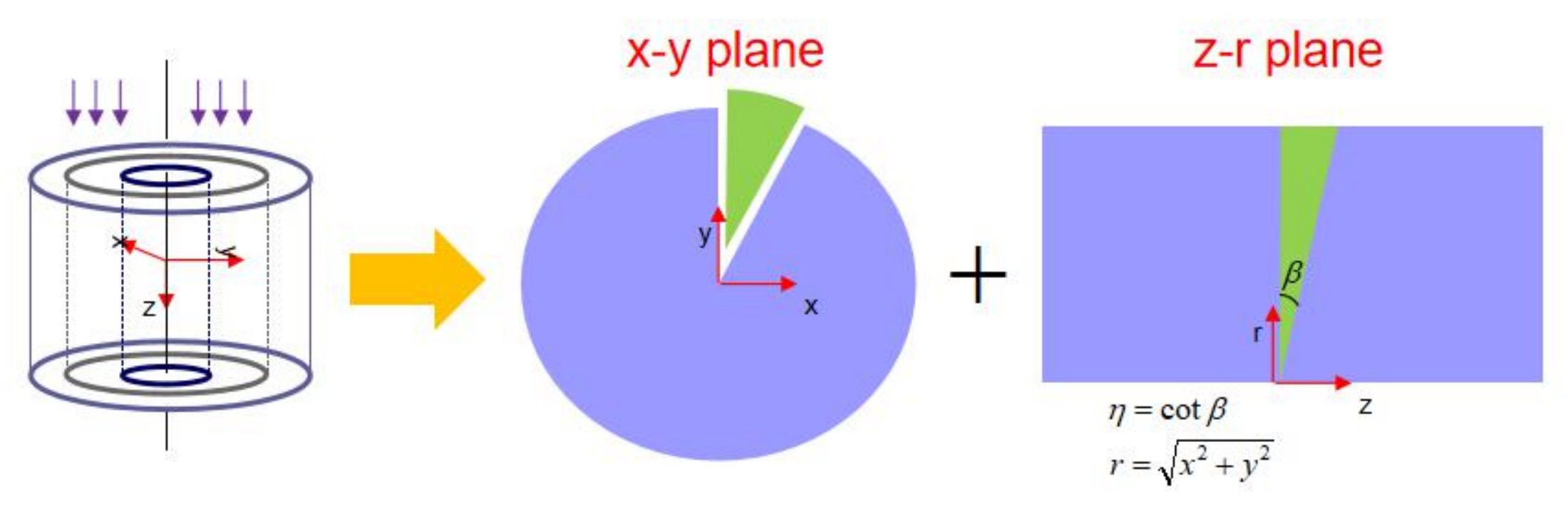


Fig.2 The graph of examples about how Iterative Retina works

To avoid to scan all the parameter space cells with the highest granularity, we iteratively run Retina increasing the parameter space granularity at each step but scanning a limited number of parameter cells around. Figure 2 shows an exemple where we run 4 times Retina on 2x2 cells to reach the equivalent precision of a 16x16 granularity in the same parameter space. The gain in processing is the following: for a granularity of the parameter space of n x n with traditional Retina, one would have to scan n^2 cells while with the iterative way one has to scan, with i iterations, i * m^2 cells, where n = m^i. This means decreasing the number of cells to scan from n^2=m^(2i) to i*m^2. At each step the region of interest is defined as the cell with the highest Retina output value.

Tracker segmentation & Iterative Retina configuration:



Divide the full tracker space into 180 sectors
(one sector show in graph with green region):
10 parts along θ_0 in x-y plane & 18 parts along η in z-r plane
For each sector one FPGA device is used to handle the track fitting & reconstruction

The parameter range for one sector we chose：

$\theta_0\subset(-\pi/20,3\pi/20)$ 1/10 of θ_0 parameter space (x-y plane)

$\eta\subset(0,0.2)$ 1/18 of the space along η (z-r plane)

Pt scan range: >=2GeV/c (1/Pt ~(0,1/2))

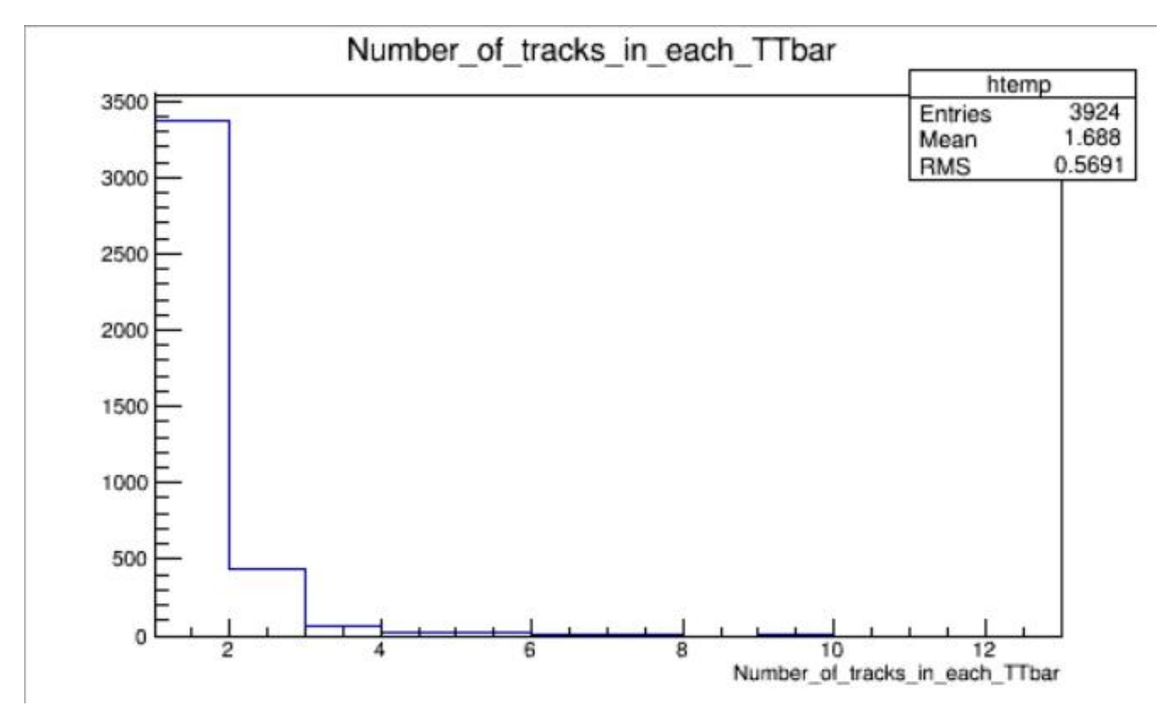
Three Iterative Retina granularities configuration：

Loop_48： 2 iterations of 6bins(1/Pt) * 8bins(θ_0) to reach 36bins * 64 bins

Loop_80： 2 iterations of 8bins(1/Pt) * 10bins(θ_0) to reach 64bins * 100 bins

Loop_200： 2 iterations of 10bins(1/Pt) * 20bins(θ_0) to reach 100bins * 400 bins

Data samples sources :
The full simulation TTbar event data from CMS Phase 2 outer tracker detector



Maxmium number of tracks indentified by iterative Retina from each TTbar event : 3 (limited by the scale of Firmware)

Firmware design of Iterative Retina

Structure Diagram of Iterative Retina in FPGA:

Top FW design block diagram of Iterative Retina :

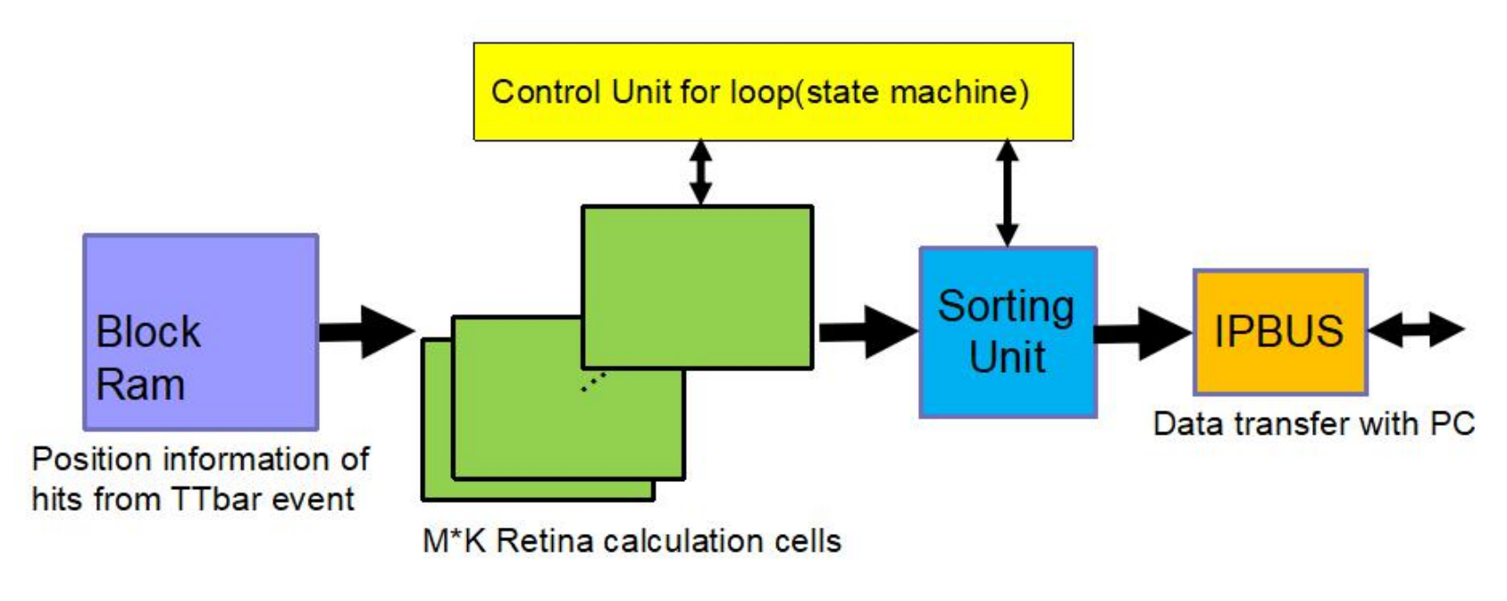
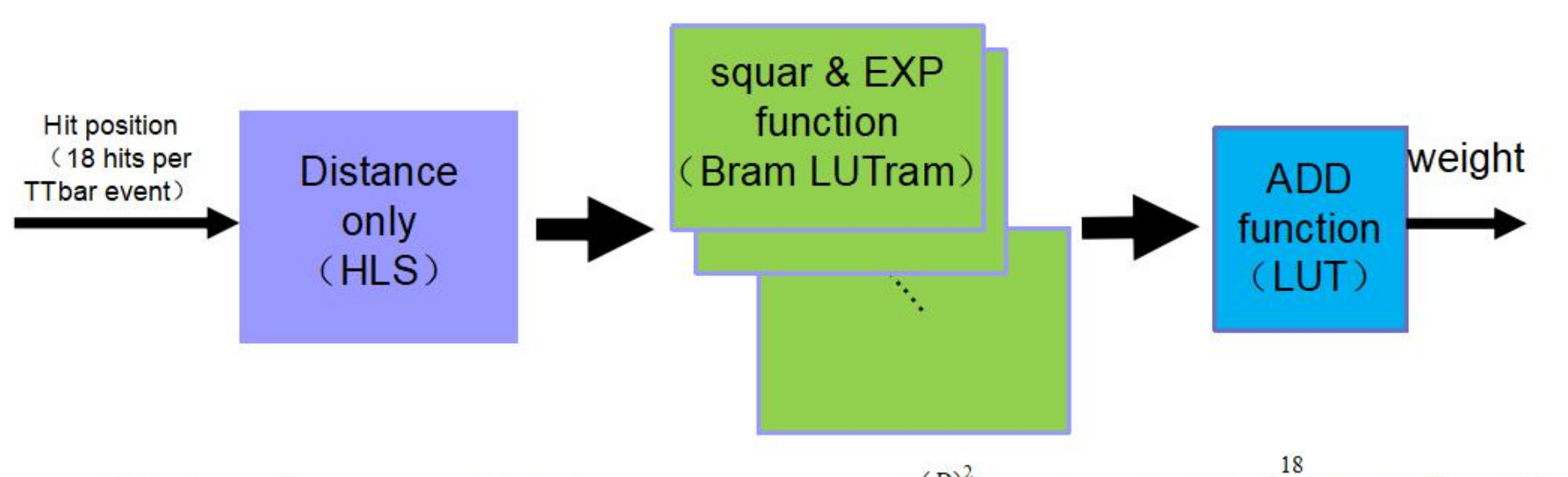

Control Unit for loop(state machine)
Block Ram
Position information of hits from TTbar event
M*K Retina calculation cells
Sorting Unit
IPBUS
Data transfer with PC

Figure left shows the block diagram of the Iterative Retina firmware, the structure of the firmware design in this case is similar to the original Retina algorithm except an additional Control Unit is added to control the loop flow of the Iterative Retina algorithm. The latency and resource usage of the Iterative Retina are shown in Table below. To process one event, it takes 312 clock cycles at the clock frequency of 200 MHz and it costs nearly half of the KC705 FPGA resources to compute 200 cells with Retina.

Block diagram of one Retina calculation cell:


Hit position（18 hits per TTbar event）
Distance only（HLS）
squar & EXP function（Bram LUTram）
ADD function（LUT）
weight

$D(m,k)=\theta\_scan_i(m,k)-\theta_i$ $Exp(D)=e^{-[\frac{(D)^2}{2\sigma^2}]}$ $Sum(m,k)=\sum_{i=1}^{18}Exp(D(m,k))_i$

Function of calculation cells:

$$Sum(m,k)=\sum_{i=1}^{18}Exp(D(m,k))=\sum_{i=1}^{18}e^{-[\frac{(\theta\_scan_i(m,k)-\theta_i)^2}{2\sigma^2}]}$$

Resource usage & latency :

Clock (Hz)	Firmware Design	DSP	LUT	BRAM	FF	Latency (Cycles) /µs
------------	KC705	840	203800	445	407600	--------------
200M	One calculation cell	1	441	0	517	--------------
200M	LOOP_200	200 (23.81%)	112326 (55.2%)	0.5 (0.11%)	58289 (14.3%)	312/1.56

Frimawre testing conditions:
Platform: KC705 evaluation board
Number of cells : 200

Result analysis

Efficiency and Purity of iterative retina:

The table below shows the efficiency and the purity of iterative Retina on ttbar events. As shown in the table the efficiency and purity in these three cases are very closly and both of them are over 90%. So with the iterative retina most of the track in ttbar event can be identified even in case of the scanning granularity is reduced.

Case	Ghost track	Mismatched track	Matched track	Duplicated tracks	Total tracks in TTbar event	Efficiency	Purity
Loop-48	460	125	1305	2362	3792	96.71	88.85
Loop-80	167	128	2312	1352	3792	96.62	95.64
Loop-200	40	238	2825	729	3792	94.78	98.89

Ghost track: The track reconstrcuted from Multiple source particles
Duplicated tracks: Multiple tracks reconstrcuted from one source particle
Matched track: The one and the only track reconstrcuted from one source particle
Mismatched track: The track not found by Iterative Retina

Adding Kalman filter as track fitter:

$T_{1\_initail}=[1/2R_{1\_Retina},\theta_{1\_Retina},1,0]$
Hits data of Track1

$T_{2\_initail}=[1/2R_{2\_Retina},\theta_{2\_Retina},1,0]$
Hits data of Track2

tracks found by Interative Retina $R_0=0.57*q/Pt$

Kalman filter

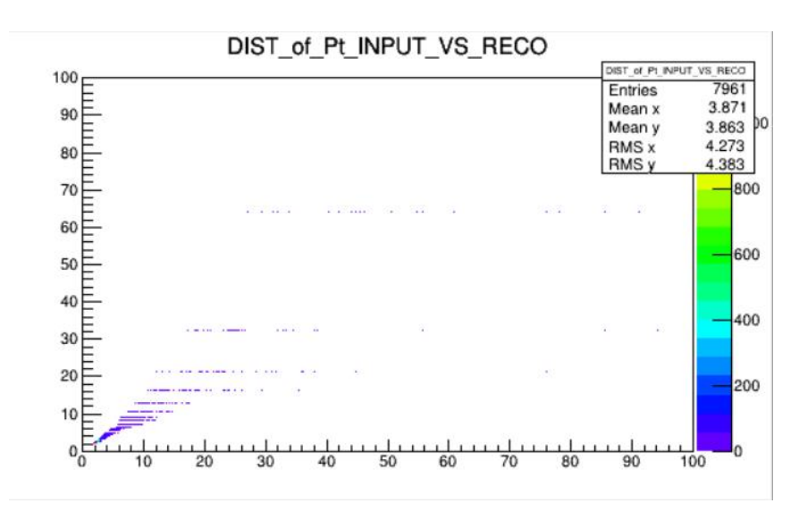
$T_1=[1/2R_1,\theta_1,\cot\beta_1,z_1]$

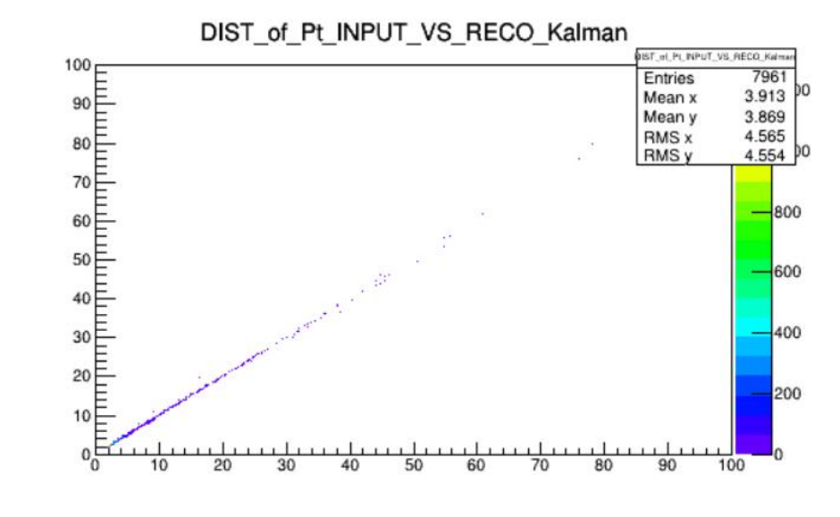
$T_2=[1/2R_2,\theta_2,\cot\beta_2,z_2]$

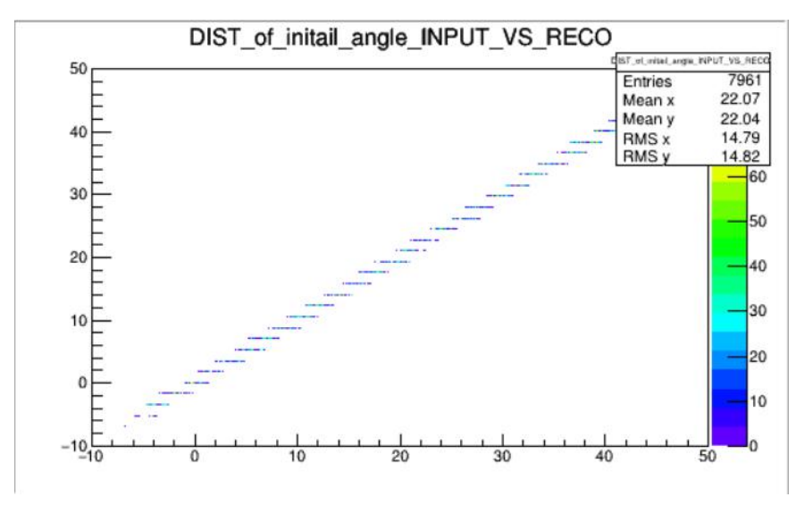
Parameters Reco by kalman filter

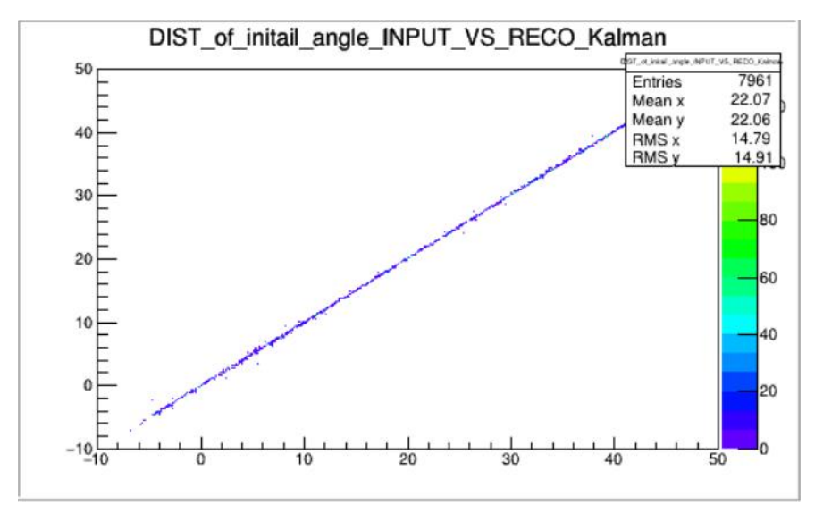
To improve further the parameters of track candidates identified by iterative Retina, we have added a Kalman filter (KF) afterwards. The initia KF track parameters are those returned by iterative Retina. KF then iterates on the hit coordinates associated to the Retina track candidate from the interaction point outwards. The results presented here are from simulations. We are currently developing the KF in firmware.

Pt & θ_0 reconstruction resolution:



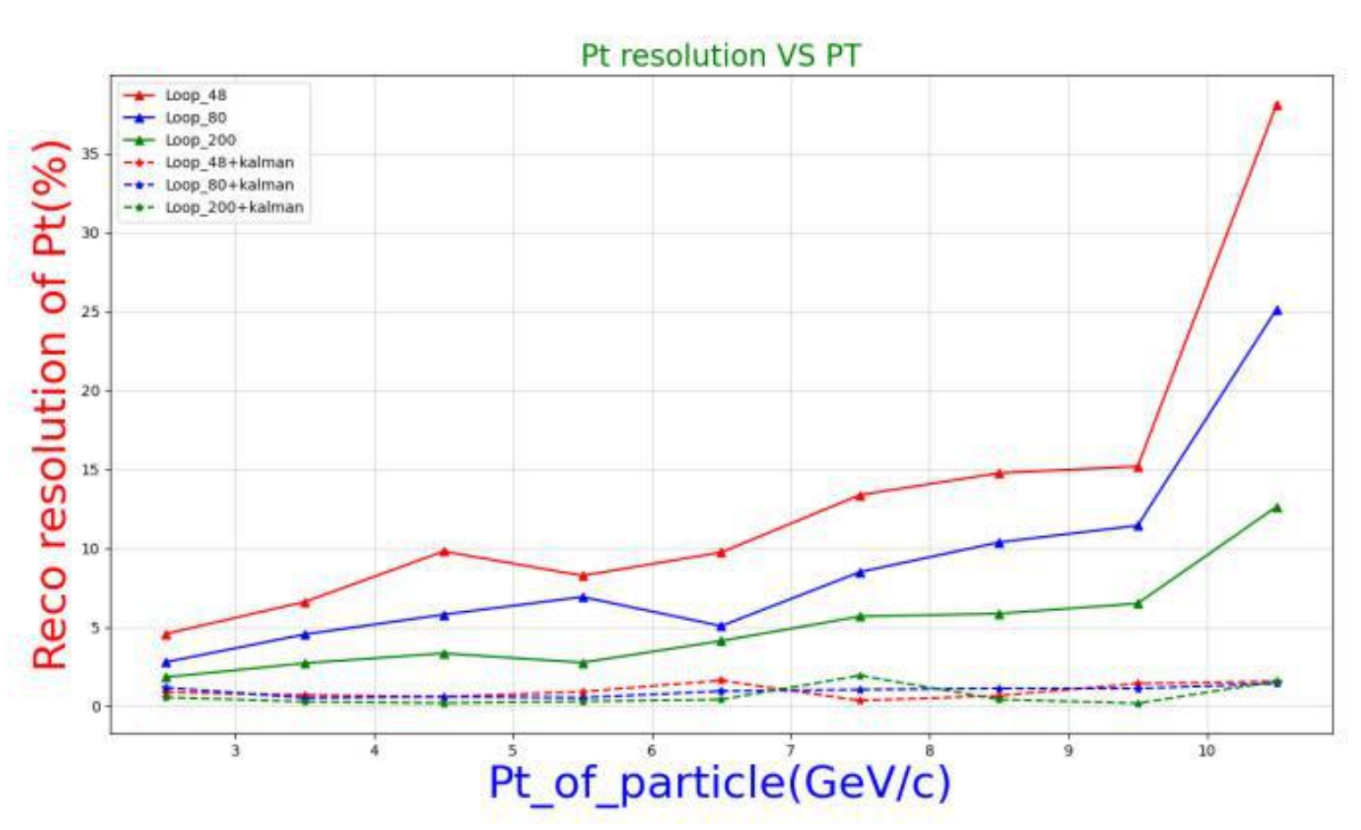


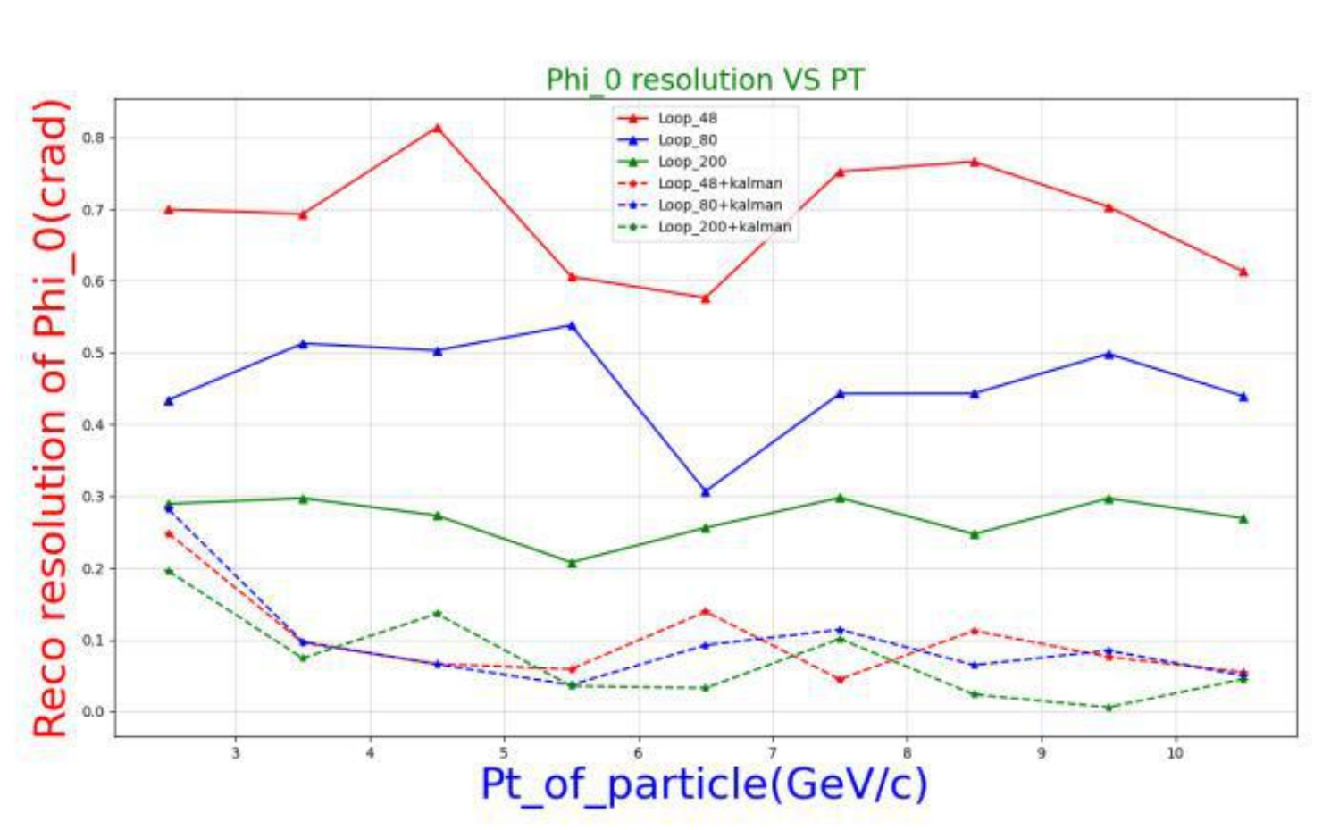




The 4 pictures on the left show the reconstructed Pt or θ_0 (y-axis) versus the simulated value (x-axis) in the case of Loop-48 scenario, with and without KF. The 2 bottom pictures show the Pt and θ_0 resolution versus the simulated Pt. 6 scenarios are shown: the 3 granularity scenarios with and without KF.

The preliminary conclusions are that the resolution improves with Retina granularity, as expected, and that the KF significantly improves it further, even in the case of low Retina granularity, which seems to indicate that we can relax Retina granularity which means reducing the FPGA resource usage or the processing time of the Retina step. Optimization is still ongoing.





References

[1] L.Ristori, "An artificial retina for fast track finding," Nucl. Instrum. Meth. A 453 (2000) 425.
[2] A.abba et al., "The artificial retina processor for track reconstruction at the LHC crossing rate," JINST 10, C03018 (2014), [arXiv: 1409.1565].
[3] Z.Song et al., Study of hardware implementation of fast tracking algorithms, 2017 JINST 12 C02068.

1. Zixuan.Song@ulb.ac.be 2. dengdandan@mails.ccnu.edu.cn