Progress on the online tracking with STT

Yutie Liang, Martin Galuska, Jifeng Hu, Wolfgang Kühn,
Jens Sören Lange, David Münchow, Björn Spruck, Hao Xu

II. Physikalisches Institut, JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN

Sep. 11 2012

Outline

1. Introduction
   Straw Tube, Conformal transformation, Hough transformation

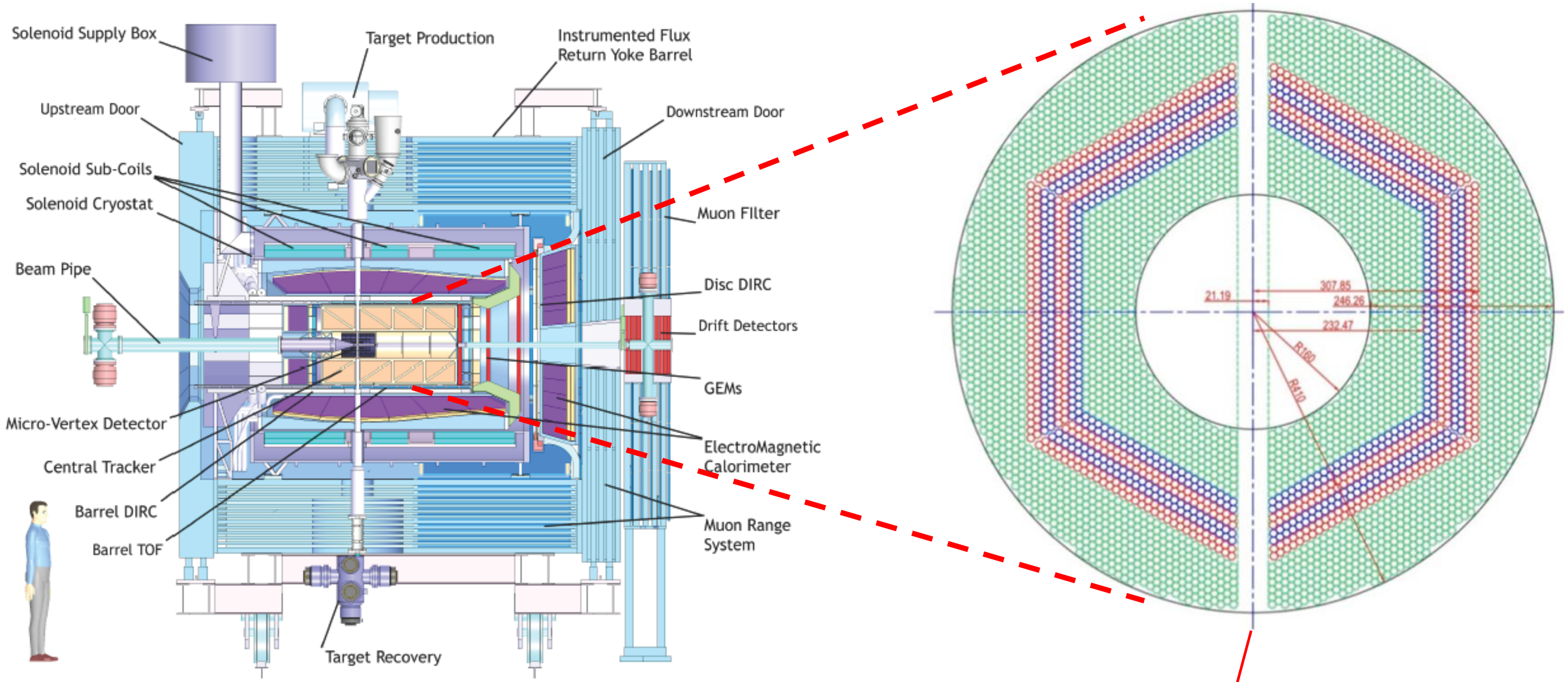
2. Progress on the tracking algorithm

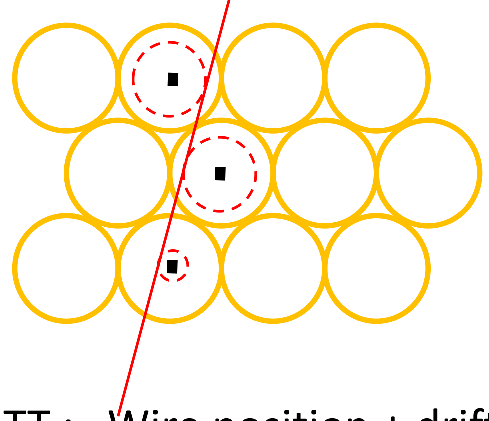
3. Progress on the VHDL implementation

4. Preliminary study of event start time

5. Summary and outlook

Straw Tube Tracker(STT)





- ➢4636 Straw tubes
- ➢23-27 planar layers
- 15-19 axial layers(green) in beam direction
- 4 stereo double-layers for 3D reconstruction, with ±2.89 skew angle(blue/red)

From STT : Wire position + drift time

Conformal transformation and Hough transformation

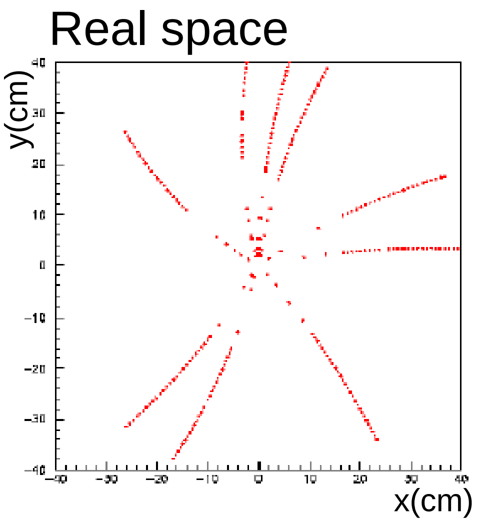
Conformal transformation:

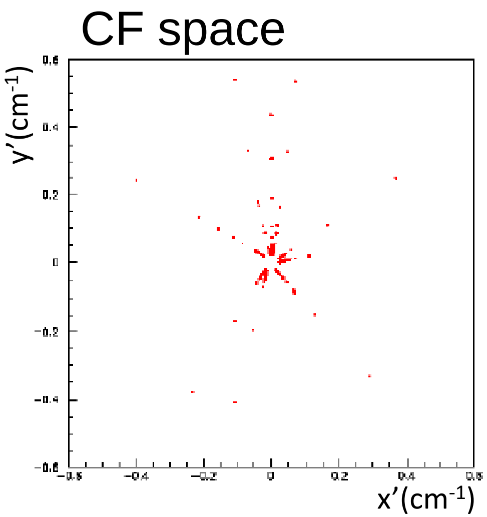
Transform circles to straight lines

$$x' = \frac{x - x_0}{r^2}$$

$$y' = \frac{y - y_0}{r^2}$$

$$r^2 = (x - x_0)^2 + (y - y_0)^2$$


Real space

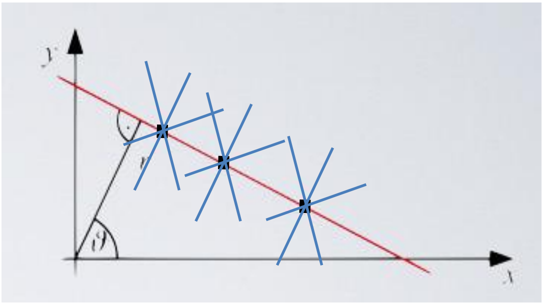

CF space

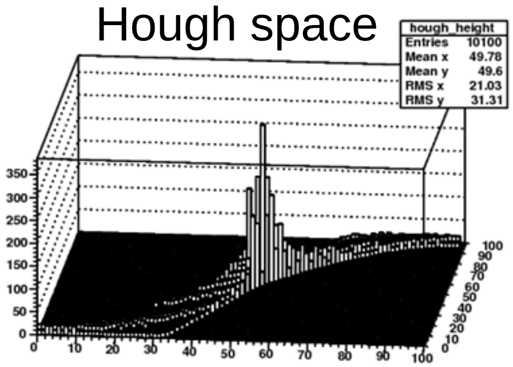
Hough transformation: Describing points in real space by parameters

For lines: y = mx + b (x1, y1), (x2, y2),…→ (m,b) or (r, θ)

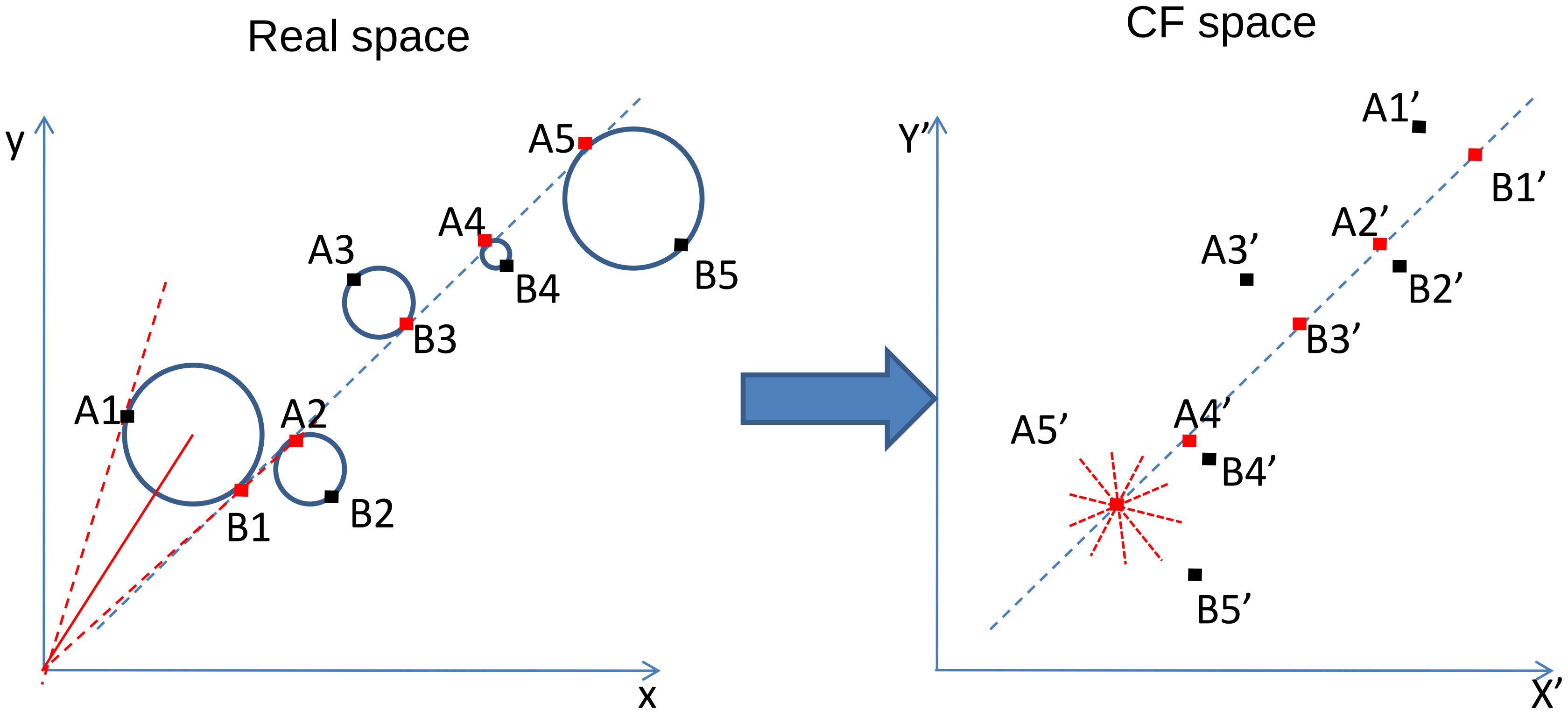
R= x cos(θ) + y sin(θ)

- ➢Use all possible angles
- ➢Save data in histogram
- ➢Peaks in histogram represent possible lines in point set




Hough space

Previous method of Pt reconstruction with STT



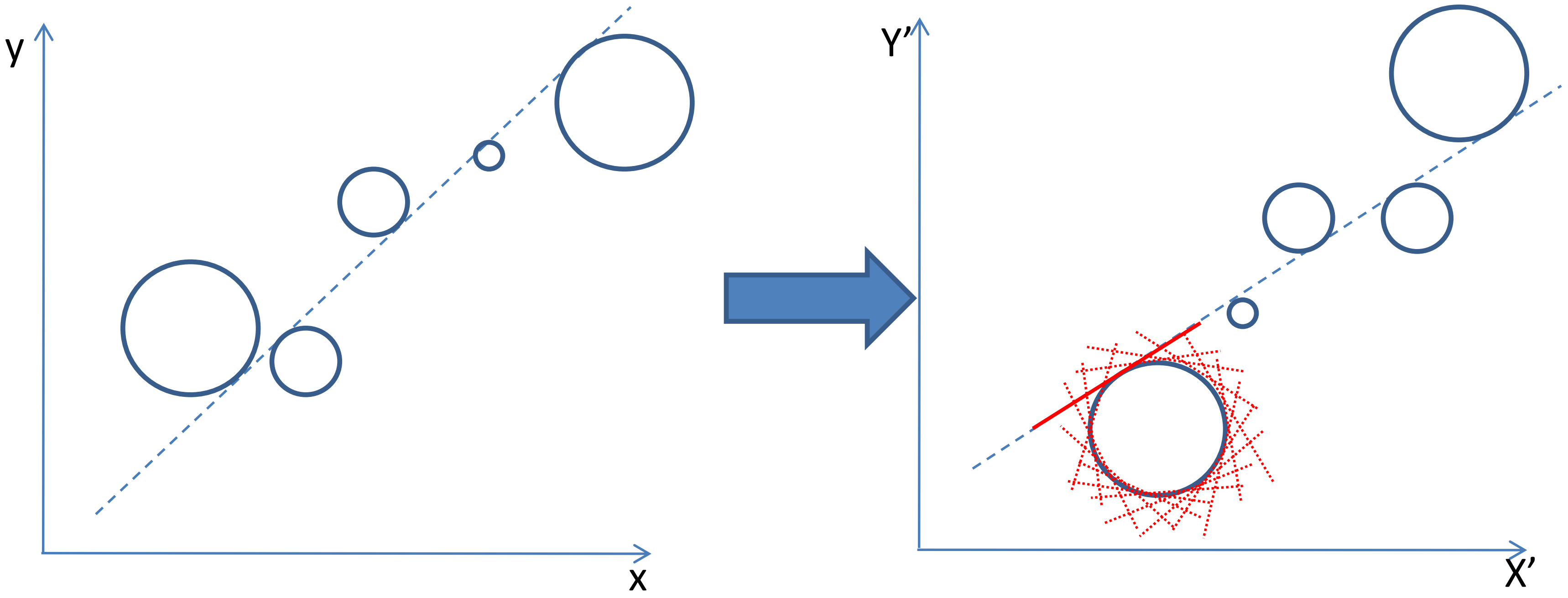
Real space

CF space

- Pick up two points around the drift circle.
- Transform these two points to CF space.
- Draw lines around each point in CF space.
- Fill the line parameter into the Hough space.

Problem:
Bad for low momentum.

New method of Pt reconstruction with STT



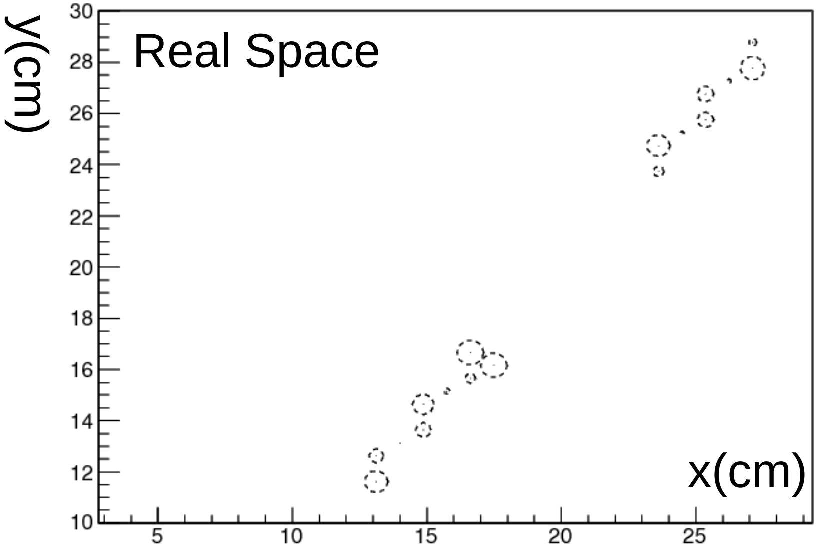
Real space

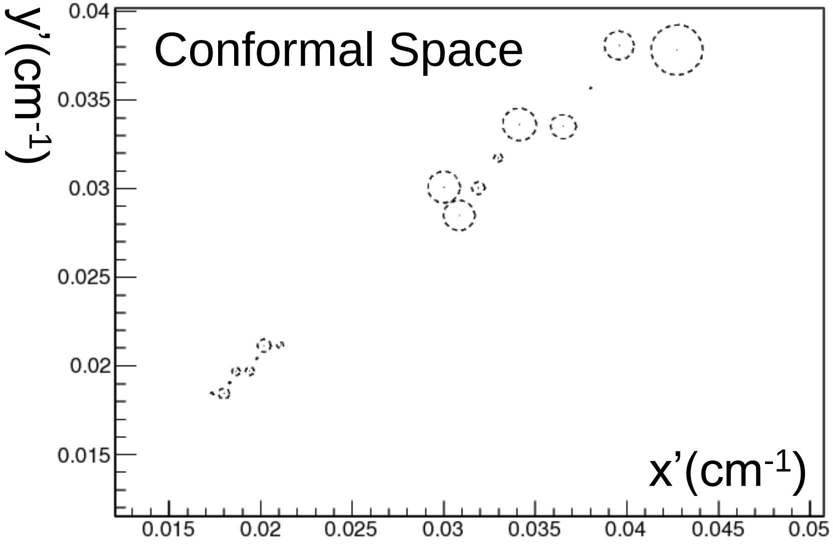
CF space

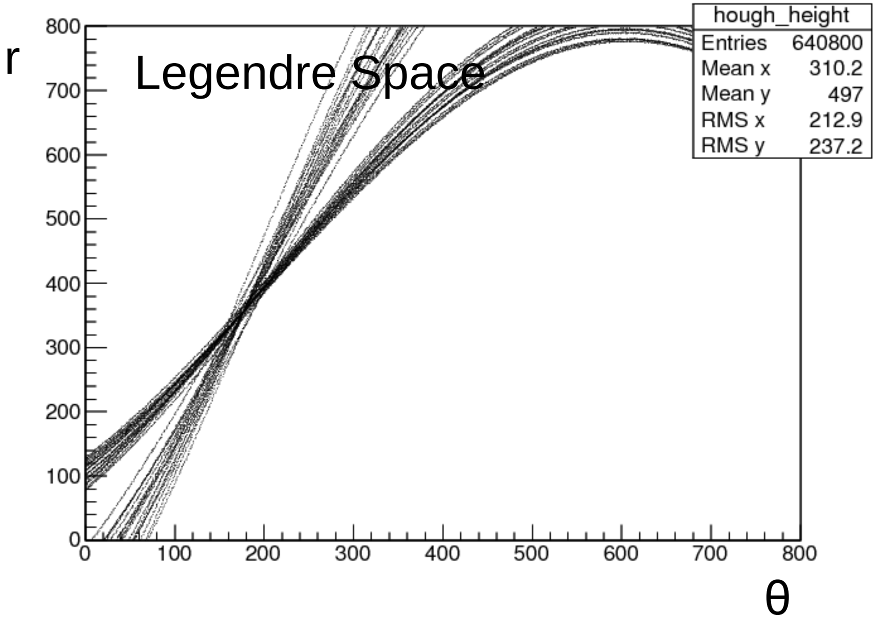
- Transform the drift circle to CF space.
- Draw lines around the “circle” in CF space.
- Fill the line parameter into the Hough space.

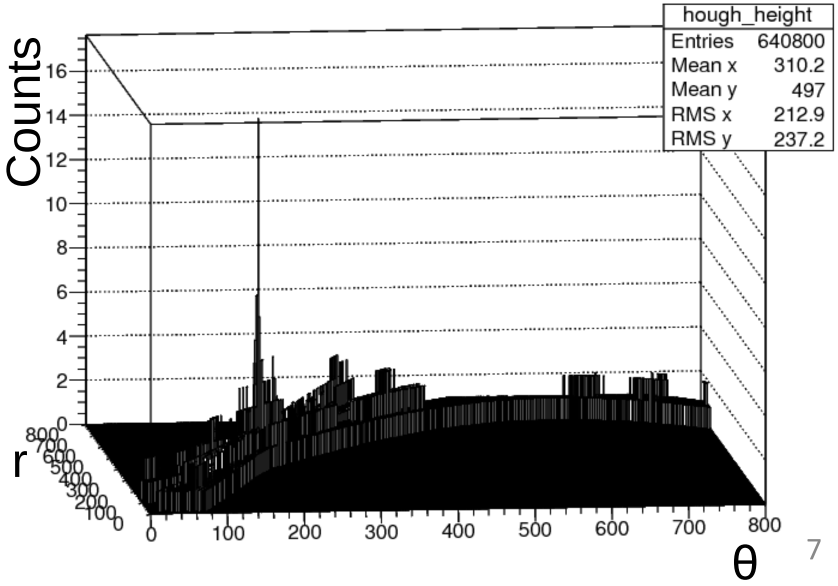
Legendre transformation

A look at one example event

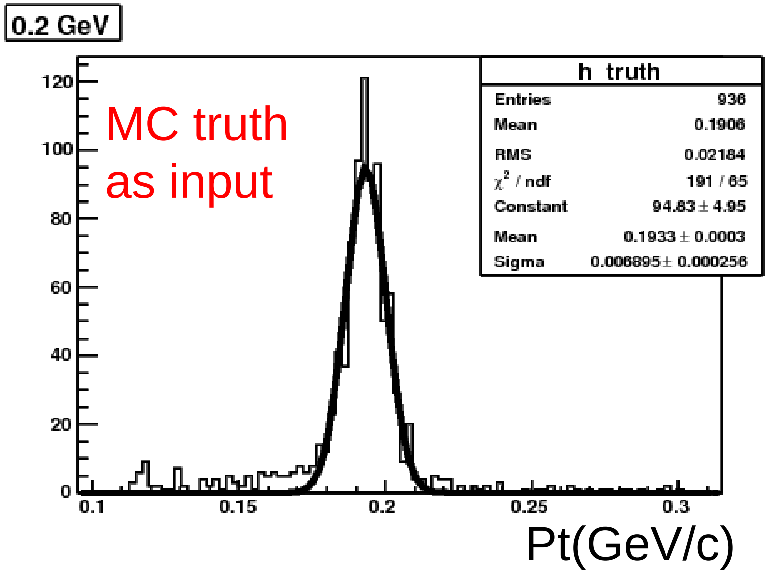




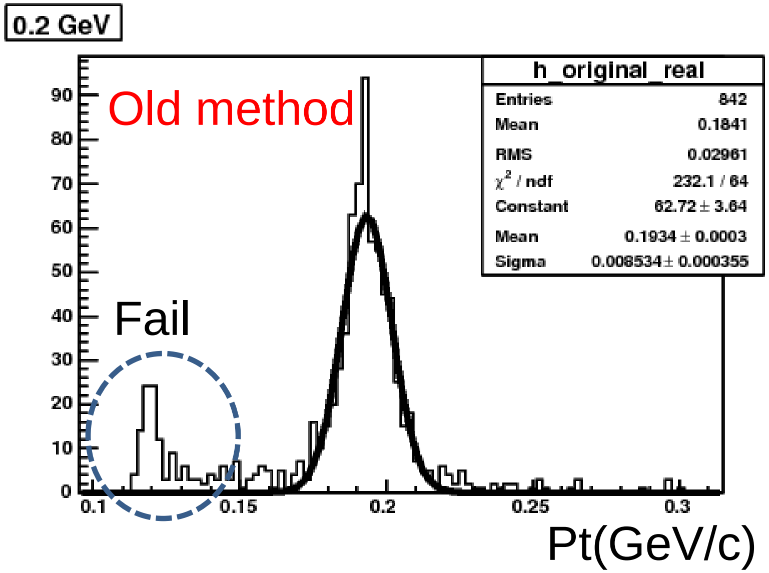




Momentum resolution

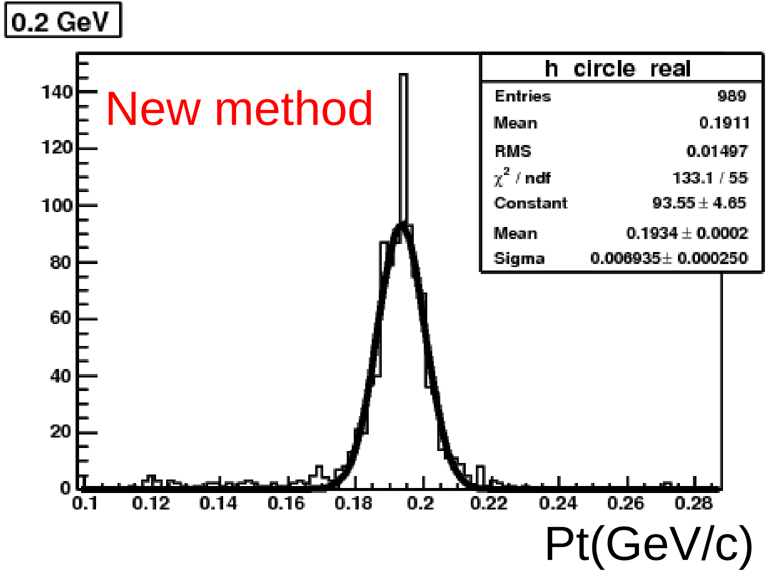


MC truth as input



Old method

Fail



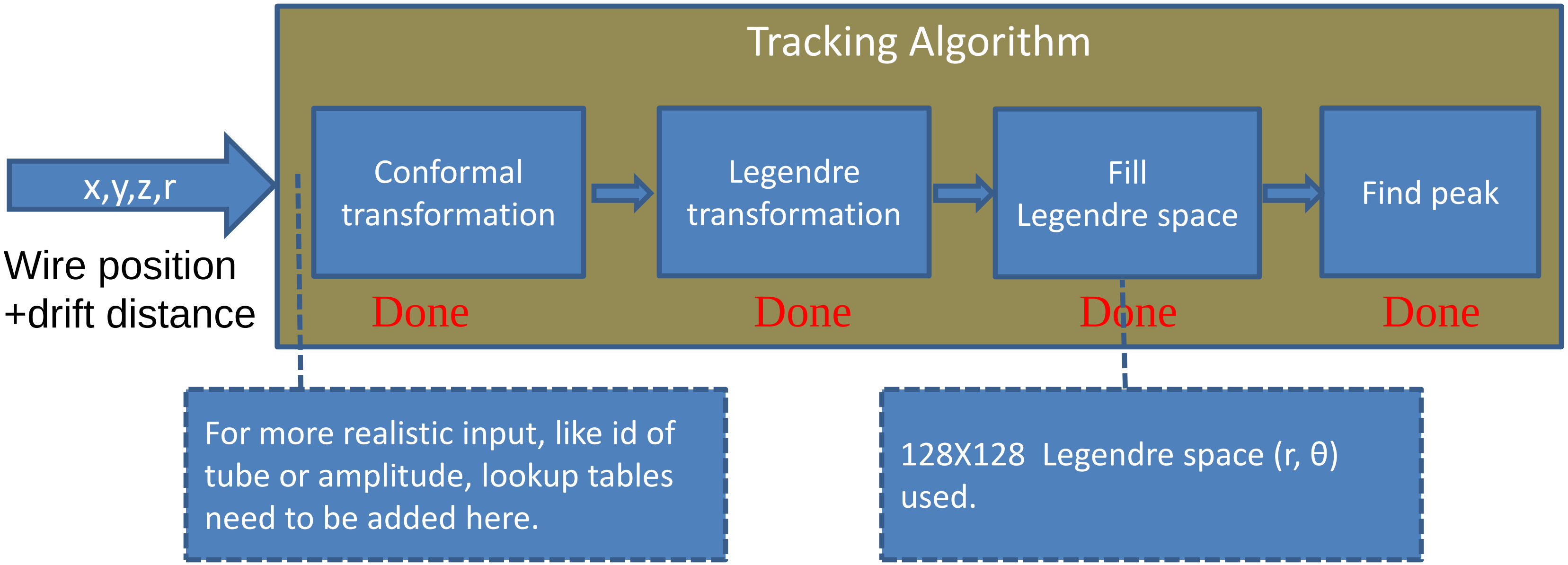
New method

σ:

	0.2 GeV/c	1 GeV/c
Truth:	3.45%	3.685%
Old:	4.27%	3.732%
New:	3.47%	3.694%

New method improves at low Pt.

Status of the tracking algorithm in VHDL



Tracking Algorithm

x,y,z,r

Wire position +drift distance

Conformal transformation

Done

Legendre transformation

Done

Fill Legendre space

Done

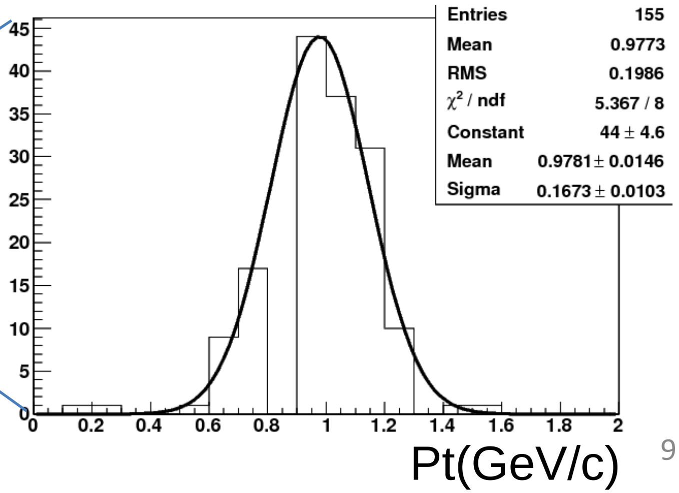
Find peak

Done

For more realistic input, like id of tube or amplitude, lookup tables need to be added here.

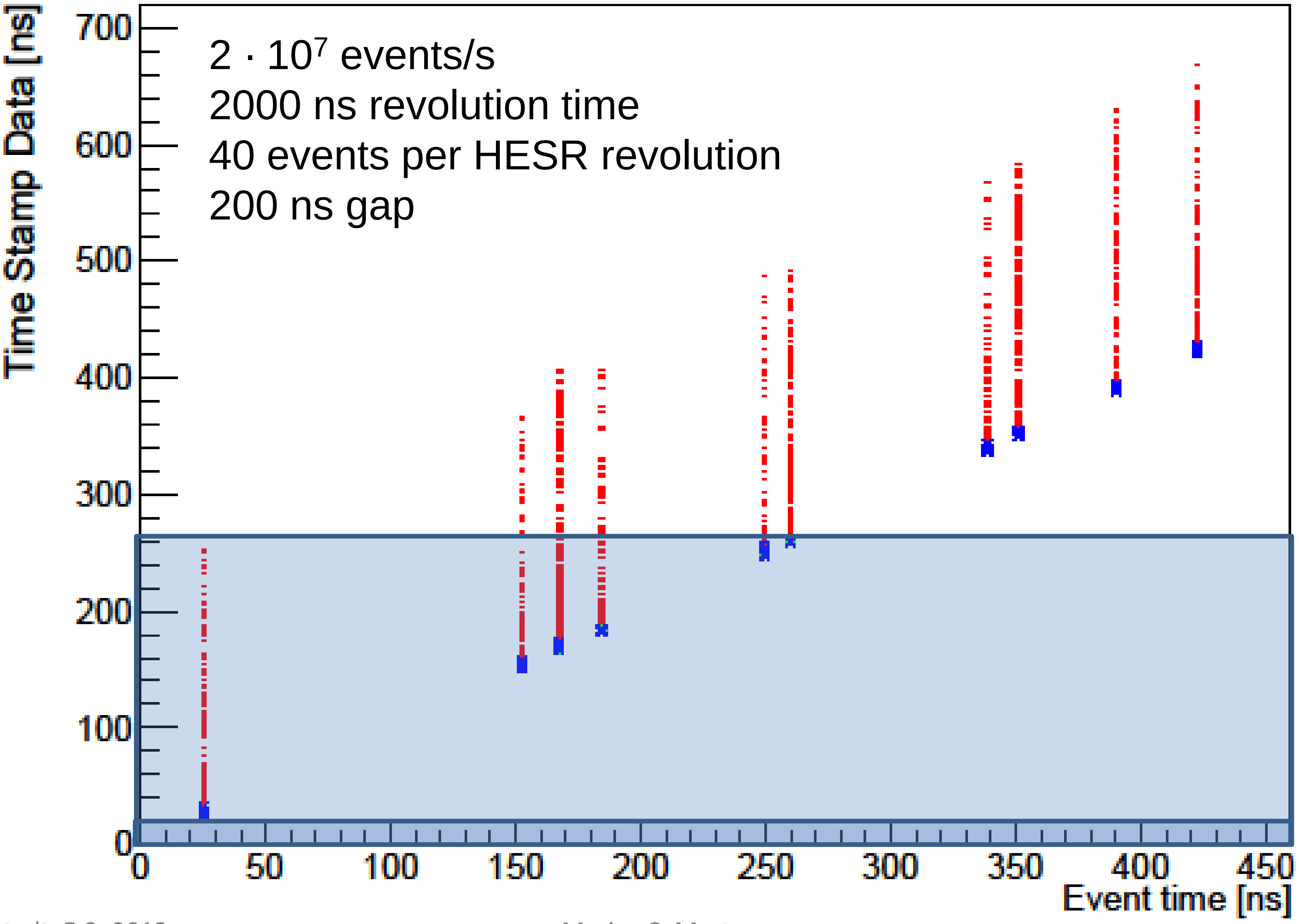
128X128 Legendre space (r, θ) used.

Simulation with ISim



Event Structure

Marius talk



$2 \cdot 10^7$ events/s

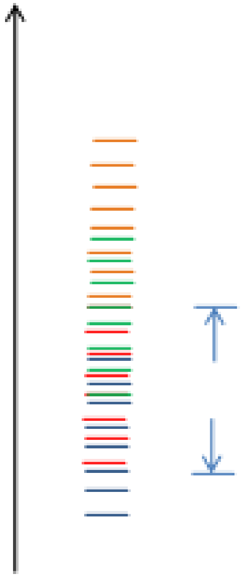
2000 ns revolution time

40 events per HESR revolution

200 ns gap

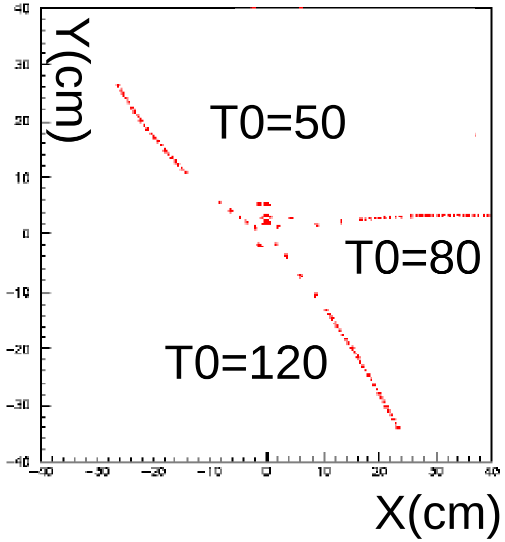
Time based simulation by T. Stockmanns

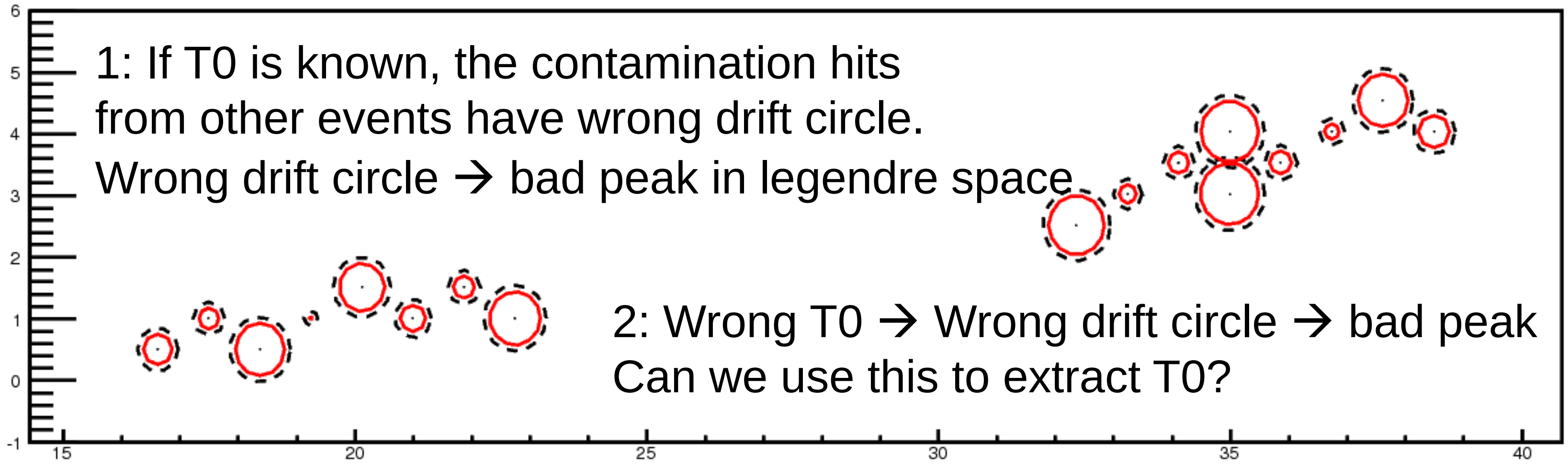
Can we extract the event start time?



- Need fast detector to provide T0.
- With T0, are the overlapped hits problem?

T0 + 220ns window size





1: If T0 is known, the contamination hits from other events have wrong drift circle.
Wrong drift circle → bad peak in legendre space

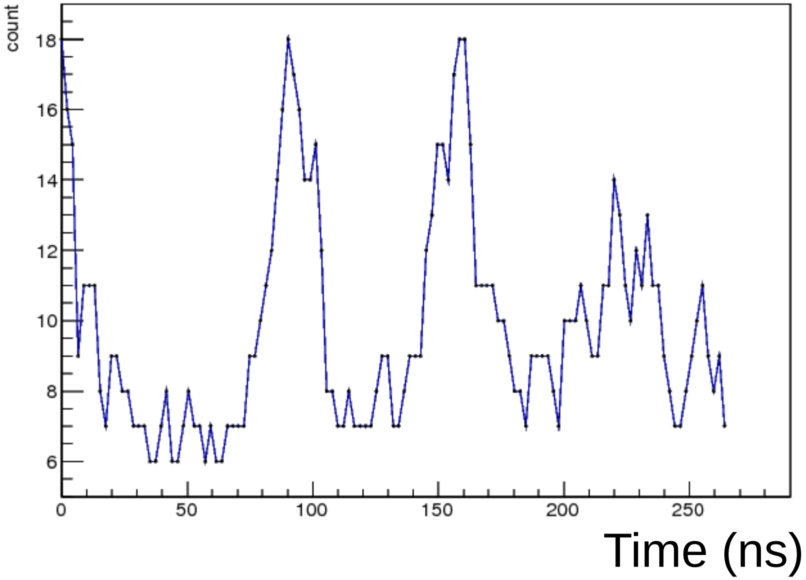
2: Wrong T0 → Wrong drift circle → bad peak
Can we use this to extract T0?

T0 extraction

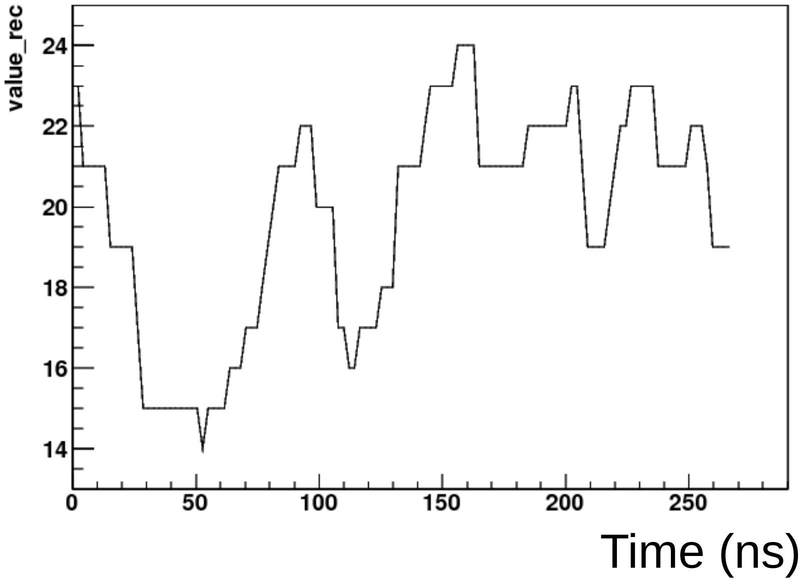
Time based simulation:
1) 4 events (single track) in one burst.
2) T0 (0, 90,160, 220) ns

T0 scan with tracking algorithm:
Step size: 2ns

C++
Hough space: 800 X 800



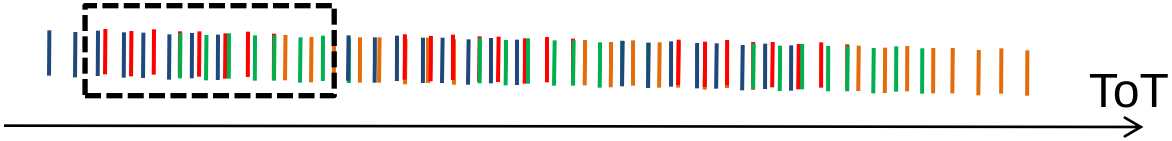
VHDL with ISim
Hough space: 128 X 128



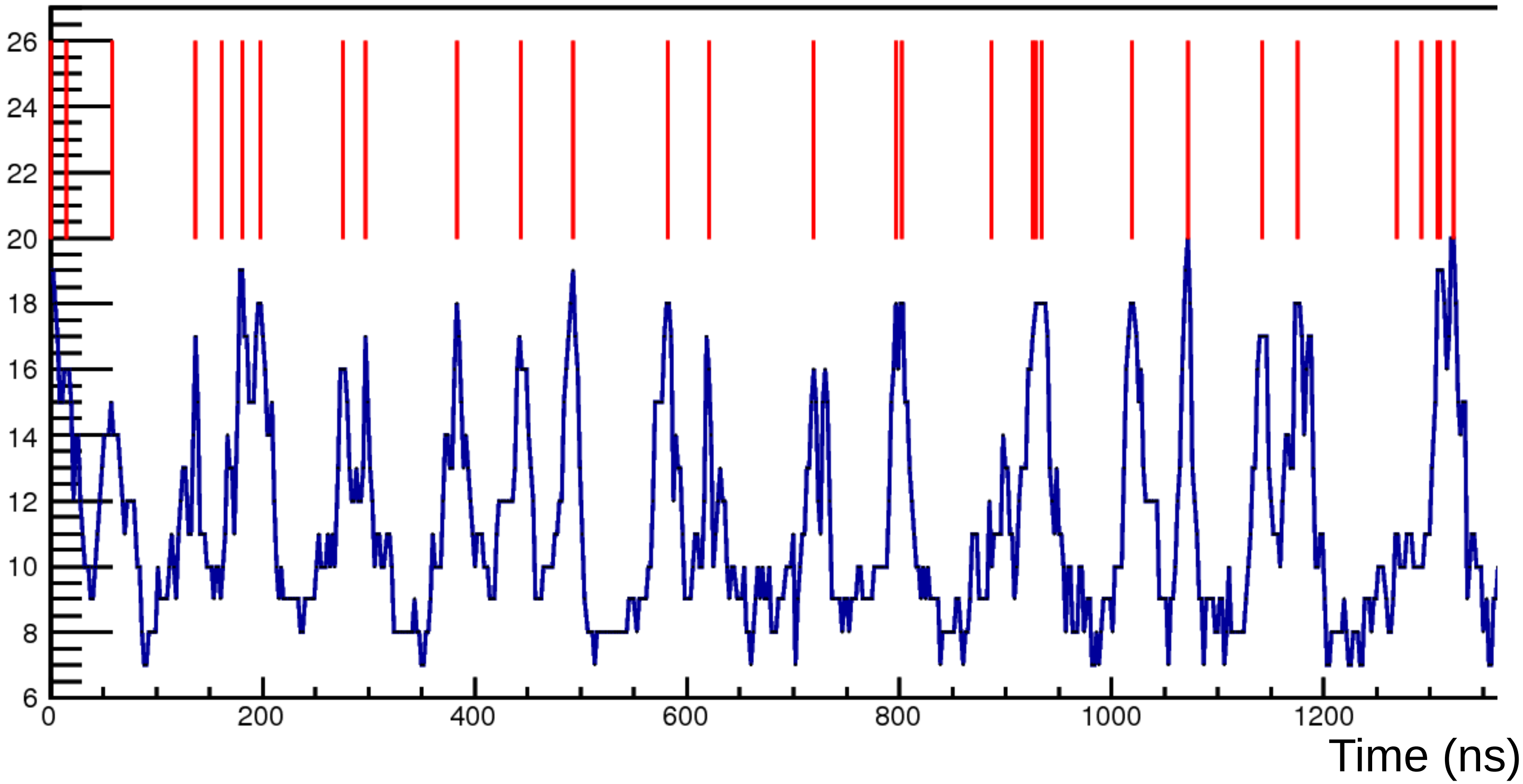
T0 extraction — for large burst

If large burst:

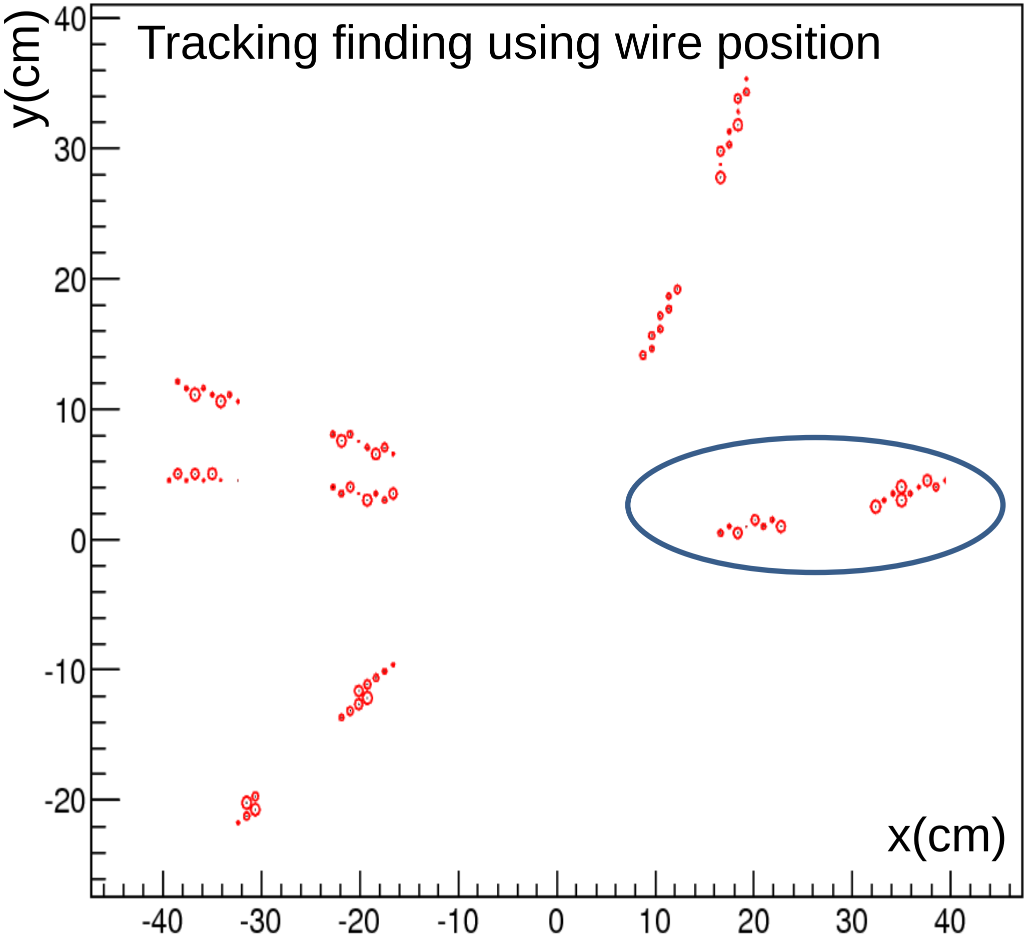
T0 scan + 220ns window size



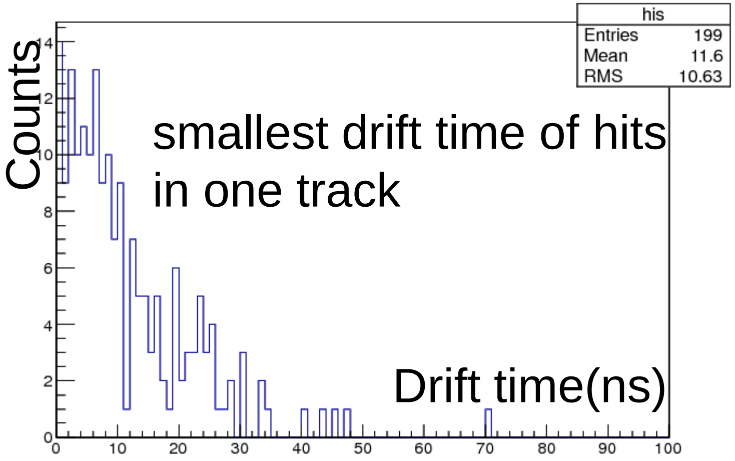
Time based simulation of one burst with 50 events



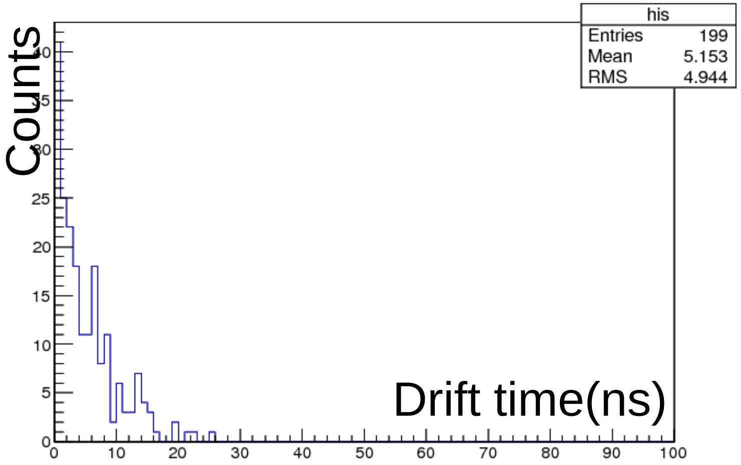
T0 extraction — another idea



Tracking finding using wire position



smallest drift time of hits in one track

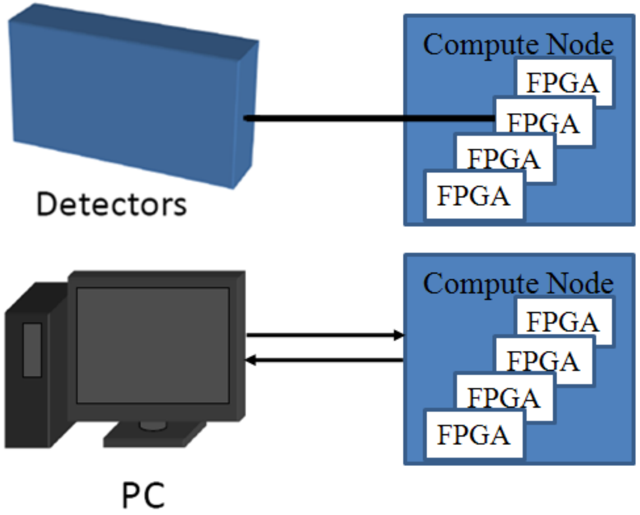


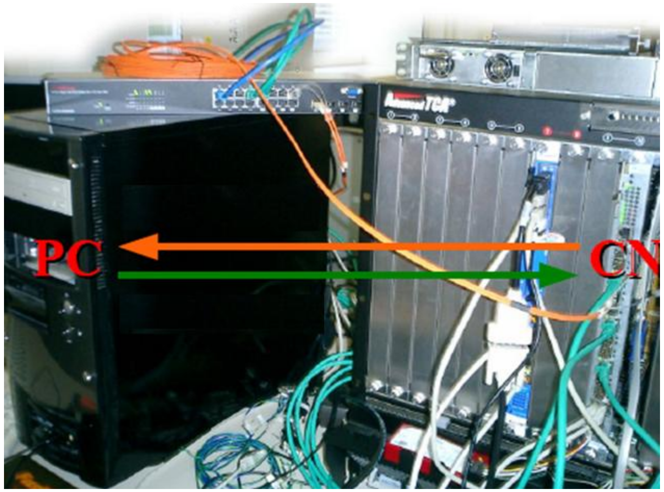
Not possible for too many hits in one burst!

Setup and test

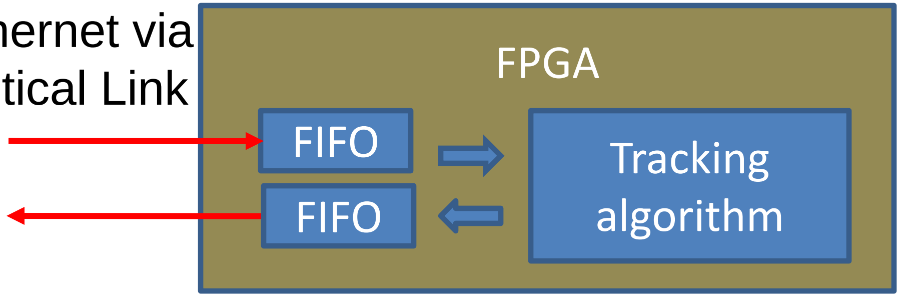
PC as data source and receiver.

- Ethernet.
- Optical link (UDP by Grzegorz Korcyl) (not integrated yet)





Ethernet via Optical Link



Summary

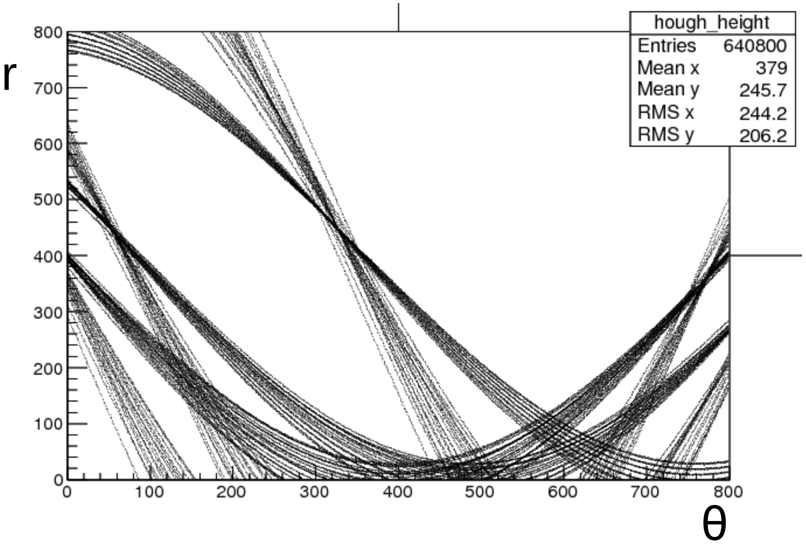
- New method studied. Low momentum reconstruction improved.
  4.27% (old method) → 3.47% (new method) @ 0.2GeV/c
- First version VHDL implementation is finished. A Legendre space of 128X128 is used.
- T0 extraction using tracking algorithm is studied. “T0 scan” is feasible to extract T0. The problem is the dramatically increased computing time. (25 times longer if using 2 ns step size.)

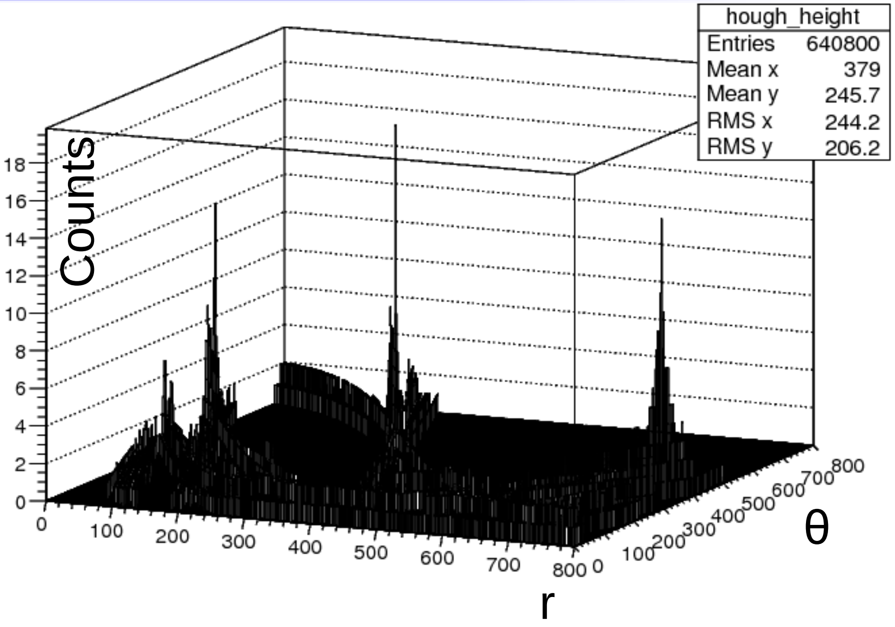
Next to do:

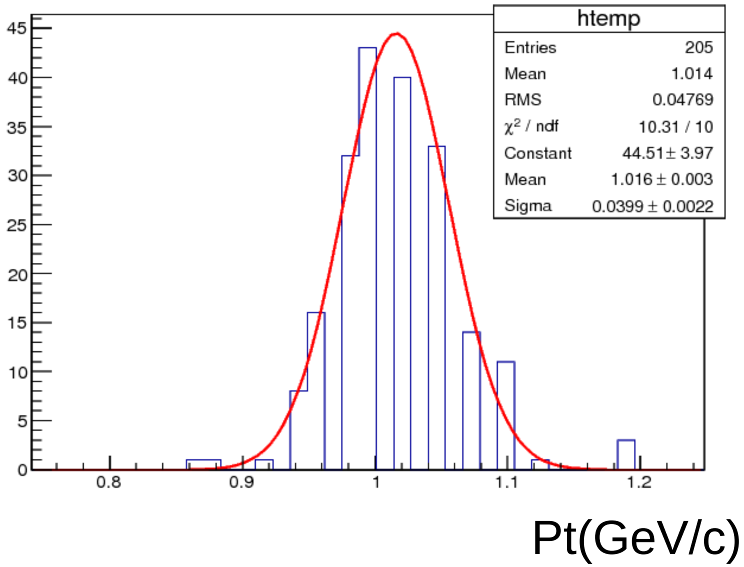
- Combine MVD information
- Optimize the algorithm.

Thank you

Multiple tracks in one event







- Momentum resolution.
- Reconstruction efficiency.
- Fake track.

If T0 is known:
For each T0, all hits within the 220ns window size need to be considered.
Assume, we have 100 hits in this window, 2 clock-cycle per hit.

100 hits * 2 = 200 clock-cycle   per event

If T0 is unknown:
We need a T0 scan.
Assume a step size of 2ns, average time between two events of 50ns.

(200 clock-cycle   per event) * 50ns/2ns   = 5000 clock-cycle per event

Conformal transformation and Hough transformation

Transform circles to straight lines

$$x' = \frac{x - x_0}{r^2}$$

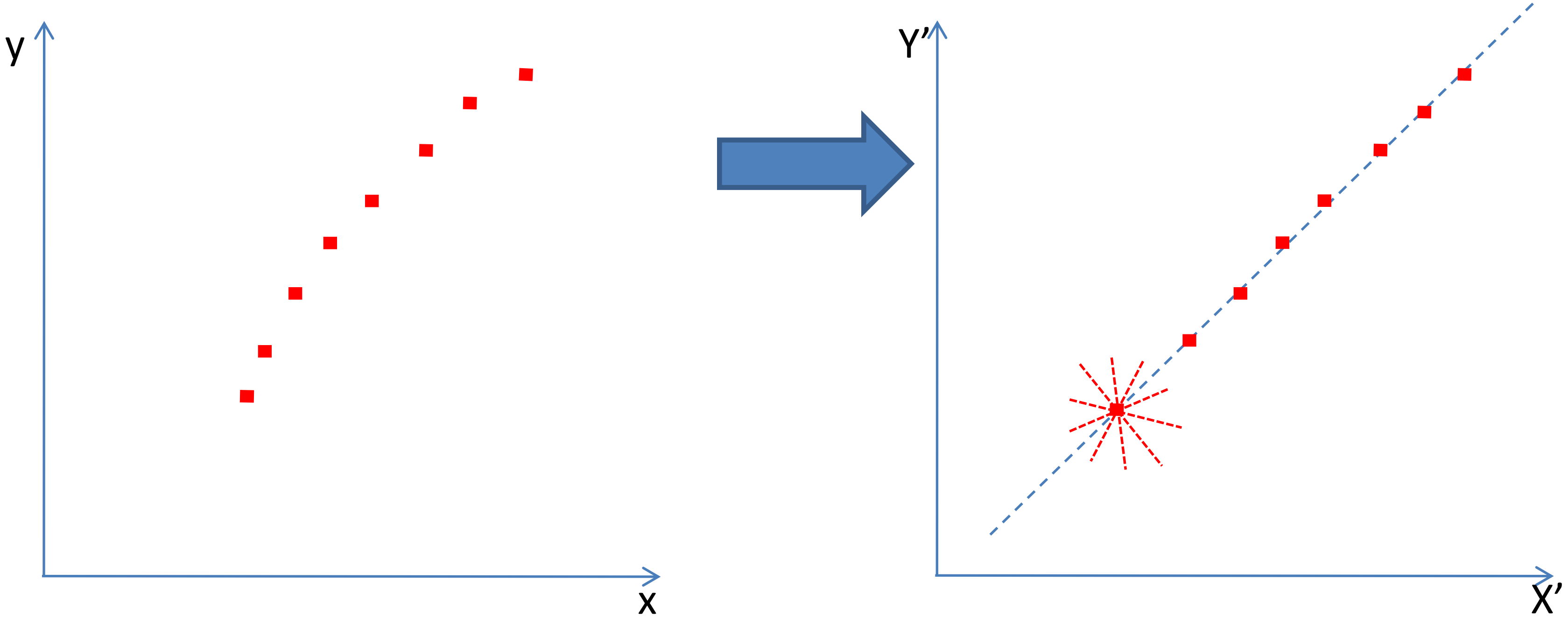
$$y' = \frac{y - y_0}{r^2}$$

$$r^2 = (x - x_0)^2 + (y - y_0)^2$$

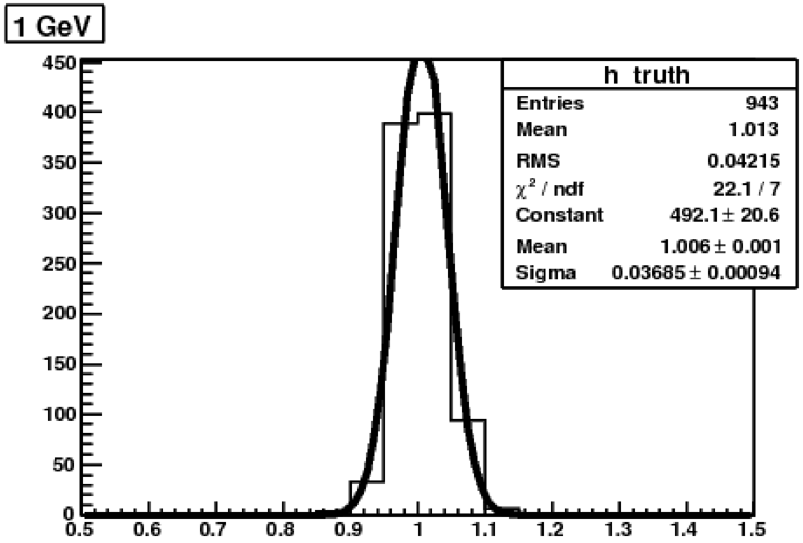
Describing lines by parameters

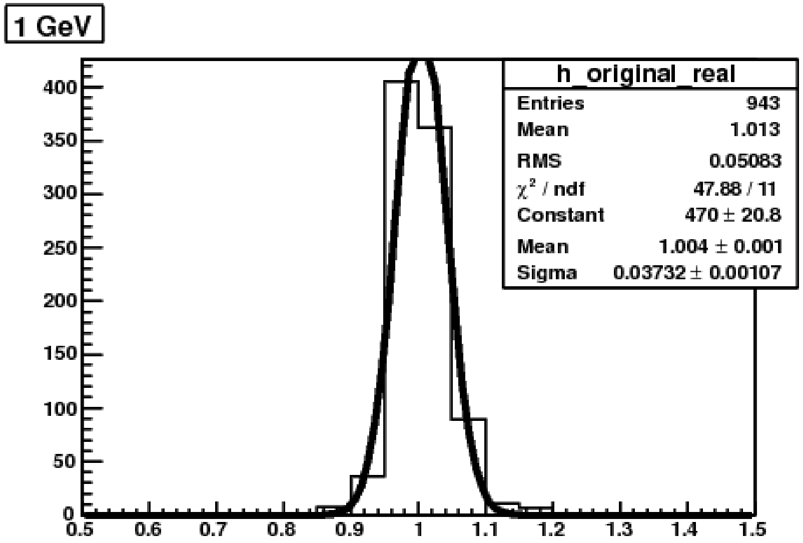
y = mx + b → (m,b) or (r, θ)

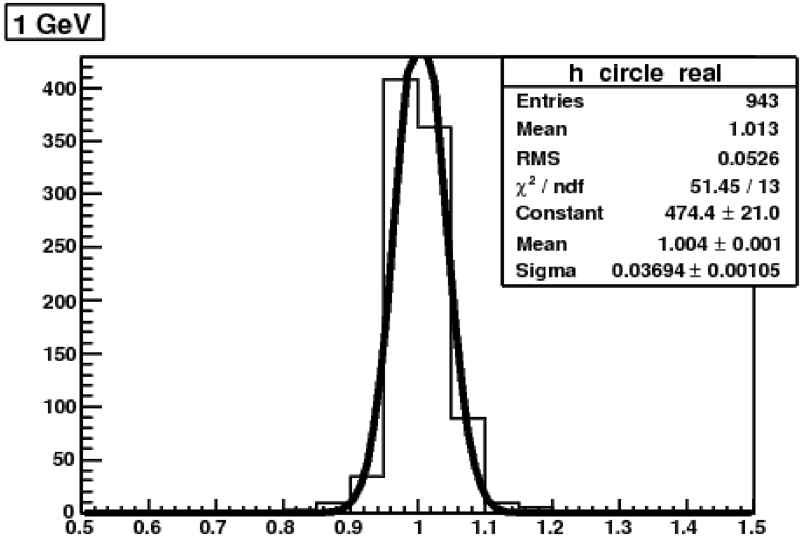
- Use all possible angles
- Save data in histogram
- Peaks in histogram represent possible lines in point set



Momentum resolution at 1GeV







σ:
Truth: 3.685%
Old: 3.732%
New: 3.694%

Precision check

One input hit.

……

000000011010100101110010000000000000110011101101000000100000000000000000000000000000011001101011

(1.6619, 0.0505, 2, 0.0250782) Unit (0.1m)

Generate two point candidates:

000000011010100110001011000000000000011001111110……………………………………………………

x1, y1 (1.66228, 0.0253601 …)   C++ calc: (1.66228,0.0254247)

000000011010100100110000000000000001001010001111 ……………………………………………………

x2, y2 (1.66089, 0.0724945 …)   C++ calc: (1.66076, 0.07555)

Conformal transformation

000000001001100111110111000000000000001001011001 ……………………………………………………

x1’, y1’ (0.601425, 0.00917053)   C++ calc: (0.601443, 0.00917575)

000000001001100111010110000000000000011010110110 ……………………………………………………

x2’, y2’ (0.600922, 0.0262146)   C++ calc: (0.600942, 0.0262299)

Hough space, not listed here.

Summary table of the resource utilization

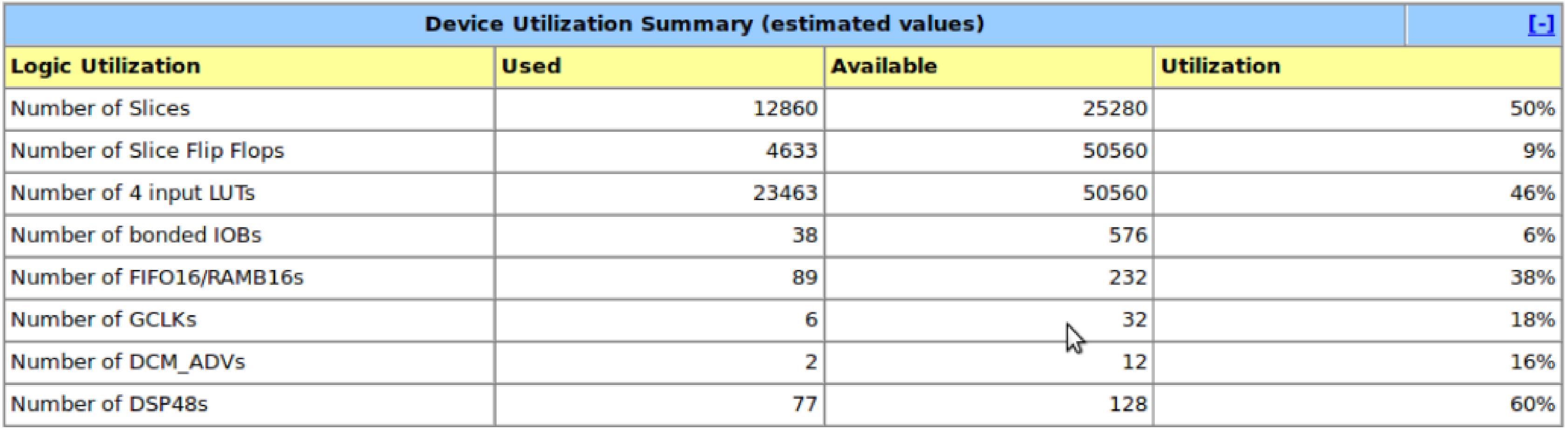
- Hough Space
  128*128 bins
- Parallelized in θ
- Serial in r
- 2 clock cycles per bin

⇒252 clockcycles

⇒2520 ns at 100 MHz

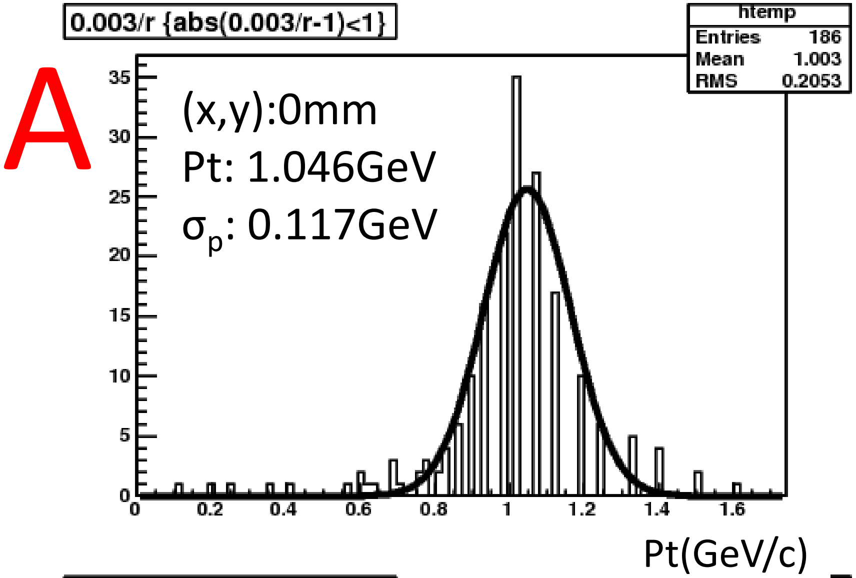
Estimation from
David’s previous design

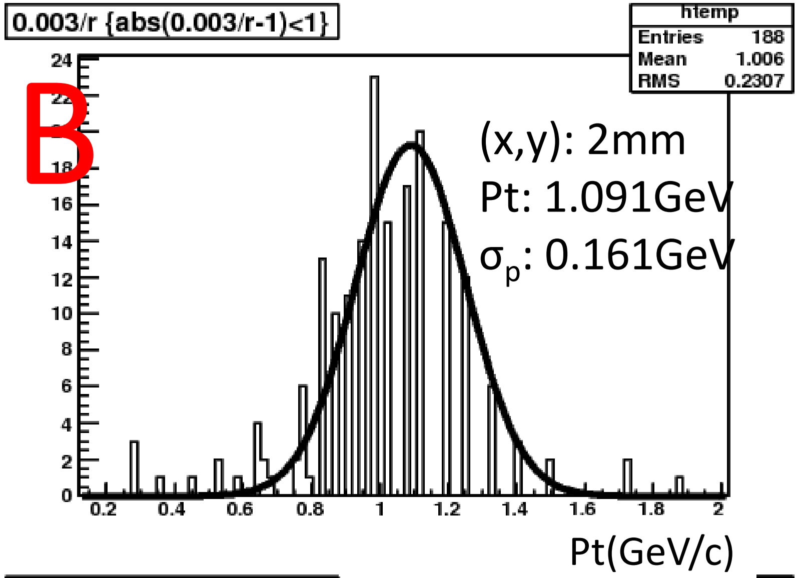
pipeline => latency.

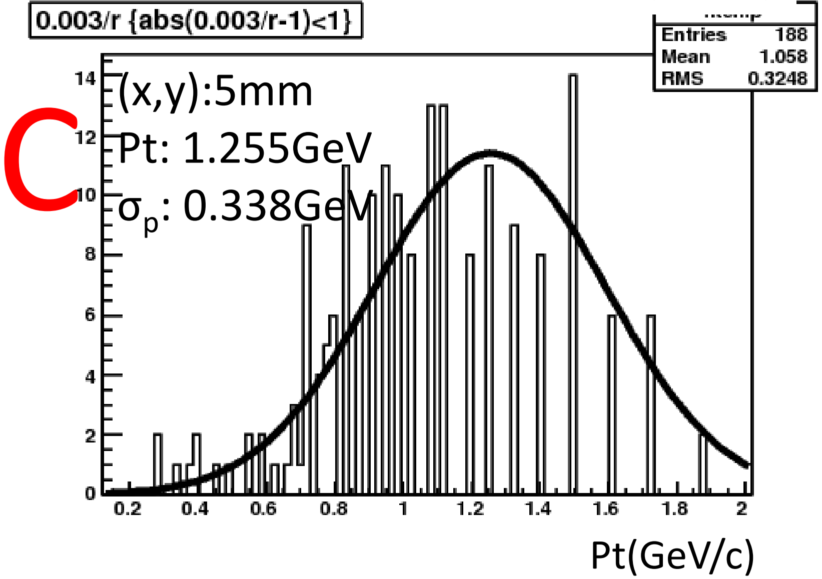


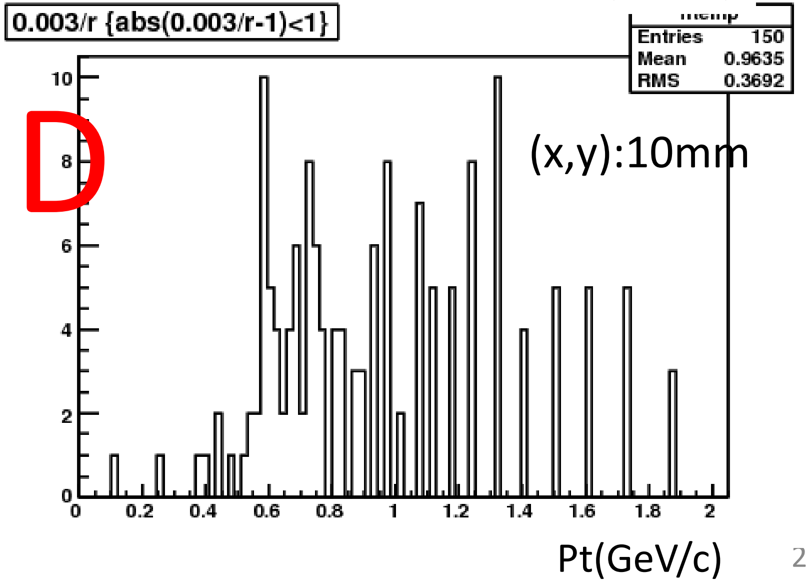
Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	12860	25280	50%
Number of Slice Flip Flops	4633	50560	9%
Number of 4 input LUTs	23463	50560	46%
Number of bonded IOBs	38	576	6%
Number of FIFO16/RAMB16s	89	232	38%
Number of GCLKs	6	32	18%
Number of DCM_ADVs	2	12	16%
Number of DSP48s	77	128	60%

Displaced vertex --old method



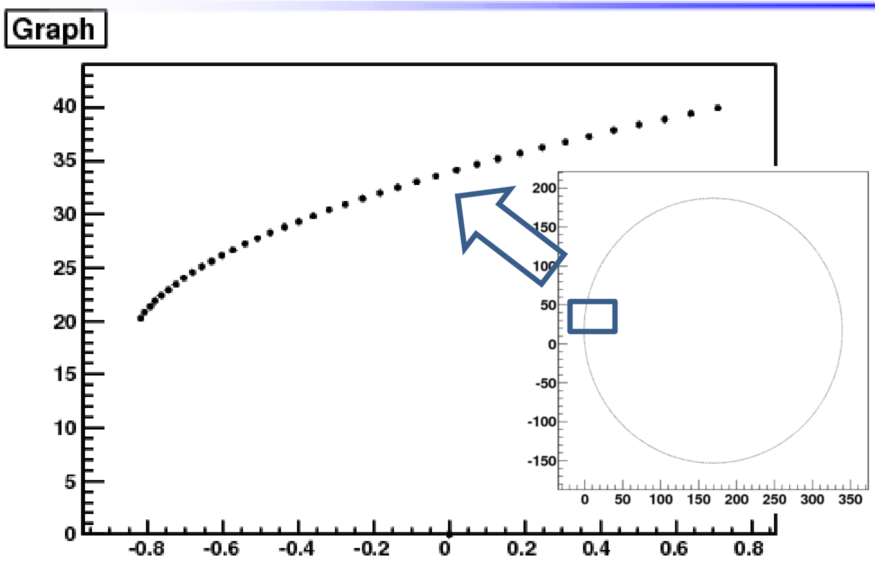


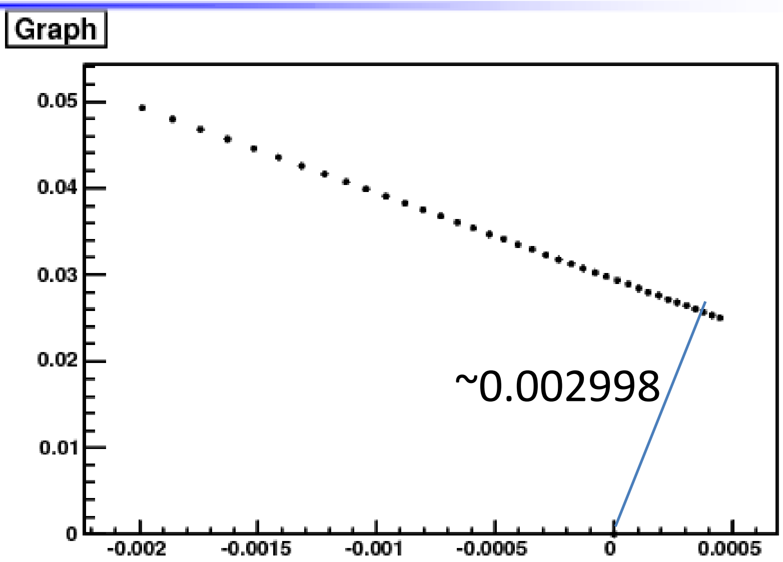




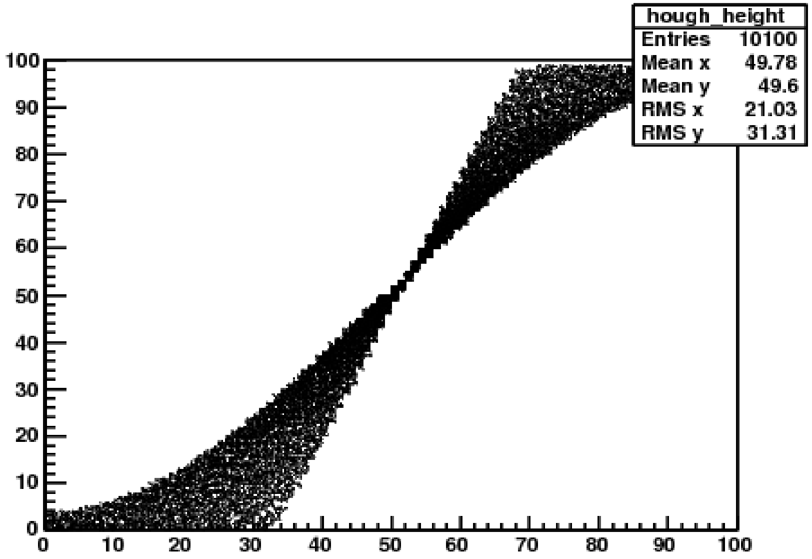
Behavior of transformations

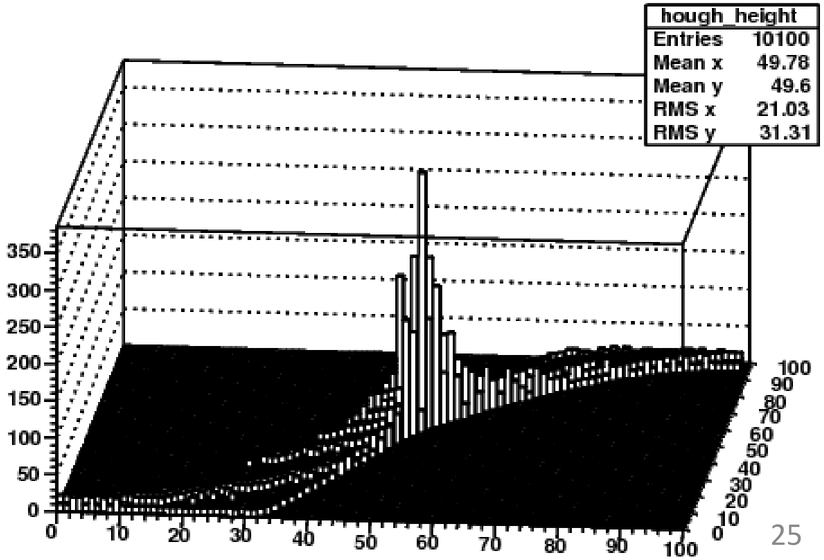
P = 1GeV phi = 100





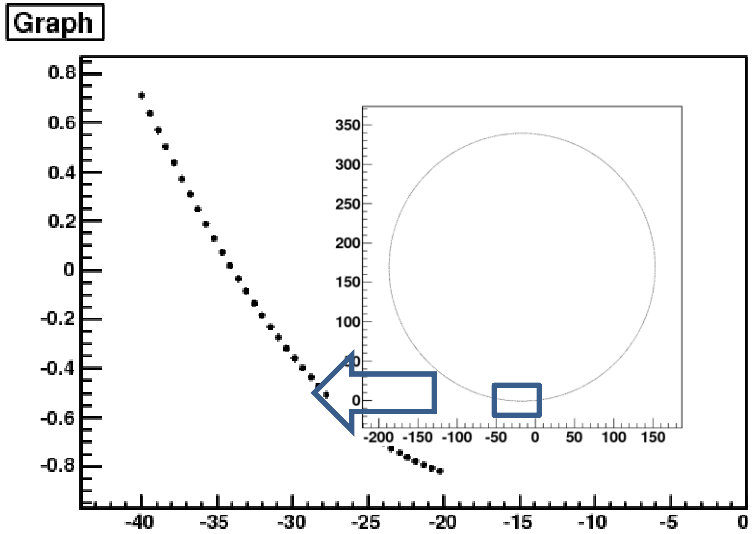
~0.002998

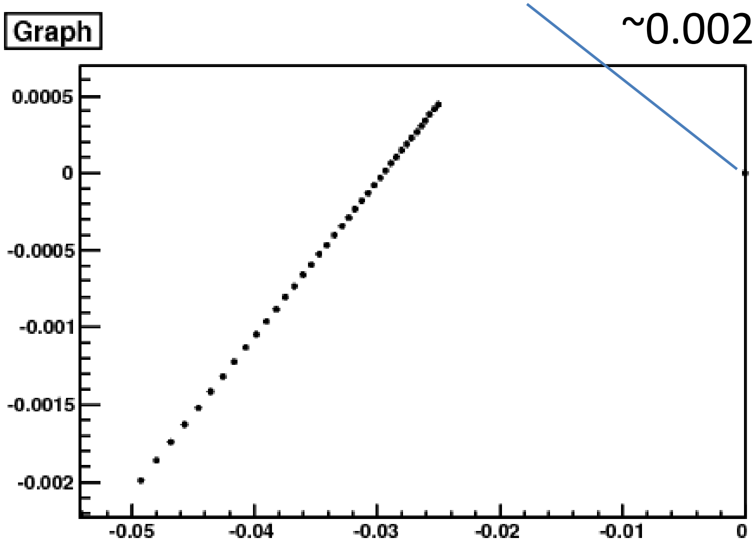




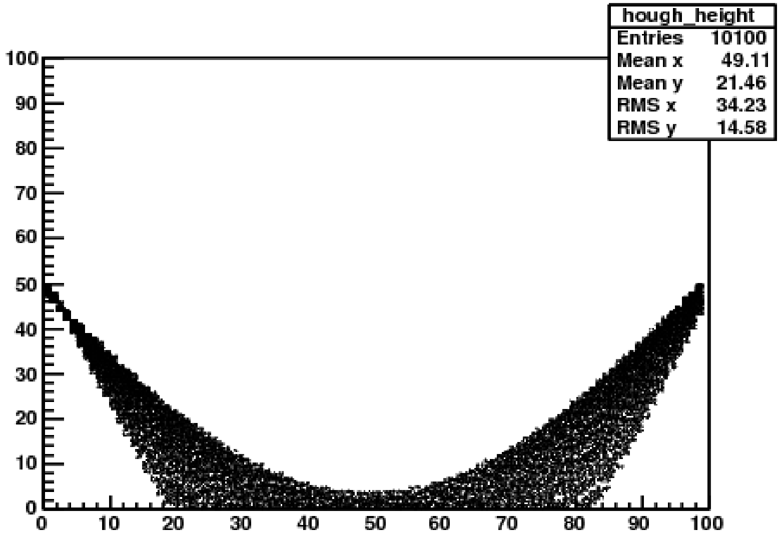
Behavior of transformations

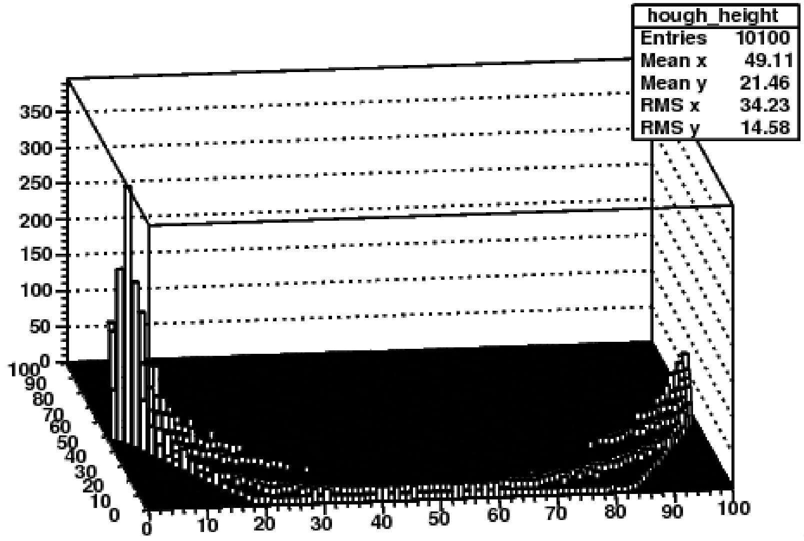
P = 1GeV phi = 190





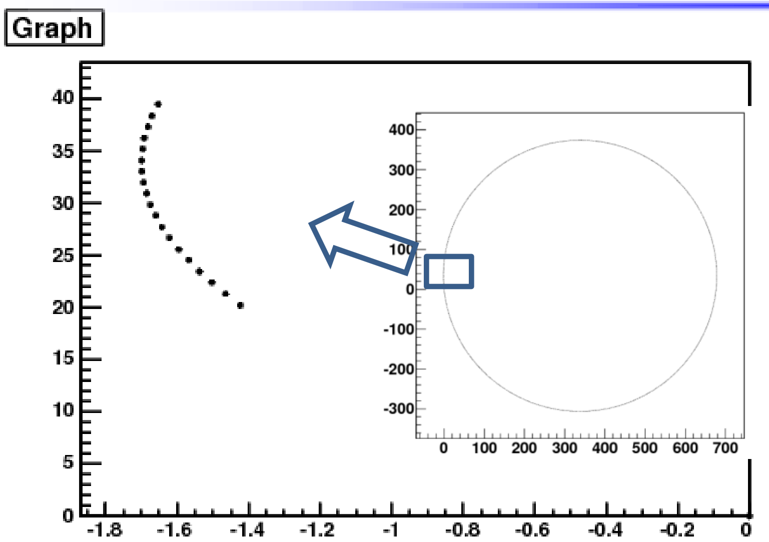
~0.002998

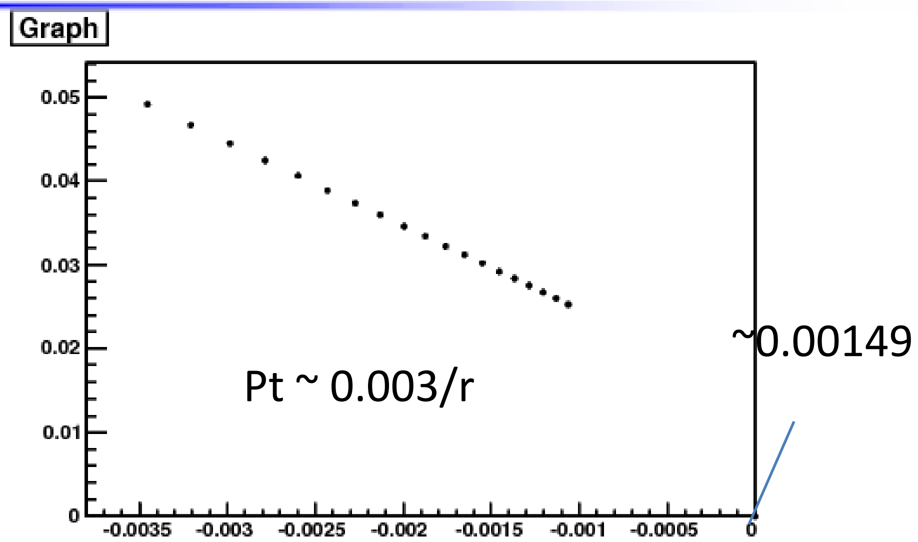




Behavior of transformations

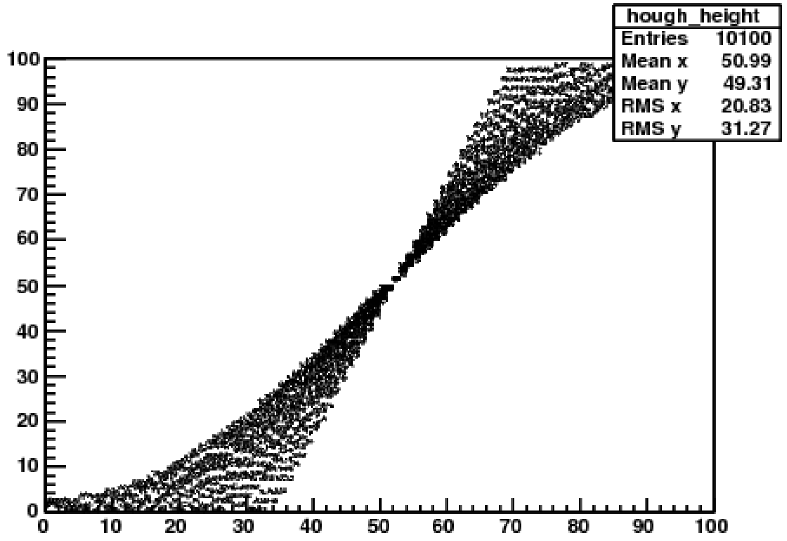
P = 2GeV phi = 100

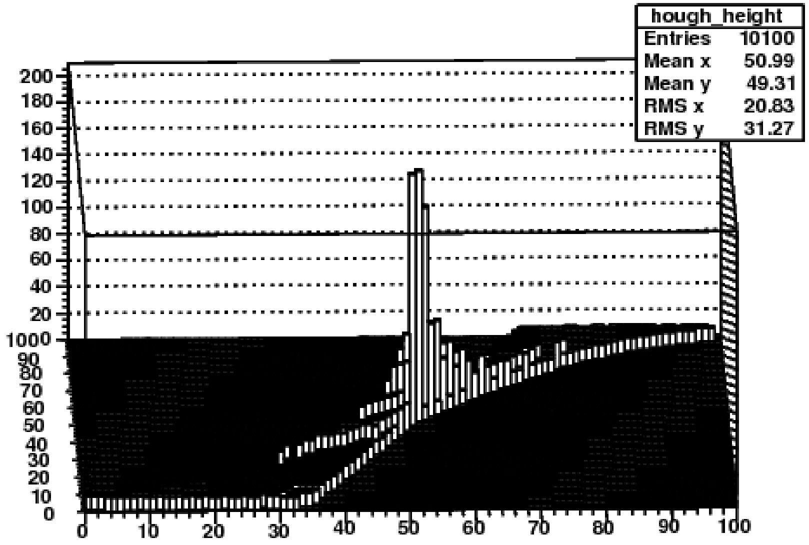


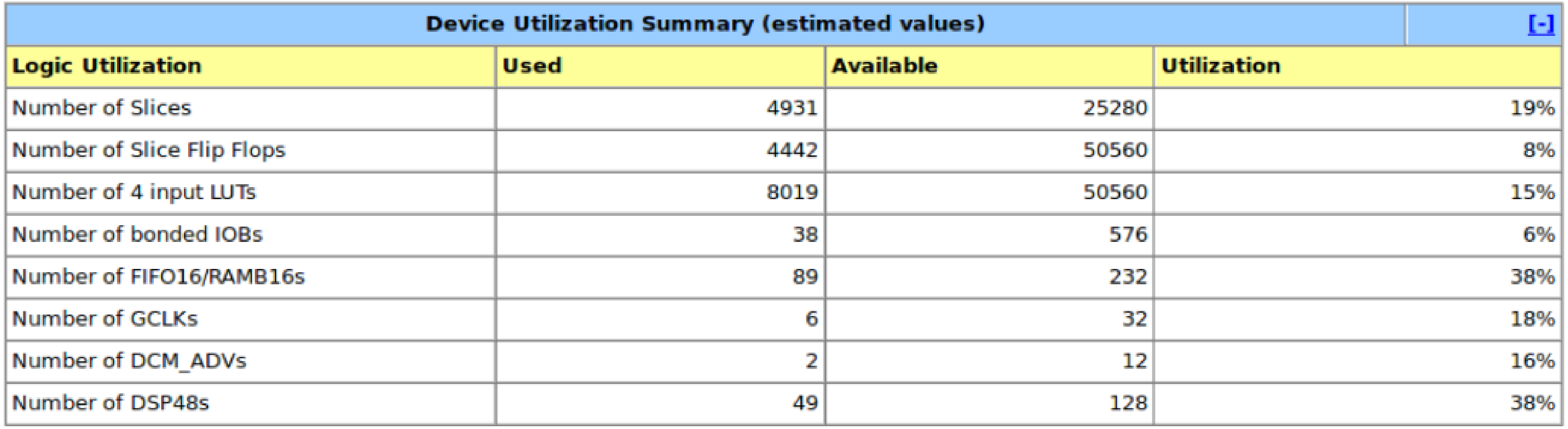


Pt ~ 0.003/r

~0.00149



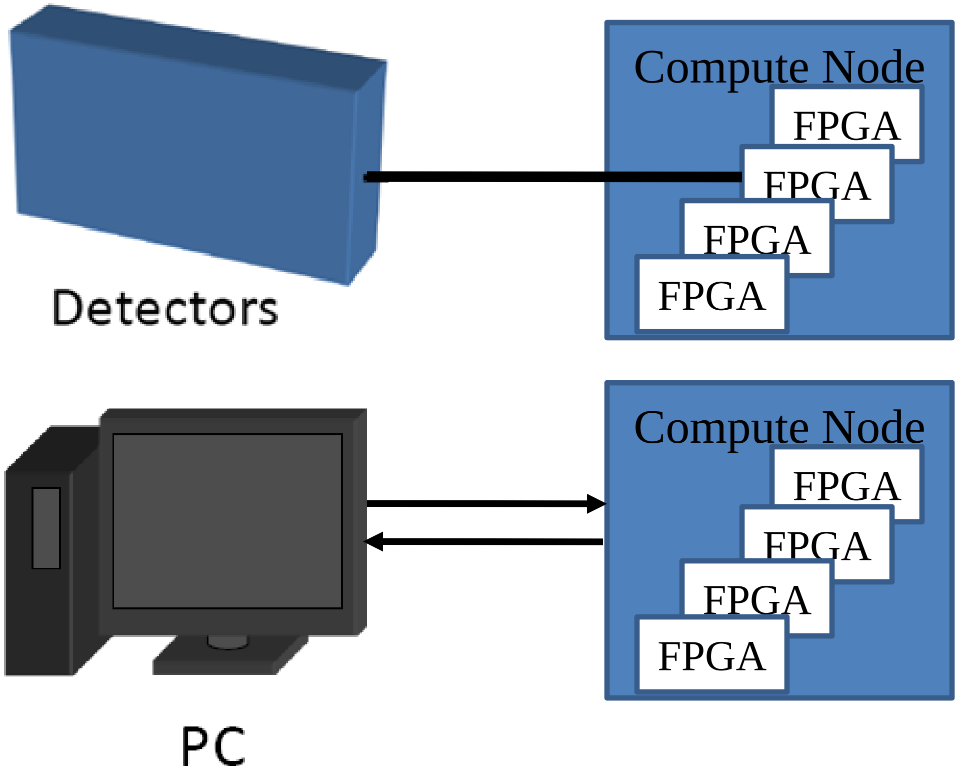




Device Utilization Summary (estimated values) [-]

Logic Utilization	Used	Available	Utilization
Number of Slices	4931	25280	19%
Number of Slice Flip Flops	4442	50560	8%
Number of 4 input LUTs	8019	50560	15%
Number of bonded IOBs	38	576	6%
Number of FIFO16/RAMB16s	89	232	38%
Number of GCLKs	6	32	18%
Number of DCM_ADVs	2	12	16%
Number of DSP48s	49	128	38%

PC as data source and receiver



Compute Node

FPGA

FPGA

FPGA

FPGA

Detectors

Compute Node

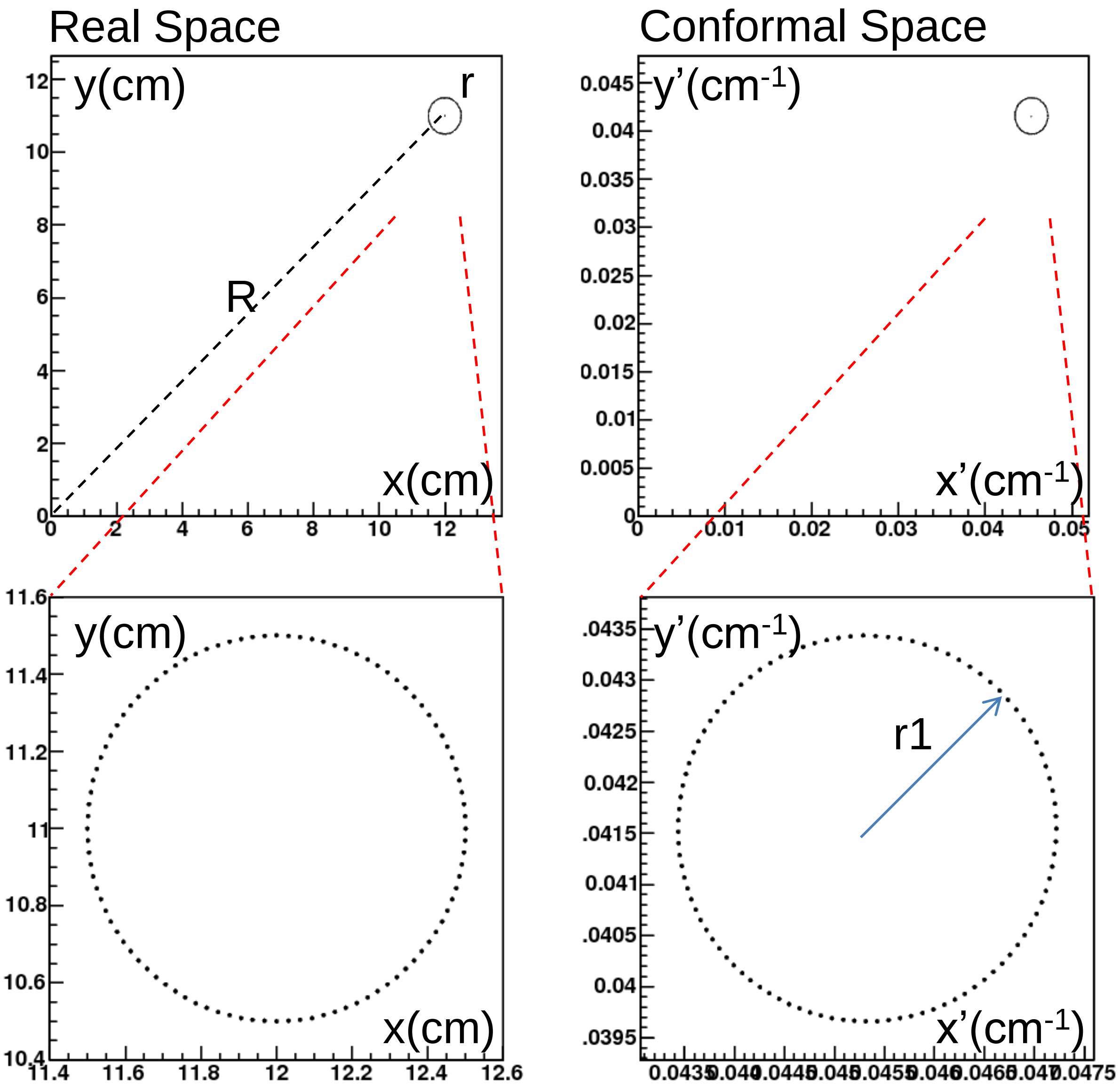
FPGA

FPGA

FPGA

FPGA

PC



Real Space

Conformal Space

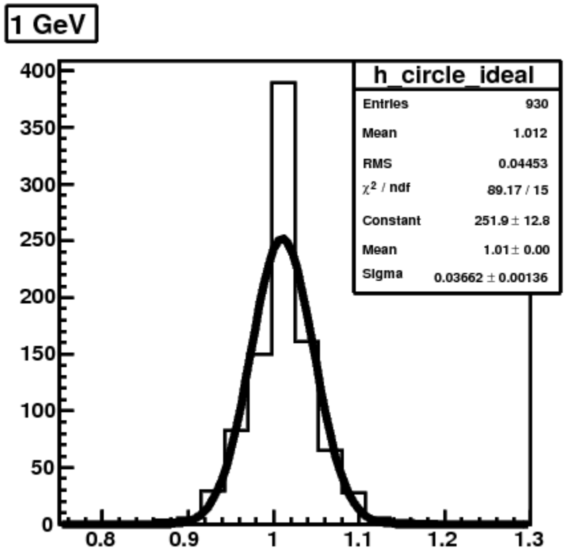
1. Circle in real space,
   center: (x0, y0)
   radius: r

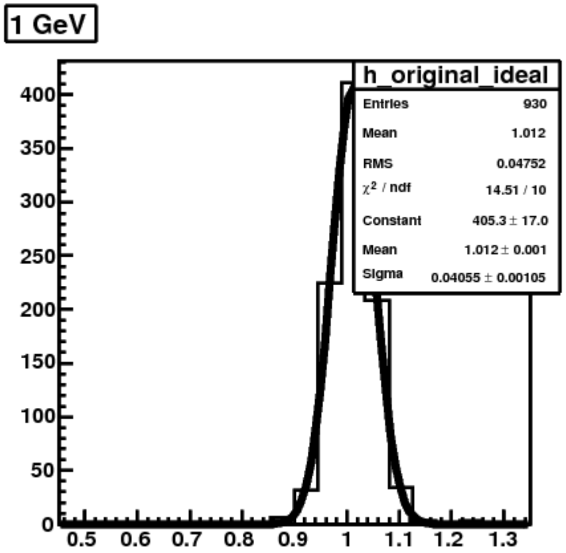
2. In CF space
   When R >> r, still a circle
   center: (x0’, y0’)
   radius: $r' = r/(R^2)$

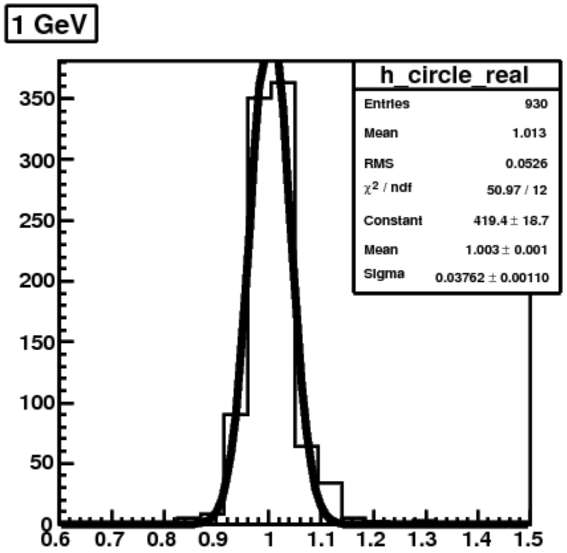
3. How good is the approximation?

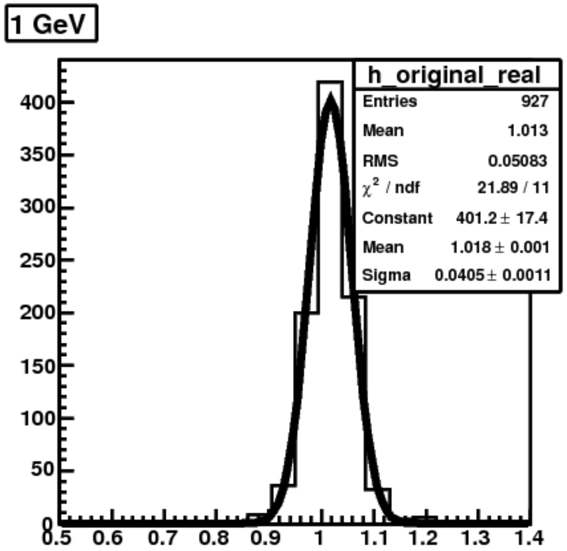
   $|r1-r'|/r' <= 3.2\ \%$

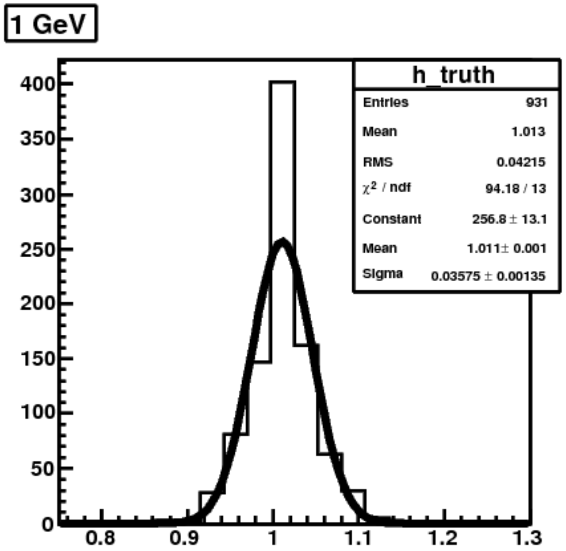
fb07:~/stt_online_tracking/points/calc_circle_cf.cpp



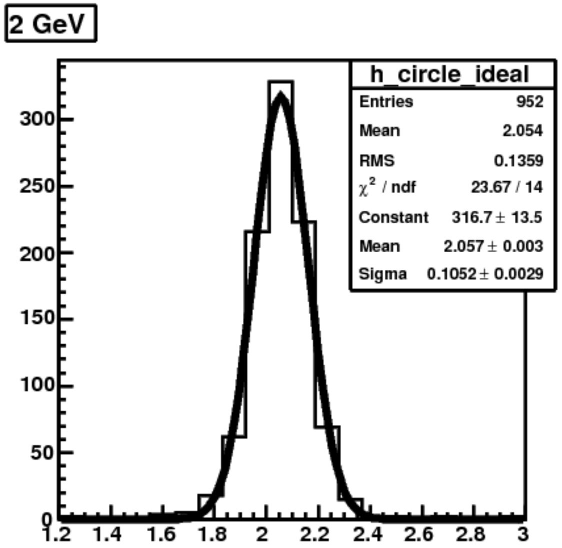


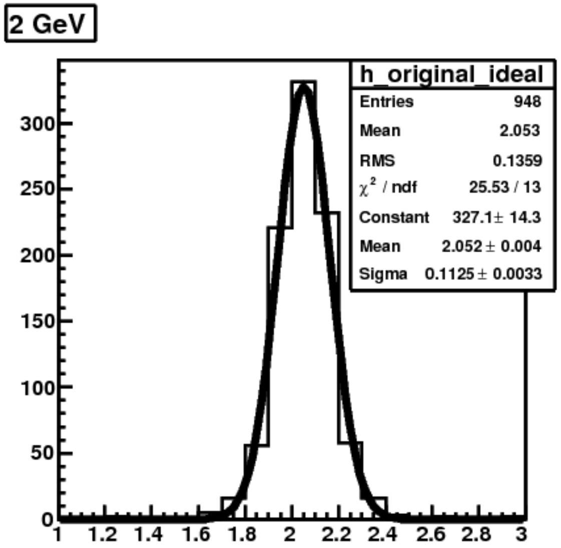


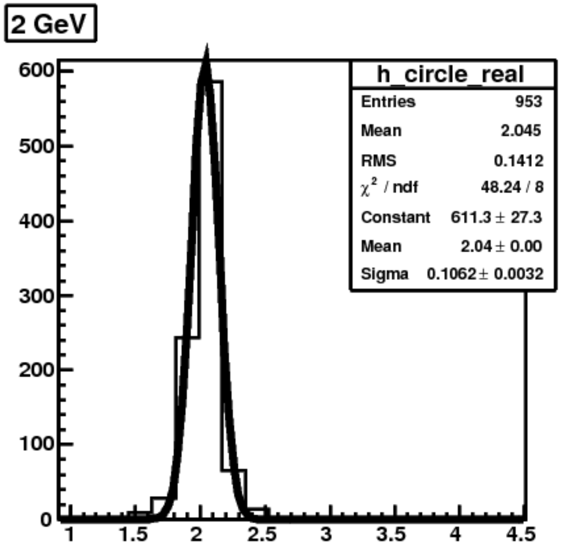


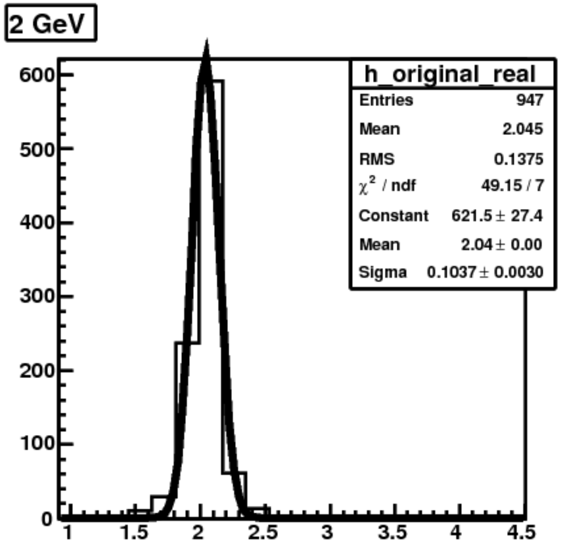


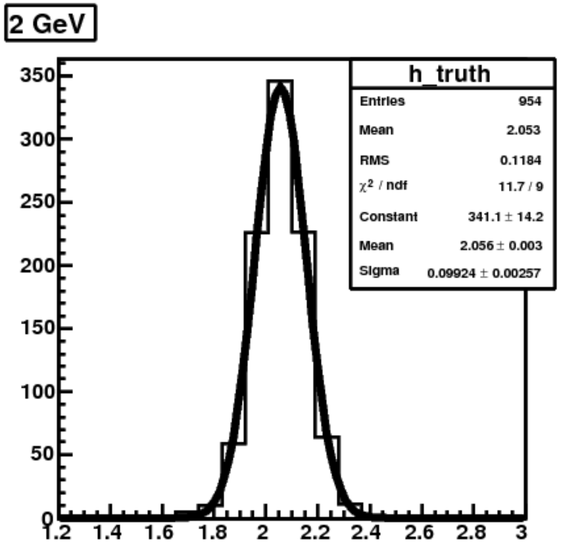
σ:
Truth: 3.58%
Old_ideal: 4.06%
Old_real: 4.05%
New_ideal: 3.66%
New_real: 3.76



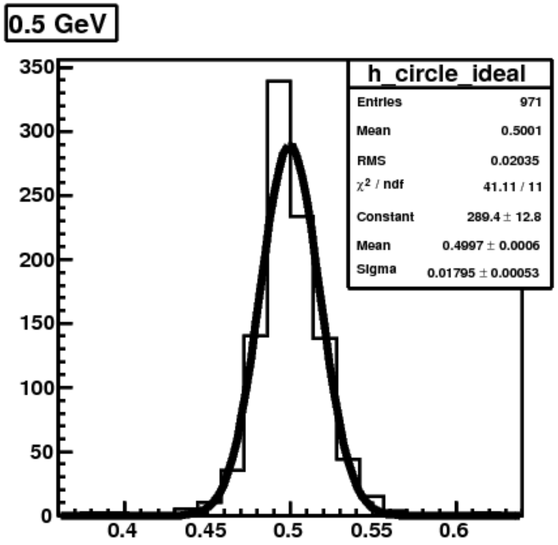


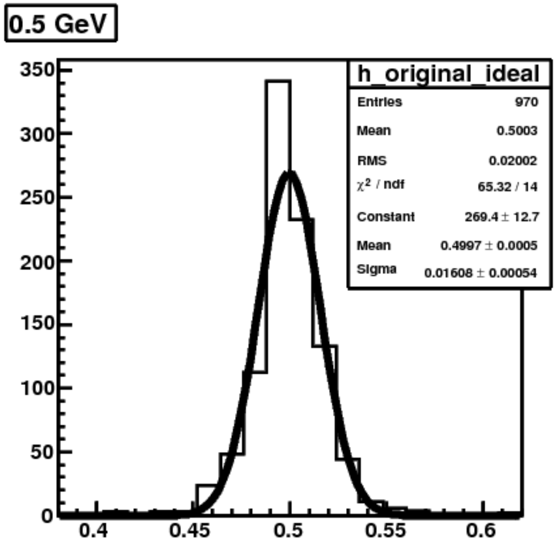


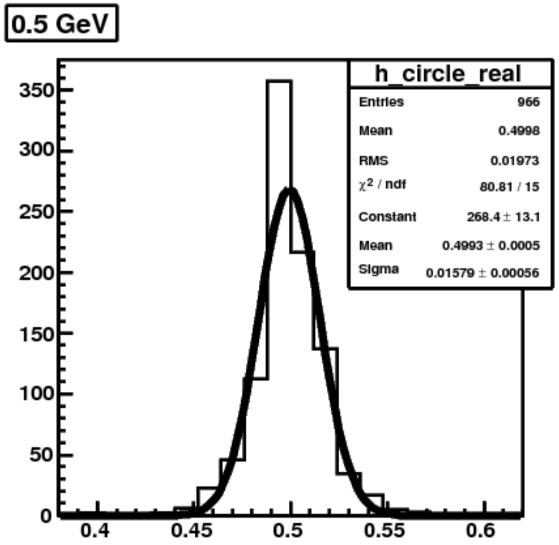


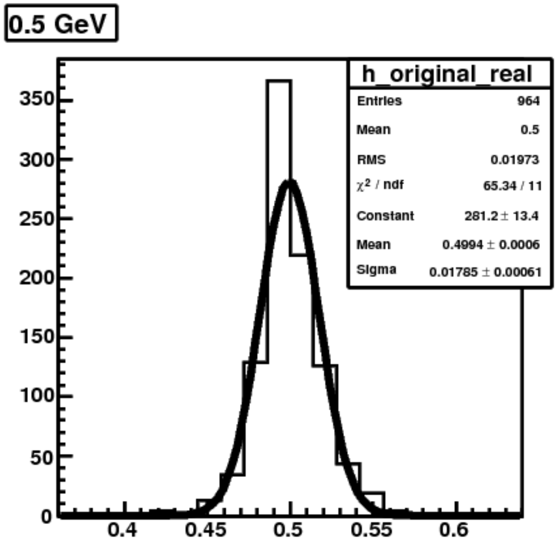


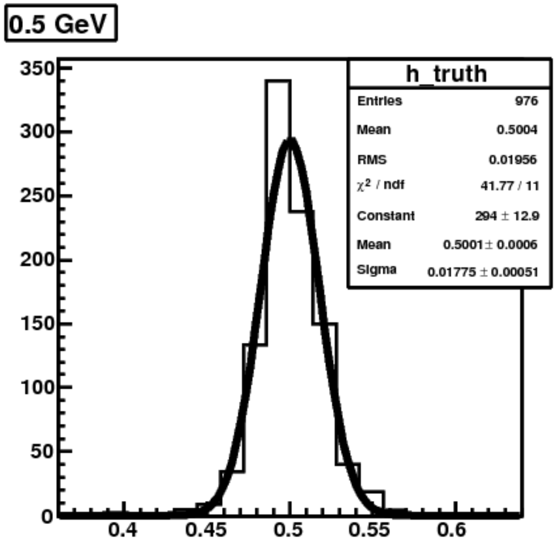
σ:
Truth: 5.0%
Old_ideal: 5.5%
Old_real: 5.2%
New_ideal: 5.3%
New_real: 5.3%



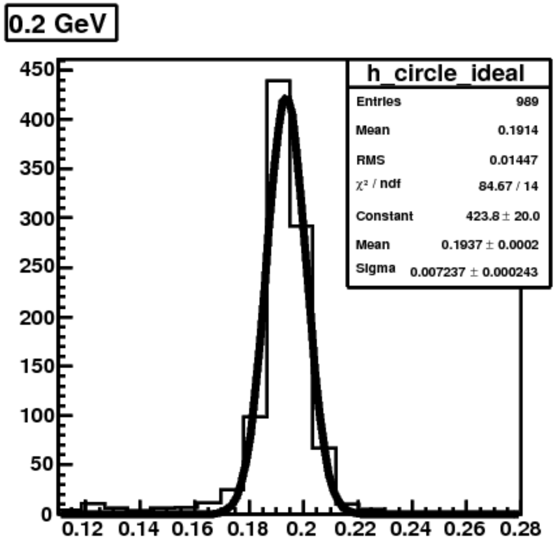


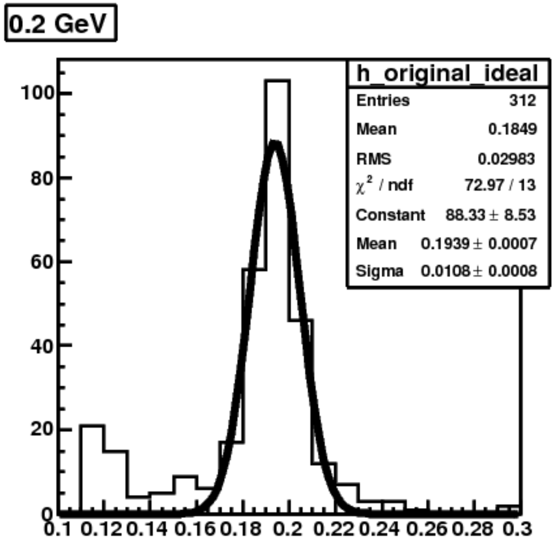


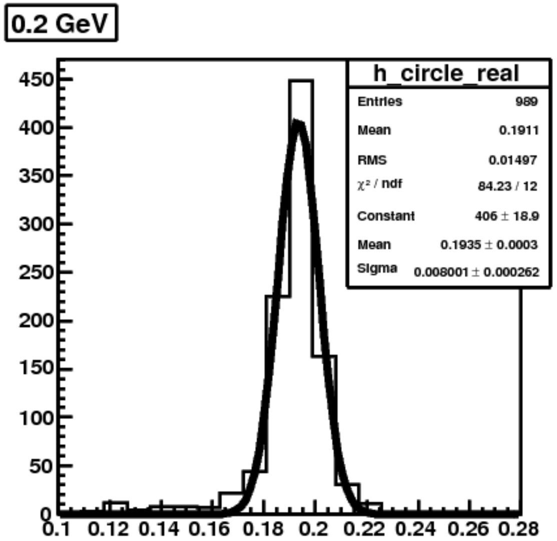


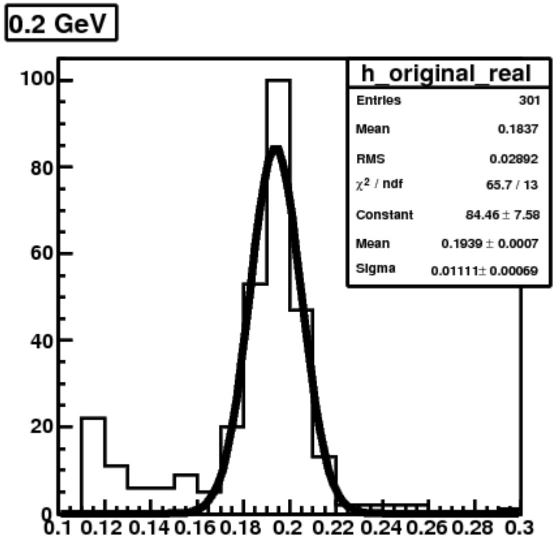


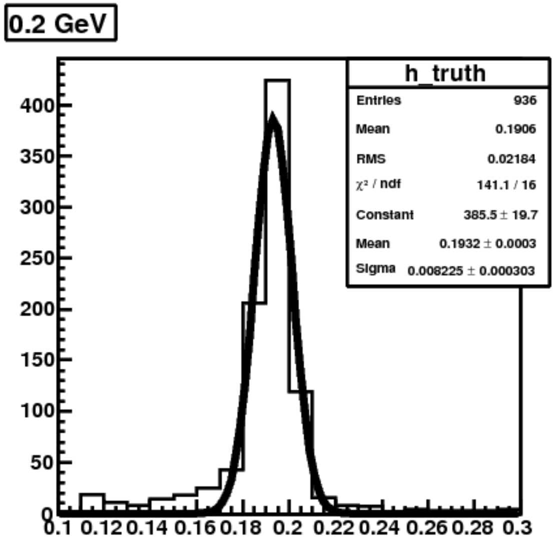
σ:
Truth: 3.54%
Old_ideal: 3.22%
Old_real: 3.57%
New_ideal: 3.59%
New_real: 3.16%





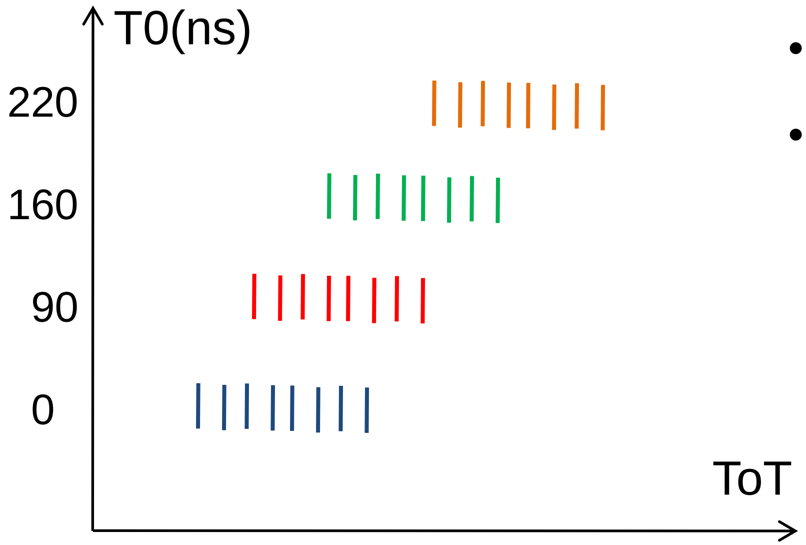




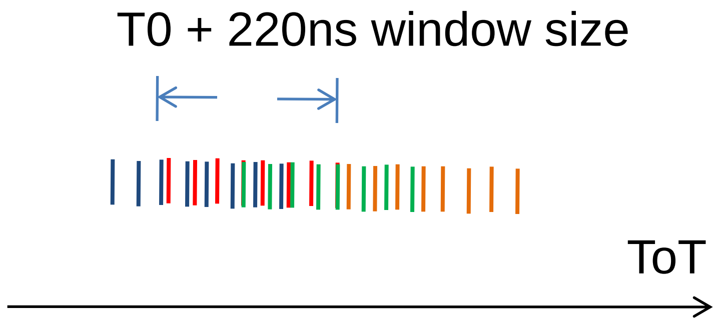


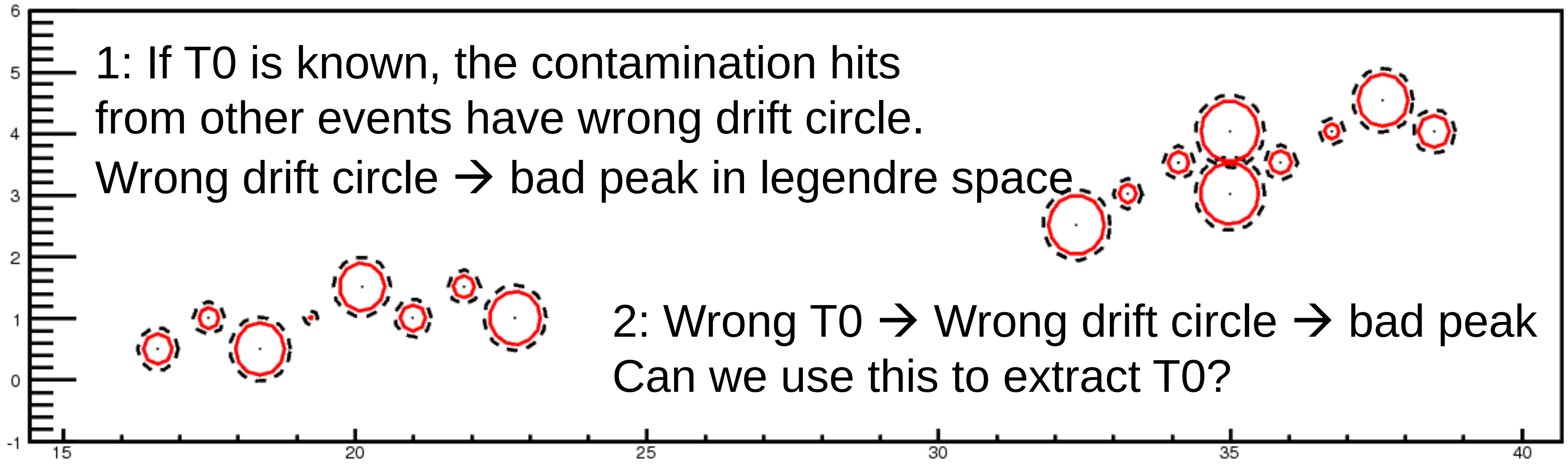
σ:
Truth: 4.1%
Old_ideal: 5.4%
Old_real: 5.6%
New_ideal: 3.6%
New_real: 4.0%

Can we extract the event start time?



- Need fast detector to provide T0.
- With T0, are the overlapped hits problem?





1: If T0 is known, the contamination hits from other events have wrong drift circle. Wrong drift circle → bad peak in legendre space

2: Wrong T0 → Wrong drift circle → bad peak
Can we use this to extract T0?