MIMOSA Analysis Framework (MAF) used in Test beam

and what a test beam analysis software should be able to do (personnal point of view)

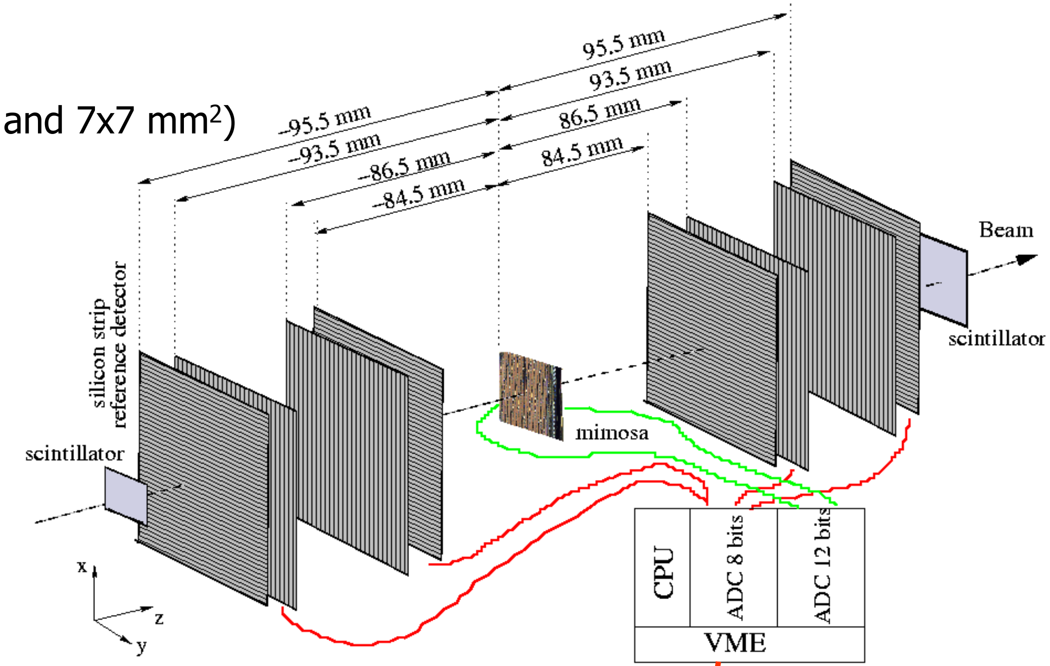
- Short overview of MAF / test beam
- Shopping list
- Alignment issues
- Data analysis issues

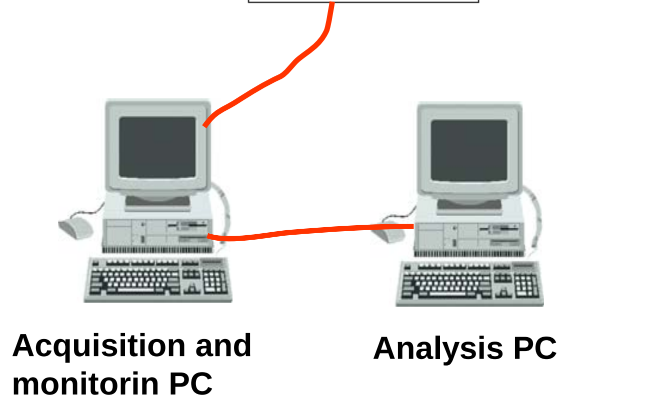
Auguste Besson (IPHC-Strasbourg)

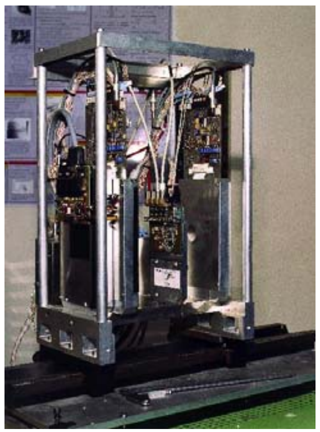
Strasbourg telescope

- Telescope
  - 8 reference planes
    - ➢ silicon microstrips, 256 x 50 µm
  - Trigger: 2 scintillitators planes (2x4 mm^2 and 7x7 mm^2)
  - Spatial resolution
    - ➢ ~2 µm/plane
  - Resolution in the DUT plane
    - ➢ ~ 1 µm
  - Cooling system
  - DUT rotation possible

- Data taking
  - Online monitoring
  - Rate
    - ➢ ~ 50 000 evts / hour (M9, 8k pixels)
    - ➢ ~ 2500 evts / h (M5, 500k pixels)
    - ➢ ~ 15-20 Runs / week
    - ➢ ~ 100ˢ Go / week.
  - Off line analysis
    - ➢ first results available in few hours



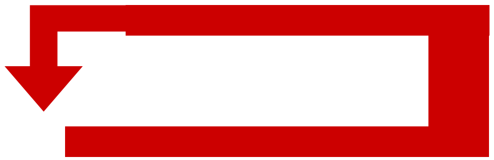




General structure of MAF (not optimal)

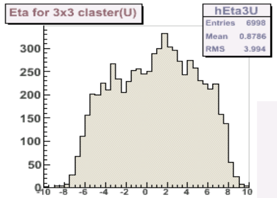
- Originally the program developped by RD42 collaboration (for strips)
  - ~ 6 different authors
    - ➢ Structure not optimized but…
    - ➢ take advantage of all the mistakes already made !

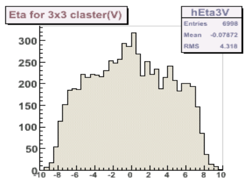
1) Generate eta function of the telescope planes
2) Alignment of telescope planes (from 2 fixed planes)
3) Reconstruction (hits in DUT and tracks selection)
4) Generate eta function of DUT
5) Alignment of DUT
6) Analysis (eff, noise, S/N, etc.)



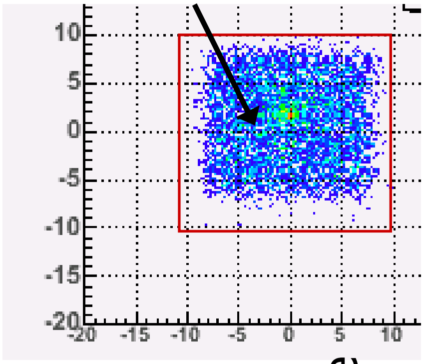
- Store:
  - Alignment parameters of the telescope planes
  - Eta functions parameter of telescope planes
  - Alignment parameters for DUT
  - Eta function parameters for DUT ?

eta functions for optimised resolution

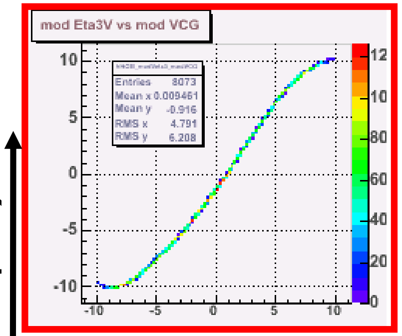




Impact position given by CoG inside a 20x20 μm² pixel



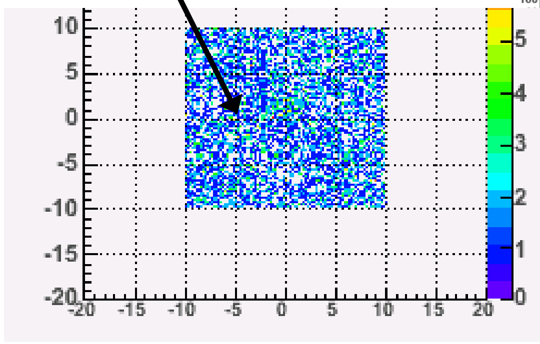
Eta function

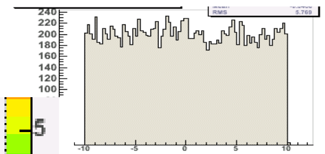


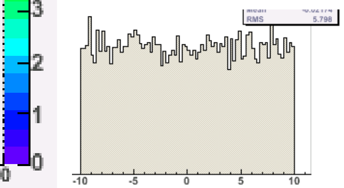
eta Value (output)

CoG Value (input)

Impact position given by eta function inside the pixel







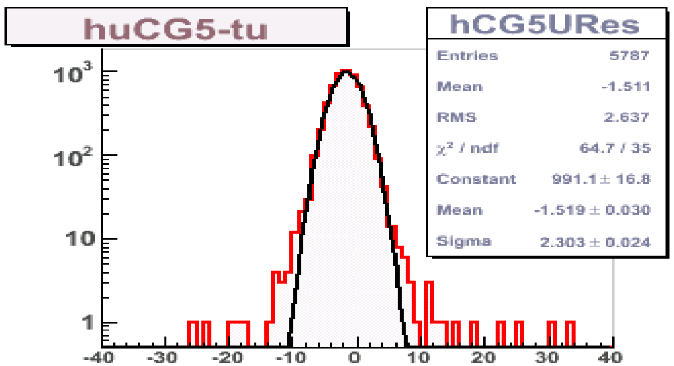
1/ CoG is not the optimal way to obtain a position

= bias introduced by non linearities of charge collection vs distance [diode-impact]

2/ eta method flatens the distributions

(probability density is expected to be flat)

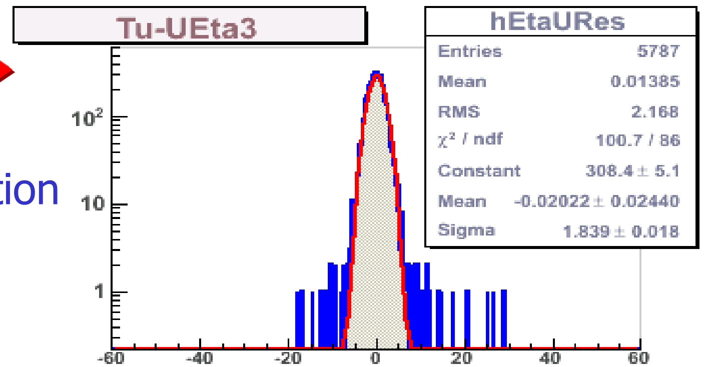
CoG(5x5) Residual: σ = 2.30 μm



3/ improves resolution significantly

(see back up slides for details)

Eta(3x3) Residual: σ = 1.84 μm



What a test beam analysis software should be able to do ? (1)

- The goal is not only to determine standard performances (efficiency, S/N, resolution, etc.) but also and mainly to determine and understand quickly any unexpected behaviour of the DUT (and there will be since we're dealing with prototypes !)
- Absolutly crucial
  - On line monitoring ⇨ other software.
    - rough alignment
    - trigger rates, beam profile/ flux setting
    - rough baseline, pedestal, noise, hit selection of DUT ⇨ detect anomalies quickly
  - Event display / scanning
    - jump to a particular event and analyse it in details (whole matrix, noise, pedestal, etc.)
  - Filters:
    - Map / mask of bad pixels, lines, etc. to tag / exclude it easily from analysis.
  - Flexibility = Complete datacard/input system
    - all possible parameters even radiation dose, Temperature, etc. so that this parameters are set in only one place and are accessible in analysis.
    - keep track of all the information in a not ambiguous way.
    - change the telescope configuration easily (positions, angles, number of planes, etc.)
    - All DUT have different formats, different numbers of sub-structures which could be analysed separatly or not.
    - **adapt reco easily < any DUT particularities.**

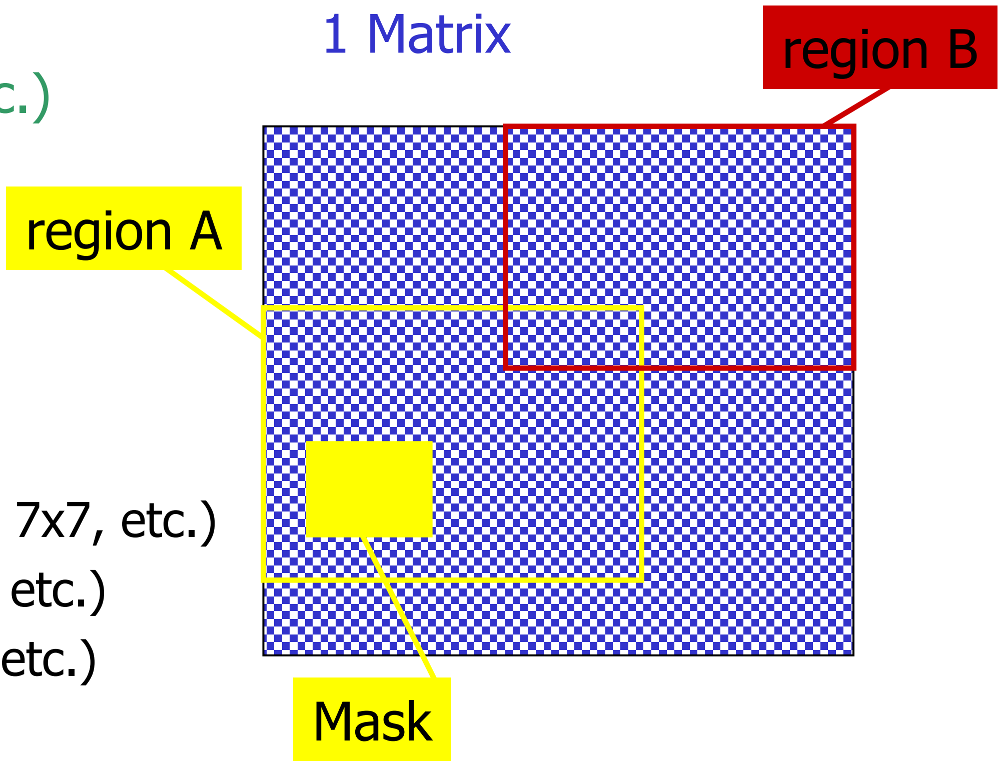
      e.g. add some common mode correction at the reconstruction level in a particular region of a given matrix for events taken between 2:12 AM and 3:47 AM.

What a test beam analysis software should be able to do ? (2)

- No crucial but <u>very</u> convenient for users
  - Complete analysis fast enough compared to data taking time to react quickly
  - Most users will want to compare different runs between them
    - ➢ some tool/framework to compare different runs easily:
    - ➢ e.g. : 5 consecutives runs @ 5 different temperatures ⇨ study Noise vs Temp.
  - Most users will need some calibration/ Noise runs
    - ➢ same framework to analyse these runs.
    - ➢ necessary to compute a fake hit rate
  - Most users will want to optimize their analysis
    - ➢ perform different hits algorithms / sets of cuts in the same reconstruction to compare performances

The dream of the user…

- Datacards options
  - Define matrices
  - Define several regions (A,B,etc.)
  - For each region define:
    - Noise: algo A, B, etc.
    - Pedestal algo A,B, etc.
    - hit selection algorithm x,y,z
    - S/N thresholds
    - cluster size (1, 2x2, 3x3, 5x5, 7x7, etc.)
    - simulated output chain (ADC, etc.)
    - filters (hot pixels, dead lines, etc.)



  - 1 root branch per region
- example M9: 4 outputs, 2 submatrices, study temperature gradient effect

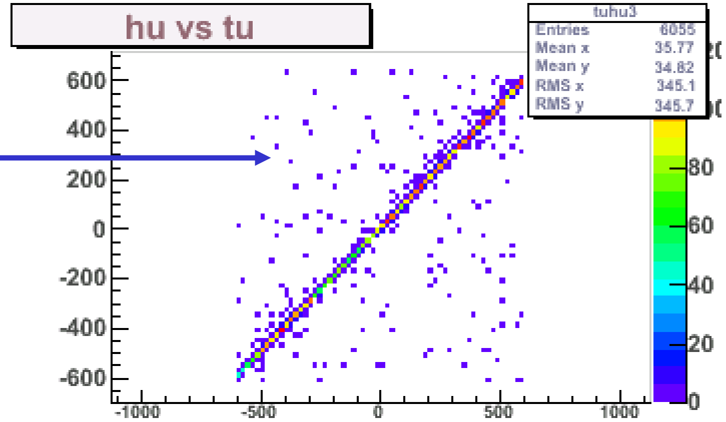
Alignment issues (1)

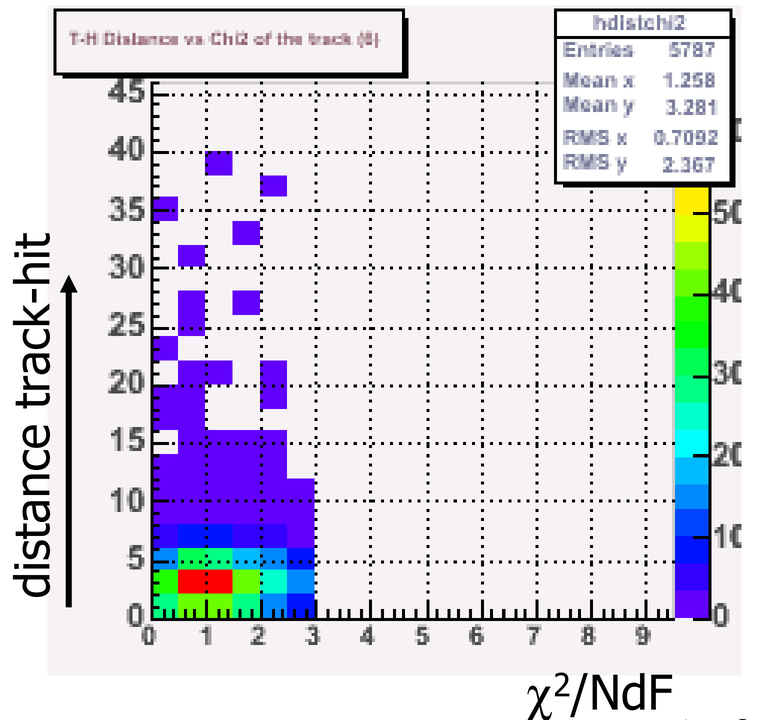
- In principle
  - Alignment of the telescope done when necessary
- In practice
  - Alignment of all planes is done for each run.
    - ➢ We are dealing with microns, so any change in temperature, any access in the test beam area could modify the alignment.
  - 2 alignments:
    - ➢ telescope alignment: relatively stable
    - ➢ DUT alignment: done for each run and possibly for each submatrix individually !
- to get the best alignment:
  - assume the algorithm needs to minimize distances between extrapolated tracks and all the associated hits in the DUT.
  - BUT:
    - ➢ you don't know a priori which hits can be associated to a given track. To know this ⇨you need to know the correct parameters of the alignment…
    - ➢ need some maximum track-hit distance cut (large before alignment)
    - ➢ some tracks don't go through the DUT itself (if the DUT is smaller than the track acceptance for instance) ⇨need to know the alignment to select the « good » tracks (=good acceptance of telescope)
    - ➢ may be necessary to do it for each submatrix individually

Alignment issues (2)

- « Recursivity » is not avoidable
  - ➢ To make the optimised alignement, need to already know it pretty well
  - – angles in the fit are very useful (study hits with large incident angle for instance) : 5 parameters (and not only 2) if position z is correctly known.
  - – Alignment of the telescope before the reconstruction
- New alignement for each configuration change

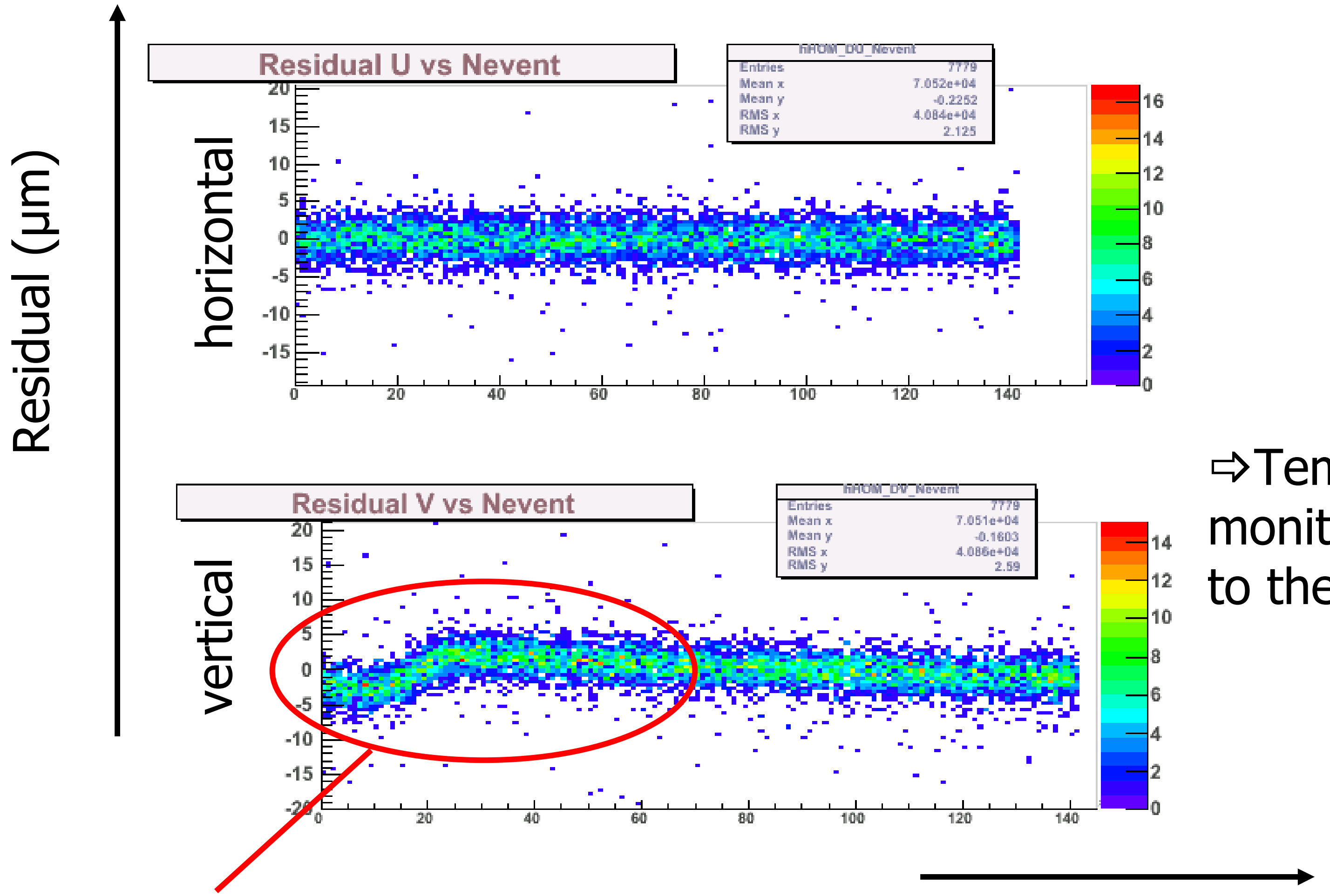
  (temperature, prototype, etc.)
- Track quality monitoring is mandatory
  - ➢ control chi², check that track selection doesn't introduce a bias, etc.
    - ➢ track hit distance, chi²





Is resolution homogenous ?

- How Mimosa breathes in CERN…



⇨Temperature monitoring coupled to the anaysis software

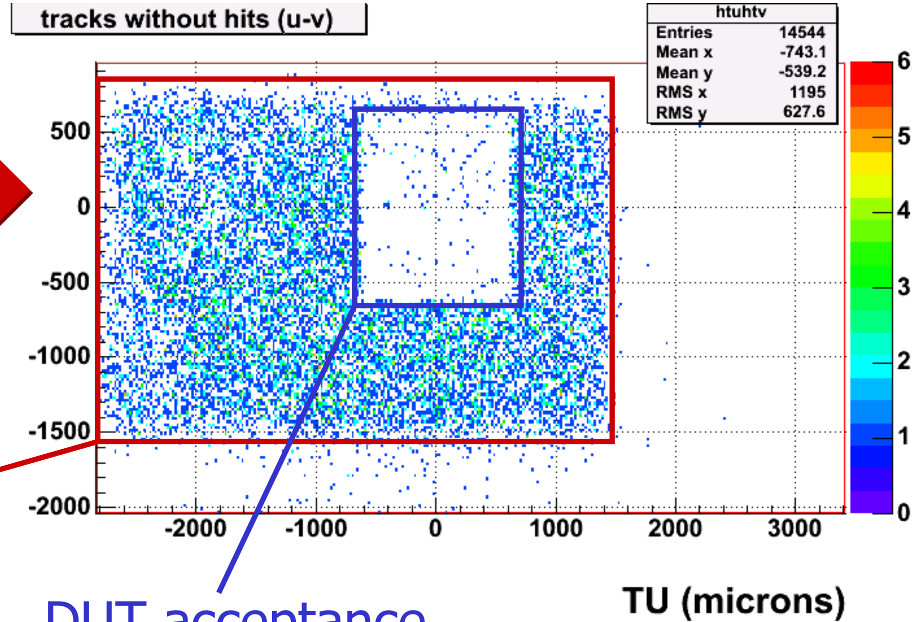
Offset ~ 4-5 µm

Event number (k)

Acceptance

- Remarks on triggers
  - Use of a large trigger to see where the DUT actually is

All track extrapolation position in the DUT plane which where not associated to a hit in the DUT



Trigger acceptance (2 x 4 mm^2)

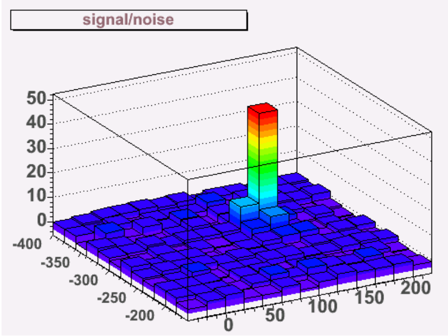
DUT acceptance

  - having an adapted trigger acceptance compared to the DUT is the easiest way to reduce data taking time.

Data issues

- The raw data can be very different from one DUT to another.
  - The user should only have to specify:
    - headers, trailers, data encoding (binary, hexa, etc.), nbits of ADC, number of submatrices, etc.
- Somehow need to reduce the amount of data via reconstruction
  - Select hits and tracks objects
    - keep only small amount of data
  - Example: Mimosa 9: 4 matrices (with different pitchs)
    - to get few 1000s events in a given matrix ⇨ ~10-20 Go runs
- Somehow need to be able to monitor everything during the run:
  - Example 1: pedestal and noise of a given number of pixels versus event number.
    - huge amount of data if it is done for all the pixels.
  - Example 2: some common mode in a given region.
- Study inefficiency:
  - Assume you reconstruct a track but have not corresponding hit in the DUT. You want to know what happened
    - access to the DUT signal AFTER alignment is done !
- Software analysis should allow to do:
  - Some standard reconstruction
  - Some « monitoring » reconstruction = Event display/scan
  - Some 2nd access to raw data after reconstruction

Some examples of monitoring



- Event display
- Efficiency vs fake rate
- Noise: vs… regions, time, etc.
- Resolution: vs method, impact in the pixel, etc.
- S/N: in seed, neighbours, etc.
- Clusters: size, charge sharing, etc.
- Double hit separation (time stamping?)
- Matrix uniformity
- Radiation, temperature, read-out frequency, etc.
- Incident angle
- Digitisation, ADC, etc.
- Edges effect

Conclusion

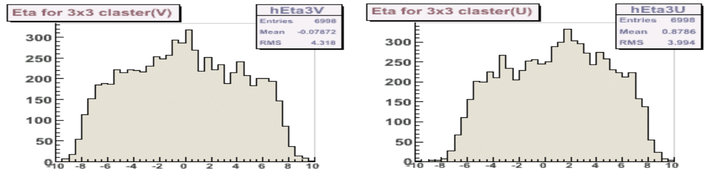
- The software will have to deal with many different configurations
  - It's not as simple as a « select hits and tracks » software.

- Users will want to study everything and in particular, things you didn't foresee.
  - The architecture has to be carefully discussed at the beginning

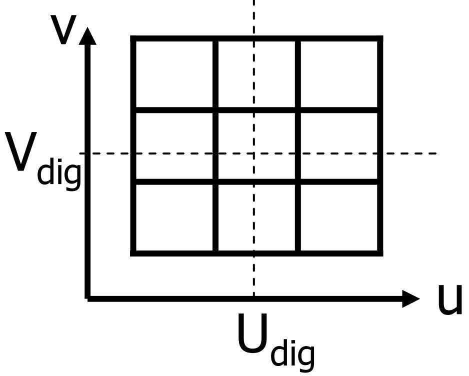
  (very bad experience with software made of 100s patchs)
  - User input is crucial

back up

Eta function principle



1/ Compute Center of gravity position from 3x3 cluster charge (Q_i) information

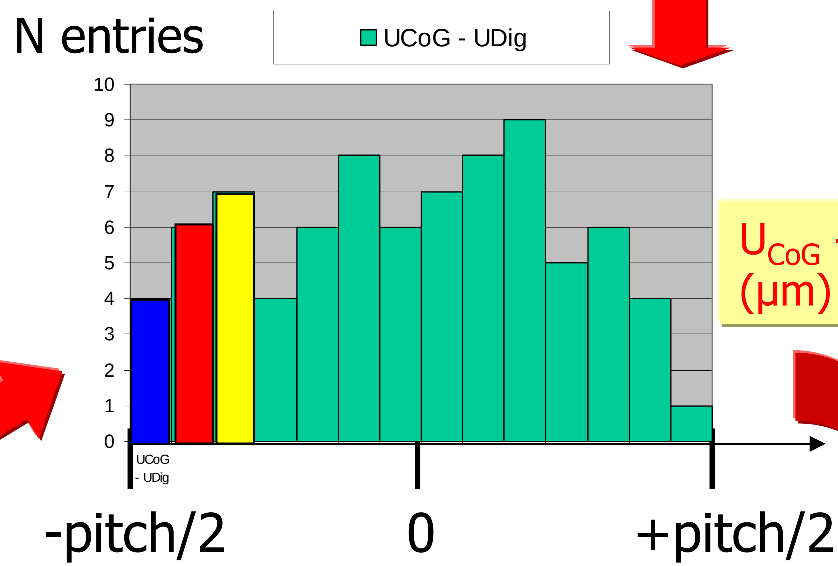


$$U_{CoG} = \frac{\sum_{i=1}^{9} Q_i u_i}{\sum_{i=1}^{9} Q_i}$$

2/ Plot Center of Gravity distance from the center of the pixel

$U_{CoG} - U_{Dig}$ (µm)

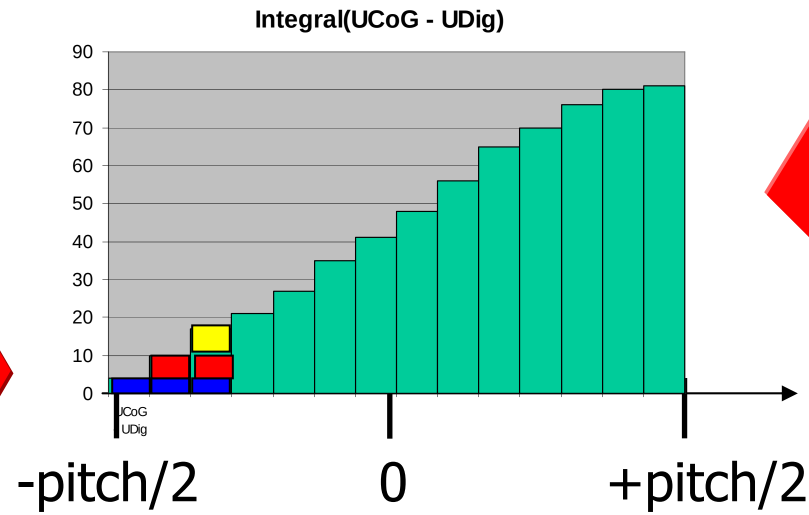
If there was no bias, this should be a flat distribution

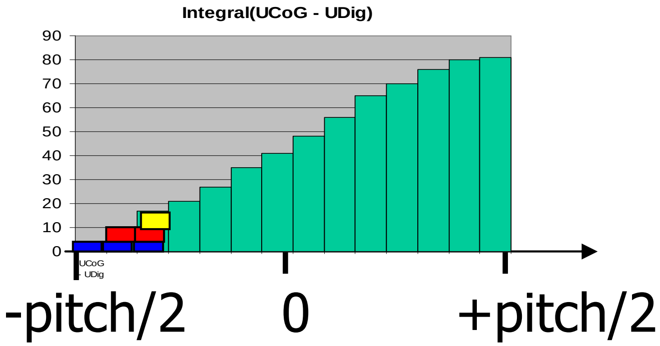


$U_{CoG} - U_{Dig}$ (µm)

3/ Integrate this distribution to get the f eta distribution function:

$$f = \int_{-pitch/2}^{x} \left(U_{CoG} - U_{Dig}\right) dx$$

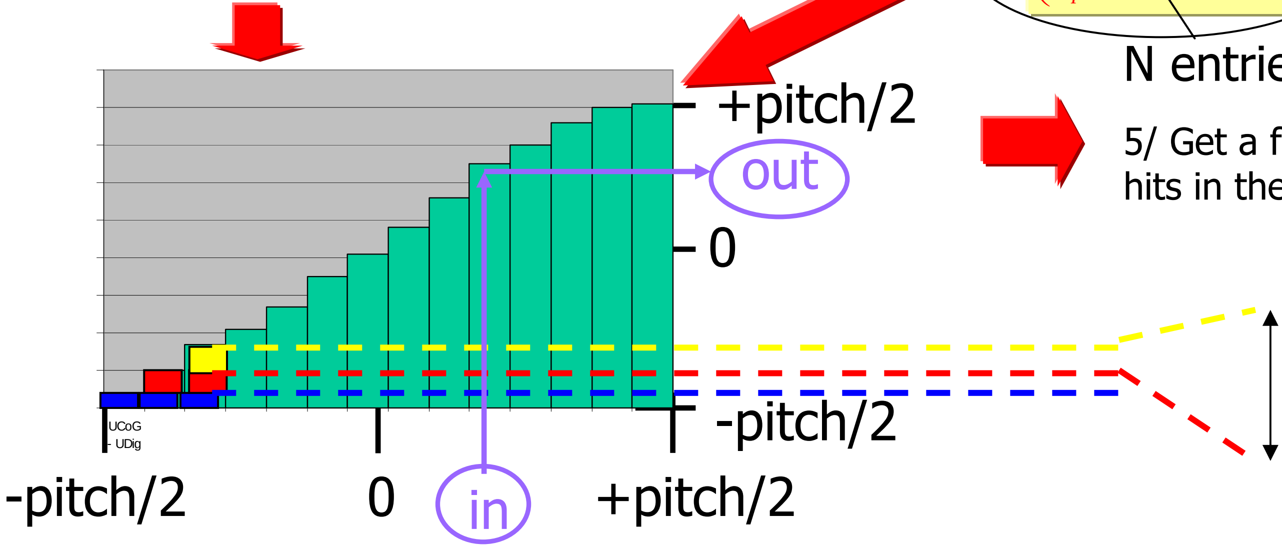




4/ Normalize by the Number of event (N entries), multiply by the pitch and shift it of - (pitch/2)

$$\left(\frac{\int_{-pitch/2}^{x} U_{CoG} - U_{Dig}\, dx}{\int_{-pitch/2}^{+pitch/2} U_{CoG} - U_{Dig}\, dx} \times pitch \right) - pitch/2$$

N entries



5/ Get a flat distribution of all hits in the pixel

This width is proportionnal to the number of entries in the yellow bin.

Consequences / issues:

- Needs to generate these eta functions…
- The bin size versus available statistics (N entries)
- The CoG can be outside the range [- pitch/2 ; + pitch/2] (happens ~1/1000)
- Number of different CoG values (from ADC so from different charge values) can not be lower than the number of bin. example: 2 bits ADC ~ 17 values; 3 bits ADC ~ 89 values; 4 bits ADC ~ 400 values
- Low statistics « in the corners of the phase space ».
- assume no correlation between U and V directions (not completly true)

Objectifs

- Tester les capteurs avec des m.i.p.
  - Reconstruction du passage de la particule grâce à un télescope
  - Alignement vis à vis du chip à tester
- Caractérisation des capteurs via:
  - ✓ Rapport signal/bruit
  - ✓ Efficacité de détection
  - ✓ Charge collectée (pixel siège et amas)
  - ✓ Résolution spatiale
- Différents paramètres:
  - ✓ Température
  - ✓ Chips irradiés (X, e^-, n)
  - ✓ Exploration de différents procédés de fabrication
  - ✓ Explorations des paramètres géométriques
    - ✓ Pitch; surface de la diode de collection de charge
    - ✓ Épaisseur de la couche épitaxiale
- Études complémentaires:
  - ✓ Critères de sélection/efficacité
  - ✓ Angle d'incidence
  - ✓ Pouvoir de séparation des impacts
  - ✓ Cartographie des matrices (uniformité des caractéristiques)
  - ✓ Uniformité entre les prototypes
  - ✓ Effets de la digitisation, etc.
- La caractérisation précise des performances des capteurs passe nécessairement par des tests en faisceau