SLHC Tracking Trigger Framework and Utilities

Andrew Rose

Summary

- A little terminology
- Simulation infrastructure
- Performance
- Conclusion

Summary

- A little terminology
- Simulation infrastructure
- Performance
- Conclusion

A little terminology

The following definitions have been agreed upon by a panel of members from the tracker upgrade and track-trigger groups.

- two or more sensors seperated by O(mm) = a stack
- the sensors within a stack = stack members.
- a high pT correlation between hits in a stack = a stub
- two, three, four, ... stacks separated by O(cm)→O(m) = a double, treble, quadruple, ... stack
- a correlation between stubs in a double, treble, quadruple, ... stack = a tracklet

Reserved for future use only:

- the basic element from which trigger primitives are produced = a station (be that a stack, a double stack, a treble stack,...)

The terms **doublet**, **superlayer** and **superstack** have all been depreciated and should not be used.

Available geometries

There are three stacked tracker geometries available:

- Strawman B - general purpose "concept" detector
- Long barrel design - 1st "baseline" design
- Hybrid geometry - 2nd "baseline" design

CVS location: CMSSW/SLHCUpgradeSimulations/Geometry

Full instructions for installation are kept up-to-date on the Wikipages:

https://twiki.cern.ch/twiki/bin/view/CMS/ExampleStrawmanB

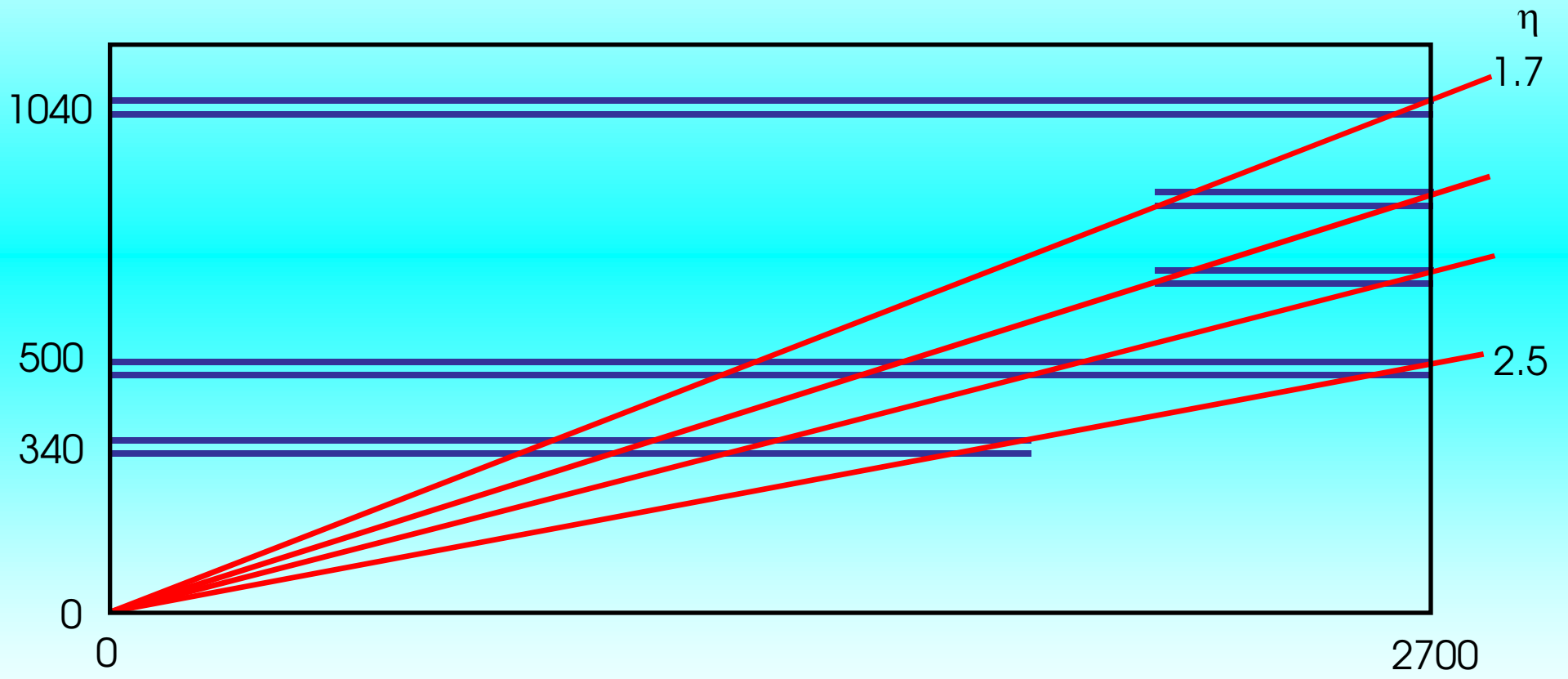
https://twiki.cern.ch/twiki/bin/view/CMS/ExampleLongBarrel

https://twiki.cern.ch/twiki/bin/view/CMS/ExampleHybridGeometry

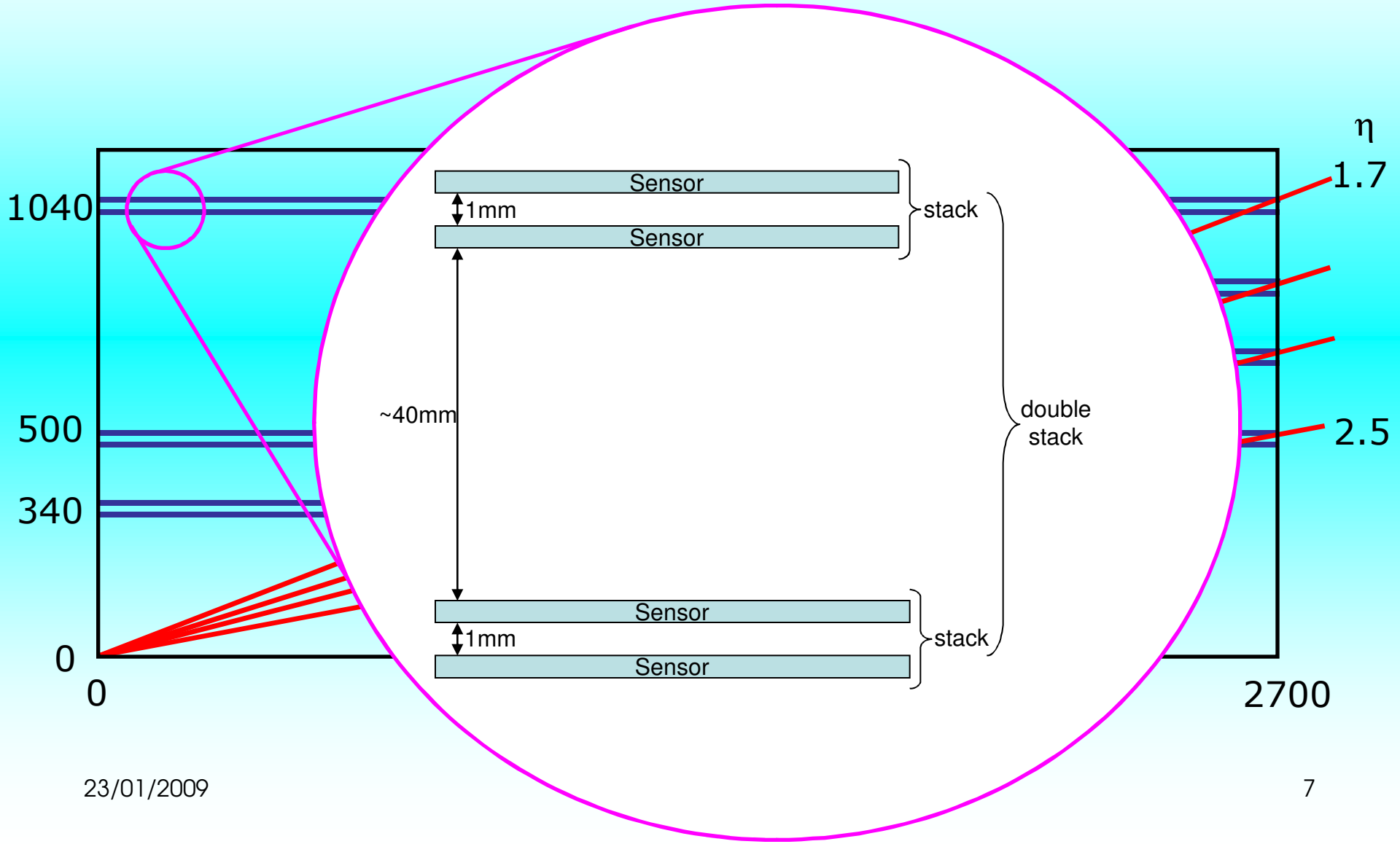
Please note: Strawman A is not a "Stacked Tracker" and is not covered in this talk.

The Long Barrel Design

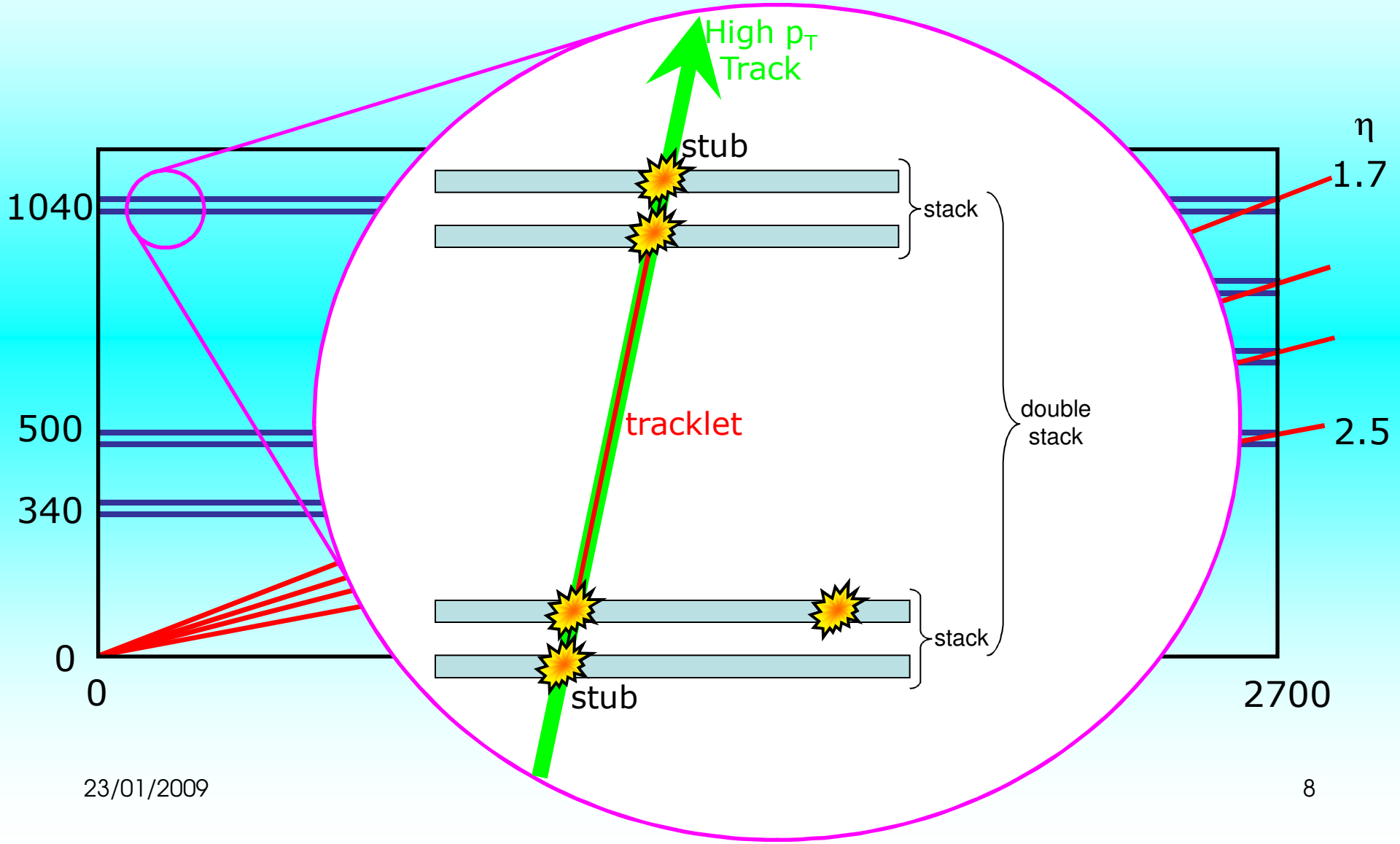
1st 'Baseline' upgrade design agreed at Fermilab workshop 24th November 2008



Looking a little closer…



Looking a little closer (2)…



Summary

- A little terminology
- Simulation infrastructure
- Performance
- Conclusion

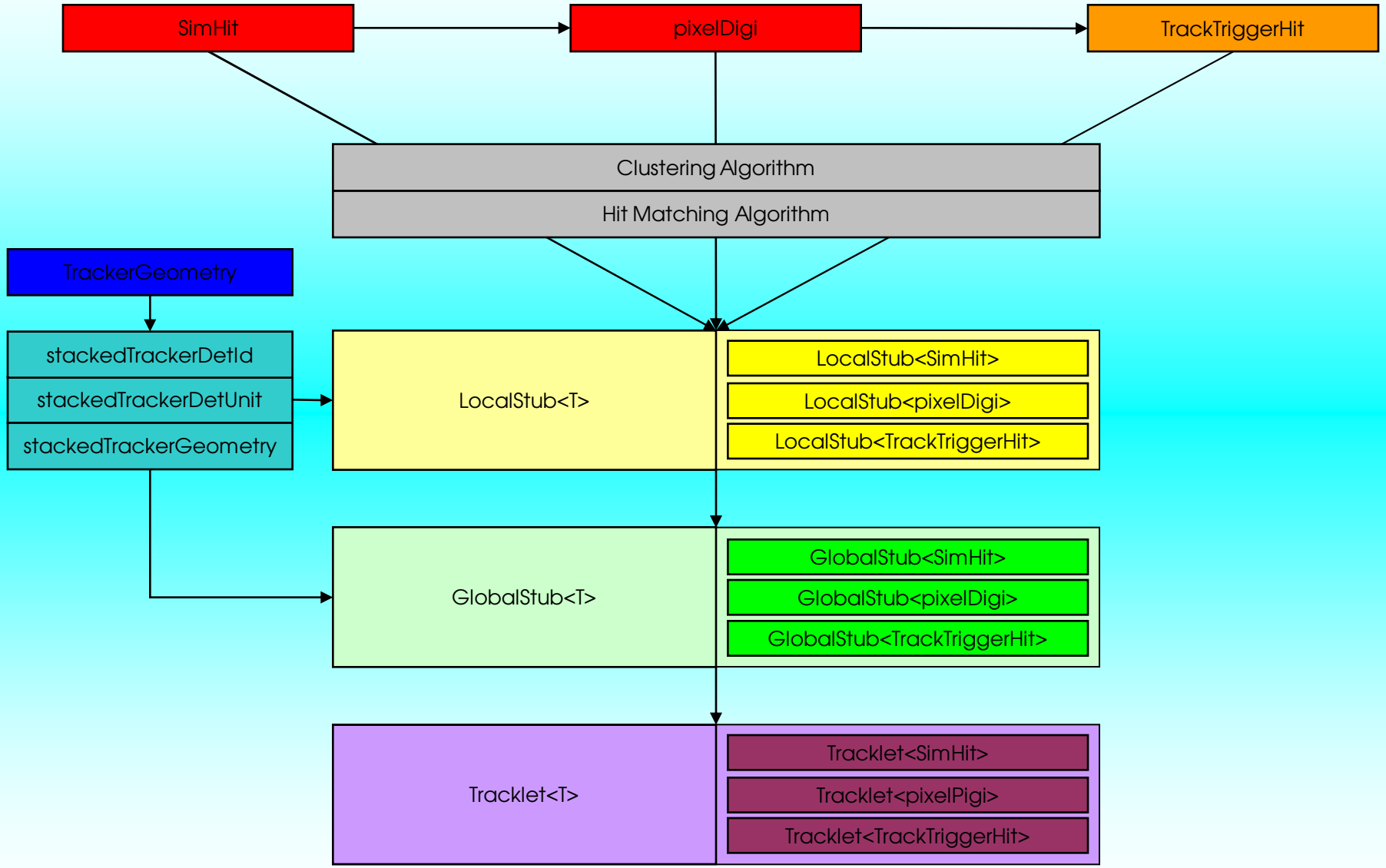
Simulation infrastructure

The strawman b family of upgrade geometries (of which the Long Barrel design is one) do not describe stacks per se, but rather a collection of individual detector element positioned as pairs.

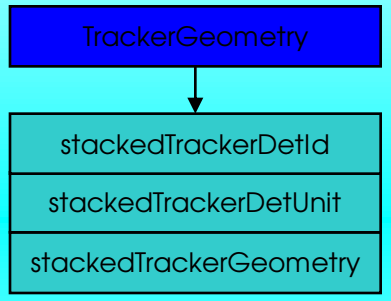
The standard framework makes no provision for handling such objects – a new type of detector object is required.

The standard framework also makes no provision for stubs or tracklets – new data formats are required.

Overview


SimHit
pixelDigi
TrackTriggerHit
Clustering Algorithm
Hit Matching Algorithm
TrackerGeometry
stackedTrackerDetId
stackedTrackerDetUnit
stackedTrackerGeometry
LocalStub<T>
LocalStub<SimHit>
LocalStub<pixelDigi>
LocalStub<TrackTriggerHit>
GlobalStub<T>
GlobalStub<SimHit>
GlobalStub<pixelDigi>
GlobalStub<TrackTriggerHit>
Tracklet<T>
Tracklet<SimHit>
Tracklet<pixelPigi>
Tracklet<TrackTriggerHit>

Stacked Tracker geometry utilities



StackedTrackerDetId: A DetId class uniquely encoding the subdetector (barrel/endcap), layer, iφ and iz.

StackedTrackerDetUnit: An object containing a sorted list of the DetIds of the stack members and having a StackedTrackerDetId.

StackedTrackerGeometry: Contains a list of stackedTrackerDetUnits and also provides various helper methods for association of StackedTrackerDetIds to StackedTrackerDetUnits.

Stacked tracker can be treated like any other piece of hardware in the detector

Stacked Tracker data formats (i)

From Geant *or* FastSim

MC truth position, energy loss, TOF, particle type, etc

∞-resolution

Also, the Track ID

From 'standard' CMSSW digitizer module

DetId, Row, Column, ADC count

finite-resolution

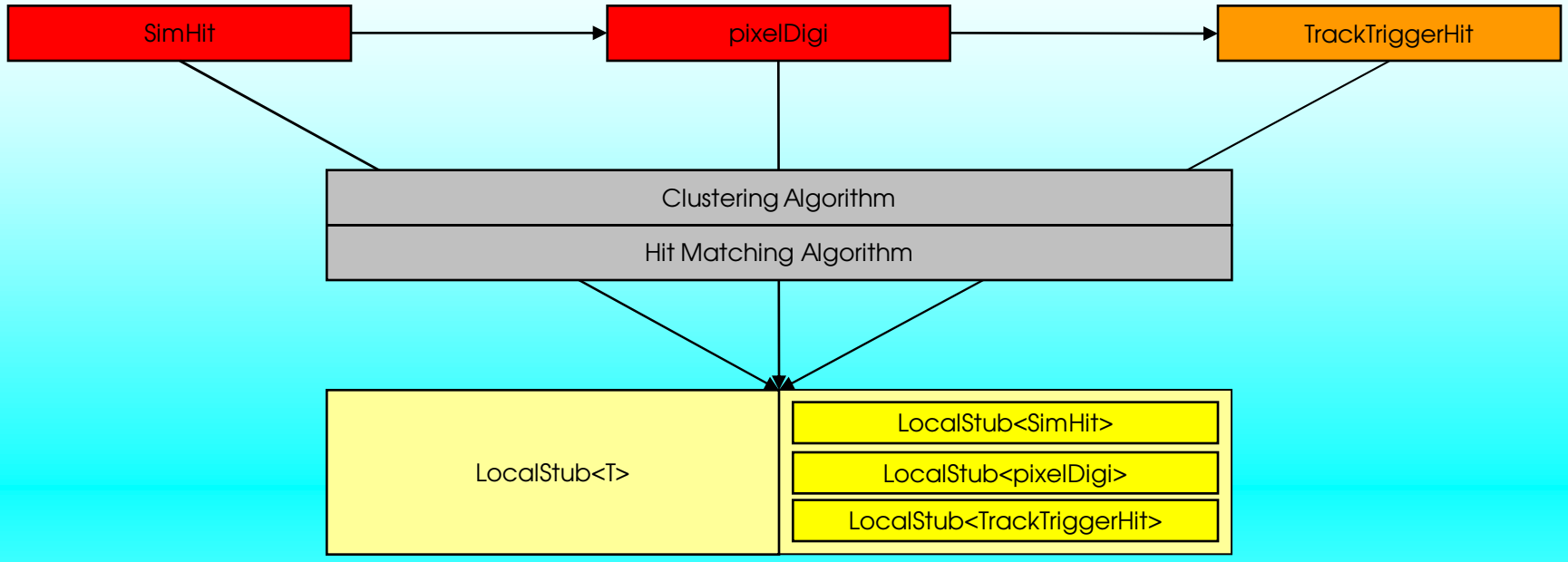
Custom data type for Track-Trigger Studies

DetId, Row, Column (assumed binary)

finite-resolution

For all practical purposes the "normal" user may ignore TrackTriggerHits

Stacked Tracker data formats (ii)

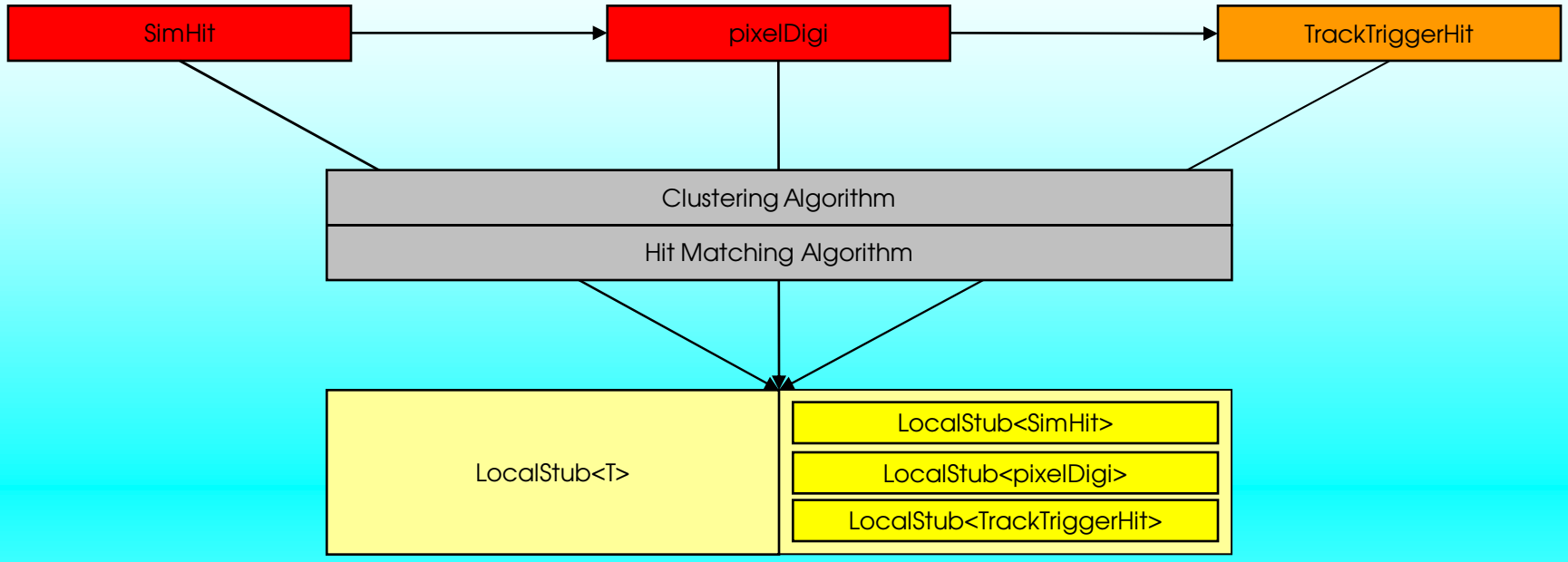


LocalStubs are sorted lists of hits† within an event, with an associated StackedTrackerDetId.

It is envisaged that this is the type of object that will be formed on-detector.

†technical note: the format actually uses persistent references to the framework hits

Clustering and Hit Matching

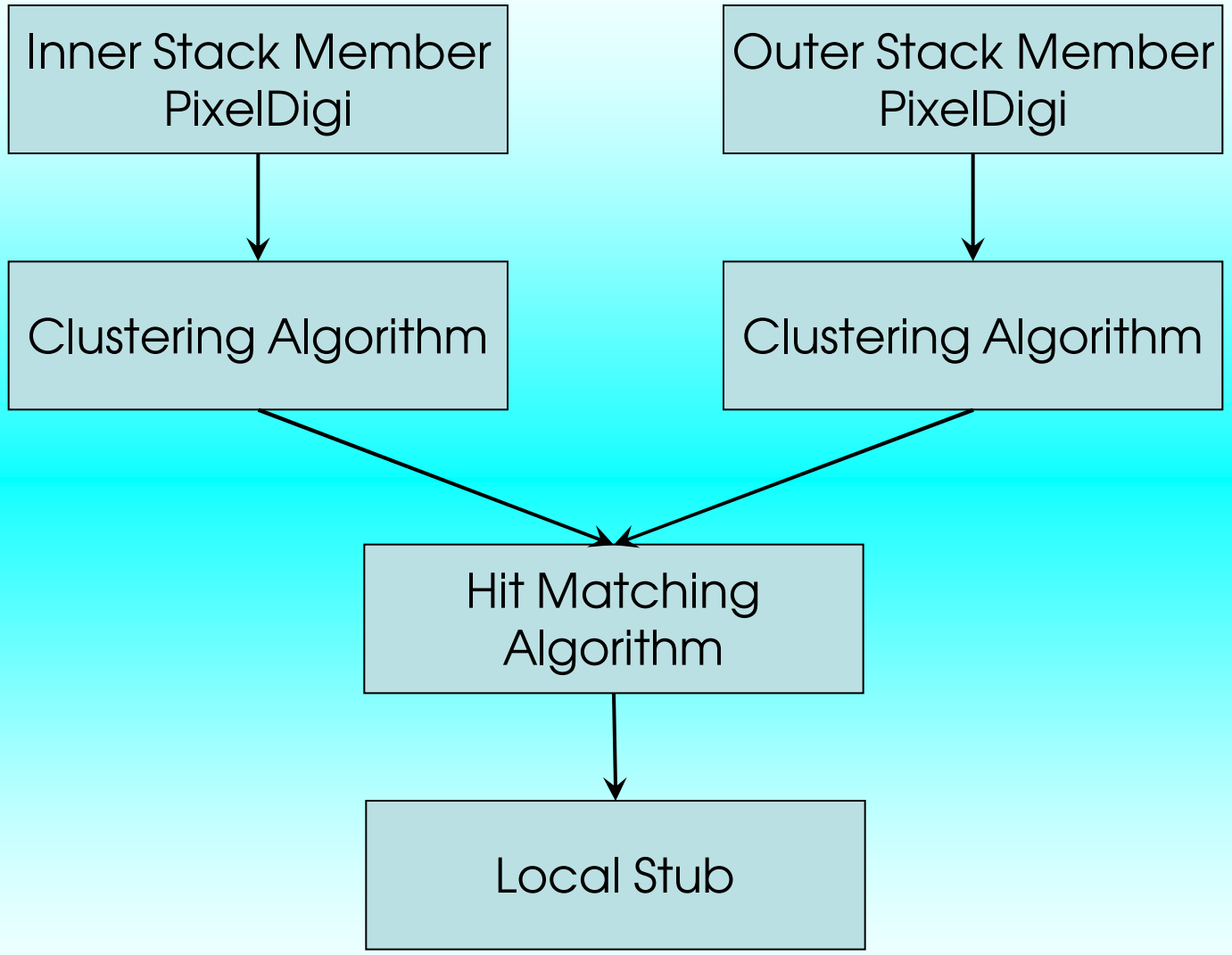


Two unknowns:

- What pixel clustering will be available on-detector?
- How will the hits will be matched into stubs?

As such the functionality is kept separate and introduced through EventSetup modules

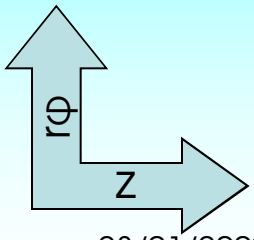
How it works


Inner Stack Member PixelDigi
Outer Stack Member PixelDigi
Clustering Algorithm
Clustering Algorithm
Hit Matching Algorithm
Local Stub

Clustering Algorithm A

- The "No Clustering" case – actually clusters of one pixel
- This is the default option
- Results in very large combinatorials and slow running for high PU

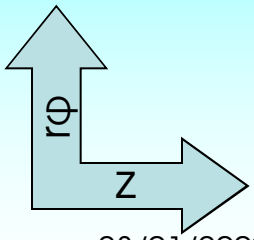
Counted here as 8 hits



Broadside Clustering Algorithm (no width cut)

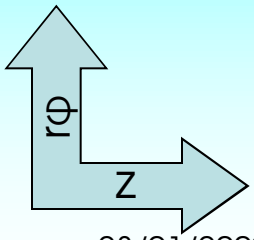
- Assume little charge sharing in z-direction due to long pixels
- Simplest clustering concept

Counted here as 3 hits



Broadside Clustering Algorithm (width cut)

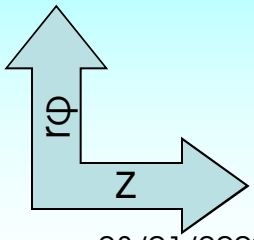
- Small modification made to previous algorithm to include the option of a configurable width cut.
- Default cut is to accept clusters ≤3 pixels wide



Counted here as 2 hits

2D Clustering Algorithm

- Considers region of 3x3 pixels with veto bit from neighbours
- Rejects clusters greater than 2 pixels wide



Counted here as 1 hit

Hit Matching Algorithms

Two basic types of hit matching:

- Global geometry
  - uses relative positions of hits in global frame (r,φ,z) to determine track momentum and interaction point and consistency cut applied
  - (Probably) not suitable on detector as requires coordinate transforms for every point
  - Best case scenario
- Local geometry
  - uses position of one hit in the local frame (row,col) and a predetermined acceptance window (Δrow, Δcol), and requires only that 2nd hit is within the window
  - Could be implemented by hardwired windows or LUTs

(Specifics of the differences in implementation are listed in spare slides at end of talk – if anyone wants more details just ask!)

Using Algorithms

Algorithms are selected in the users configuration file

Hit Matching Algorithms are labelled:

- HitMatchingAlgorithm_globalgeometry_xxx_ - Global Geometry (current default)
- HitMatchingAlgorithm_pixelray_xxx_ - Global Geometry
- HitMatchingAlgorithm_window_xxx_ - Local Geometry

Clustering Algorithms are labelled:

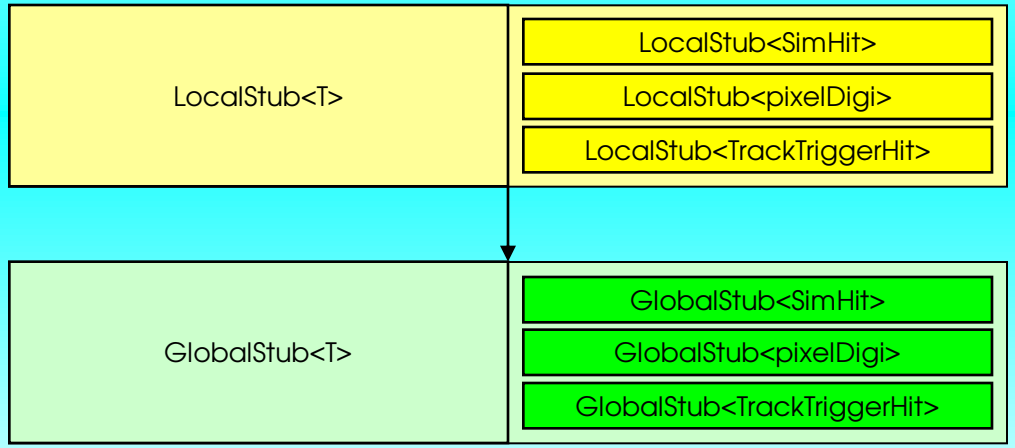
- ClusteringAlgorithm_a_xxx_ - (current default)
- ClusteringAlgorithm_broadside_xxx_
- ClusteringAlgorithm_2d_xxx_

Examples and instructions at
https://twiki.cern.ch/twiki/bin/view/CMS/TrackTriggerHitsAndStubs

Stacked Tracker data formats (iii)

GlobalStubs are geometric objects with a global position (the average global position of the two constituent hits) and global direction (the vector between the two constituent hits).

It is envisaged that this is the type of object that will be used in the level-1 trigger for association of hits between stacks.



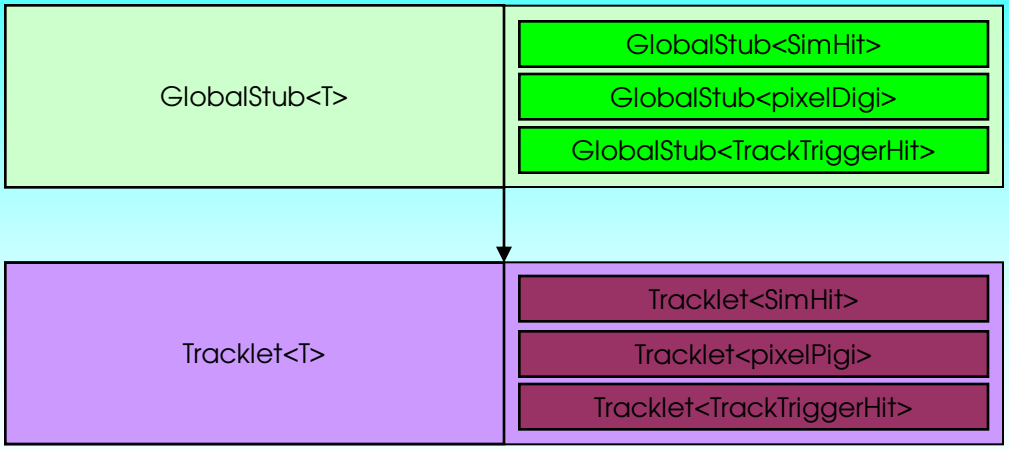
Currently they are produced one-for-one from the local stubs.

Stacked Tracker data formats (iv)

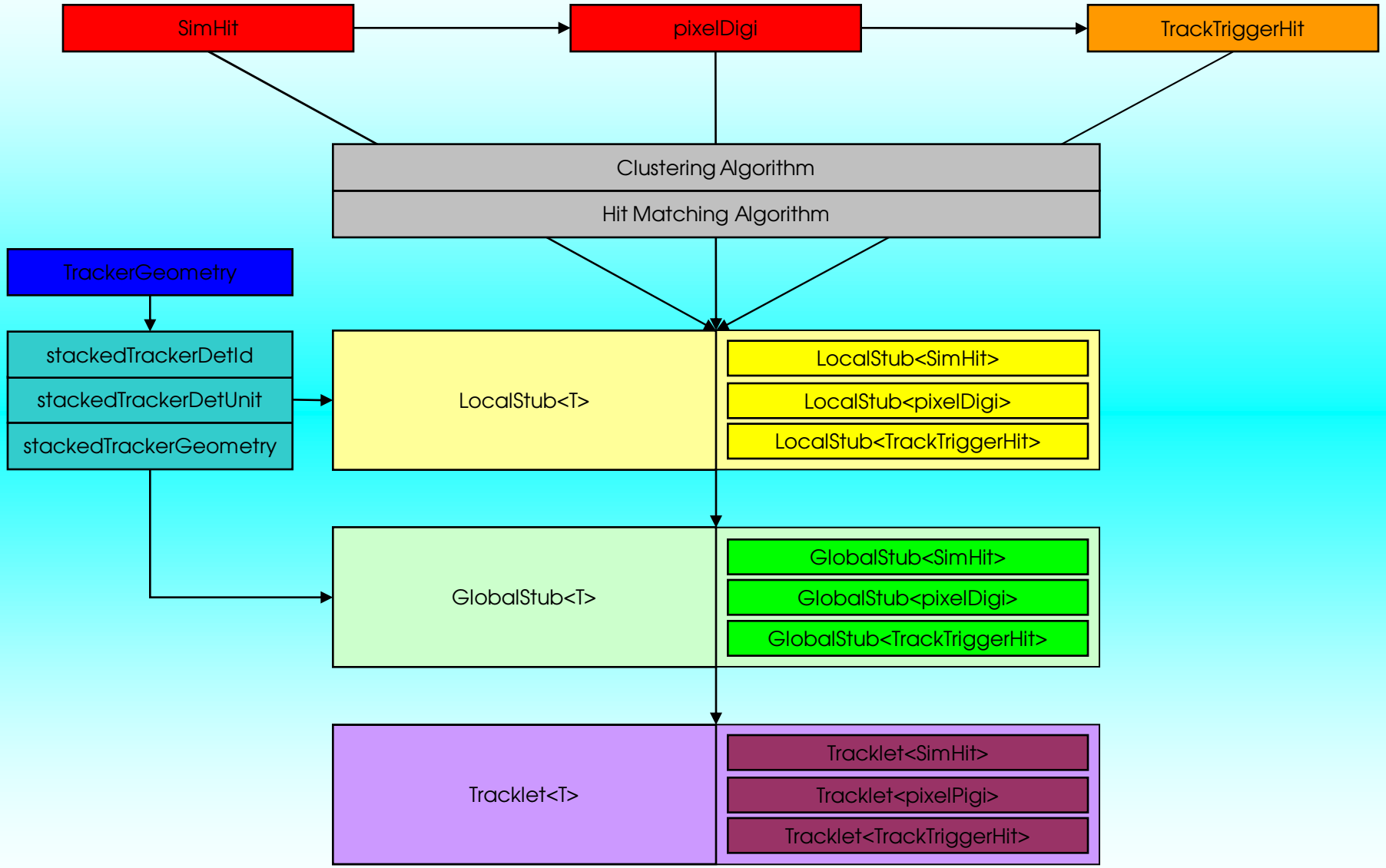
Tracklets are "track-like" objects which allow a P_T measurement

They are formed geometrically from global stubs by placing a cut on $\Delta\Phi$ based on a p_T threshold and a cut on the projected vertex position

Currently only performed between pairs of consecutive layers although framework allows for any number of stubs in a tracklet



Overview


SimHit
pixelDigi
TrackTriggerHit
Clustering Algorithm
Hit Matching Algorithm
TrackerGeometry
stackedTrackerDetId
stackedTrackerDetUnit
stackedTrackerGeometry
LocalStub<T>
LocalStub<SimHit>
LocalStub<pixelDigi>
LocalStub<TrackTriggerHit>
GlobalStub<T>
GlobalStub<SimHit>
GlobalStub<pixelDigi>
GlobalStub<TrackTriggerHit>
Tracklet<T>
Tracklet<SimHit>
Tracklet<pixelPigi>
Tracklet<TrackTriggerHit>

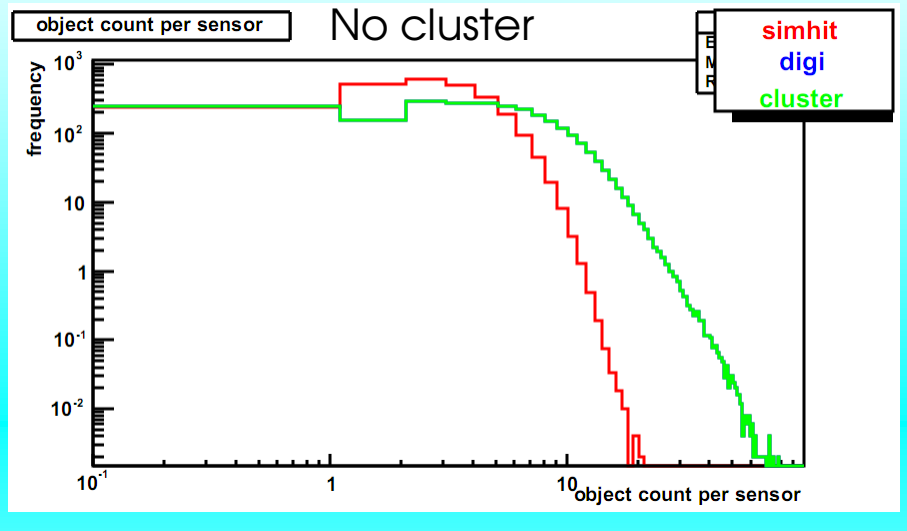
Summary

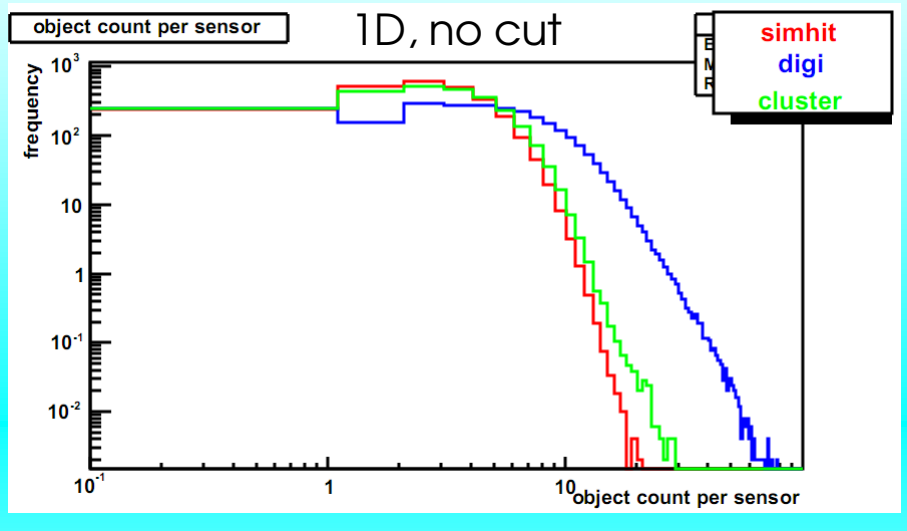
- A little terminology
- Simulation infrastructure
- Performance
- Conclusion

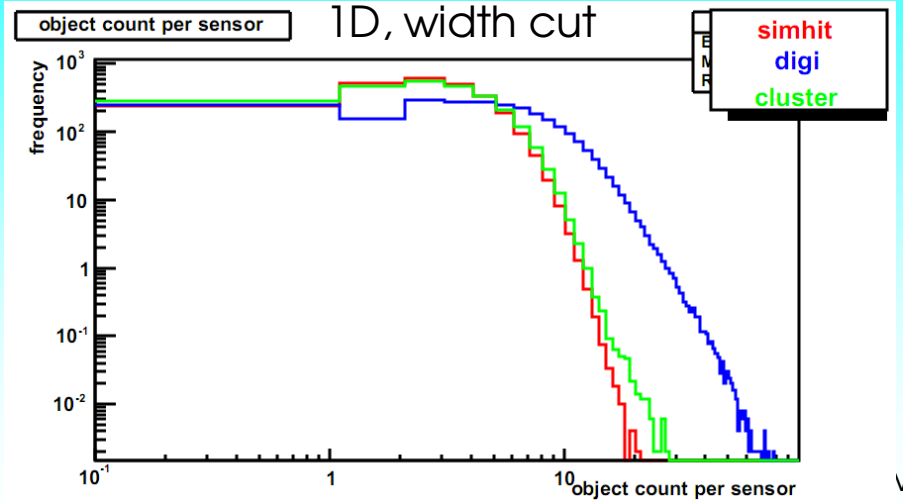
Clustering performance: rates

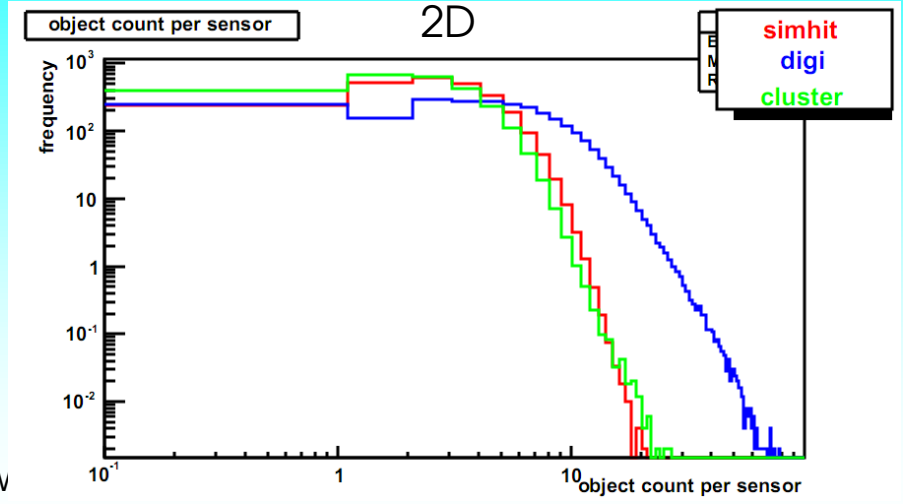
green is the line of interest - red and blue are for reference

Y-axis scale is per event for layer at r=25cm



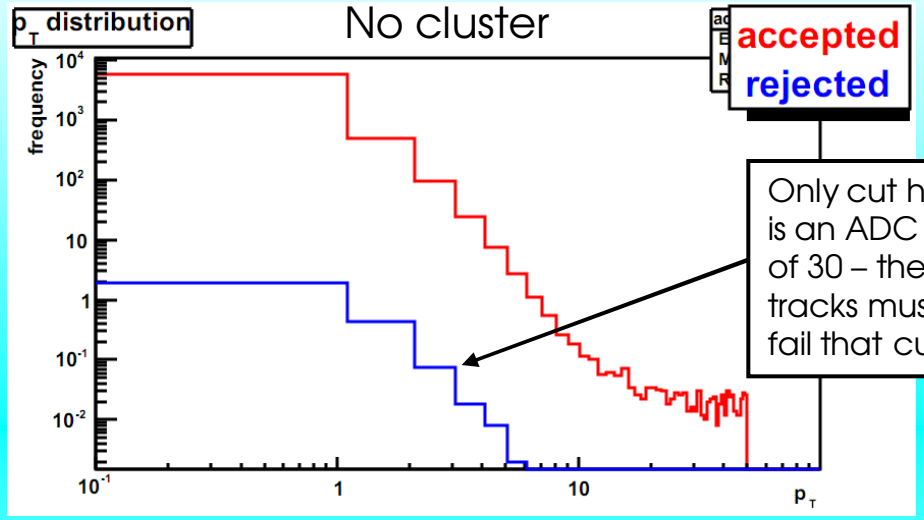




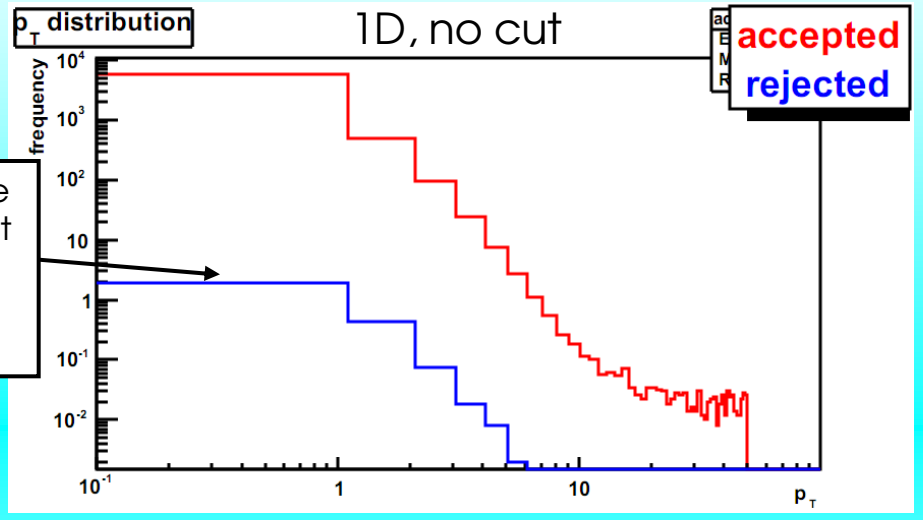


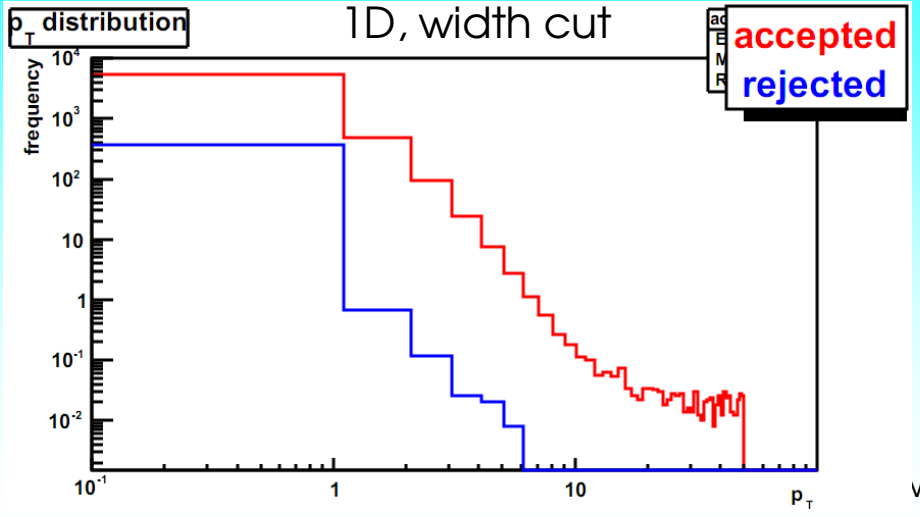
Clustering performance: P_T distribution of accepted and rejected MC tracks

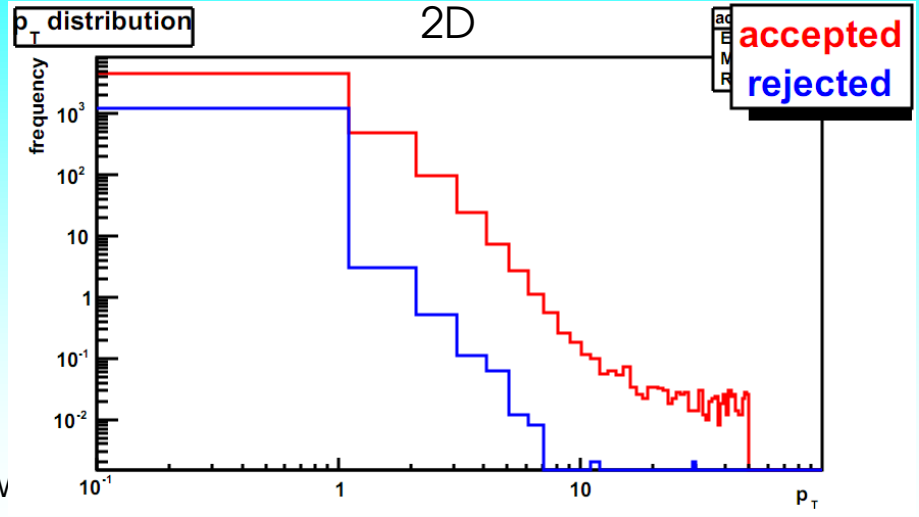
Y-axis scale is per event for layer at r=25cm



Only cut here is an ADC cut of 30 – these tracks must fail that cut

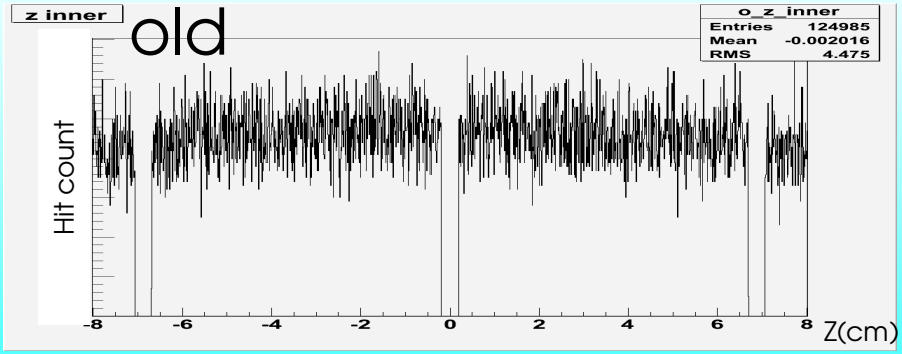


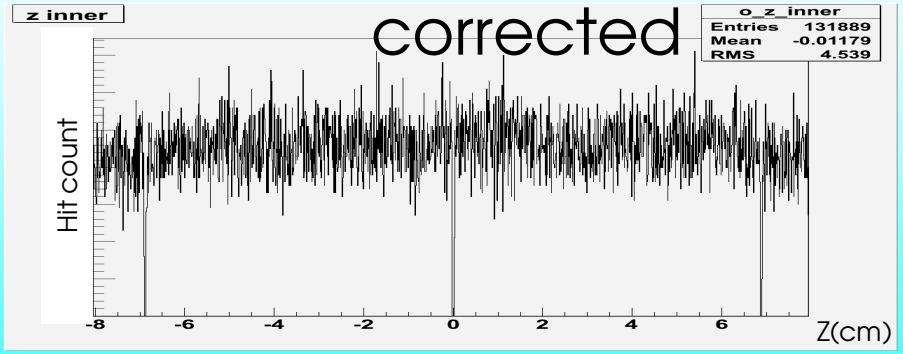




Performance

Warning: there are "gaps" in the standard geometry that limit efficiency!





CVS should be updated soon!

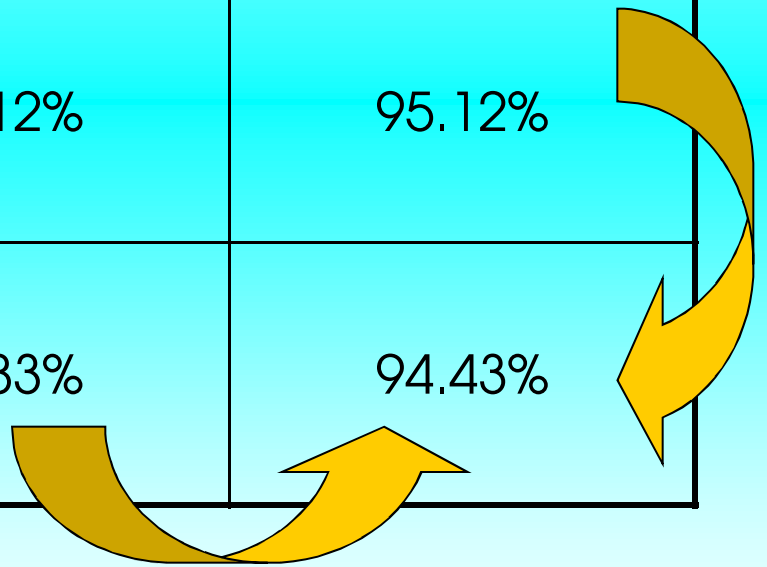
50GeV electron gun efficiency

Stack gap in z	standard (4mm)	corrected (500 μm)	corrected (800μm)
Digi Stub efficiency (globalgeometry hit matching, no clustering)	93.88%	96.91%	96.12%

Algorithm Performance

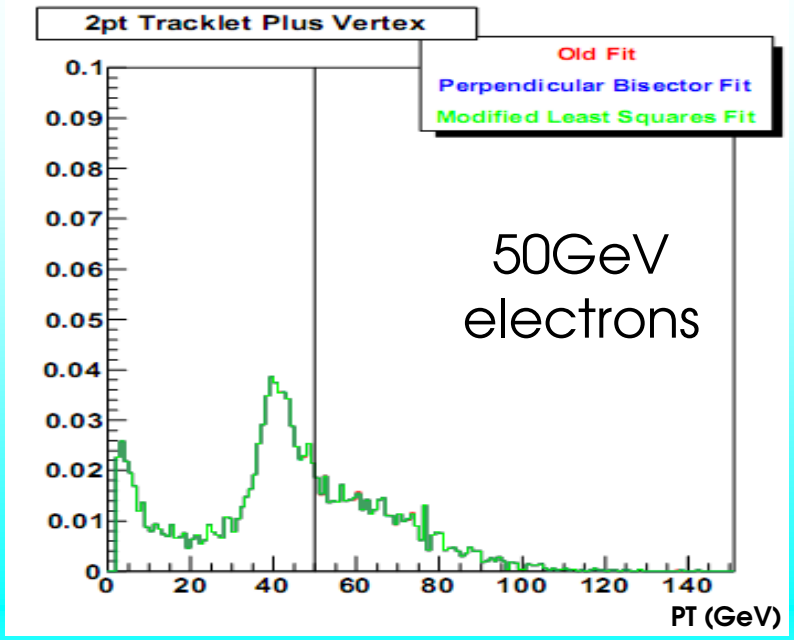
50GeV electron gun efficiency (800μm corrected geometry)

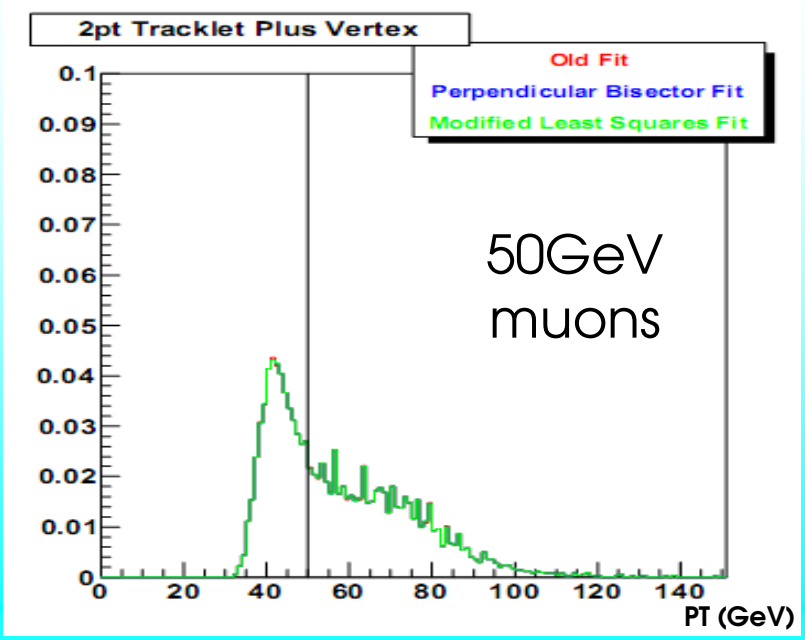
Clustering Algo \ Hit Matching Algo	global geometry	window
a (no clustering)	96.12%	95.12%
2d	95.83%	94.43%



Still trying to understand source of losses!!

Tracklet momentum resolution





- Resolution is poor due to close spacing of layers (using more tracker points improves this)
- Bremsstrahlung and Material Interactions cause problems for electron reconstruction – work ongoing to improve this!

Summary

- A little terminology
- Simulation infrastructure
- Performance
- Conclusion

Conclusion

- A comprehensive suite of tools exists for the modelling of a triggering tracker and treating it like a an other "standard" detector system
- A generic and highly configurable suite of tools exists to generate stubs and tracklets
- These tools are stable and being used by a large tracking trigger group
- There is an active Hypernews forum with many questions already answered and many people willing and able to answer any other questions
- Work is ongoing to optimise both design and algorithm performance

Full instructions for the geometries at
https://twiki.cern.ch/twiki/bin/view/CMS/ExampleStrawmanB
https://twiki.cern.ch/twiki/bin/view/CMS/ExampleLongBarrel
https://twiki.cern.ch/twiki/bin/view/CMS/ExampleHybridGeometry

The geometry utilities are described at
https://twiki.cern.ch/twiki/bin/view/CMS/SLHCStackedTrackerTools

Instructions on how to use the geometry utilities can be found at
https://twiki.cern.ch/twiki/bin/view/CMS/SLHCStackedTrackerToolsTutorial

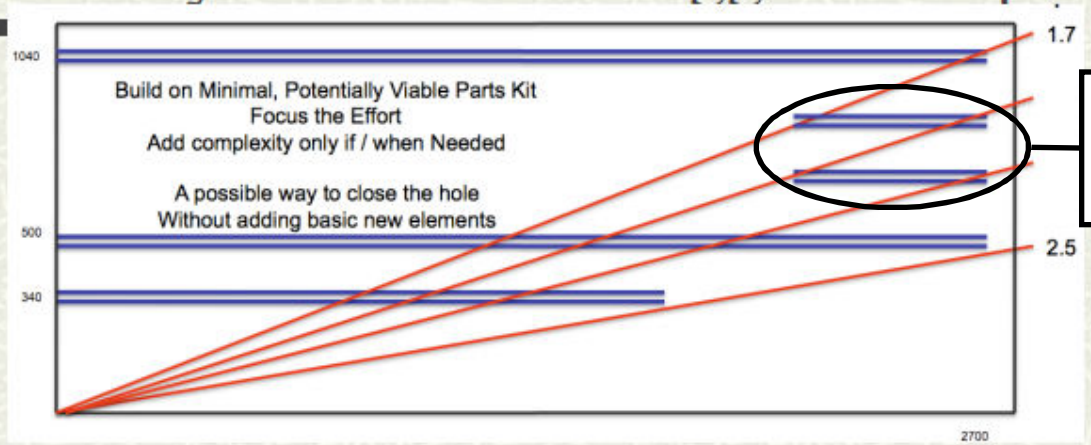
More info about data formats and instructions on how to generate stubs can be found at
https://twiki.cern.ch/twiki/bin/view/CMS/TrackTriggerHitsAndStubs

Spares

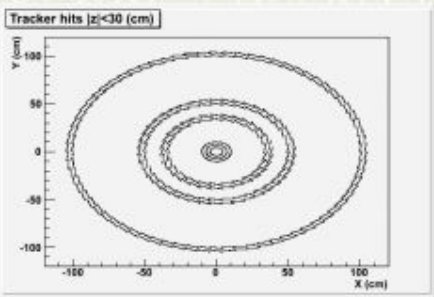
Current geometry model status

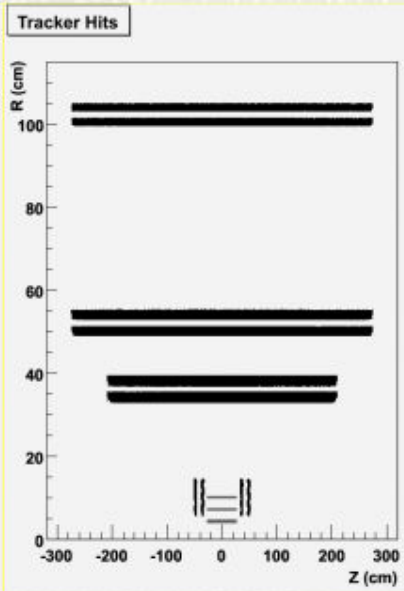
Baseline Geometry from Track Trigger Group

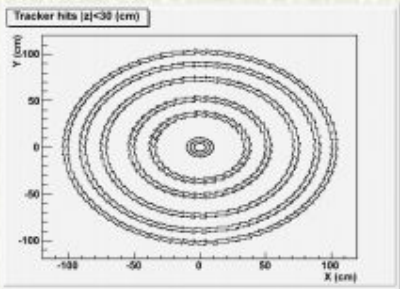
Start with StrawB
Remove TEC
Add in extra Layers
Extend Barrel Length

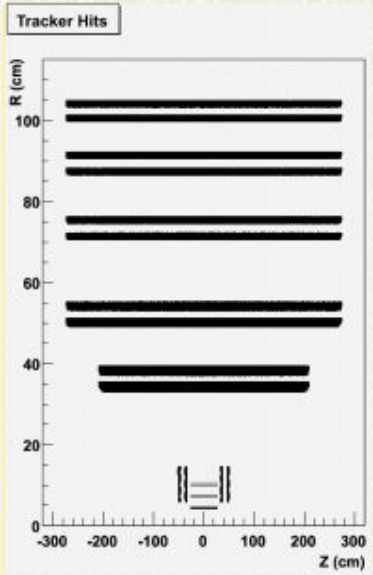


Short layers

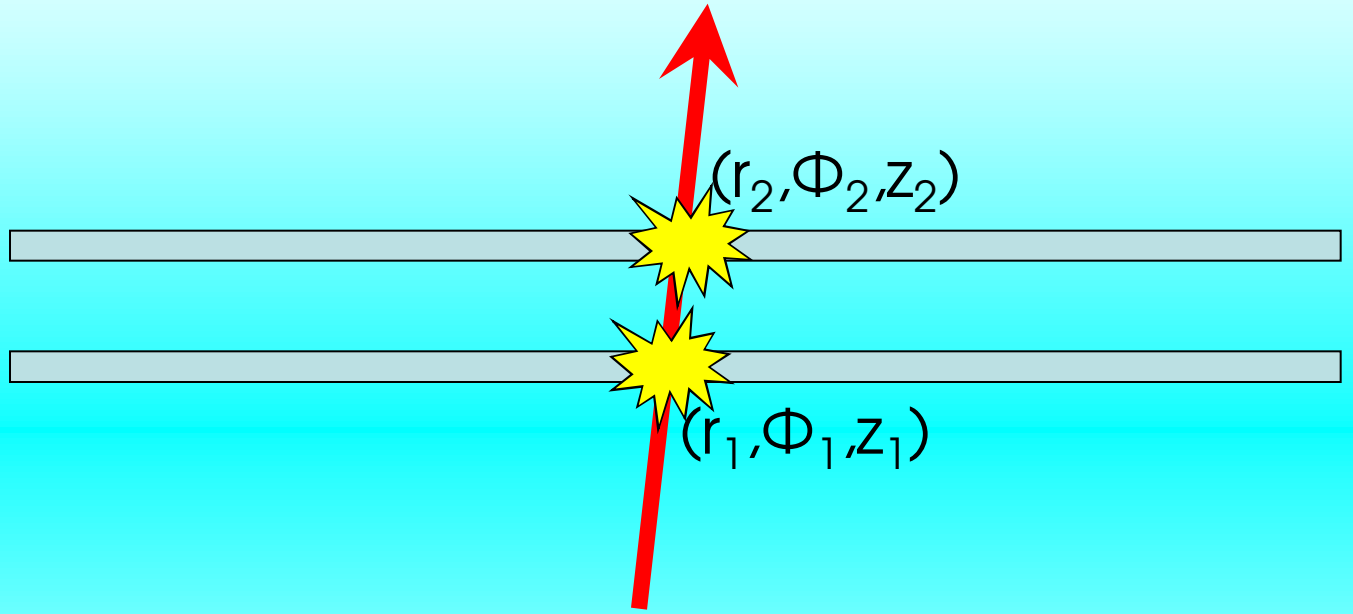








Global Geometry Algorithms



subtraction

(x_1,y_1) → (r_1,Φ_1,z_1) → Δz → compare with $\Delta z_{thresh.}(z_{IP},r_1,r_2)$

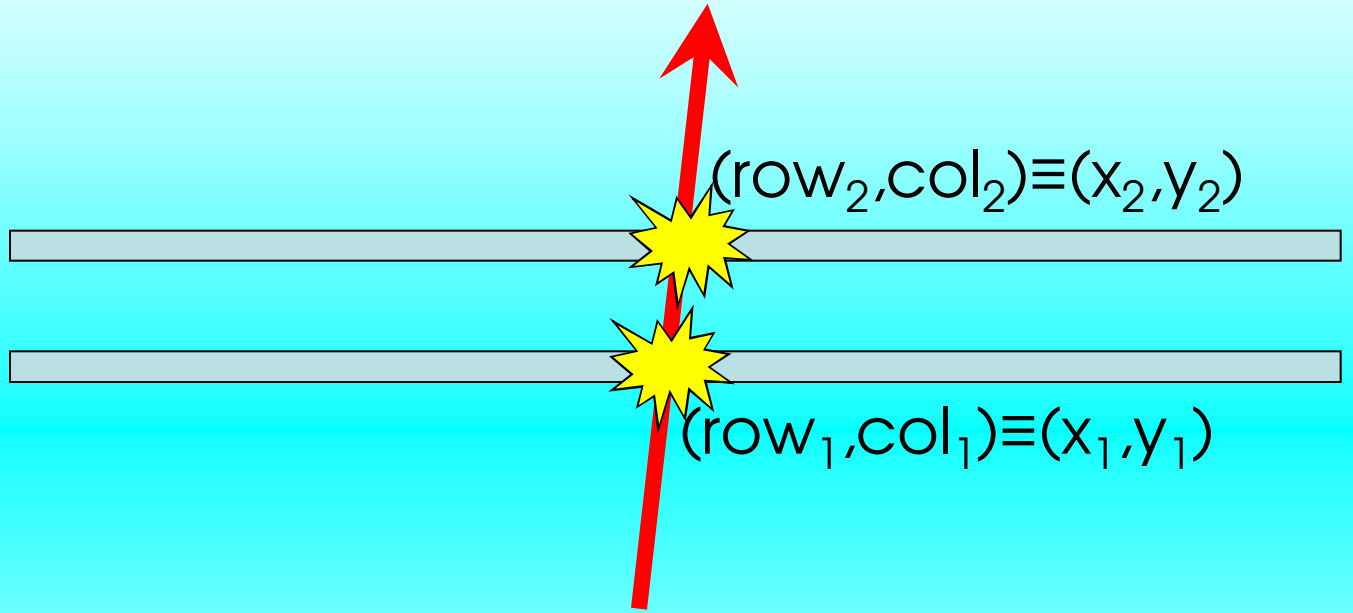
LUT

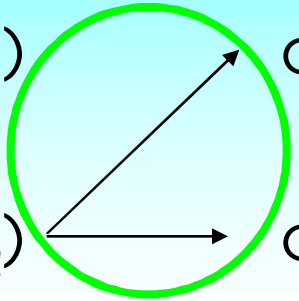
LUT

(x_2,y_2) → (r_2,Φ_2,z_2) → $\Delta\Phi$ → compare with $\Delta\Phi_{thresh.}(p_T^{cut},r_1,r_2)$

comparison

Local Geometry Algorithms





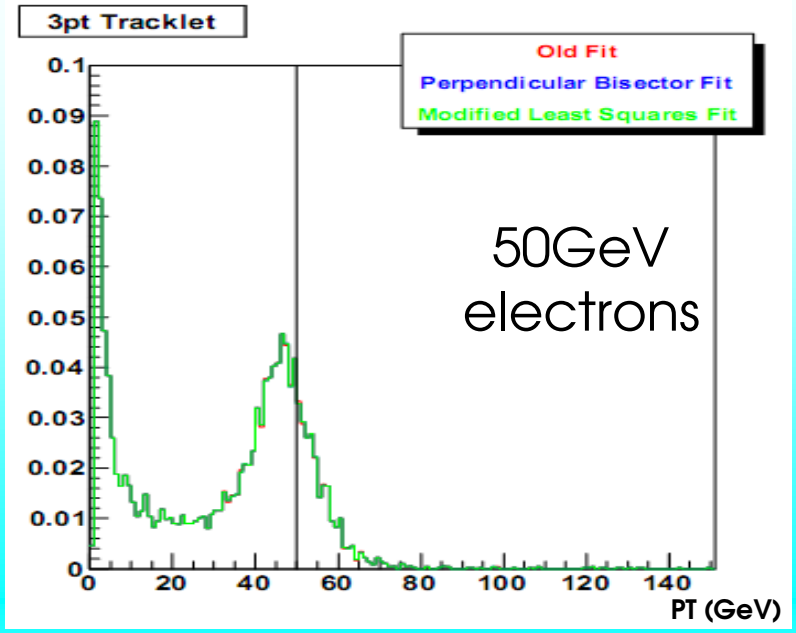
(row_1,col_1) compare with $col^{\pm}_{thresh.}(z_{IP},row_1,col_1)$

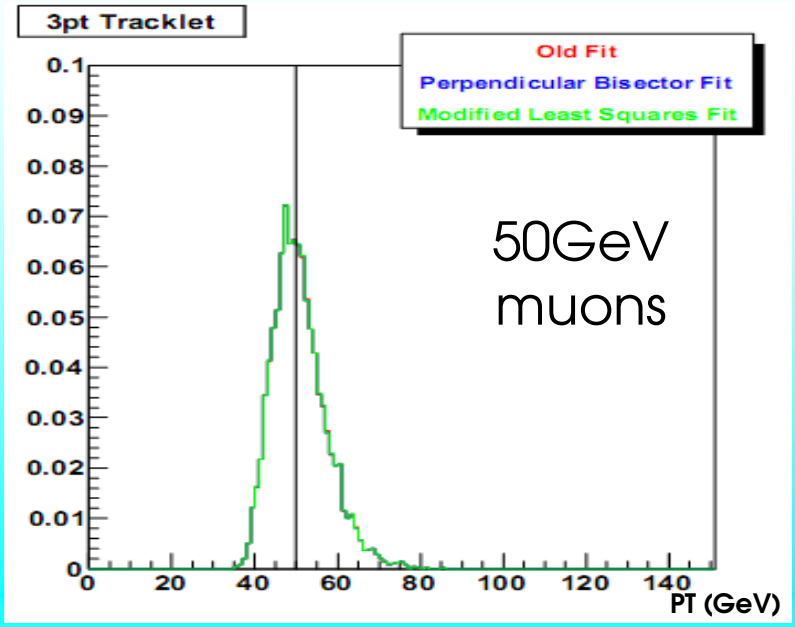
(row_2,col_2) compare with $row^{\pm}_{thresh.}(p_T^{cut},row_1,col_1)$

comparison

LUT or hardwired

Tracklet momentum resolution





Using three points in the tracker improves muon momentum resolution