Pixel Vertexing in 3D

Outline

- Definitions
- Samples studied
- Beam Position estimated using 3D pixel-vertexing
- Timing issues
- Conclusion and Outlook

Trigger Meeting
February 25, 2010

Suvadeep Bose (UNL)
Aaron Dominguez (UNL)

Vertex fitting

- Finding a set of vertexes using *Divisive Vertex Finder.*
- Each Vz is calculated by finding the weighted average of the tracks' z,

using the error from the z-estimation of PixelTracks fitting

- The set of vertexes is sorted in decreasing $\sum p_T^2$
- "Tagged" vertex has largest sum
- The logic used for vertex fitting:

$p_T = 0$ if $p_T < 2.5$ // Don't count tracks below 2.5 GeV

$= p_T$ if $2.5 < p_T < 10.0$

$= 10$ if $p_T > 10.0$

❖ Use the *Divisive Vertex Finder* Algorithm .

Data Sample and Event Selection

- Used Reco data:
- Run 124120 (2.3 TeV ; physcis bit ON for lumi blocks 1-59; Interacting BX = 51).
- **/MinimumBias/BeamCommissioning09-Feb9ReReco_v2/RECO**
- The vertexing is performed on an HLT path based on L1 technical bits:

```
NOT(Tech[36] OR Tech[37] OR Tech[38] OR Tech[39])(no beam halo)
AND (Tech[40] OR Tech[41]) (minimum bias)
AND NOT(Tech[42] XOR Tech[43]) (no single splash)
```

I re-reconstructed this data through the pixeltrack, pixelvertex path to ensure correct 3D pixelVertexing.

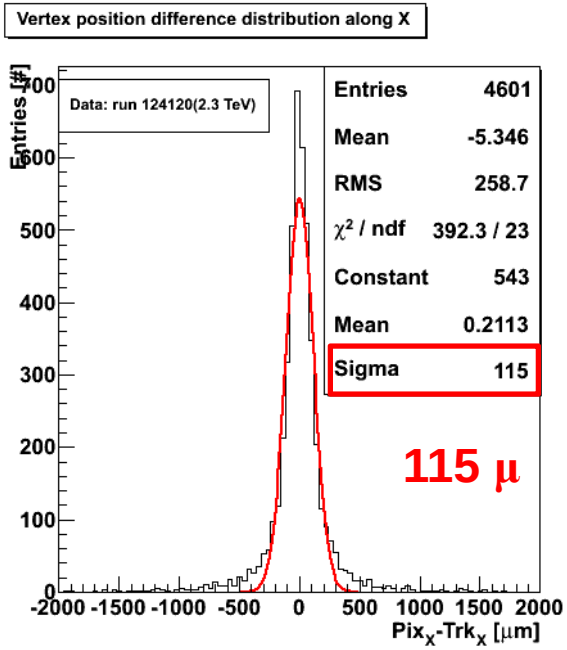
- Parameters in config file:
- MaxNormalizedChi2 = 100
- p_T > 1.0 GeV
- Max No. Of Vertices: 1 / 2
- Vertices are expected in a cylinder of volume 4 cm x30 cm
- ZSeparation = 0.05
- Vertex Finder Algo: DivisiveVertexFinder (1D)
- Comb. Kalman Fitter is used for 3D vertexing.

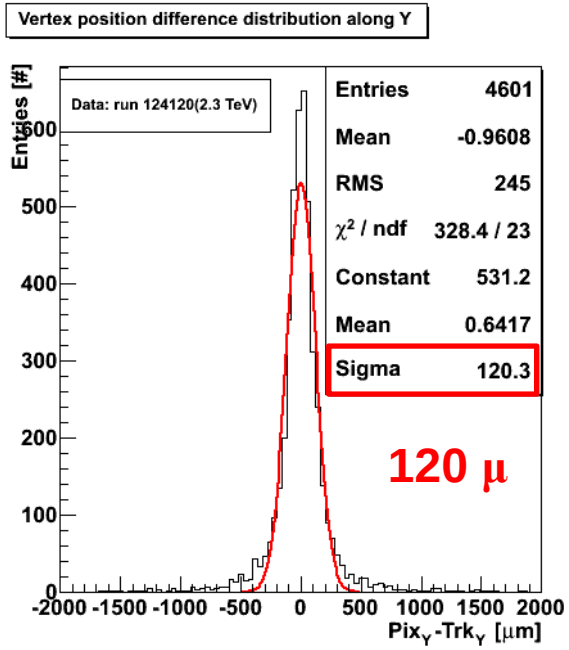
- Monte Carlo Sample:

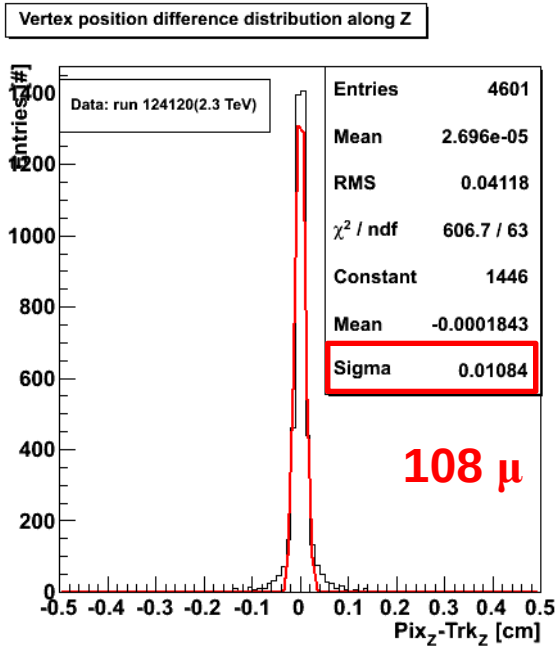
D6TMC START3X_V16D: /MinBias/Summer09-V16D_900GeV-v1/GEN-SIM-RECO

3D pixel vertexing (MinBias)

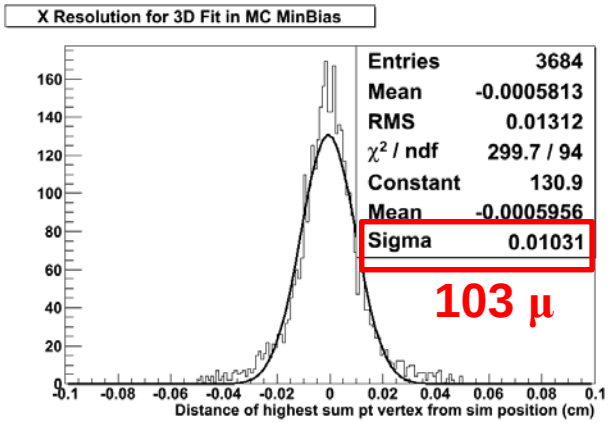
- Difference between the pixel vertex and the "offline" vertex – in Data:

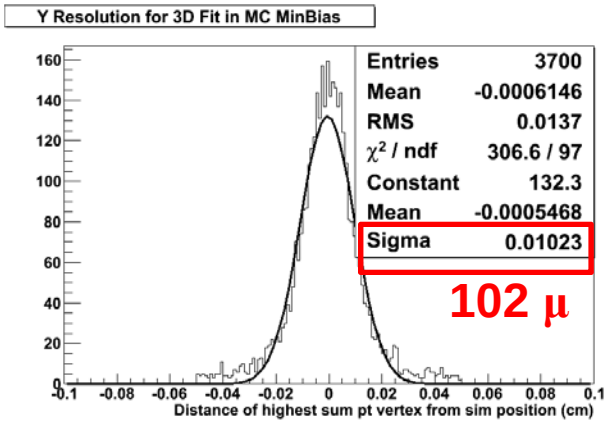


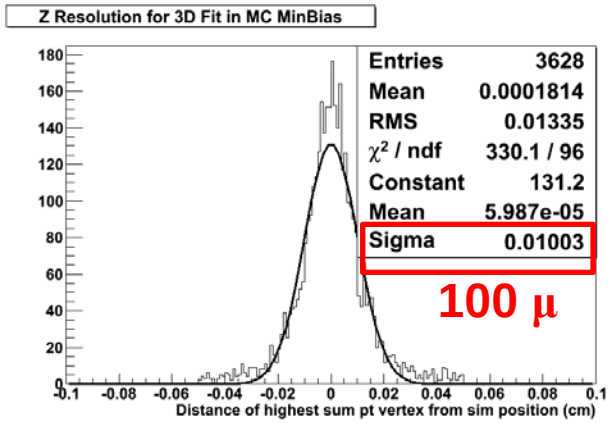




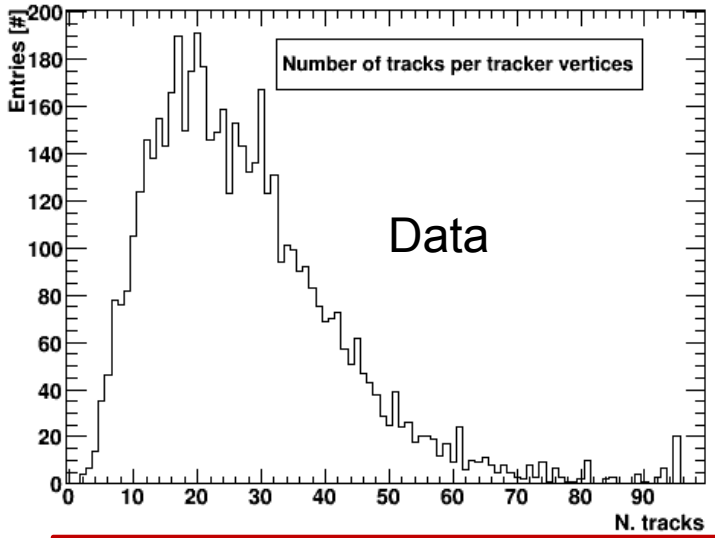
- Difference between the pixel vertex and MC Truth (SimTrack) – in Monte Carlo:

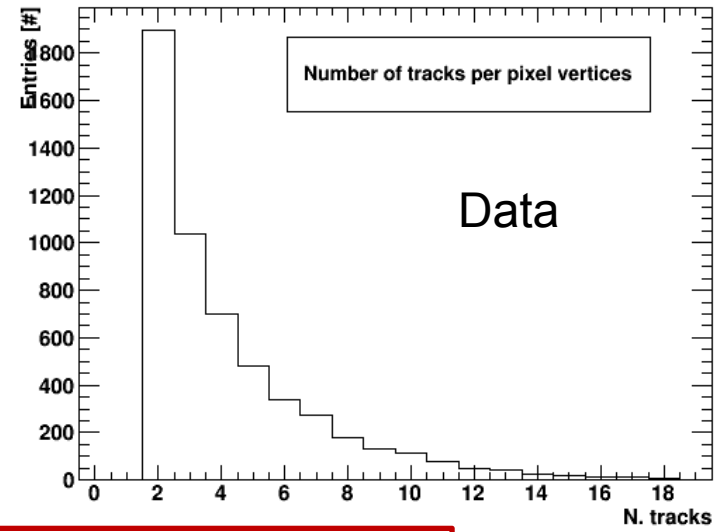




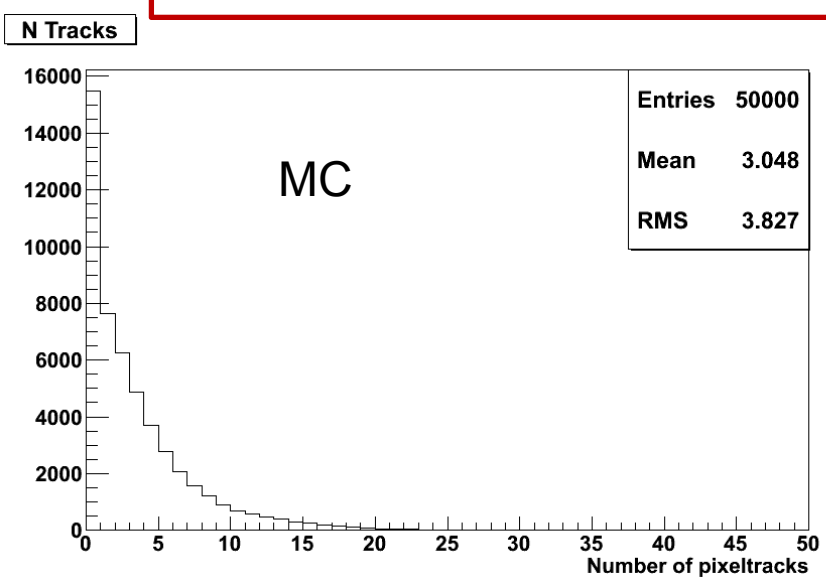


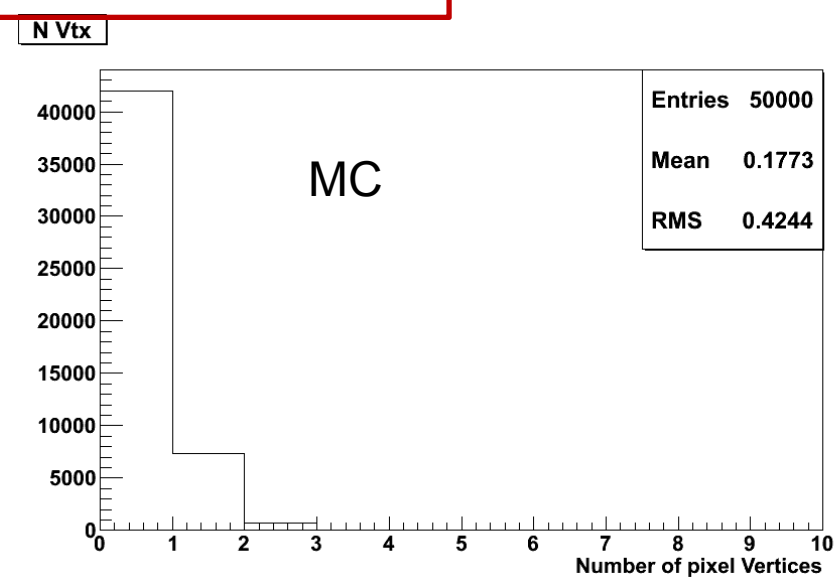
Number of Pixel Tracks and Vertices (MinBias)



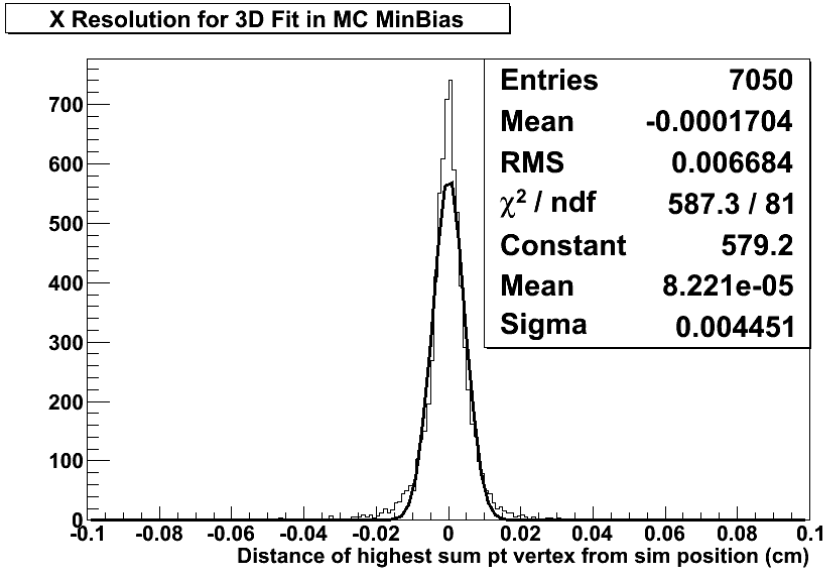


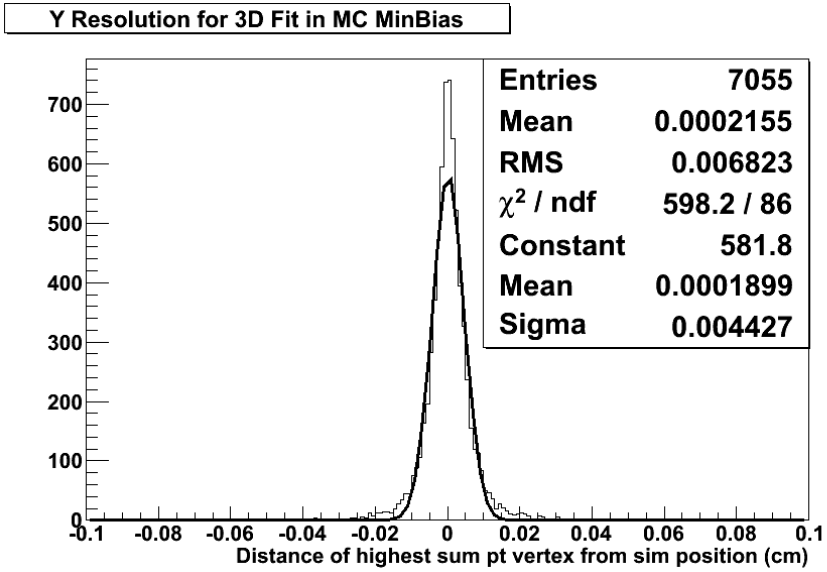
- p_T > 1 GeV
- Minimum number of Tracks for Vertex matching >2



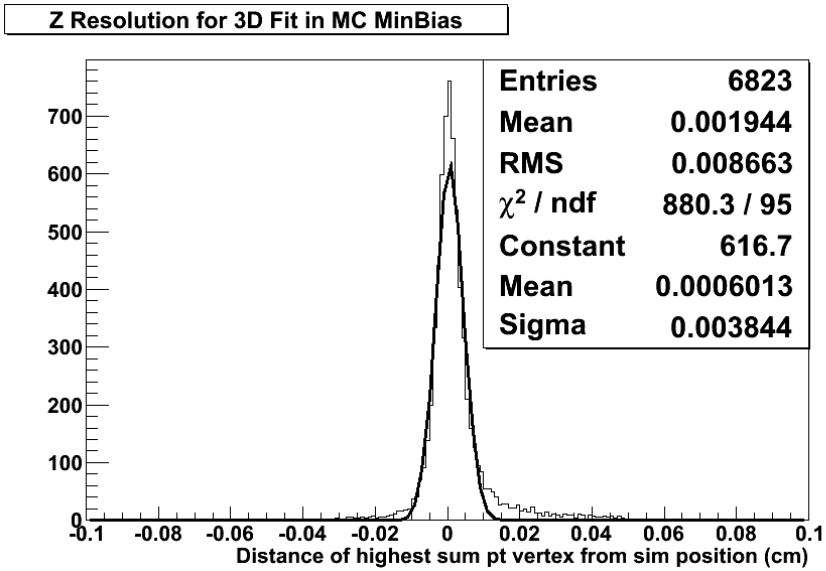


3D pixel vertexing (ttbar)





Resolution (Sigma) (micron)		
x	y	Z
44.5	44.3	38.4



Looking at ttbar samples - MC
with a beam line nicely centered at 0,0,0.

- Is it why resolutions are much better?

Timing

	Module	Time / event (CPU) (ms)	Times / event (Real) (ms)
Data (run 124120)	Pixel tracks	0.46	0.77
	Pixel Vertexes	0.08	0.09
MC (MinBias 900 GeV)	Pixel tracks	0.55	0.66
	Pixel Vertexes	0.11	0.12
MC (RelVal ttbar)	Pixel tracks	6.44	7.16
	Pixel Vertexes	1.65	1.65

- $p_T > 1$ GeV
- Minimum number of Tracks for Vertex matching >2

Conclusion and Outlook

- The resolution for ttbar sample ~ 40 micron
- For MinBias samples – resolutions ~ 100 micron
- Data and MC (MinBias) show comparable resolutions.
- Timing is very fast in Divisive method.
- Can be used in HLT path in StartUp.

(minbias)	CPU (ms)	Real (ms)
Pixel track	0.5	0.6
Pixel Vertex	0.1	0.1

❖ OUTLOOK:

- Currently one-pass Kalman Filter is being used. Needs to investigate whether an offset in beam line requires special care (as well as the track collection). Need to look deeper into other Kalman Vertex Fitter that uses the beam line as an estimate for the starting point.
- The bad MC resolution in MinBias samples may be because of poor accuracy by including tracks in the vertex which don't belong!
- Try tuning the parameters (p_T, no. of tracks).

Back Up (Divisive method)

- Start from a unique cluster of tracks and iteratively discard not compatible tracks
- recover discarded tracks to make another cluster
- iteration stops when the number of discarded tracks is less than the minimal number required to make a vertex
- The luminosity region is initially split in a set of sub-regions.

- Parameters:
  - Minimal number of track in a vertex (D=3)
  - Tolerance of track-vertex distance (D=5σ)
  - Maximal allowed distance between two tracks in a same region (D=0.05 cm)