BeamCal electron reconstruction

Aura Rosca

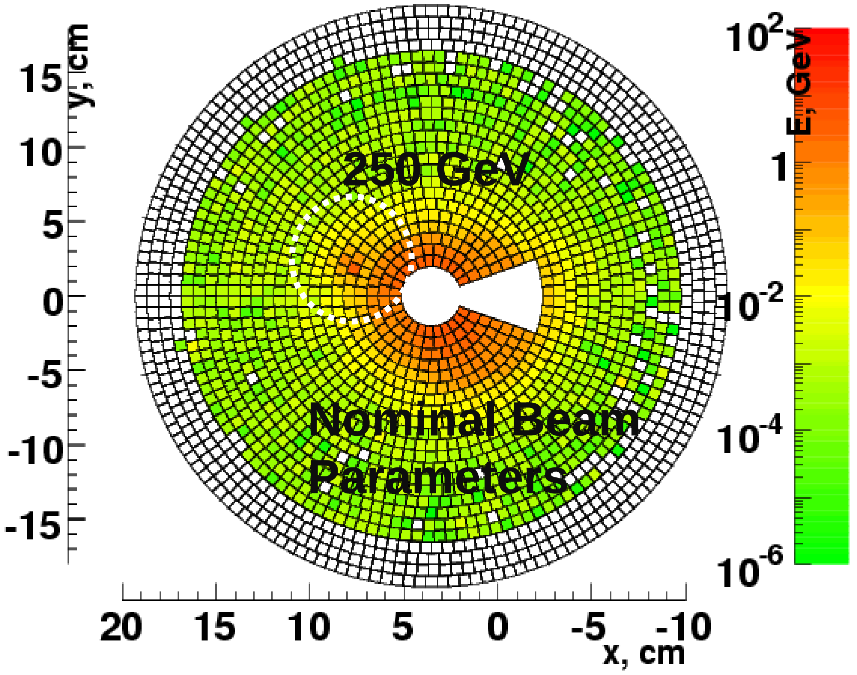
6th April 2011

Short reminder: Cluster reconstruction algorithm in BeamCal

(Developed by Olga N. and Wolfgang L.)

1. Background subtraction procedure

- Calculate average and rms of the energy deposition of the background in each pad of the BeamCal, from 10 BX;
- Superimpose 1 BX background + 1 high energy electron;
- Subtract the value of the background average from the superposition;

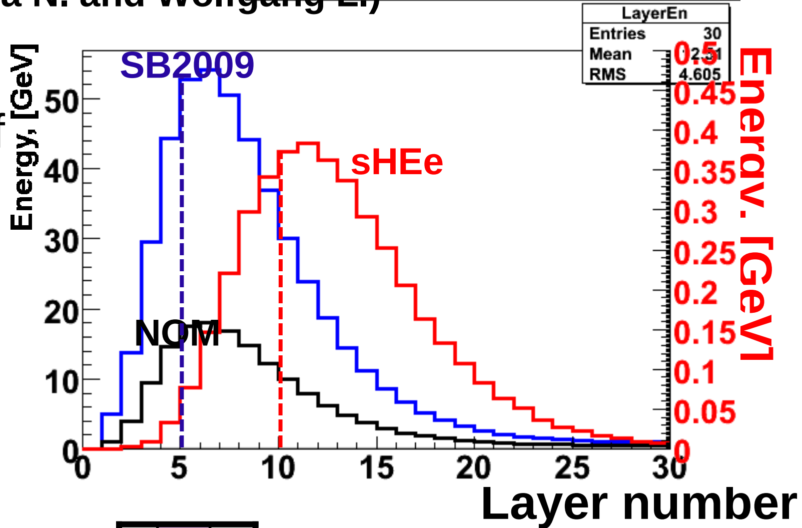


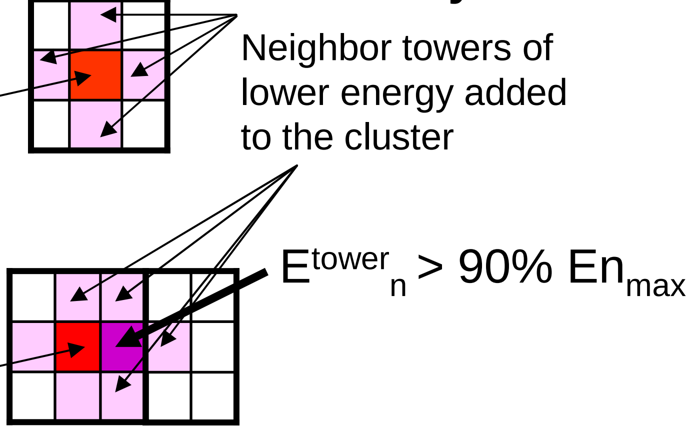
Short reminder: Cluster reconstruction algorithm in BeamCal

2. Cluster search

(Developed by Olga N. and Wolfgang L.)

- Identify towers after the 5-th layer as chains of 10 consecutive fired pads;
- Search for the tower with the maximum deposited energy;
- Add neighbor towers, in a 3×3 matrix around the tower with the highest energy;
- If such a neighbor tower has an energy larger than 90% of the energy of the central tower, En_{max} add this tower neighbors as well;
- Output: energy of the cluster, corresponding ring number, pad number.
- SW used for efficiency studies in FW BeamCal



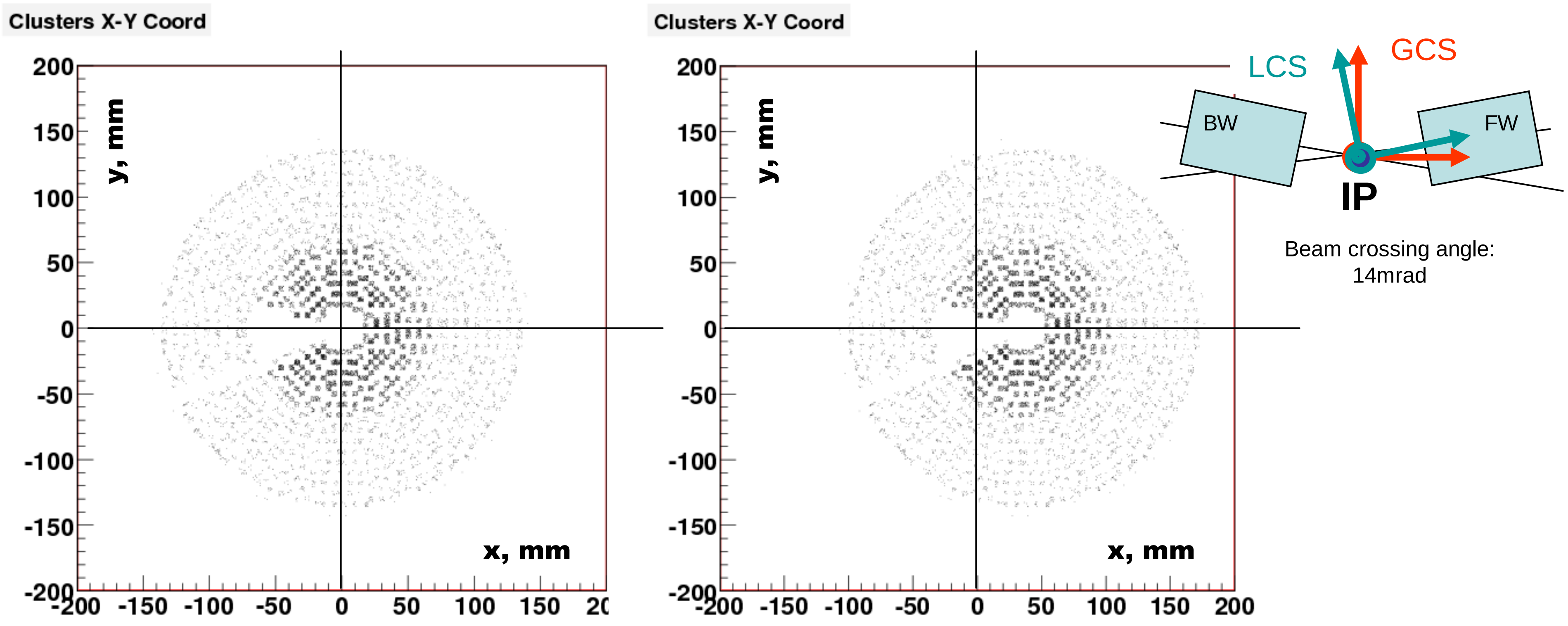

Neighbor towers of lower energy added to the cluster
E^{tower}_{n} > 90% En_{max}
En_{max}

Standalone SW reconstructed clusters

X-Y Coordinates of Clusters in the FW BeamCal

BeamCal Local CS (LCS)

Global ILD CS (GCS)



Beam crossing angle:
14mrad

Implementation into central reconstruction

- Jenny List proposed the integration of the standalone BeamCal reconstruction software into the central reconstruction framework through a two step process:
  - a simulation of the beamstrahlung background by cell-by-cell Poisson fluctuations of the average occupancy taken from a background map produced previously.
  - a redesign of the reconstruction code to read the BeamCal collection of hits instead of root files and produce two new collections, a cluster collection and a reconstructed particle collection, respectively;
- We have started with the second step;
- Step one still to be developed in the future,

Redesign of the reconstruction code

- Develop a BeamCal reconstruction class, BCalReconstruction:
  - Destructor: ~BCalReconstruction()
  - Functions:
    - RecCorr GetReconstrCoordinates (int number_layers, int number_rings, int number_pads[], CellType ***info_detector);
      - typedef struct {

```
   int side; // 0,1,-1 -> no, FW, BW reconstruction
  double RecEne, ErrEne, CoordX, CoordY, CoordZ, RecRad, RecPhi;
} RecCorr;
```

  - Protected member functions:
    - int** SearchTowers (int the_Chains[maxrings][maxphis][maxlayers]);
    - RecCorr SearchClustersFW (CellType ***info_detector);
    - RecCorr SearchClustersBW (CellType ***info_detector);
    - double GetEnergyCalib (double energy);
    - double GetEnergyErr (int ring, int pad);
    - double GetCoordRotX (int ring, int pad, float IP, float angle);
    - double GetCoordY (int ring, int pad);
    - double GetCoordRotZ (int ring, int pad, float IP, float angle);
    - void Free2DArray (int **p2DArray), void Free3DArray (CellType ***p3DArray);

New Marlin processor

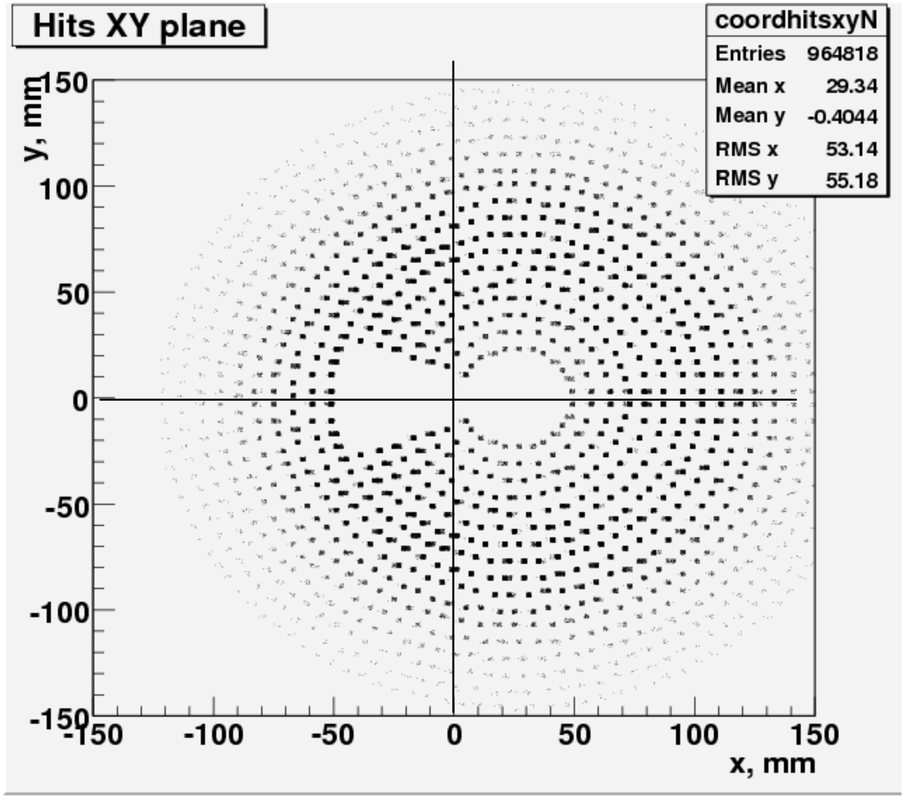
- Develop a Marlin processor, BCalReco, to perform the following tasks:
  - read the BeamCal collection of hits and put the information into a 3D dynamic array of structures, CellType ***info_detector:

    -
      ```
      typedef struct {
            double sRin,sRout,sZstart,sZend,sSphi,sDphi,sEdepNeg,sEdepPos;
            int sPos[3];
      } CellType;
      ```
  - call the reconstruction code (linked to the processor as a static library, libbcreco.a):
    - `bcal_reco = new BCalReconstruction();`
    - `bcal_electron = bcal_reco->GetReconstrCoordinates(nLayers,nRings,nbPhis,cells);`
  - output the relevant collections (clusters, reconstructed particles) – task not yet done.

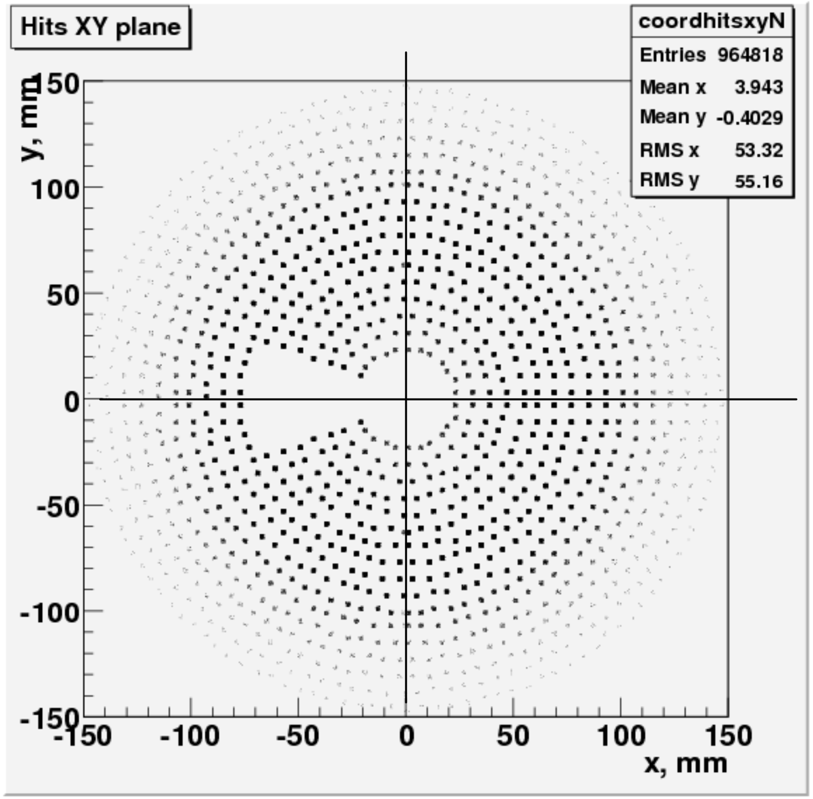
Hits in the BW BeamCal

X-Y Coordinates of the hits in the BW BeamCal

Global ILD CS



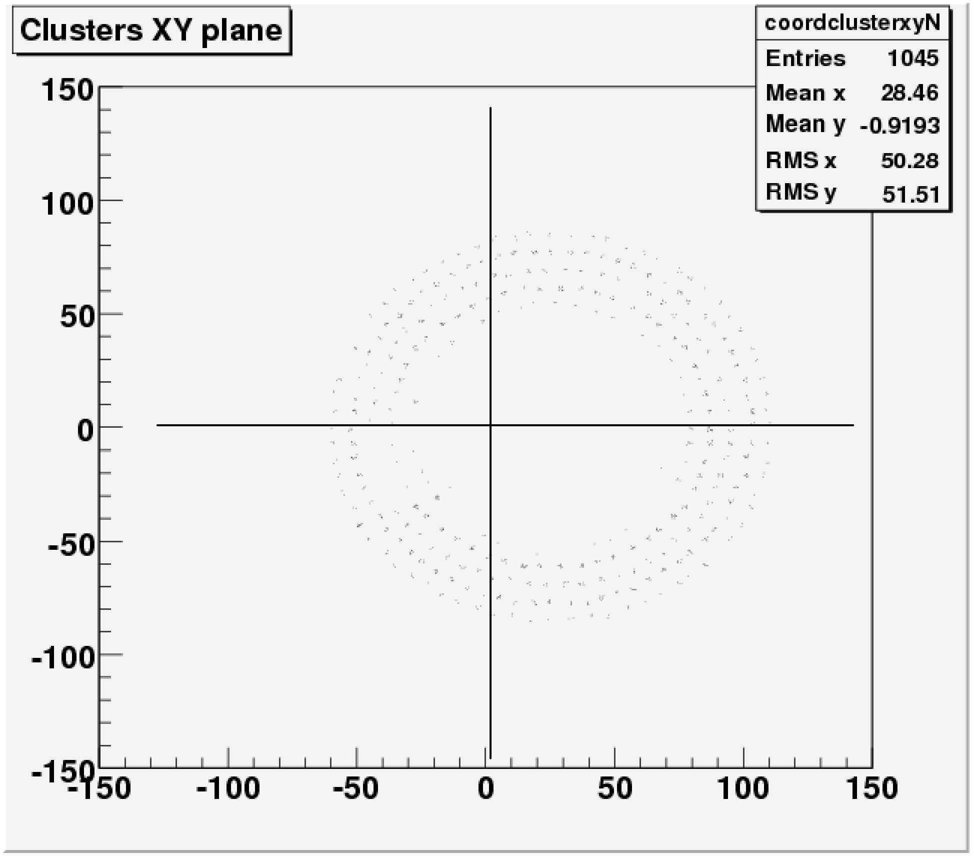
BeamCal Local CS



Clusters in the BW BeamCal

X-Y Coordinates of Clusters in the BW BeamCal

Global ILD CS



Future work still to be done

- Currently the reconstruction code assumes a cluster is found either in the FW BeamCal, or in the BW BeamCal respectively. The case where two clusters are found, one in the FW and another in the BW BeamCals should be next implemented.

- Also, the case when more than one single cluster in each BeamCal is found should be foreseen.

- Convert the output of the reconstruction code into the relevant collections, inside the processor.

- Test the SW with physics events, for now only tests with single electrons sent either to the FW or BW Beamcal have been done.

- Generate documentation from the source code.

Summary

- Electron reconstruction algorithm for BeamCal was redesigned to be included into the central reconstruction software
- New Marlin processor exists, BCalReco
- Several issues to be addressed in the near future: simulation of the background, production of the relevant output collections.