BeamCalClustering Implementation and Testing

André Sailer, Andrei Sapronov

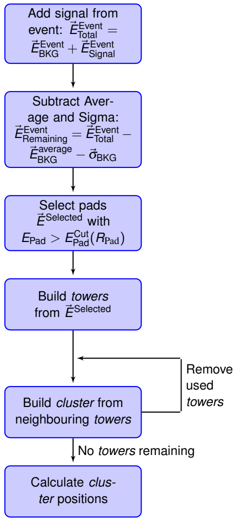
CERN-PH-LCD

ILD Software Phonemeeting
June 10, 2015

BeamCal Electron Clustering Algorithm

General Idea of Clustering

1. Energy in the BeamCal Pads is $\vec{E}^{\text{Event}}_{\text{Total}} = \vec{E}^{\text{Event}}_{\text{Signal}} + \vec{E}^{\text{Event}}_{\text{BKG}}$
2. Subtract the average background plus one standard deviation for each cell $\vec{E}^{\text{Event}}_{\text{Remaining}} = \vec{E}^{\text{Event}}_{\text{Total}} - (\vec{E}^{\text{average}}_{\text{BKG}} + \vec{\sigma}_{\text{BKG}})$
3. Select the pads with sufficiently large remaining energy, e.g., $E^{i}_{\text{Remaining}} > \sigma^{i}_{\text{BKG}}$
4. Find *towers* in the selected pads: Pads with the same position in different layers
5. Merge neighbouring *towers* into *cluster*
6. Calculate *cluster* position



Add signal from event: $\vec{E}^{\text{Event}}_{\text{Total}} = \vec{E}^{\text{Event}}_{\text{BKG}} + \vec{E}^{\text{Event}}_{\text{Signal}}$
Subtract Average and Sigma: $\vec{E}^{\text{Event}}_{\text{Remaining}} = \vec{E}^{\text{Event}}_{\text{Total}} - \vec{E}^{\text{average}}_{\text{BKG}} - \vec{\sigma}_{\text{BKG}}$
Select pads $\vec{E}^{\text{Selected}}$ with $E_{\text{Pad}} > E^{\text{Cut}}_{\text{Pad}}(R_{\text{Pad}})$
Build *towers* from $\vec{E}^{\text{Selected}}$
Build *cluster* from neighbouring *towers*
Remove used *towers*
No *towers* remaining
Calculate *cluster* positions

Motivation for New Implementation

- Existing Marlin Processor for BeamCal Reco is basically unmaintainable
- Wanted to study ElectronTagging efficiency for CLIC
    - Different background, different geometry, number of bunch crossings per event, more realistic background fluctuations,. . .
    - Goal was to have a more generic and maintainable BeamCalReconstruction

Example from the Code: `doClustering`

```
BeamCalClusterList doClustering(BCPadEnergies totalEnergy ,
                                BCPadEnergies background ,
                                BCPadEnergies backgroundSigma ,
                                BCPadCuts cuts ) {
  BeamCalClusterList beamCalClusters;
  totalEnergy.subtractEnergies(background);

  PadIndexList selectedPads = getPadsAboveSigma( totalEnergy , backgroundSigma , cuts );

  while( not selectedPads.empty() ) {
    PadIndexList padsForNextRound;
    TowerIndexList myTowerIndices = getTowersFromPads( selectedPads );
    TowerIndex largestTower = max_element( myTowerIndices );
    forall( TowerIndex testTower in myTowerIndices ) {
      if ( not areNeighbours( largestTower , testTower ) ) {
        selectedPads.removeTower( testTower );
        padsForNextRound.addTower( testTower );
      }
    }
    beamCalClusters.addCluster( getClusterFromAcceptedPads( totalEnergy , selectedPads ) );
    //try again for the remaining pads without the cluster that was already found
    selectedPads = padsForNextRound;
  }//while there are towers

  return beamCalClusters;
}
```

Highlights of the new Implementation

- Geometry information can be read from Gear or soon DD4hep
- Several methods for including background
  - Individual background bunch crossings for most realism
    - Bunch crossing is randomly picked from a set
    - Processor to create suitable root files only containing BeamCal background included
  - Parametrisation of background (A. Sapronov)
    - Suitable processor to create parametrisation also included
  - Averaged background (as in old implementation)
    - Can read the `bg_aver...` files used for ILD reconstruction
- “Event Display” for understanding events in the BeamCal
- Write out efficiency histograms
- Steerable Marlin Processor
- To add new clustering you only have to write a new ‘doClustering’ function

How to Install

- Source code in FCAL svn repository:
  `https://svnsrv.desy.de/svn/FCAL/Software/FCalClusterer/`
- Standard installation (Needs Marlin, lcio, Gear, and Root)

```
svn co https://svnsrv.desy.de/public/FCAL/Software/FCalClusterer/trunk FCalCluster
cd FCalCluster
mkdir build
cd build
cmake -C $ILCSOFT/ILCSoft.cmake ..
make install
export MARLIN_DLL=...
```

- Will make a (pre)-release branch and tag soon

How to run 1: Input/Output

```
<processor name="MyBeamCalClusterReco" type="BeamCalClusterReco">

  <!--Name of BeamCal Collection-->
  <parameter name="BeamCalCollectionName" type="string"
             lcioInType="SimCalorimeterHit"> BCAL </parameter>

  <!--Name of the MCParticle Collection, only needed and used to estimate
      efficiencies-->
  <parameter name="MCParticleCollectionName" type="string"
             lcioInType="MCParticle">MCParticle </parameter>

  <!--Name of the Reconstructed Cluster collection-->
  <parameter name="RecoClusterCollectionname" type="string"
             lcioOutType="Cluster">BeamCalClusters </parameter>

  <!--Name of the Reconstructed Particle collection-->
  <parameter name="RecoParticleCollectionname" type="string"
             lcioOutType="ReconstructedParticle">
  BeamCalRecoParticle </parameter>
```

How to run 2: Background Method

```xml
<!--How to estimate background [Parametrised, Fixed, Real]-->
<parameter name="BackgroundMethod" type="string"> Averaged </parameter>

<!--Root Inputfile(s)-->
<parameter name="InputFileBackgrounds" type="StringVec">
  bg_aver.sv01-14-01-p00_fieldX02.mILD_o1_v05.E250-TDR_ws.PBeamstr-pairs.I270000.root
</parameter>

<!--Number of Bunch Crossings of Background per event-->
<parameter name="NumberOfBX" type="int"> 1 </parameter>
```

How to run 3: Pad Selection

```
<!--Use the cuts for the pads specified in ETPad, if false, the variance in
    each pad is used times SigmaPad Factor. If false the first entry in ETPad is
    used as a minimum energy to consider a pad at all-->
<parameter name="UseConstPadCuts" type="bool"> False </parameter>

<!--If not using ConstPadCuts, each pad SigmaCut*variance is considered for clusters-->
<parameter name="SigmaCut" type="double"> 1.0 </parameter>

<!--Layer (inclusive) from which on we start looking for signal clusters-->
<parameter name="StartLookingInLayer" type="int"> 1 </parameter>

<!--Rings from which onwards the outside Thresholds are used-->
<parameter name="StartingRing" type="FloatVec"> 0 1  </parameter>

<!--Energy in a Pad, after subtraction of background required to consider it
     for signal-->
<parameter name="ETPad" type="FloatVec"> 0.005 0.3   </parameter>
```

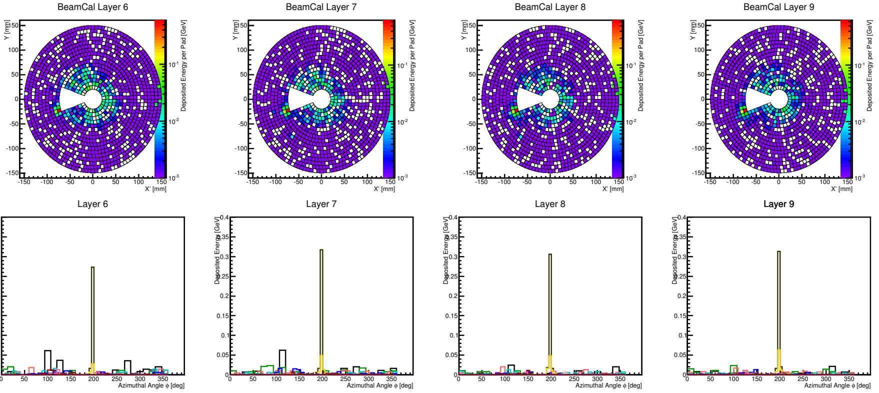
How to run 4: Energy Reconstruction, Cluster Selection

```
<!--Energy in a Cluster to consider it an electron-->
<parameter name="ETCluster" type="FloatVec"> 0.5 0.5  </parameter>

<!--Minimum number of (non-consecutive) pads in a single tower to be considered for sigr
<parameter name="MinimumTowerSize" type="int"> 6 </parameter>

<!-- Calibration factor to account for sampling fraction -->
<parameter name="LinearCalibrationFactor" type="double"> 72.76 </parameter>
```

How to run 5: Debugging

```
  <!--Flag to create the TEfficiency for fast tagging library-->
  <parameter name="CreateEfficiencyFile" type="bool"> true </parameter>

  <!--The name of the rootFile which will contain the TEfficiency objects-->
  <parameter name="EfficiencyFilename" type="string">TaggingEfficiency.root
  </parameter>

  <!--Number of Event that should be printed to PDF File-->
  <parameter name="PrintThisEvent" type="int"> -1 </parameter>

  <!--verbosity level of this processor
       ("DEBUG0-4,MESSAGE0-4,WARNING0-4,ERROR0-4,SILENT")-->
  <parameter name="Verbosity" type="string"> DEBUG2 </parameter>

</processor>
```

Event Display



ImpactAngle θ: -1.0, Energy Deposit: 4.4 GeV
Max Deposit: 0.3 GeV in Layer 7

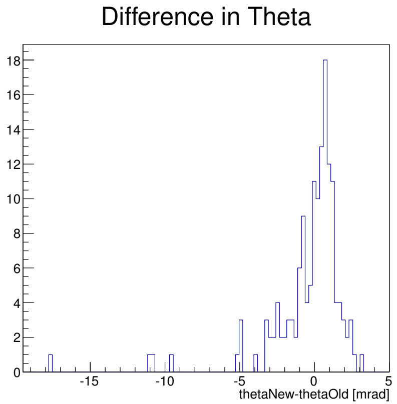
Comparison

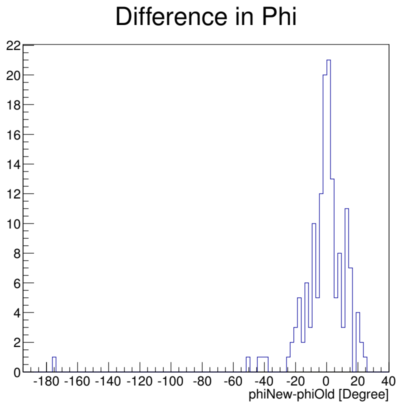
- Simulated some 250 GeV Electrons in Mokka
  - Harder than I thought, because Mokka just ignored my random seed so I could not run Mokka in parallel on my desktop PC
- Reconstruct with `/cvmfs/ilc.desy.de/sw/ILDConfig/v01-16-p10_250/StandardConfig/current` running only BCalReco and new BeamCalClusterReco (parameters as above)
- Reading `bg_aver.sv01-14-01-p00_fieldX02.mILD_o1_v05.E250-TDR_ws.PBeamstr-pairs.I270000.root` as background source

Preliminary Results

- With the parameters as described above the new Reconstruction finds clusters in every event where the Old reconstruction found a cluster as well
- In most cases the positions are close to each other and to the actual electron cluster
- The reconstructed energies are far off in many cases
- Needs some tuning, understanding fake rates, calibration

Reconstruction Differences





Summary

- New BeamCal reconstruction implementation is usable in ILD reconstruction
- Needs tuning