New EMC softwares planned for Release 15

- Cluster centroid algorithm
- Track-bump matching method
- Conclusions

Gagan Bihari Mohanty
Special EMC Software Meeting
May 19, 2004 @ SLAC
Cluster centroid algorithm

Introduction

- Use Depth position method instead of Surface Position to shift the digi position by 12.5 cm
- The bump centroid should be accordingly repositioned inside the crystal
- $\Delta \theta(\theta_{\text{bump}} - \theta_{\text{track}})$ and $\Delta \phi(\phi_{\text{bump}} - \phi_{\text{track}})$ will be calculated inside crystals compared to the current centroid position method where they are calculated @ intersection points
- Track-bump matching algorithm needs to be developed keeping the new definition of $\Delta \theta$ and $\Delta \phi$ in mind
- It is expected, the new centroid algorithm and accompanying change in track-bump matching will affect many other ongoing Emc works
EmcDigi is a Class Template to hold the Emc DIGIs

Here one can specify which centroid algorithm to choose (Surface or Depth position method)

The later option is selected with the digi position shifted by 12.5 cm inside the crystal

The cluster/bump centroid should be automatically repositioned while accessing it via `where()` or `position()`

However, we found out more needs to be done to get it work
Comparison of Old vs. New module

```c++
//cout <<" Using surface Position "<<endl;
//cout <<" Before shifting "<<(HepPoint) myGeom->frontCentre()<<endl;
//cout <<" Using depth Position "<<endl;
//cout <<" Before shifting "<<(pos)<<endl;
//cout <<" After shifting "<<(pos)<<endl;
static HepPoint (*pointer) ( const EmcXtal* ) = EmcDigi::surfacePosition;
---
// static HepPoint (*pointer) ( const EmcXtal* ) = EmcDigi::surfacePosition;
// modified on October 22, 2003
static HepPoint (*pointer) ( const EmcXtal* ) = EmcDigi::depthPosition;
---
// double EmcDigi::_rescaleFactor = 1.;
// double EmcDigi::_positionDepth = 0.;
---
// double EmcDigi::_rescaleFactor = 1.08;
// double EmcDigi::_positionDepth = 12.5; // in cm
---
// modified on October 22, 2003
// double EmcDigi::_rescaleFactor = 1.08;
// double EmcDigi::_positionDepth = 12.5; // in cm
```

< Old > New
EmcClusterLiloPos is a combined linear and logarithmic digi-weighting algorithm used for cluster (bump) position reconstruction.

In the current version, this module is not concurrent with the digi shifting i.e. it does not care whether the digi position is shifted or not.

Rather it always calculates the cluster position projecting it onto the crystal surface (does not work for the new method).

We commented it out and asked to use the centroid position as it is, project only if it lies outside a crystal.
Comparison of Old vs. New module

```cpp
const Hep3Vector *lNormal = lGeom->normalToFrontFace();

const Hep3Vector lLiloVector = lLiloPoint - HepPoint(0, 0, 0);
const Hep3Vector lCentre = lGeom->frontCentre() - HepPoint(0, 0, 0);

const double lLength = lNormal->dot(lVector);

const double lMag = lNormal->dot(lCentre) / lNormal->dot(lUnit);

if (lLength < 0.0) {
  // Point is outside crystal. Not sure its correct for all cases?
  // Anyhow, project point back onto front-face.
  cout << "Point is outside crystal, projected back" << endl;
  const Hep3Vector lUnit = lLiloVector.unit();
  const double lMag = (lNormal->dot(lCentre) / lNormal->dot(lUnit));
  lLiloPoint.setMag(lMag);
} else {
  // Don't project onto front-face. Keep the centroid inside the crystal.
  if (lLogNum > 1) {
    // Use logarithmic centroid position
    lLiloPoint.setMag(lLogSum.mag());
    cout << "With log centroid method, lilo-position" << lLiloPoint.mag() << endl;
  } else {
    // Use linear centroid position
    lLiloPoint.setMag(lLinSum.mag());
    cout << "Use lin centroid method, lilo-position" << lLiloPoint.mag() << endl;
  }
}
```

< Old > New
Cluster centroid algorithm

Bump Centroid: Surface vs. Depth

![Graph of Bump XY radius vs. Z value]

Entries 5029
Mean x 87.02
Mean y 87.88
RMS x 94.6
RMS y 10.34

Entries 5025
Mean x 92.76
Mean y 96.04
RMS x 101.1
RMS y 12.4

Track Match Study

Gagan Mohanty
First of all, the bump centroid position is shifted

The trend in shifting is exactly as one would expect:

- Barrel shows maximum possible shifting (12.5 cm) where the track hits at normal incidence
- The shifting gradually decreases as one moves away from the barrel in either way
- Forward endcap follows the expected trend

There were some worries how well the barrel-endcap transition region would behave

However, we do not notice any drastic jump anywhere including the transition region
Cluster centroid algorithm

**Timeframe for new centroid algorithm**

- Required modules (EmcDigi.cc/EmcClusterLiloPos.cc) are revised for the new algorithm and tested to be reliable
- Would be delivered to Reco Manager to integrate with 15.0.x
- We (Rolf and I) need to sit down together to see how it can be accomplished
- Expect to be done by next week
- As the new algorithm has immediate impact on the track-bump matching (TM), the study is being prioritized to get the new set of TM parameters asap
- This is planned to be finished before undertaking more comprehensive step such as introducing new variable to take care of non-IP tracks *etc.*
Track-bump matching method

**Introduction**

- Split up $\Delta \theta$ and $\Delta \phi$ distributions obtained with the new centroid algorithm in:
  - polar angle $\cos(\theta)$
  - momentum $|\vec{p}|$
  - charge species (+/−) for different particles ($e/\mu/\pi$)

- Fit those distributions to a double Gaussian with common mean,
  \[
  f(x) = N(e^{-\frac{(x-\mu)^2}{2\sigma_1^2}} + R \cdot e^{-\frac{(x-\mu)^2}{2\sigma_2^2}})
  \]

- Parameterize the obtained fit variables ($\mu$, $\sigma_1$, $R$ and $\sigma_2$) as function of momentum for different fiducial regions

- Calculate the significance level for a specific particle hypothesis and corresponding track-bump matching efficiency

- Obtained fit parameters for the $\pi-$hypothesis are in fact the new set of TM parameters and will go to the database
At this point, our study does not require any major changes in the basic structure of the TM module (inherited from original developer John Back).

Main issue is, however, to get hold of a high statistics control sample data to have reasonable entries in each bin of $\Delta \theta / \Delta \phi$ distribution.

Now I am running on the CM2 converted dataset on default cache mode both at SLAC and Lyon.

Shall begin fitting the histograms from early next week with RooTuples that would have been done by this weekend.

Although may not be over the full 1+2+3 dataset, I expect them it to be reasonable enough for the study.
Implementation issue and timeframe(2)

- Carrying on this study till we have the new set of TM parameters, I would keep 4 weeks at hand (+1 week contingency)
- However, it should be made clear that the TM parameters one is talking here are the ones adjusted to new centroid algorithm and planned for 15.0.x
- New, and hopefully better parameterizations involving new binning variable (track entrance angle $\psi$) etc. will follow and is targeted for July 15.x.x
Conclusions

- Bump centroid algorithm part is ready
- Now concentrating effort on the track-bump matching part
- Hope to get it done by the stipulated deadline