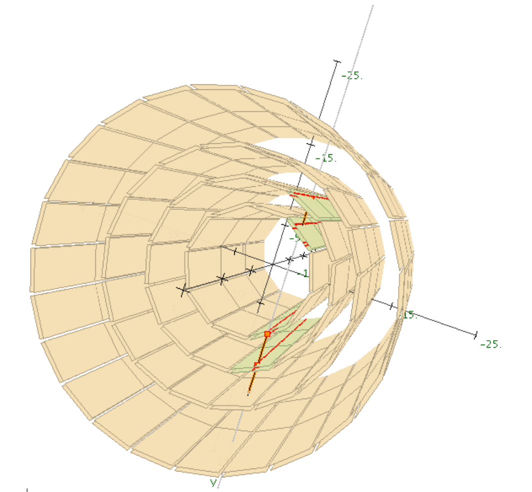
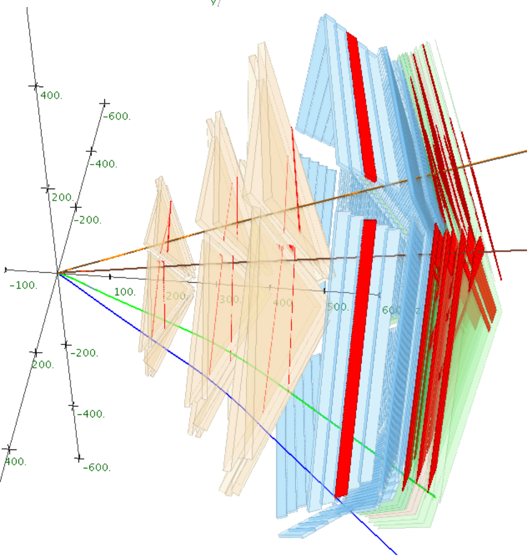
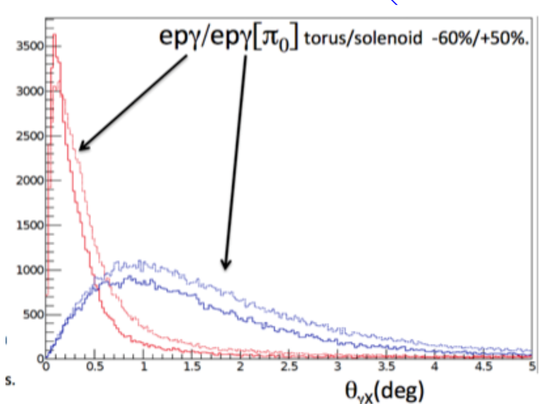






CLAS12 Event Reconstruction Status

Veronique Ziegler
Jefferson Laboratory

CLAS12 Collaboration Meeting
October 21, 2015

Overview

- Current Status of CLAS12 Software
  - Model and Architecture
    - Reminder & Status
  - Common tools
    - Reminder, status
    - Kinematic Fitter & PID Cuts
  - Simulation
    - Latest GEMC features
  - Reconstruction
    - Status & plans for the next release
  - Code Management
- Summary

Computing Model and Architecture

- ClaRA
  - SOA for reconstruction application deployment, does event-level parallelism
  - Data challenges on 24 core Haswell machine to validate scaling, performance and reconstruction speed
- Reconstruction code framework and running environment
  - version 1.0 release, version 2.0 in development
    - Detector reconstruction and event building framework
    - Reconstruction & validation
    - Calibration & Monitoring suite
- Common Tools packages
  - I/O, CCDB-access
  - Geometry
  - Calibration & Monitoring packages
  - Kinematic Fitter
  - Tracking tools: FastMC, MagField, Swimmer, snr (noise finding), Fitting libs

CLAS12 Software Components

- **Common Tools Packages**

- **Simulation (GEMC)**
  - Compliant with new version of GEANT4
  - GEMC now includes time propagation of light along strips for ECAL and PCAL & properly simulates gain matching algorithm for FTOF

- **Event Reconstruction**
  - Central & Forward Tracking
  - Time-of-Flight Particle Identification
  - Cerenkov counters and Electromagnetic shower counters

- **Visualization Tools**
  - ced includes tracking, calorimetry, TOF reconstruction information,
    - many views, detailed magnetic field information
  - 3-D views

Common Tools Packages

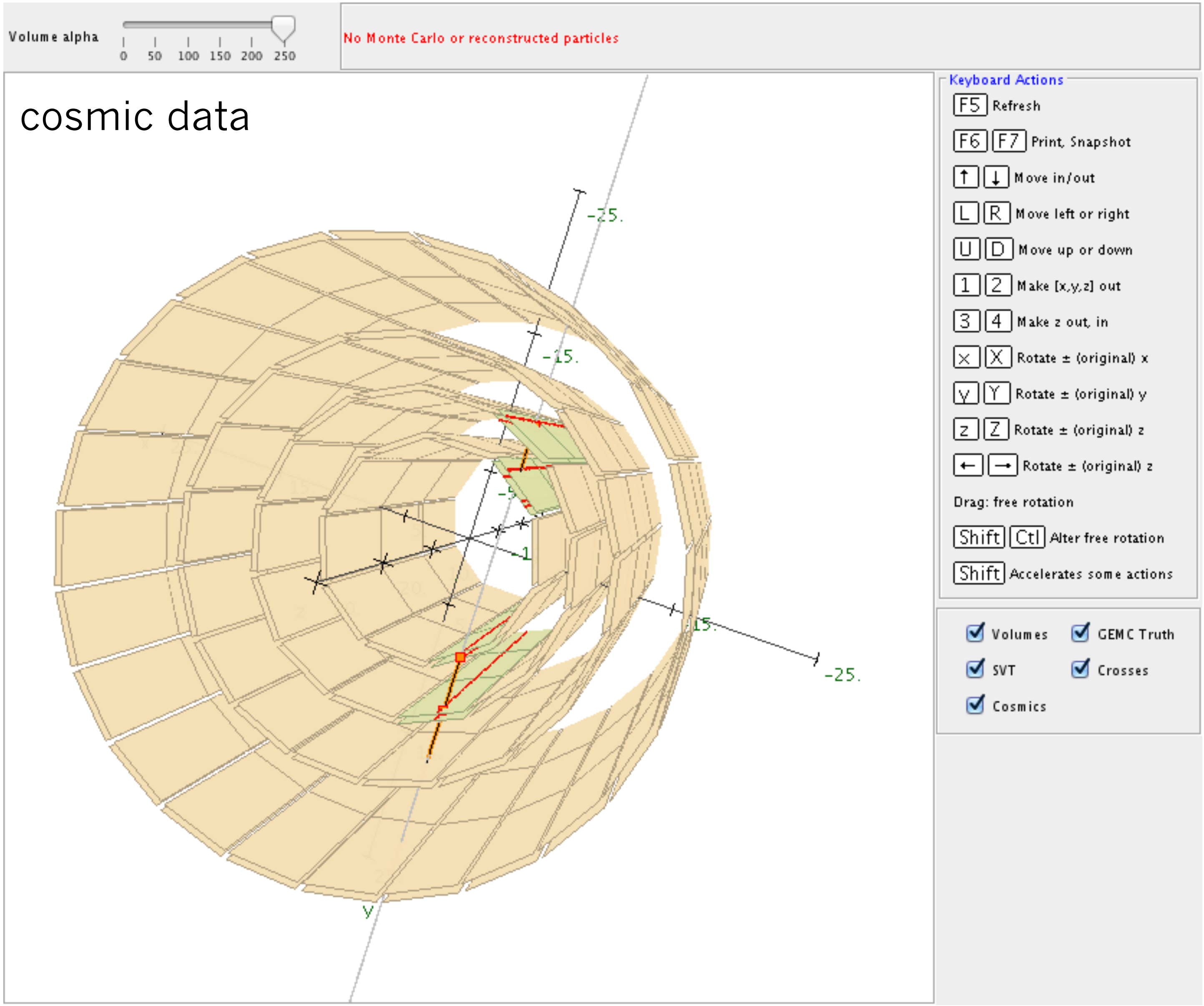
✓ **Detector Geometry Package provides detector geometry for :**

- Forward Time-of-Flight
- Calorimeters
- Drift Chambers
  - Updated Geometry document and implementation (previous mismatch between GEMC and hardware → fixed)
- Silicon Vertex Tracker
- Central Neutron Detector
- In development:
  - GEMC volumes
  - RICH
  - CTOF

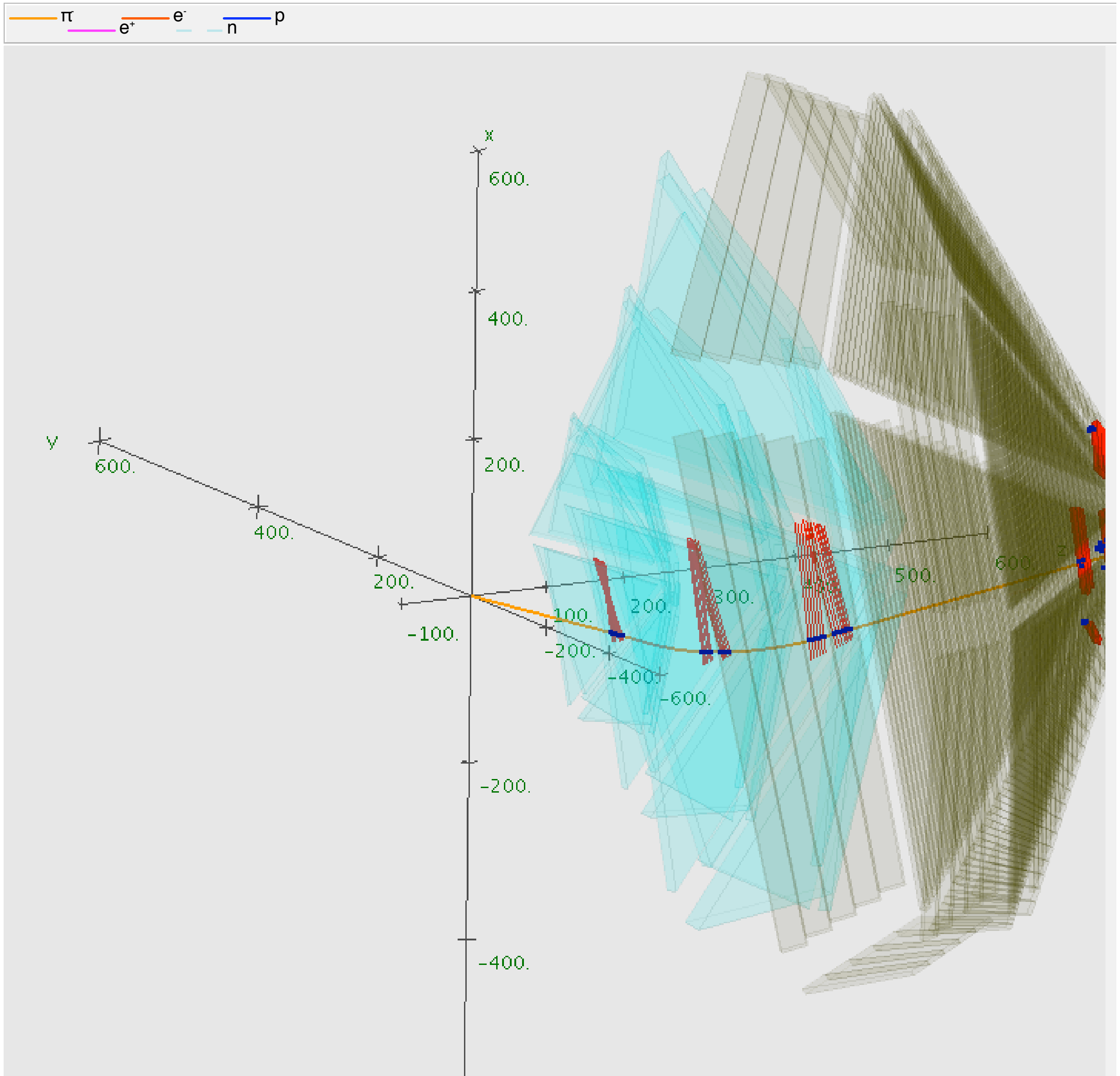
✓ **Geometry tools and utilities:**

- Drawing package for 2D detector representation
- Detector components for Fast Monte Carlo
- 3D shapes for event display 3D viewer

ced 3-D (D. Heddle)



ced 3-D



π
e+
e-
n
p
x
y
600.
400.
200.
-200.
-400.
600.
400.
200.
-100.
100.
200.
300.
500.
600.
-200.
-400.
-600.

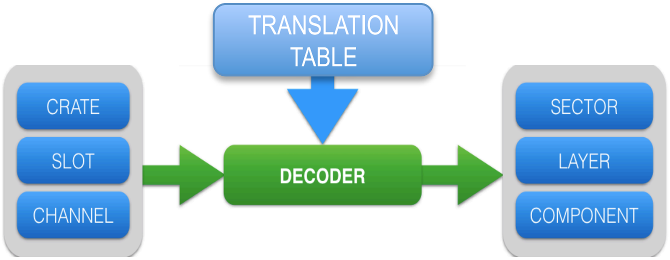
Common Tools Packages

EVIO Data I/O and Utilities

G. Gavalian

✓ **Raw Data I/O:**

- Software for reading coda data for different modes (MODE=1,3,7) automated translation tables for detectors (plugins), standardized hit bank generation
    - * translation tables provided by detector groups
    - * In use by several systems (SVT,TOF, EC, FT, now also DC…)



✓ **EVIO utilities:**

- Utility for splitting and merging files
- GUI for viewing GEMC generated banks (in CED)
- GUI for viewing and fitting RAW ADC spectra Reading data from ET ring (online)

✓ **BOS utilities:**

- Reader/Convertors for BOS (CLAS6) data to EVIO format
- Unified Interface for analyzing data from CLAS6
- Framework for PID, cuts and corrections for CLAS12 and CLAS6,
  Data format from CLAS6 can be passed to CLAS12 calibration and monitoring.

Common Tools Packages

DB Utilities

```
 Options:
 -i          :   input file name
 -l          :   number of events to skip
 -n          :   number of events to run
 -o          :   output file name
 -s          :   service list to run (e.g. BST:FTOF:EB )

  FLAGS : use -config SYSTEM::ITEM=value syntax to pass configuration.


 Available Configurations :

  -config CCDB::GEOMRUN=10          : set ccdb run number to 10 for loading geometry
  -config CCDB::GEOMVAR='custom'    : set ccdb variation to 'custom' for loading geometry
  -config CCDB::CALIBRUN=10         : set ccdb run number to 10 for calibration constant
  -config CCDB::CALIBVAR='custom'   : set ccdb variation to 'custom' for calibration constants
  -config MAG::torus=0.75           : set scale for TORUS magnet to 3/4
  -config MAG::solenoid=0.5         : set scale for SOLENOID magnet to 1/2
  -config DCTB::kalman=true         : enable kalman filter in time based tracking



DATABASE OPTIONS:
   to use local sqlite database type : setenv CCDB_DATABASE etc/database/clas12database.db



SHOW AVAILABLE DETECTORS : 12


  *****************************************************************
  * MODULE       * AUTHOR                      *  VERSION * LANGUAGE *
  *****************************************************************
  * BST          * ziegler                     *      1.0 *     java *
  * CTOF         * kenjo                       *      1.0 *     java *
  * DCHB         * ziegler                     *      2.0 *     java *
  * DCTB         * ziegler                     *      2.0 *     java *
  * EB           * gavalian                    *      1.0 *     java *
  * EC           * gavalian                    *      1.0 *     java *
  * FMT          * ziegler                     *      1.0 *     java *
  * FTCAL        * devita                      *      1.0 *     java *
  * FTHODO       * devita                      *      1.0 *     java *
  * FTMATCH      * devita                      *      1.0 *     java *
  * FTOF         * gavalian                    *      1.0 *     java *
  * HTCC         * henkins                     *      1.0 *     java *
  *****************************************************************
```

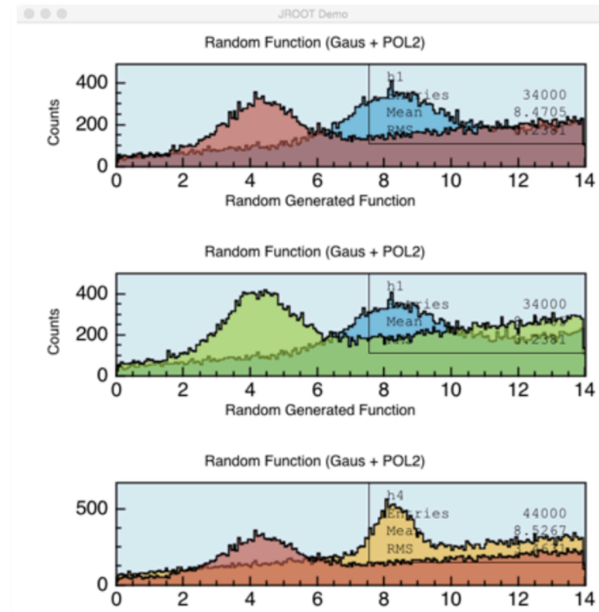
database variations at run-time

Common Tools Packages

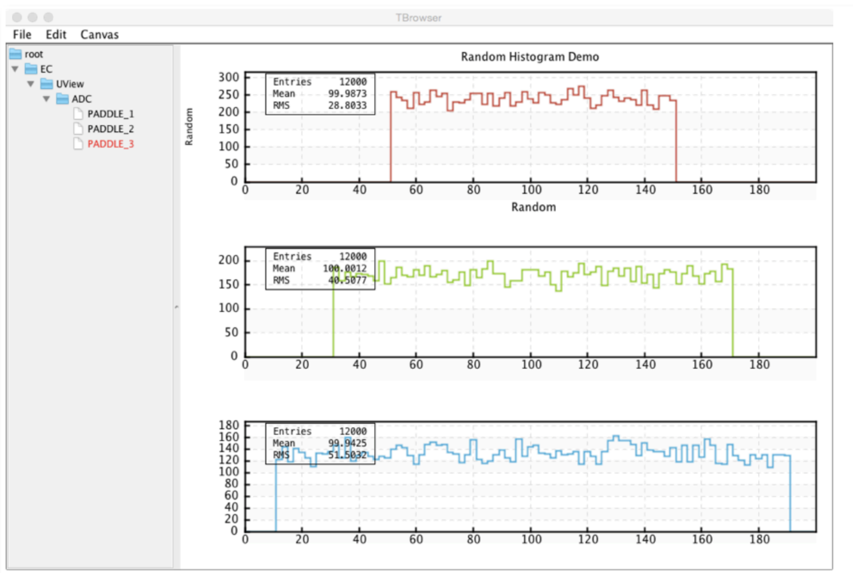
G. Gavalian

✓ Plotting Library:

- Histogram & graph classes, Ntuple class for analysis, NTuple I/O implementation in progress
- Fitting added using Minuet library
- Latex parsing added for titles and texts



- Histogram object browser



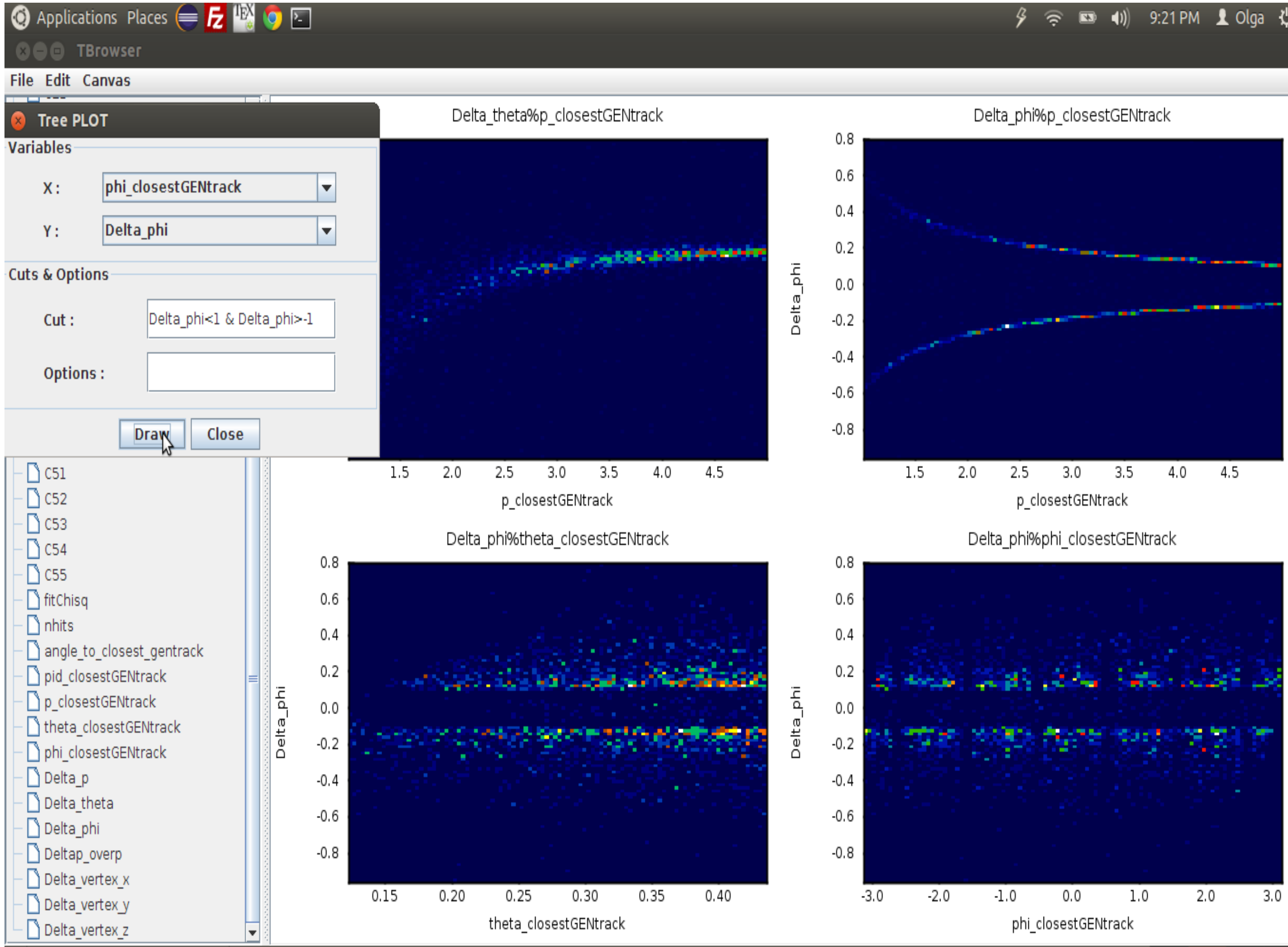
- Evio-root viewer upgrade to be compliant with new version of ROOT

DC Tracking N-Tuples

Olga Cortes- ISU

Tree Selector:
*1D/2D plots
*Cuts in any entry of the n-tuple

Ntuple entries



Ntuples→	Hits	Crosses	Clusters	Tracks/ Resolution
Time Based	✓	✓	✓	✓
Hit Based	✓	✓	✓	✓

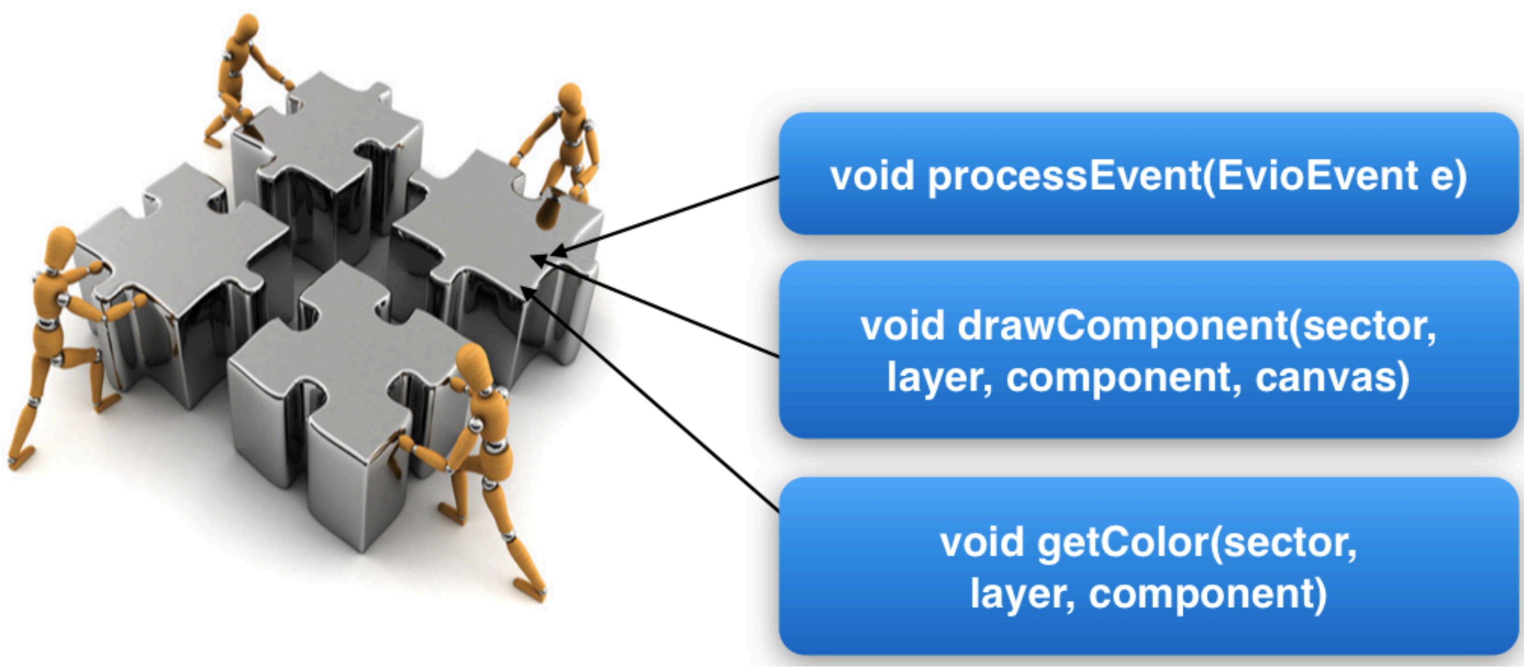
Common Tools Packages

G. Gavalian

✓ **Calibration & Monitoring Software:**

- Plugin based software framework
- standard interface for passing data through modules
- standard representation of the detector components
- interface to draw relevant histograms for each component
- automated plugin discovery from the package

Detailed examples in CALCOM Report (next talk)



Kinematic Fitter & PID Cuts (A. Kim)

Usage:

```
>> git clone https://github.com/eg1dvcs  // clone eg1dvcs repo

import org.jlab.clas6.eg1dvcs.*
...
ExperimentalKinematicFitter fitter = new ExperimentalKinematicFitter();

fitter.setPID(11, new nt22ElectronCut())
//fitter.setPID(11, new InclusiveElectronCut())
//fitter.setPID(11, new ExclusiveElectronCut())
```

//Collections of electron cuts maintained by experimental group for different reactions

```
//import org.jlab.clas6.eg1dvcs.kenjo.*
//fitter.setPID(11, new kenjoElectronCut())
...
```

(not always reliable, could be outdated)

//Personal collections of cuts

```
public interface PIDcut{
    public bool test(DetectorPa
    public void debug(Detecto

    ...
}

//e.g.
public class eg1dvcsSamplingFractionCut i
//EC sampling fraction cut for electron in e
    public bool test(DetectorParticle part){
        //e.g EC sampling fraction > 0.3
    }

    public void debug(DetectorParticle part
        //Plot EC sf vs momentum and cuts
    }
    ...
}
```

Different classes correspond to different cuts, e.g.
- eg1dvcsSamplingFractionCut (default)
- nt22SamplingFractionCut (nt22 skim)
- exclusiveSamplingFractionCut (loose)
- inclusiveSamplingFractionCut (strict)
- MySamplingFractionCut (unique)

All of them are stored in **git repo** specific to experiment:
- easy to share
  (simply clone from git)
- easy to cross-check and fix
  (pull request)
- easy to migrate:
  (just fork the repo to
  e.g. eg1dvcs2)

EASY TO FIND WHO IS RESPONSIBLE FOR THE CUT (git log)

- Framework to standardize the implementation of analysis cuts and event selection
- Tested first on CLAS6 data

GEMC
Detector signal model

M. Ungaro

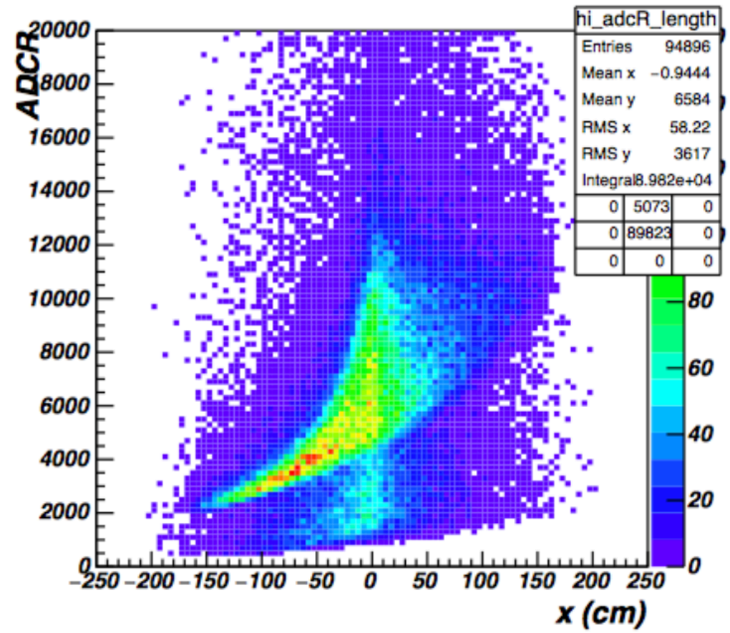
from CCDB:

ADC:

- Attenuation according to exponential law
- Conversion from energy to ADC based on MIP signal (dEdxMIP=2 MeV/cm, countsForAMinimumIonizing=2000)

TDC:
- Delay due to light propagation in the paddle (effective velocity)
- Parameterized Time Walk
- Gaussian time spread based on parameters that will be matched to data (σ2=σ02+σ12/√EPMT Conversion from time to TDC (time2tdc=20ns−1)

Output: both “smeared” and “unsmeared” TDCs



Status:
0 – fully functioning
1 – noADC
2 – noTDC
3 – noADC, noTDC(PMT i sdead)
5 – any other reconstruction problem

GEMC

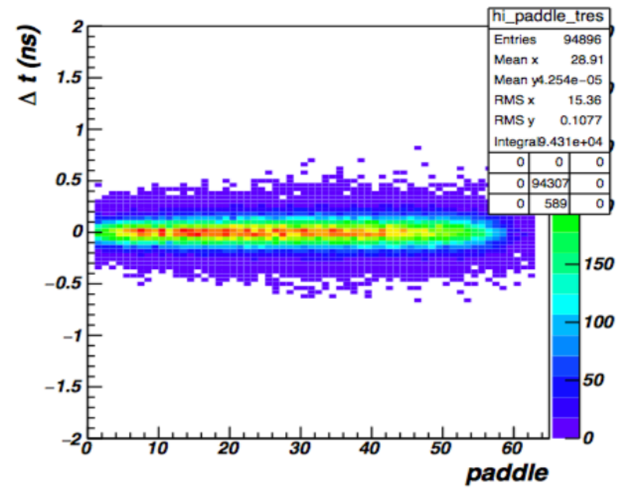
Detector signal model

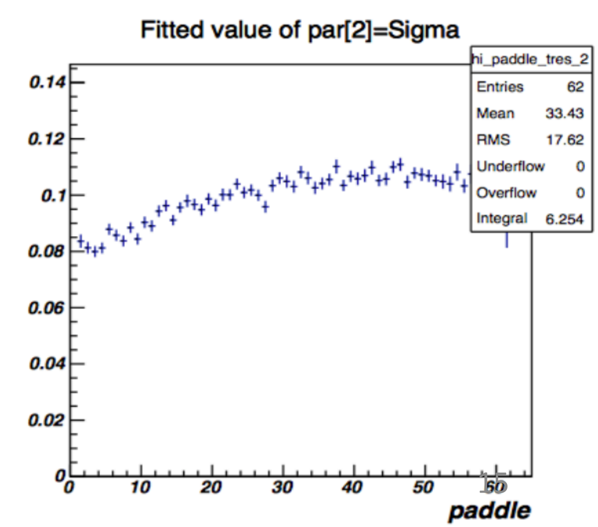
M. Ungaro

ADC:

- Attenuation according to exponential law
- Conversion from energy to ADC based on MIP signal (dEdxMIP=2 MeV/cm, countsForAMinimumIonizing=2000)

TDC:
- Delay due to light propagation in the paddle (effective velocity)
- Parameterized Time Walk
- Gaussian time spread based on parameters that will be matched to data (σ2=σ02+σ12/√EPMT Conversion from time to TDC (time2tdc=20ns−1)





Output: both "smeared" and "unsmeared" TDCs

Status:
0 – fully functioning
1 – noADC
2 – noTDC
3 – noADC, noTDC(PMT i sdead)
5 – any other reconstruction problem

What’s new in the “devel” version (future 2.3)

-RUN_WEIGHTS= “runs.txt”

2	0.1
13	0.6
22	0.2
30	0.1

-N=100,000 generates:

10,043 with constants from run 2
59,901 with constants from run 13
20,034 with constants from run 22
10,022 with constants from run 30

Run number put in header bank

(events are also ordered by run)

Event Reconstruction Status

- **Track Reconstruction**
  - Central Tracking
    - SVT reconstruction → Global fitting method, Kalman Filter fitting method implemented (validation stage)
    - CTOF reconstruction → beta from pathlength of track from SVT to CTOF
  - Forward Tracking
    - DC Hit-Based Tracking, DC Time-Based Tracking → Kalman Filter fitting method validated
    - FMT pattern recognition → refit using reconstructed FMT 3-D points
    - FTOF reconstruction→ beta from pathlength of track from DC to FTOF paddle
      - **Digitization code change to take into account attenuation length requires modifications to existing code (inv. digi.)**
    - FT reconstruction using calorimeter and hodoscope to id low angle electrons and reconstruct pi0's
    - EC/PCAL reconstruction → detector responses used to obtain PID, neutrals reconstruction
    - HTCC reconstruction → e- ID [HTCC will be in version 2.0]
- Event Builder with Likelihood-based PID

Developments

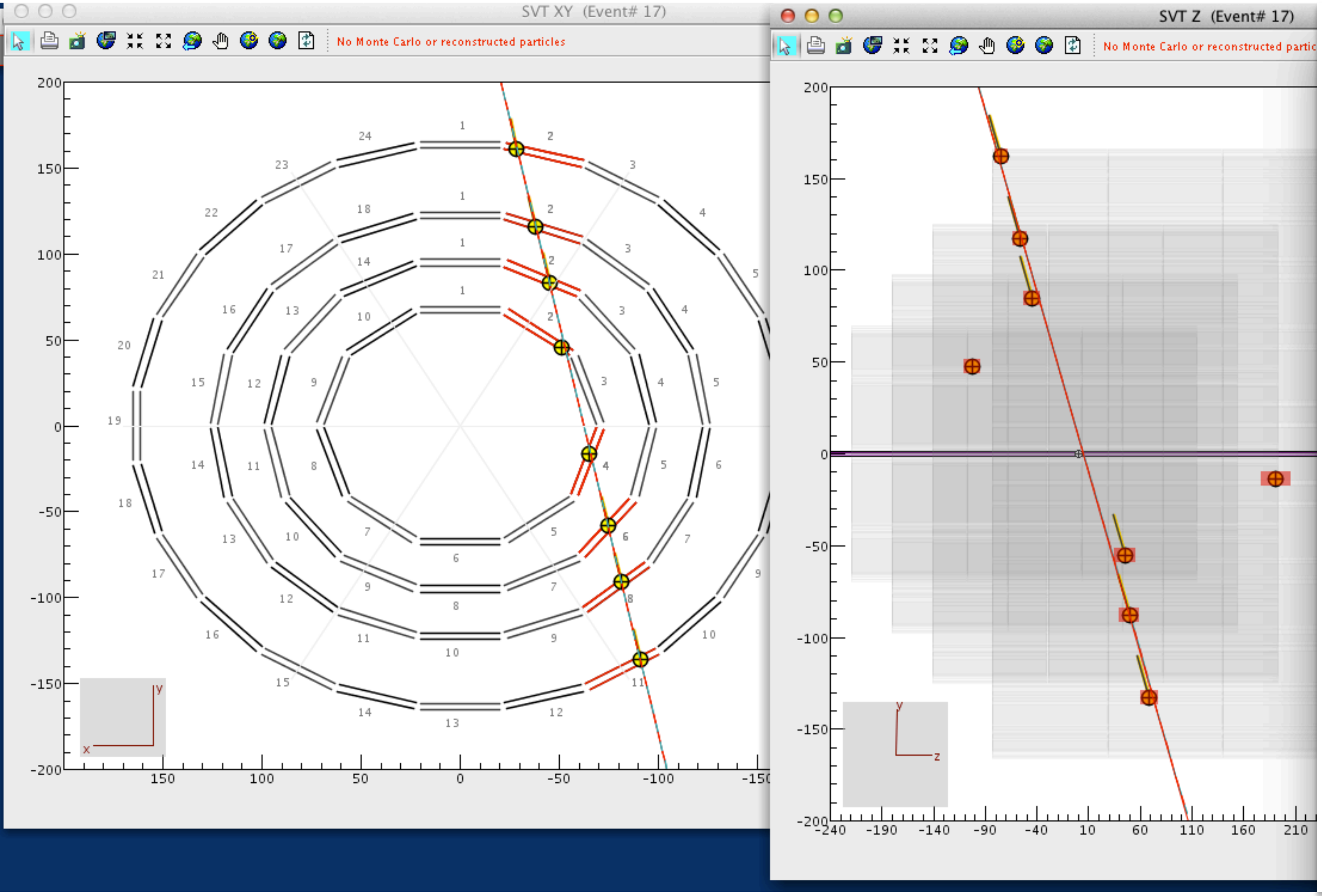
* Tracking
* Neutrals, PID detector reconstruction

Validations & Fixes

SVT Tracking with Cosmics

- Use SVT cosmics data to test the reconstruction software & algorithms and to identify hardware issues (e.g. dead strips)
- Useful to find bugs, fine-tune the code, tune the MC

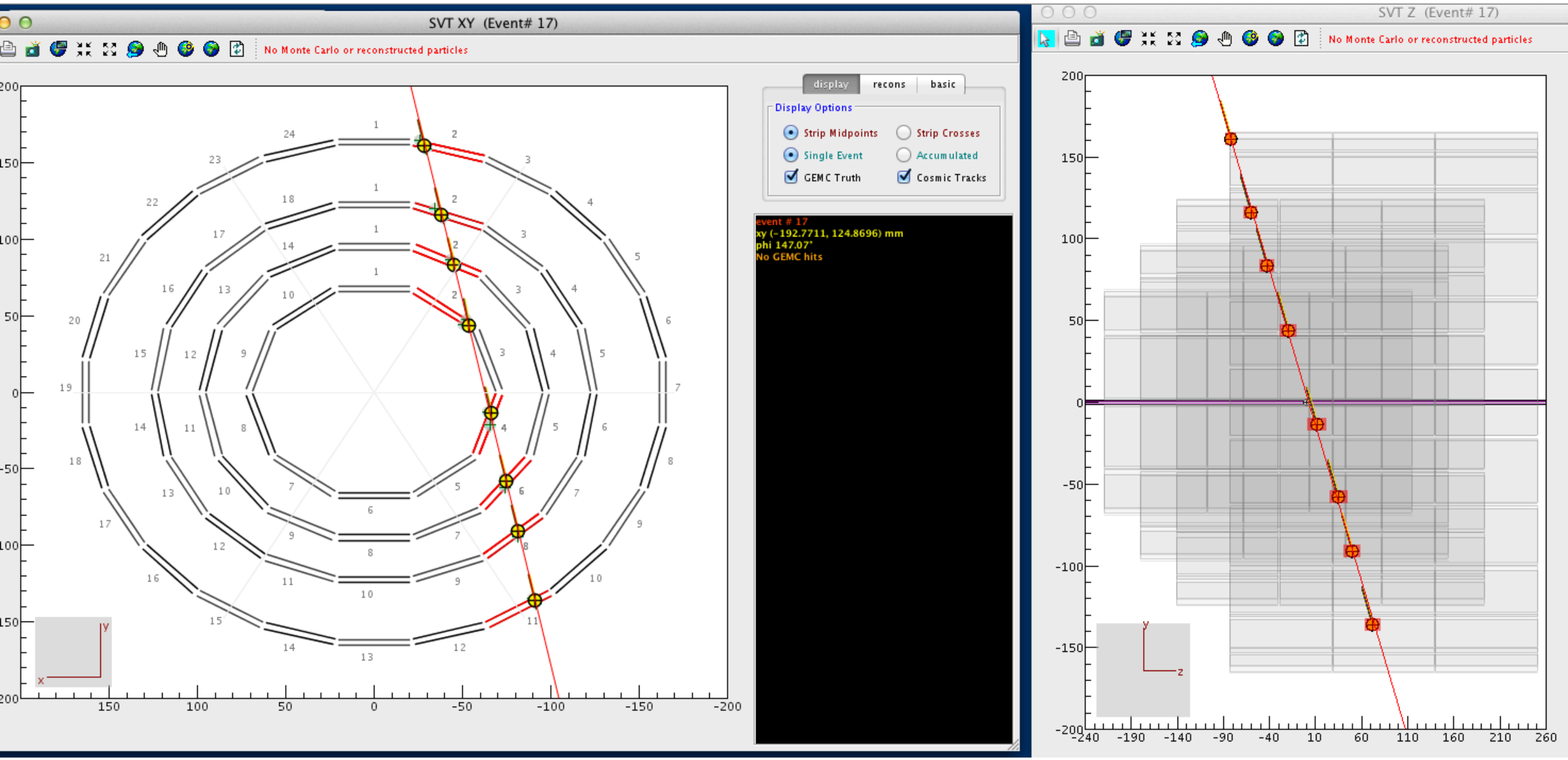
COSMICS DATA (RUN 257_17) EVENT 17



2 crosses not on track → z correction failed !!!

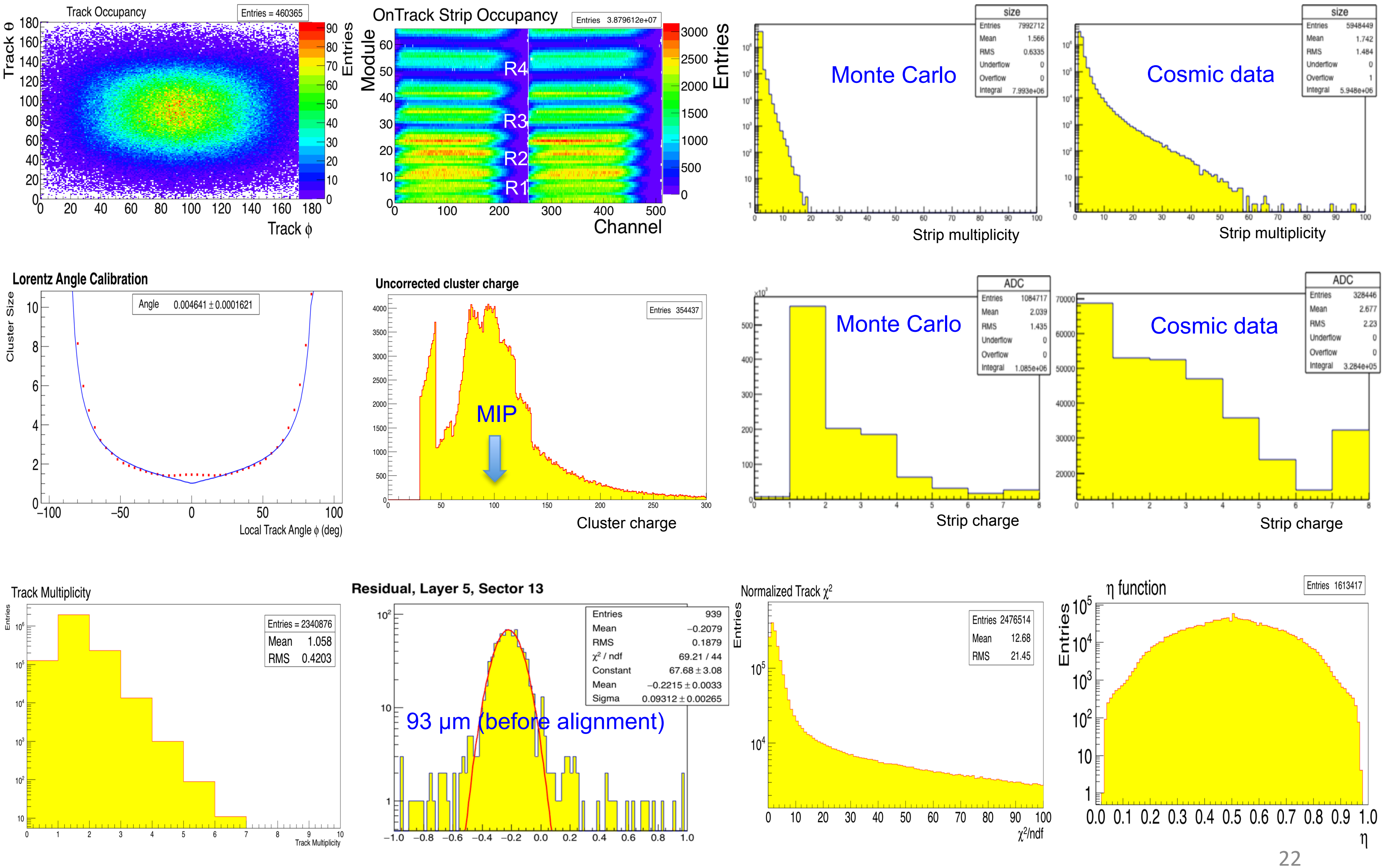
➔ Simulate this event to debug…

COSMICS DATA (RUN 257_17) EVENT 17



After bug fix

SVT Data Validation Sample Plots (Y.Gotra)



SVT Data Validation Suite Development

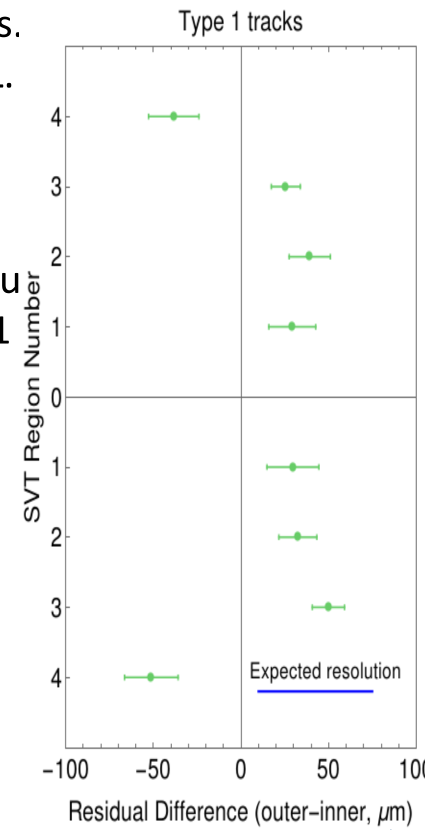
(Y. Gotra)

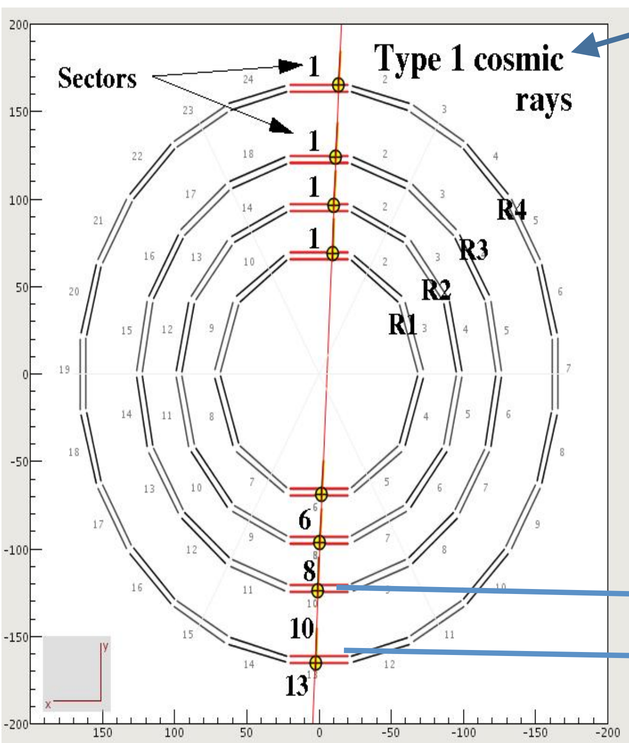
- Run on reconstructed data and Monte Carlo samples using CoatJava based package
- Track structured event object (track-trajectory-cross-cluster-hit-digi) with cross-links
- Component based detector performance plots (barrel-region-sector-layer-chip-strip)
- Local reconstruction validation plots (cluster size, seed strip etc.)
- Track reconstruction validation plots (angular distributions, residuals, χ^2, on-track vs. off-track etc.)
- Calculation of standard parameters (resolutions, multiplicities, Lorentz angle, η-function etc.)
- Color coded presentation of monitored observables (occupancies, efficiencies)
- Automated bad channel mapping
- Comparison between the data and simulated events
- Online data quality monitoring
- Inter/intra-run stability validation for critical observables
- Automated software release validation
- Automated run validation
- Batch processing on the farm

Status: developing event and histogram classes and interfaces, profiling

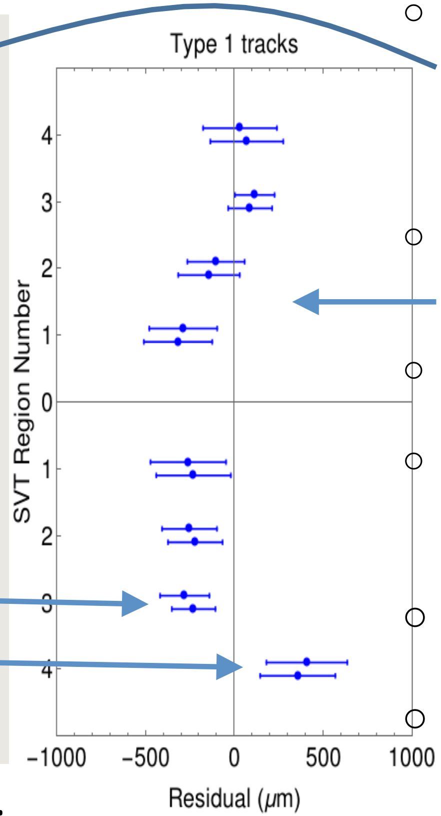
Alignment of the Silicon Vertex Tracker (J. Gilfoyle)

- Correcting detector misalignments essential for reaching design specifications.
- Construction of the SVT is complete and it is taking cosmic ray data in the EEL.
  - Straight tracks from cosmic rays can be used to measure alignment.
  - Procedures developed here will be used for other detector subsystems.
- Recent results.
  - Cosmic rays come from all over the scan so many different tracks. See Yu
  - To simplify the analysis we focus on a subset of the cosmic rays – Type 1





Each region has two layers.

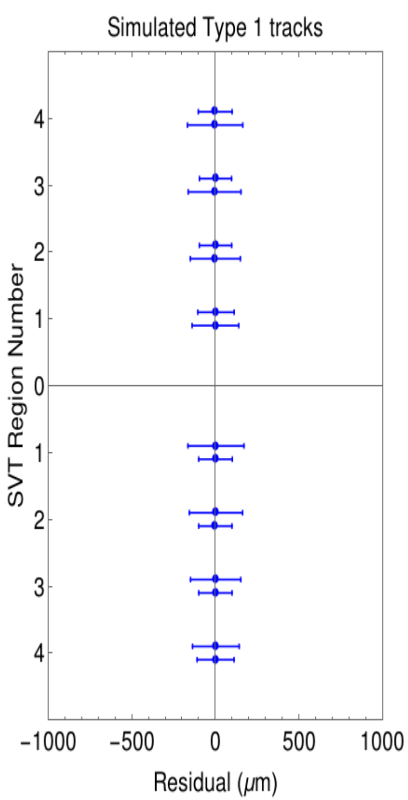


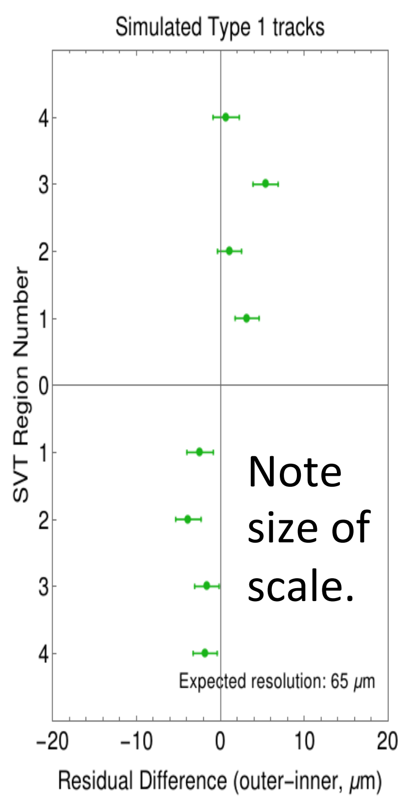
- Select cosmic ray events with 16 layers, two in each region and through the central sectors.
- Extract residuals layer-by-layer and plot by region.
- Error bars are RMS of residual spectrum.
- Regions here coincide with Type 1 track figure.
- Residuals show misalignments of a few hundred microns.
- Residual difference within each region show signs of a shift along the beam axis, but small in size.

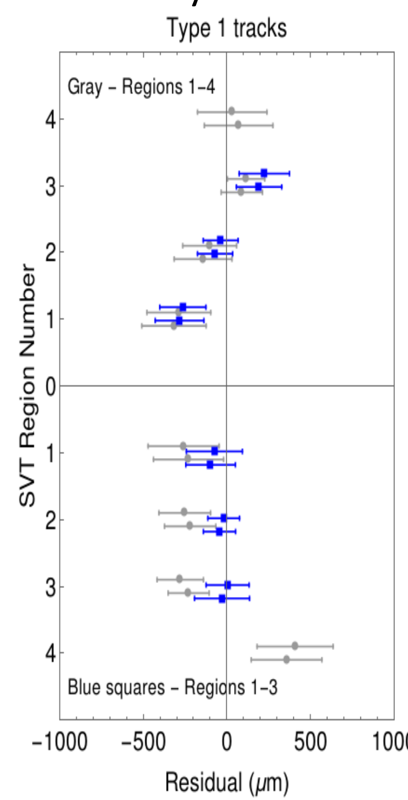
Alignment of the Silicon Vertex Tracker

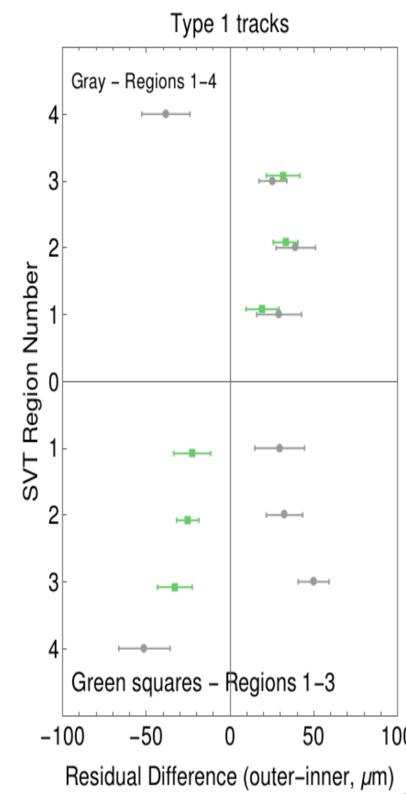
(J. Gilfoyle)

- Checks on the alignment analysis – use cosmic ray simulation.
  - Simulation is done with ideal geometry so the residuals should be consistent with zero.
  - Tests the reconstruction.
- Regions 1-3 and Region 4 are attached to different structures – possible different alignment parameters.
  - Use same set of Type 1 events, but now leave Region 4 out of the track fitting.
  - Most, but not all, residuals improve.
  - Change in residual difference still under study.









Validations & Fixes

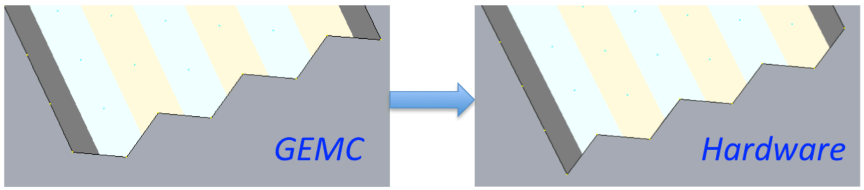
DC Tracking with Cosmics

- Use DC cosmics data to test the reconstruction software & algorithms and to identify hardware issues (e.g. cable swaps)
- Useful to find bugs, fine-tune the code, get Time-To-Distance function → adding special calibration and validation banks

Fixes & To Do List

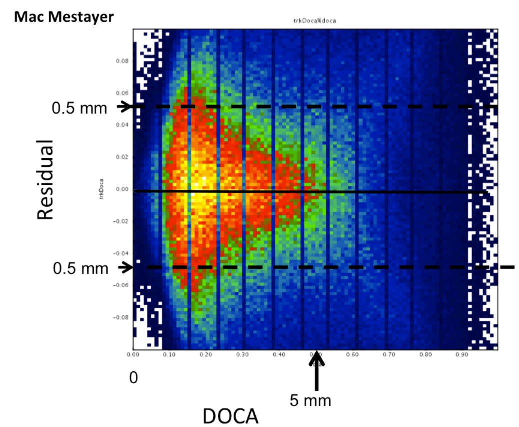
– Insured proper geometry called for data ✓

DC Geometry fix: *brick-wall pattern*



☛ also change in GEMC

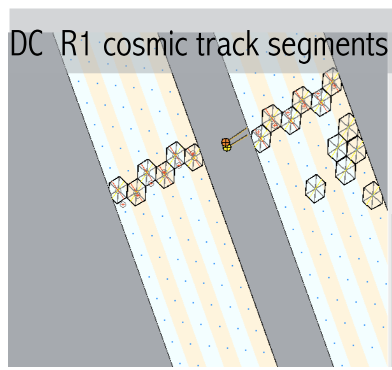
– Layer efficiency bank output ✓

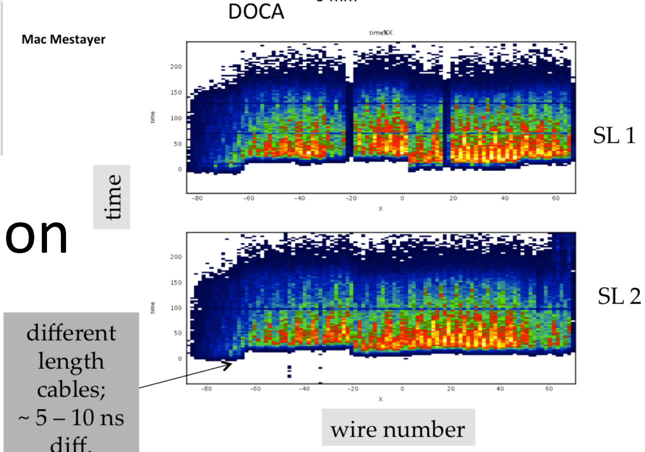


Residuals from un-calibrated data; using same T → X curve as GEMC !

– TBLA banks for time-to-distance analysis

- time residuals ✓
- track doca (TB)
- …





different length cables; ~ 5 – 10 ns diff.

– Include cable delays (& all calibration constants) in reconstruction

FT Software Status

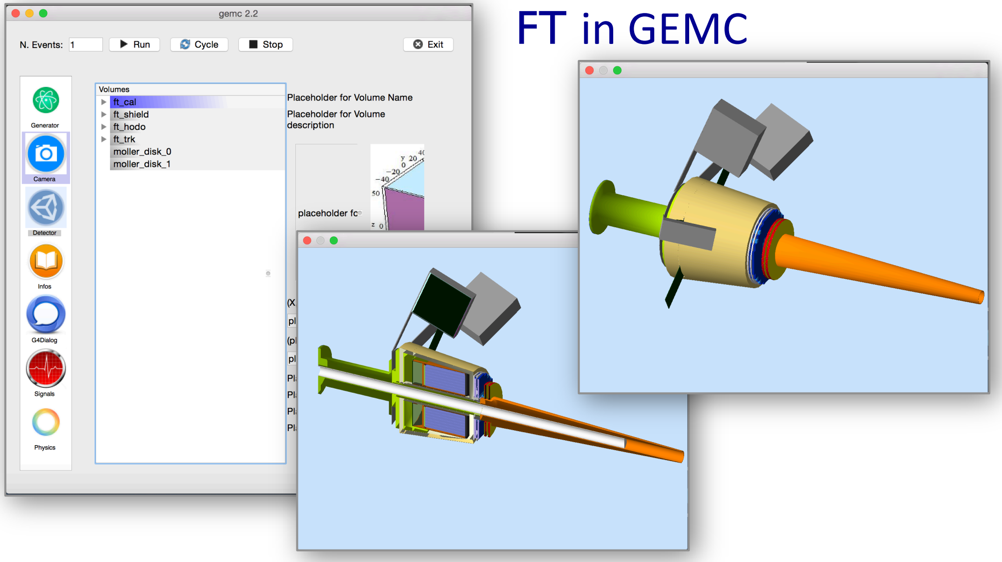
R. de Vita

Simulations:
- Realistic geometry description of all FT components, including active and passive elements/materials
- Digitization routines implemented
- Output bank structures defined

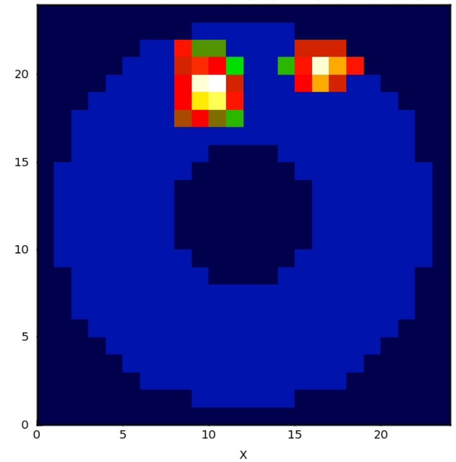
Reconstruction:
- Algorithms developed for all three subsystems
- Calorimeter and hodoscope reconstruction available in first release
- Tracker reconstruction implementation in progress

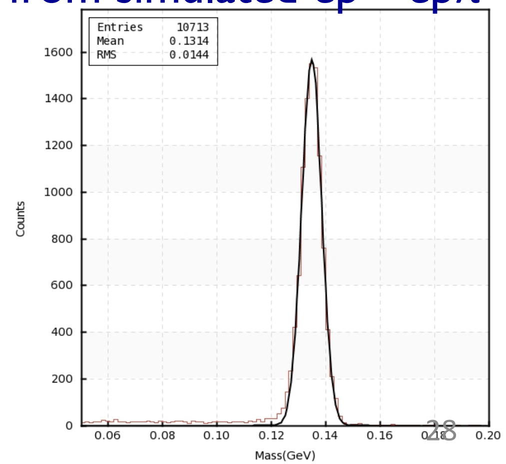
Database:
- Algorithms developed for all three subsystems

FT in GEMC



FT π^0 reconstruction from simulated ep→epπ^0



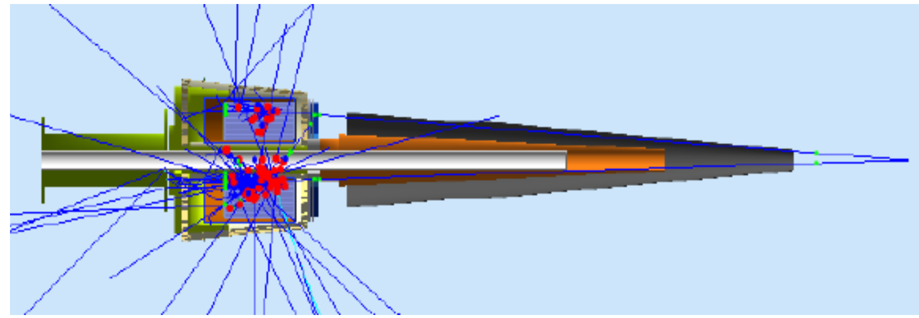


FT-Cal Energy Corrections

R. de Vita

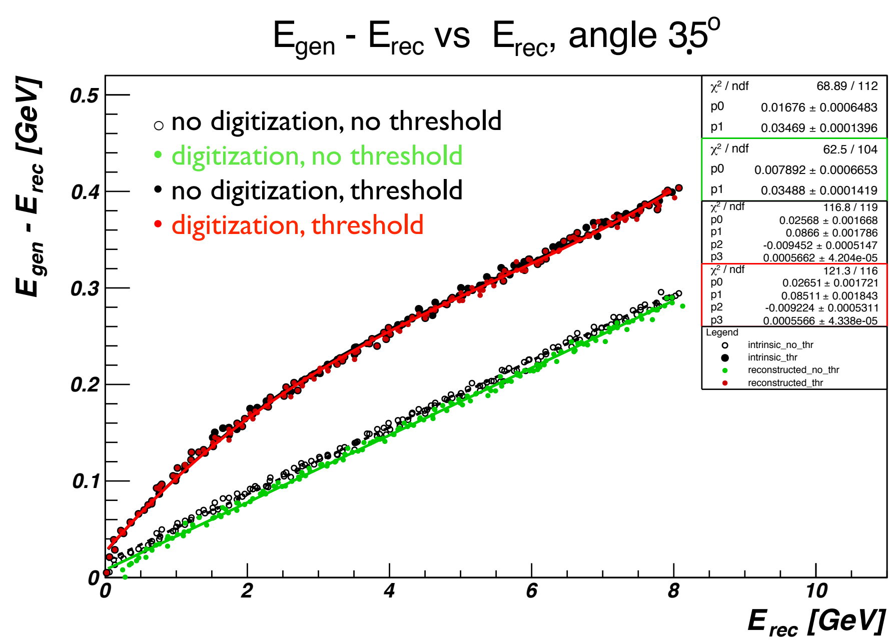
Reconstructed energy from clustering is less than the incoming electron/ photon energy due to:

- EM shower leakages due to finite calorimeter size
- Channel threshold (5 MeV)



Energy corrections:

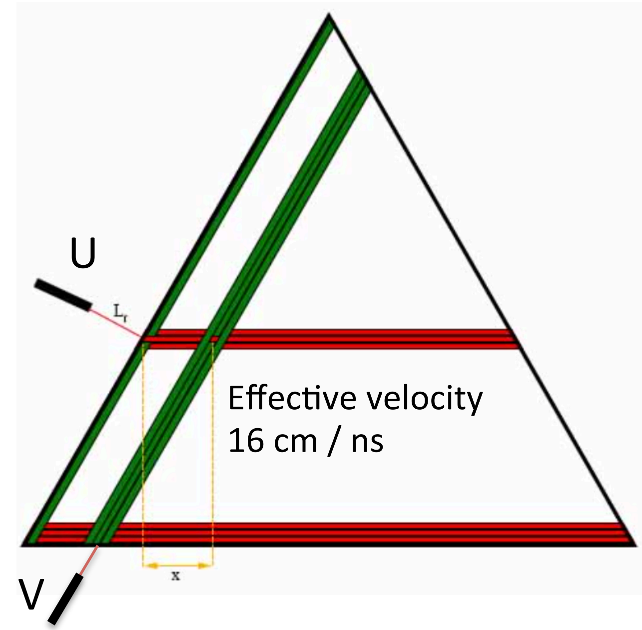
- Derived from simulations as a function of the incoming particle energy and impact point
- E_{gen}-E_{rec} vs E_{rec} for fixed θ parameterized with analytic function and applied to cluster energy to obtain the "true" particle energy
- 10 points in θ from 2.5° to 4.5°, 5 MeV energy bins from 0 to 8 GeV
- Channel threshold is the dominant contribution



Work by Lucilla Lanza (U. Roma Tor Vergata)

EC/PCAL Software Status

Cole Smith

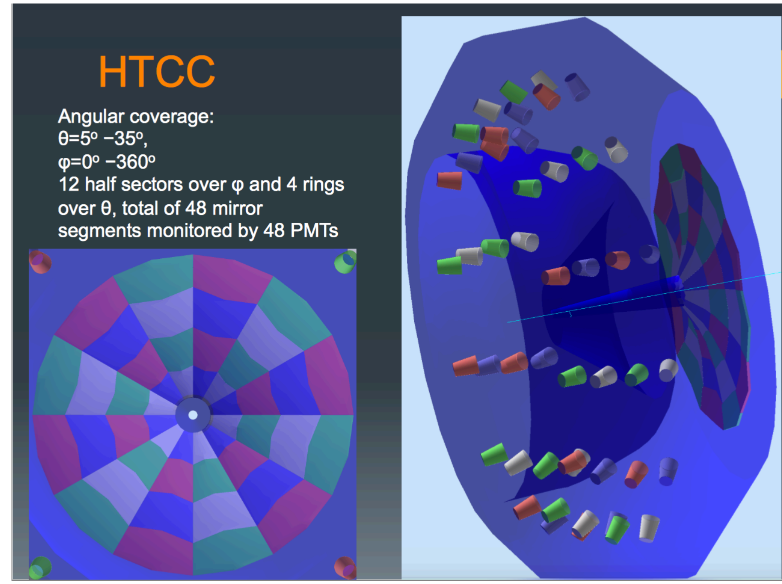


- DGTZ banks: Raw counts x 24 ps/count to simulate both V1190 / V1290 TDCs
- **Simulation checked against data using time differences between cosmic muon hits in different U,V,W layers**
- Forward Carriage data show effect of different cable and light guide lengths and **provide estimate of time resolution for simulation**

- EC reconstruction package currently being evaluated by C. Smith in collaboration with Gagik
- Under COAT-Java EC is a unified detector with 3 superlayers:
  0=PCAL 1=ECAL inner 2=ECAL outer
- Output bank consists of lists of hits, peaks and clusters for each superlayer
- ECReconstruction class does not attempt cluster matching between superlayers
- Ongoing work to introduce ccdb constants, attenuation corrections, iterations & reconstructed objects associations in reconstruction, and methods for handling two-cluster identification with shared energy in peaks for π^0 id (Cole)

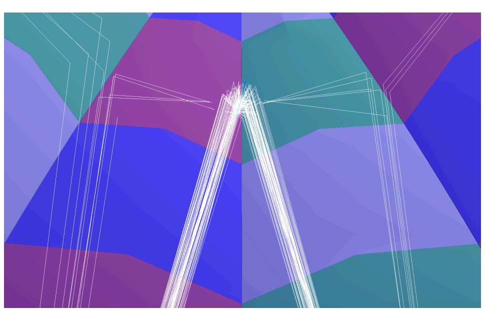
HTCC Software Status

(N. Markov)

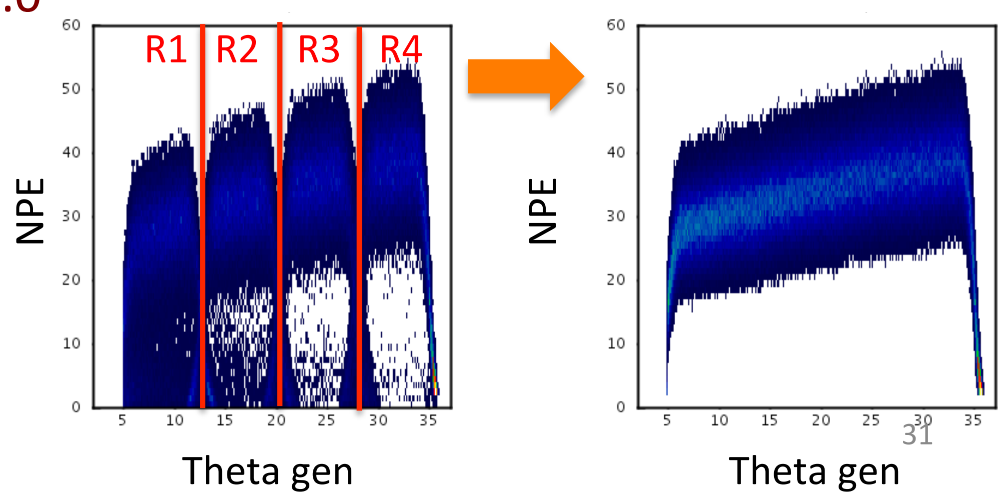


- ❑ Exiting code (original author: A. Puckett, Java: J.Hankins) ported into COAT-Java, validated & further developed by Nick Markov
- ❑ Simulation and reconstruction incorporated in CLAS12 software framework
- ❑ Clustering algorithms in place and being validated
- ❑ Timing to be completed
- ❑ Investigating fiducial cuts
- ❑ Service in reco. chain for Release 2.0

Clusters are reconstructed for events with hits in several mirrors

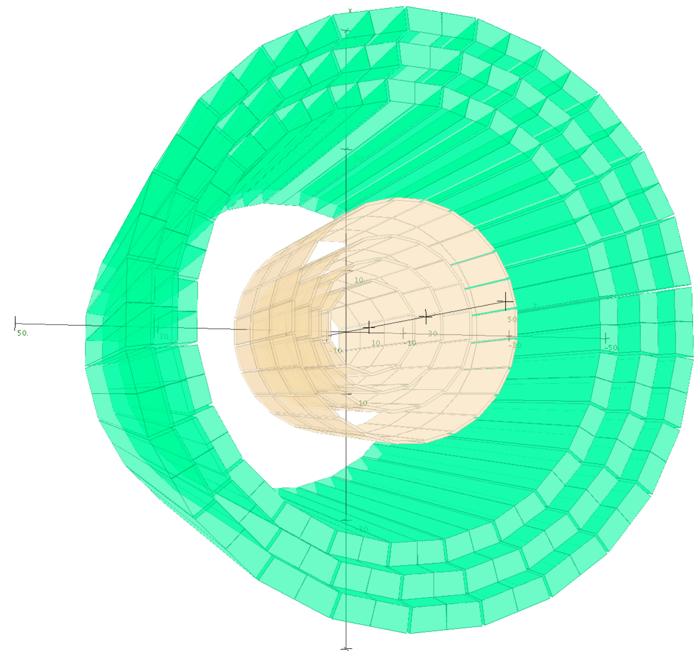


Simulation | Reconstruction



CND Software Status

Daria Sokhan, Gavin Murdoch (Glasgow)

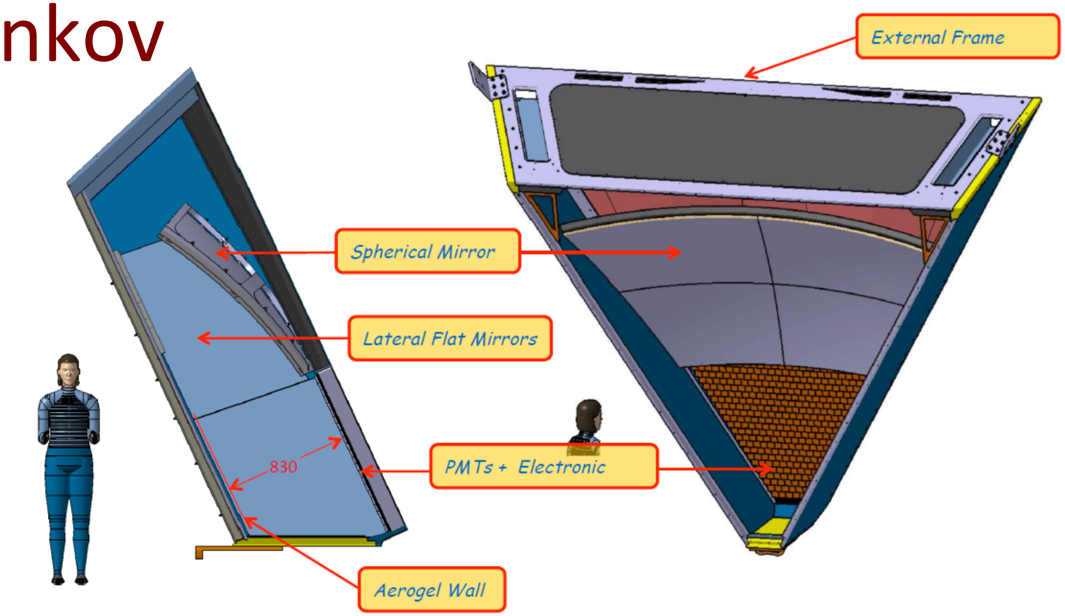


- Geometry and digitization implemented in GEMC
- Implemented in ced (D. Heddle)

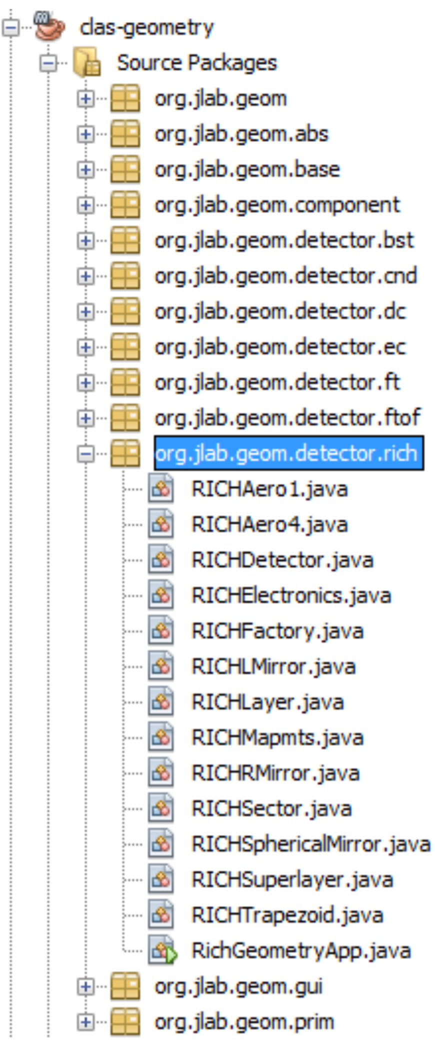
- Reconstruction code developed within COAT-Java framework at Glasgow
  - Algorithms developed originally in ROOT an ported into CLAS12 framework

- ADC to energy algorithm in place
- Time and hit position in paddle reconstruction algorithm in place
  - Identify direct / neighbor signal on basis of timing, reconstruct time and position of hit within the paddle

The Ring Imaging CHerenkov (RICH)

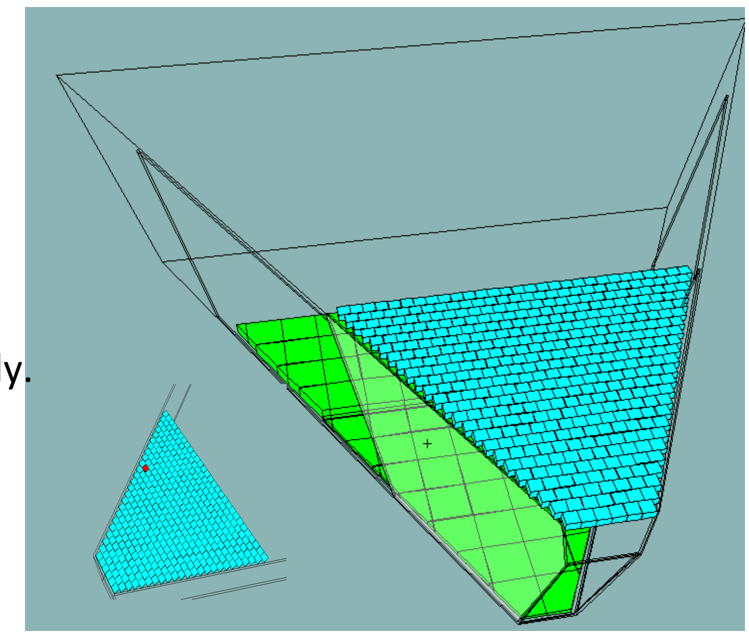


Addition of the RICH to CLAS12 Geometry



- MAPMTs
- Aerogel tiles
- planar mirrors
- Lateral Mirrors
- Electronic panel
- Spherical Mirror: underway
- Ready for material budget study.

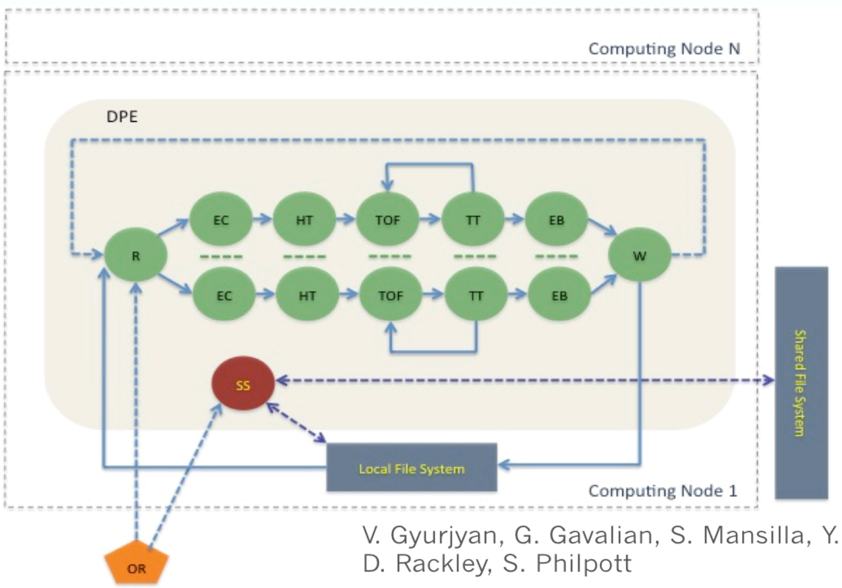
Rotatable, zoomable, translatable and option to choose an mapmt



Work done by F. Benmokhtar and Duquesne's students

Data Challenges

- Production of large data samples with background including 4 tracks generated in the CLAS12 fiducial volumes
- Exclusive access to Haswell machine to run test
- Scaling studies running in multithreaded mode



V. Gyurjyan, G. Gavalian, S. Mansilla, Y. Chen, D. Rackley, S. Philpott

Results of Data Challenges

- 1,500 files
- 30,000,000 events total
- 7 JLAB batch farm nodes (Haswell, total 168/336 cores)
- 28,809 sec. total processing time
- 1,041.6 Hz processing rate
  - Estimated rate : 1,120 Hz (ram disk IO)
  - Time lose due to File IO (local file system IO) and File staging
  - Linear scaling up to 24 cores
- Reconstruction runs 6 ms/event on 24 core Haswell (50 machines to keep up with DAQ)

Milestones and Plans

- 2015 Milestones
  - **January**: ClaRA linear scaling with number of thread
  - **May**: First Release of CLAS12 software package including simulation, reconstruction, analysis and visualization applications available for beta testers
  - **June**: Calibration and Monitoring suite (FTOF, FT, EC plugins under development using suite)
  - **September:** ClaRA switch from cMsg to ZeroMQ (xMsg) [network public subscribe & data transfer protocols]
  - **Mid-December 2015**: Second release
- Second Release
  - Thread-safe
  - More realistic digitization for Calorimeters and FTOF
  - HTCC used for electron ID
  - Improved Central Tracking
  - Improved FT-Cal reco. (energy corrections in place)
- Data Challenges
  - Test full reco. chain running in multithreading mode, DB accessing, scaling...

Code Management

- git repository
  - all projects included in release are now on github
  - GEMC also on git (github.com/gemc)
- Maven for version control & release
  - first release version (1.0) and development version
- Information on downloading the package, running the reconstruction and analyses codes is here:

http://clasweb.jlab.org/clas12offline/docs/software/html/index.html

Summary

- GEMC 2.3 to be released with detector signal model
- Release 2.0 expected to be released right on time for Christmas
  - Ongoing validations, algorithm improvements with cosmics & simulated events
  - Will include HTCC for e- ID
- Stress tests ongoing
  - First test successful
  - Optimization of code for speed planned
- Common tools for analysis (PID cuts)

BACKUP SLIDES

gemc collaboration (gemc.github.io)

How to contribute

Feel free to ask

So let's say that you have an idea for a great feature. It's a good idea to open an issue describing the feature and its implementation and ask the code author's opinion. If they agree, go for it! They might even have some good suggestions for changes or additions to the feature as well.

If it's a bug you found, occasionally it can be ok to just create a **pull request** (PR), as long as it's clearly a bug with a straightforward fix, but it's also not a bad idea to file the bug as an issue first.

Finally, if you want to contribute but are not sure where, you can ask the author if they need help with anything – it could be as simple as helping improve the documentation.

Forking the repo

Ok, so we have a great feature idea (or we found a bug), we opened an issue to check with the author, and they signed off on it. Whoo! Time to get to coding. First thing you do is create a fork, that is a copy of the main repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Forking a repository is a simple two steps process:

1. On GitHub, navigate to the gemc repository.
2. In the top-right corner of the page, click Fork.

You now have a copy of the repo you just forked, available in your GitHub account; its **fork-url-address** can be found on the right menu.

You can create a pull request based on this fork. If you are working on several new features at once, you can create a branch for each feature.

Code Standards

When writing both commits and code, it's important to do so in harmony with a project's existing style. If the project uses camelCase variable naming, this is how you should name your variables as well. If the project has a test suite, you should be writing tests for any changes you make.

Even if you don't agree with some of the author's stylistic decisions, you should adhere to them in your PR. If you have a solid reason why they should be changed, open up an issue and discuss it there. Never ever change an author's existing code style to something you prefer, this is in extremely poor taste.

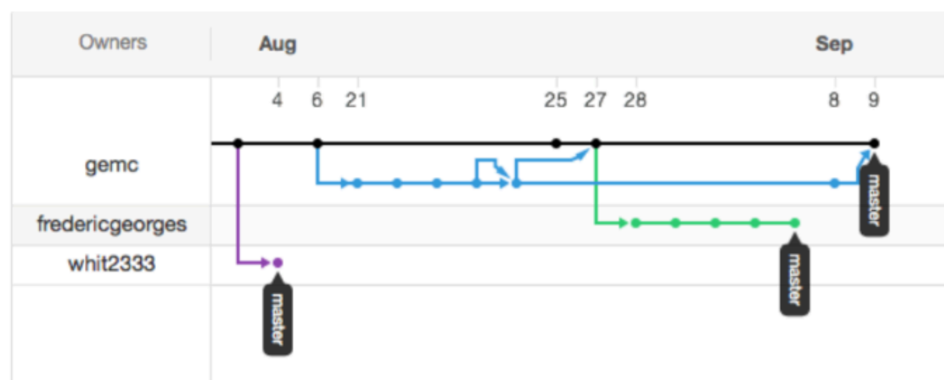
Create a Pull Request

To create the pull request, navigate in github to your fork, and click on the PR button:

You will be presented with a page with a summary of your changes. Once you're ready, go ahead and press the PR button to provide additional informations:

- Make sure you selected the correct branch name ("master" if it's the main fork)
- Make sure the title and description are clear and concise
- If the change is visual, make sure to include a screenshot or gif
- If the PR closes an issue, make sure to put Closes #X at the end of the

1. Create an "issue"
2. Fork
3. Modify
4. Pull request



source: https://github.com/gemc

FADC, multi-hit signal types

Detector Time Window (DTW)

Individual geant4 steps are integrated over DTW

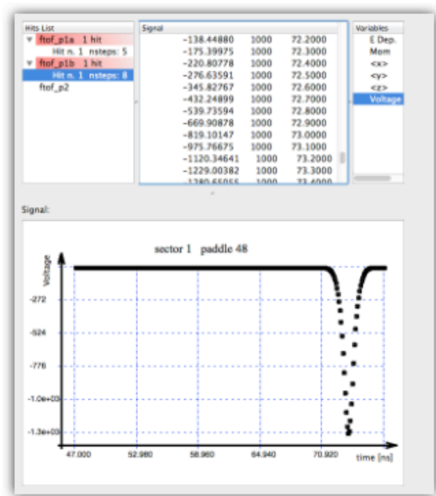
```
> BST  100, 0

  > True Step by Step infos   (101, 0)
      - Edep                  (101, 1)
      - Pid                   (101, 2)
      - positions             (101, 3)

  > Dgtz Step by Step infos   (102, 0)
      - ADCL                  (102, 1)
      - ADCR                  (102, 2)

  > True Integrated infos     (103, 0)
      - Edep                  (103, 1)
      - Pid                   (103, 2)
      - positions             (103, 3)

  > Dgtz Integrated infos     (104, 0)
      - ADCL                  (104, 1)
      - ADCR                  (104, 2)

  > Voltage as a function of time (105, 0)
      - Identifier                 (105, 1)
      - Time                       (105, 2)
      - Voltage                    (105, 3)

  > FADC Bank                (106, 0)
      - Identifier            (106, 1)
      - Time                  (106, 2)
      - Voltage               (106, 3)
```

```
 pid",      "Ri", "ID of the first particle entering the sensitive volume");
"mpid",     "Ri", "ID of the mother of the first particle entering the sensitive volume");
"tid",      "Ri", "Track ID of the first particle entering the sensitive volume");
"mtid",     "Ri", "Track ID of the mother of the first particle entering the sensitive volume");
"otid",     "Ri", "Track ID of the original track that generated the first particle entering the sensitive volume");
"trackE",   "Rd", "Energy of the track");
"totEdep",  "Rd", "Total Energy Deposited");
"avg_x",    "Rd", "Average X position in global reference system");
"avg_y",    "Rd", "Average Y position in global reference system");
"avg_z",    "Rd", "Average Z position in global reference system");
"avg_lx",   "Rd", "Average X position in local reference system");
"avg_ly",   "Rd", "Average Y position in local reference system");
"avg_lz",   "Rd", "Average Z position in local reference system");
"px",       "Rd", "x component of momentum of the particle entering the sensitive volume");
"py",       "Rd", "y component of momentum of the particle entering the sensitive volume");
"pz",       "Rd", "z component of momentum of the particle entering the sensitive volume");
"vx",       "Rd", "x component of primary vertex of the particle entering the sensitive volume");
"vy",       "Rd", "y component of primary vertex of the particle entering the sensitive volume");
"vz",       "Rd", "z component of primary vertex of the particle entering the sensitive volume");
"mvx",      "Rd", "x component of primary vertex of the mother of the particle entering the sensitive volume");
"mvy",      "Rd", "y component of primary vertex of the mother of the particle entering the sensitive volume");
"mvz",      "Rd", "z component of primary vertex of the mother of the particle entering the sensitive volume");
"avg_t",    "Rd", "Average time");
"hitn",     "Ri", "Hit Number");
```

Automatic

sector
SuperLayer
Layer
wire
LR
Doca
SDoca
time
Stime

user-defined



TOF Reconstruction Status

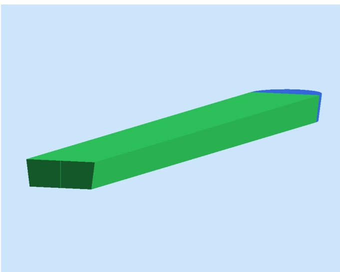
- GEMC now properly simulates gain matching algorithm for FTOF
- FTOF reconstruction package trimmed down version of A. Colvill's code and ported into COAT-Java framework by Gagik (idem CTOF)
- Current algorithm does independent clustering but no linking between panel 1A and 1B
- No calibrations constants incorporated into the code at this stage (idem CTOF)

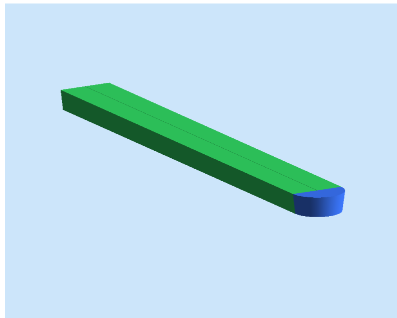
- Linking algorithms and use tracking information to improve matching to be completed…

CND geometry in GEMC

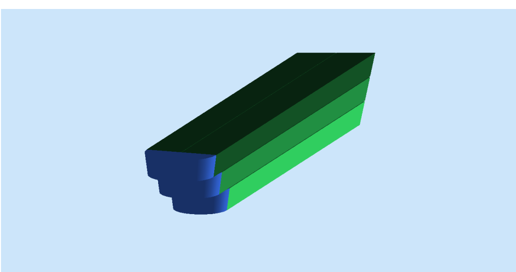
Daria Sokhan, Gavin Murdoch (Glasgow)

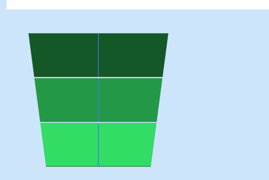
Trapezoid scintillator paddles (green) assembled in pairs and coupled with u-turn light guide (blue) at downstream end.



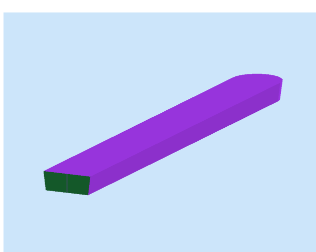


Three layers of pairs form a segment (block):

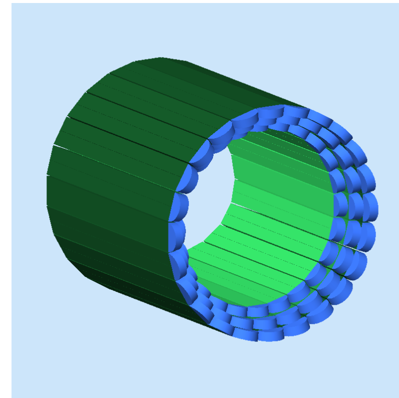


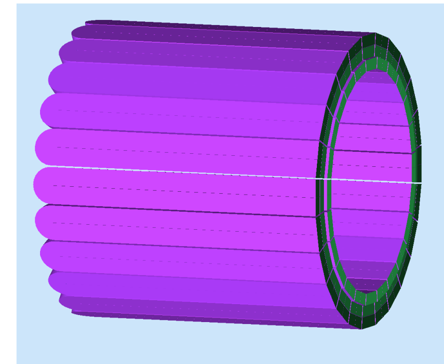


The paddles are additionally wrapped in Al (magenta):

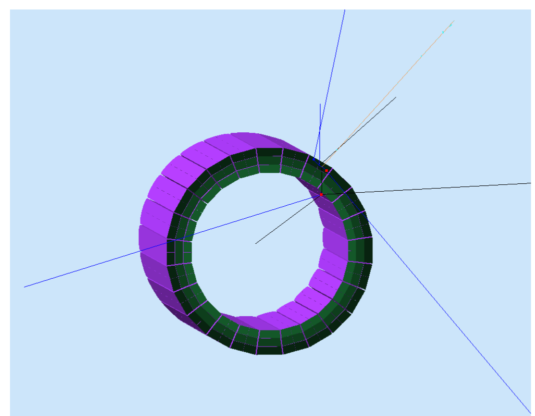


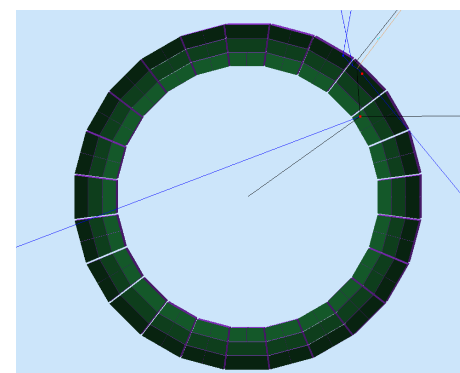
28 blocks assemble to form the full CND barrel:





Full CND geometry has been implemented in GEMC since v. 2.2





Hit digitisation currently being ported into Java.

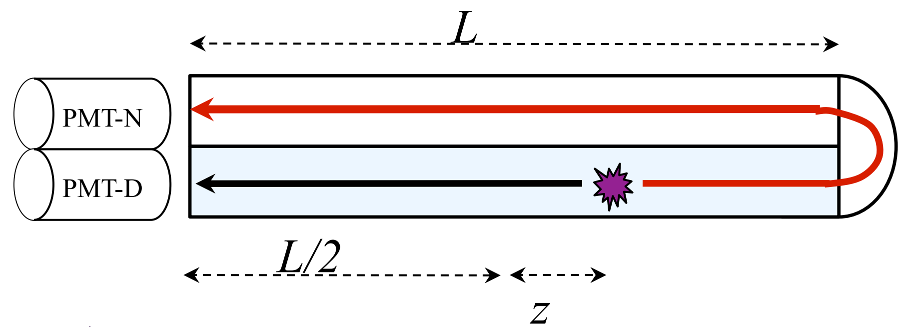
CND reconstruction

Daria Sokhan, Gavin Murdoch (Glasgow)

★ Convert TDC and ADC counts into time and energy:

$$t_{D,N} = \frac{TDC_{D,N}}{C_{TDC}} - t_{offset} \qquad E_{D,N} = \frac{ADC_{D,N} - P}{C_{ADC} Y G Q q}$$

C_{TDC}: Conversion from TDC channels to time
t_{offset}: electronic time offsets between the paddles
P: ADC pedestal
C_{ADC}: Conversion from ADC channels charge
q: charge of an electron
G: gain of the PMT
Q: quantum efficiency of the PMT
Y: light yield in scintillator (optical photons per deposited energy)



★ Identify direct / neighbour signal on basis of timing, reconstruct time and position of hit within the paddle:

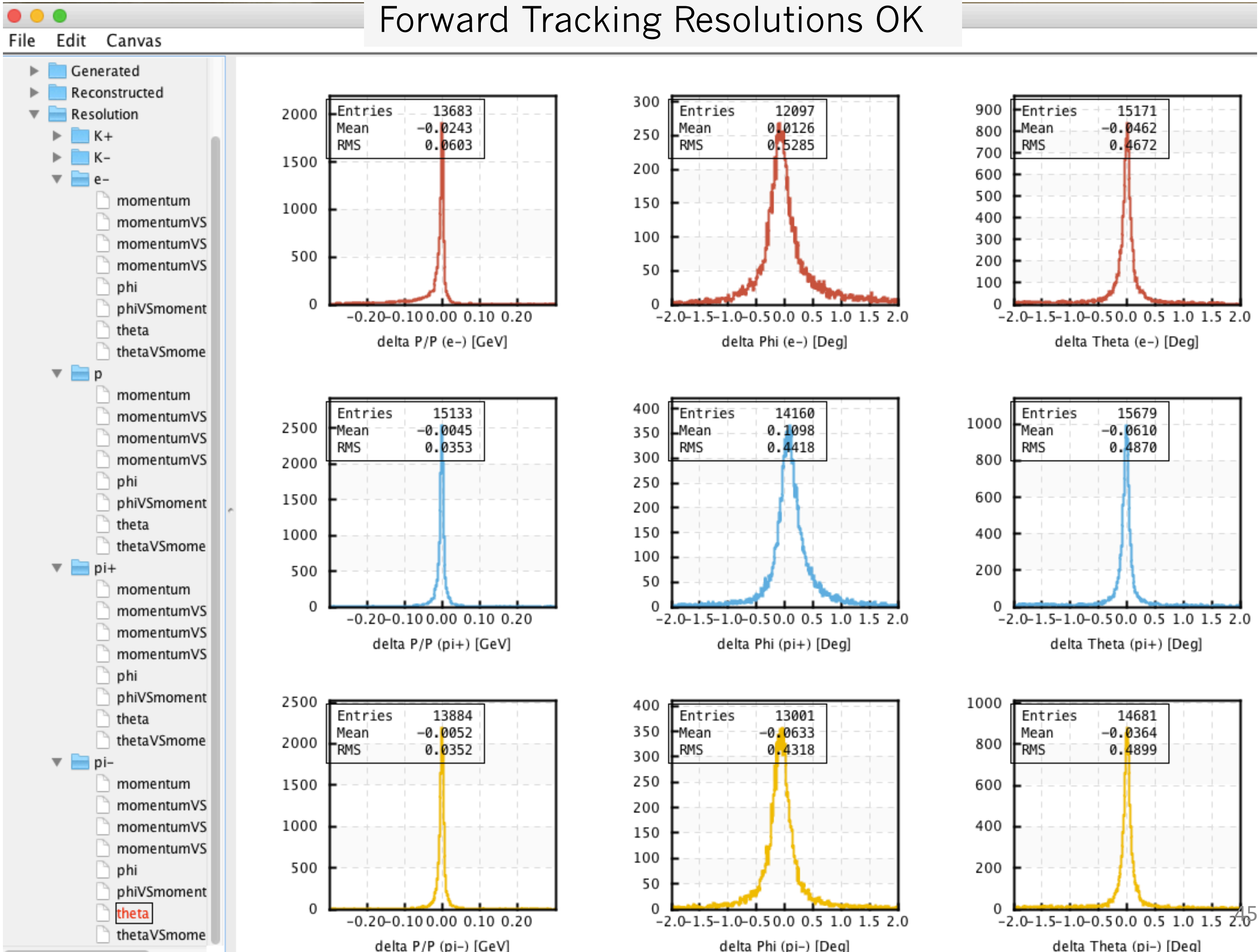
$$t = \frac{1}{2}(t_D - T_D + t_N - T_N - t_u) - \frac{L}{2}\left(\frac{1}{v_{effD}} + \frac{1}{v_{effN}}\right) \qquad z = \frac{v_{effD}}{2}(t_D - T_D - t_N + T_N + t_u) - \frac{L}{2}\frac{v_{effD}}{v_{effN}}$$

L: length of paddle
t_u: time for light signal to travel round the u-turn
$T_{D,N}$: time for light signal to travel through long lightguides
$v_{effD,N}$: effective velocity of light in the scintillator

Reconstruction code is functional in ROOT, being translated into Java

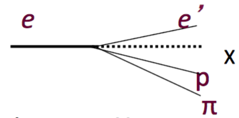
Validations

Forward Tracking Resolutions OK



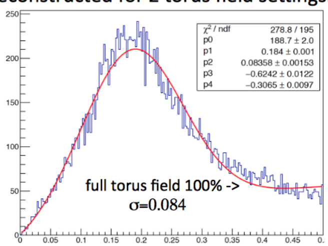
Validations (Version 1.0)

CLAS12 reconstruction: physics impact

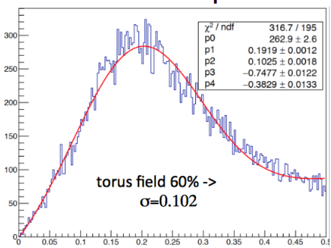


The same set of exclusive events were reconstructed for 2 torus field settings.

exclusive epπ+X



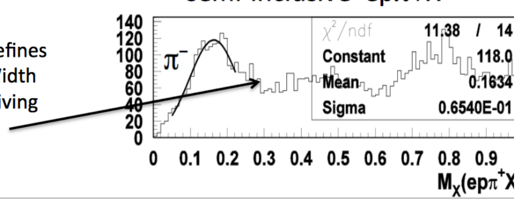
$M_X(ep\pi+X)$



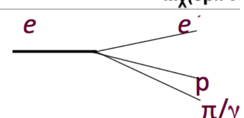
$M_X(ep\pi+X)$

semi-inclusive epπ+X

Resolution in the missing mass defines the signal to background ratio. Width increases by 20%(60% field) or giving ~15% increase in time

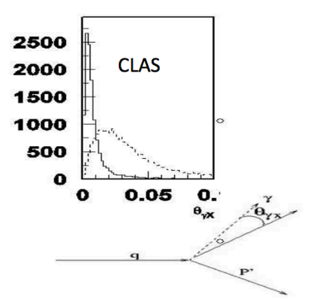


CLAS12 reconstruction: physics impact

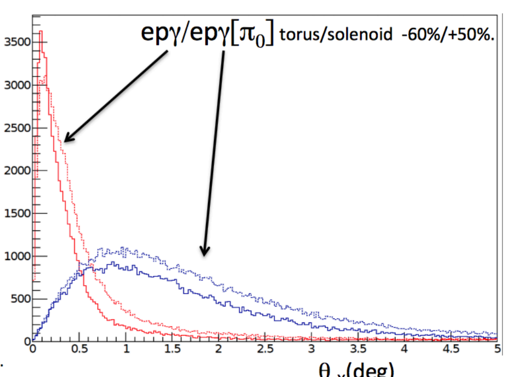


The same set of exclusive events were reconstructed for 2 configurations
torus/solenoid -100%/+100% (solid)
torus/solenoid -60%/+50%. (dashed)

exclusive epγ/epπ$_0$



Background rejection in DVC analysis relies on the cut on the angle between measured and calculated (from electron and proton) photons.



Lower field option may be even better for DVCS studies.

CLAS12 software chain validation

H. Avakian, R. De Vita, J. Gilfoyle, A. Kim

CLAS Collaboration Meeting, June 17, 2015

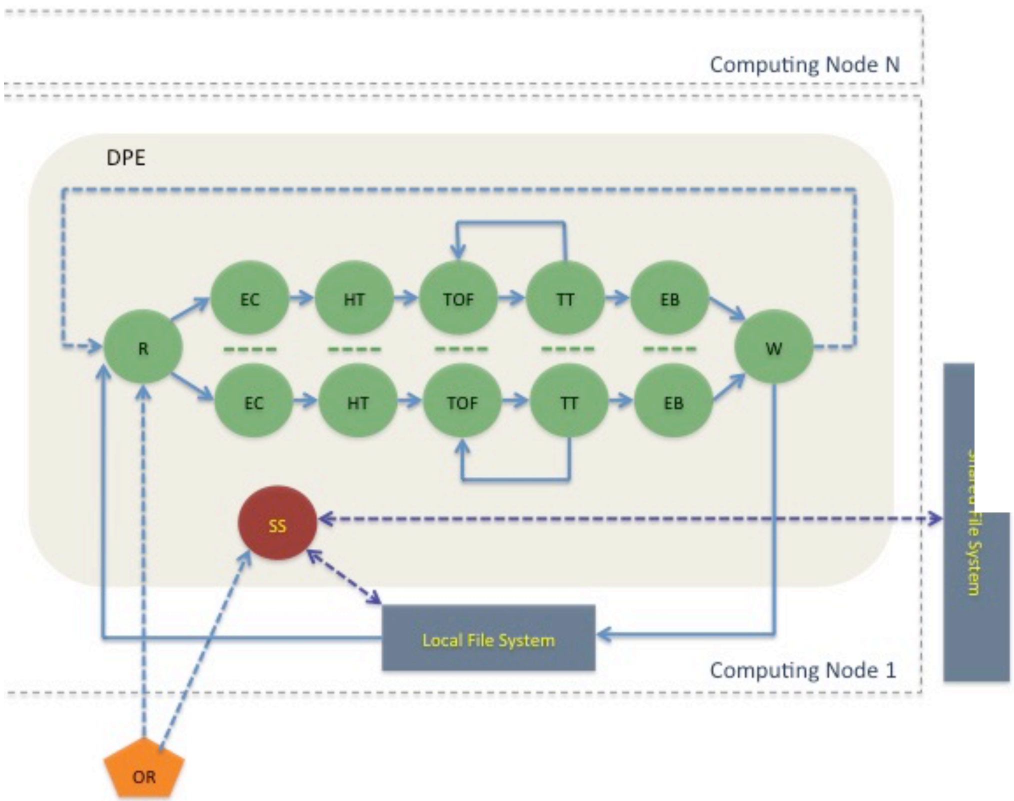
TODO list

- Preparation for a full scale test/challenge:
  - improve central tracking ✓
  - improve low momentum track reconstruction ✓
  - improve neutral reconstruction ongoing
  - micromegas geometry checks required
  - optimize reconstruction in strong background conditions ongoing

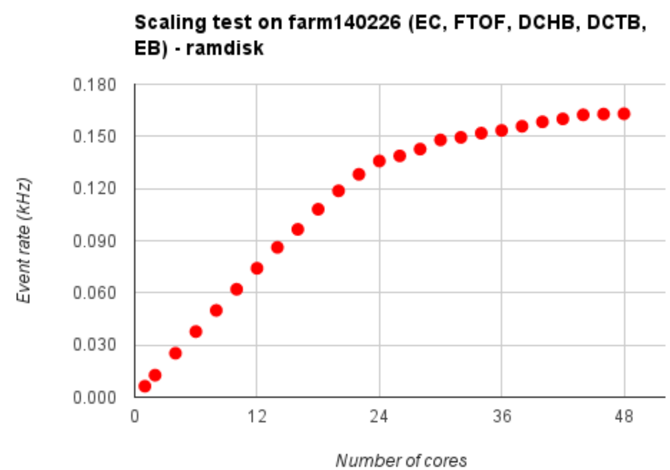
- Repeat for Version 2.0
- Redo low momentum tracks reconstruction analysis with new Central reconstruction code

Stress Tests Results

- **J-Lab batch system Clara deployment and data flow: data staging service (SS), reader service (R), electromagnetic calorimeter reconstruction service (EC), charge particle hit based tracking (HT), time of flight reconstruction (TOF), charge**

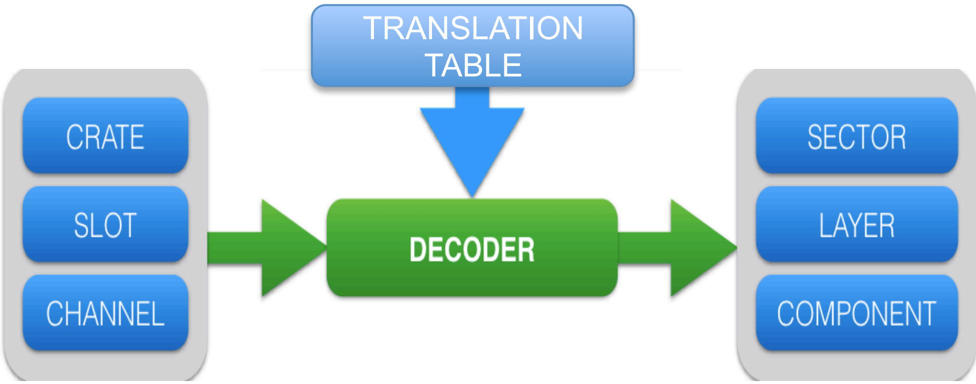


- **Multi-core stress tests of coatjava-2.0 on the node farm140226**
- **Consistent with expectations for compute intensive services on hyper-threaded cores**



S. Mancilla

Common Tools Packages



- ✓ **Standard Raw evio composite → Jevio bug fix**
- ✓ **Detector Groups provide Translation Tables**
- ✓ **Decoder new bank format**

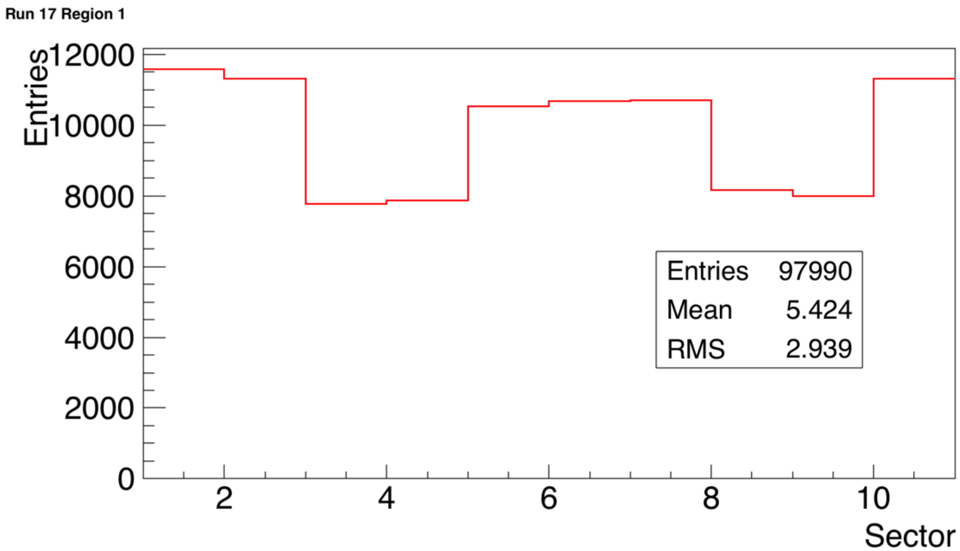
G. Gavalian

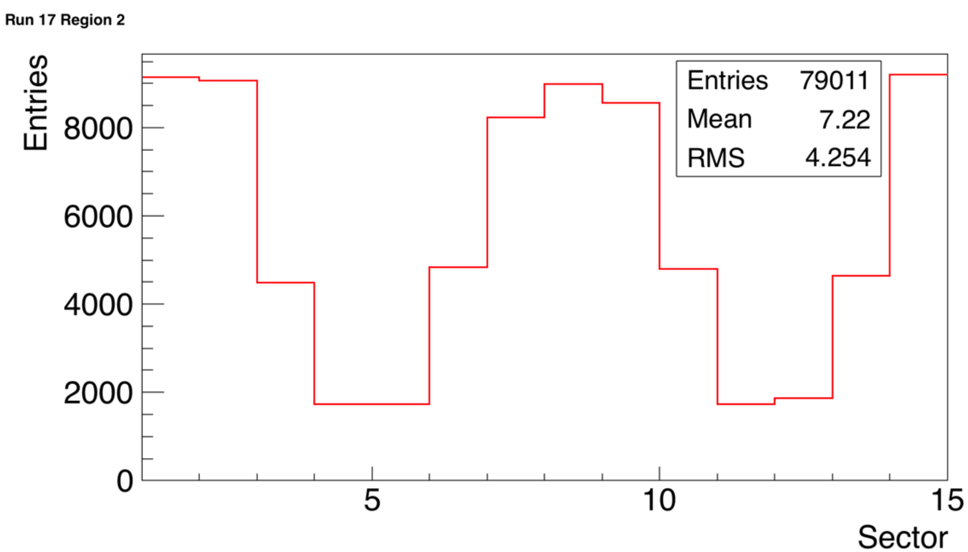
FORMAT, by columns:
1) name of the detector
2,3,4) Crate / slot / channel (notice that previously first 3 numbers were sector, layer, component)
5,6,7) sector / layer / component
8) (0 - ADCL, 1 - ADCR, 2 - TDCL, 3 - TDCR)

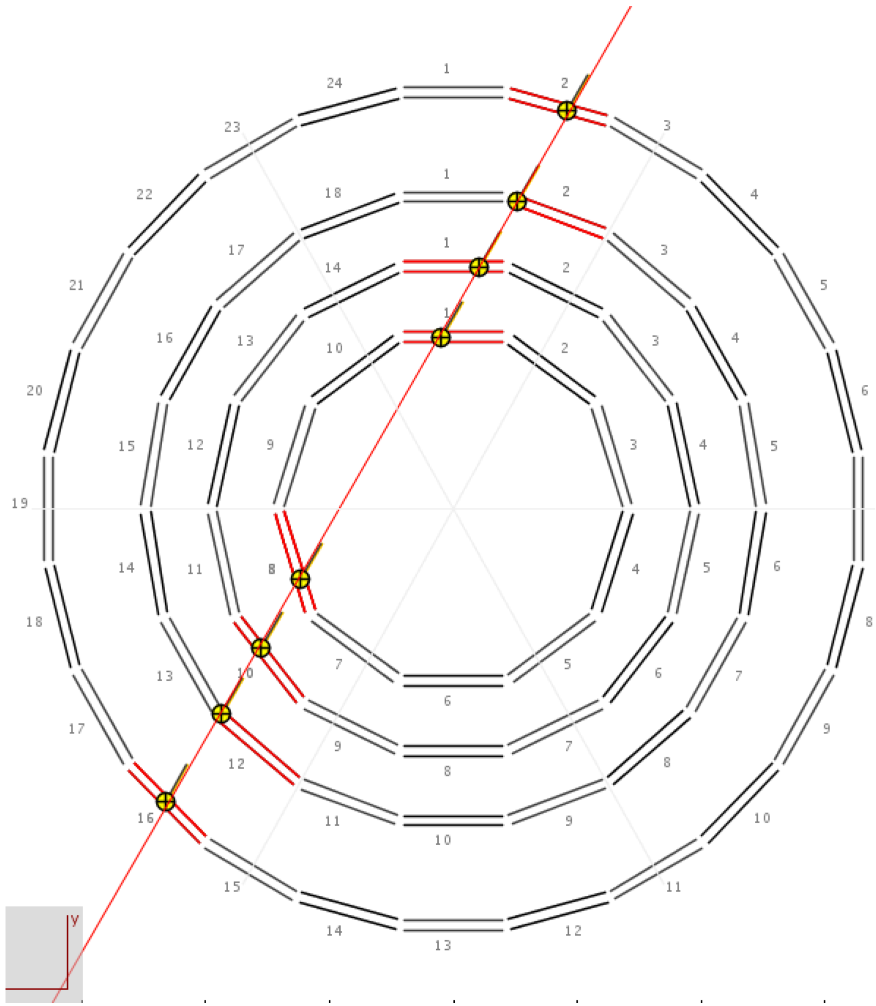
```
#------------------------------------------------------------------------
# TRANSLATION TABLE
#------------------------------------------------------------------------
# Detector - CRATE - SLOT - CHANNEL - SECTOR - LAYER - COMPONENT - ORDER
# ORDER 0 = ADCL, 1 = ADCR, 2 = TDCL, 3 = TDCR
#------------------------------------------------------------------------
CTOF    68    13    0    1    1    1    0
CTOF    68    13    1    1    1    1    1
CTOF    68    14    0    1    1    1    2
CTOF    68    14    1    1    1    1    3
#------------------------------------------------------------------------
# END OF Translation TABLE
```


48

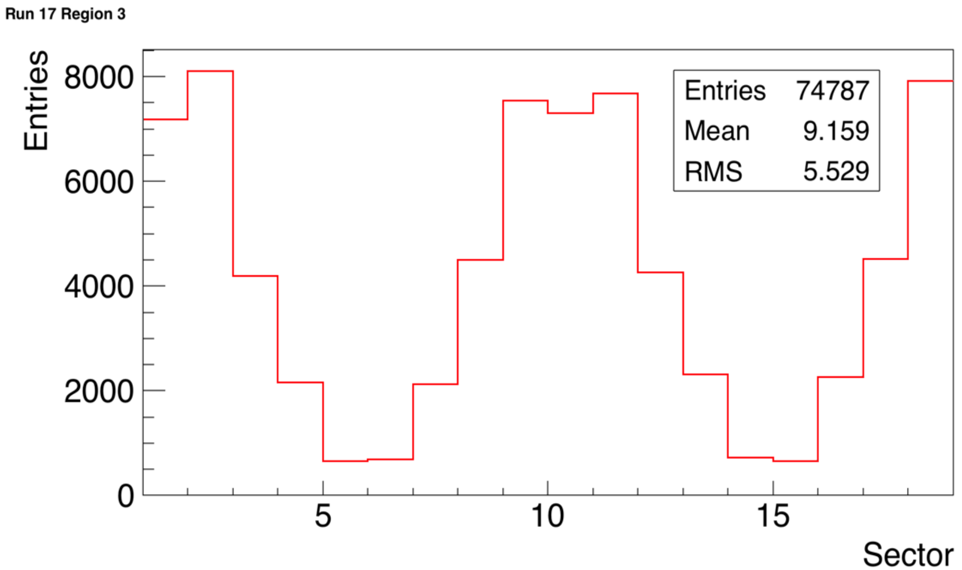
Sector Occupancy for Events with 8 OnTrack Crosses

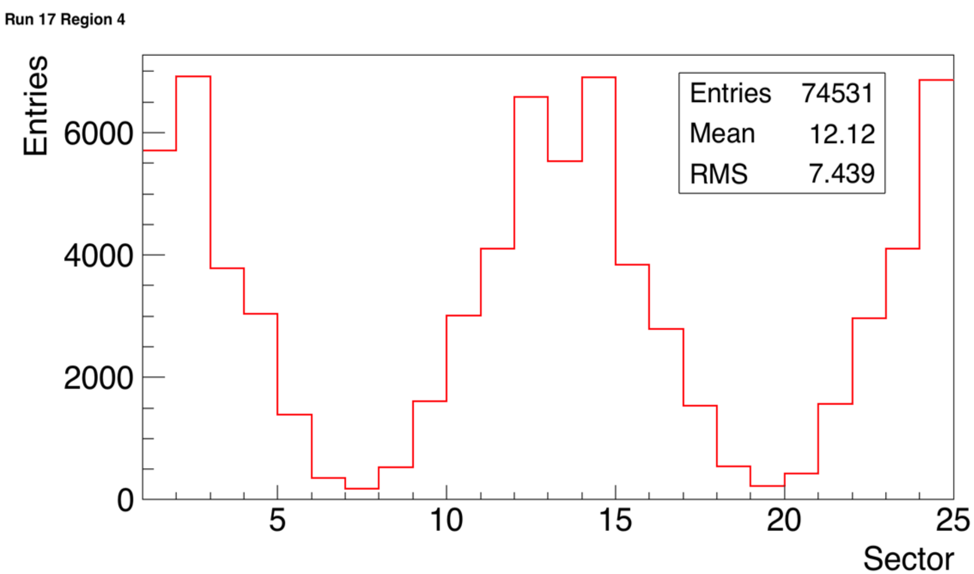


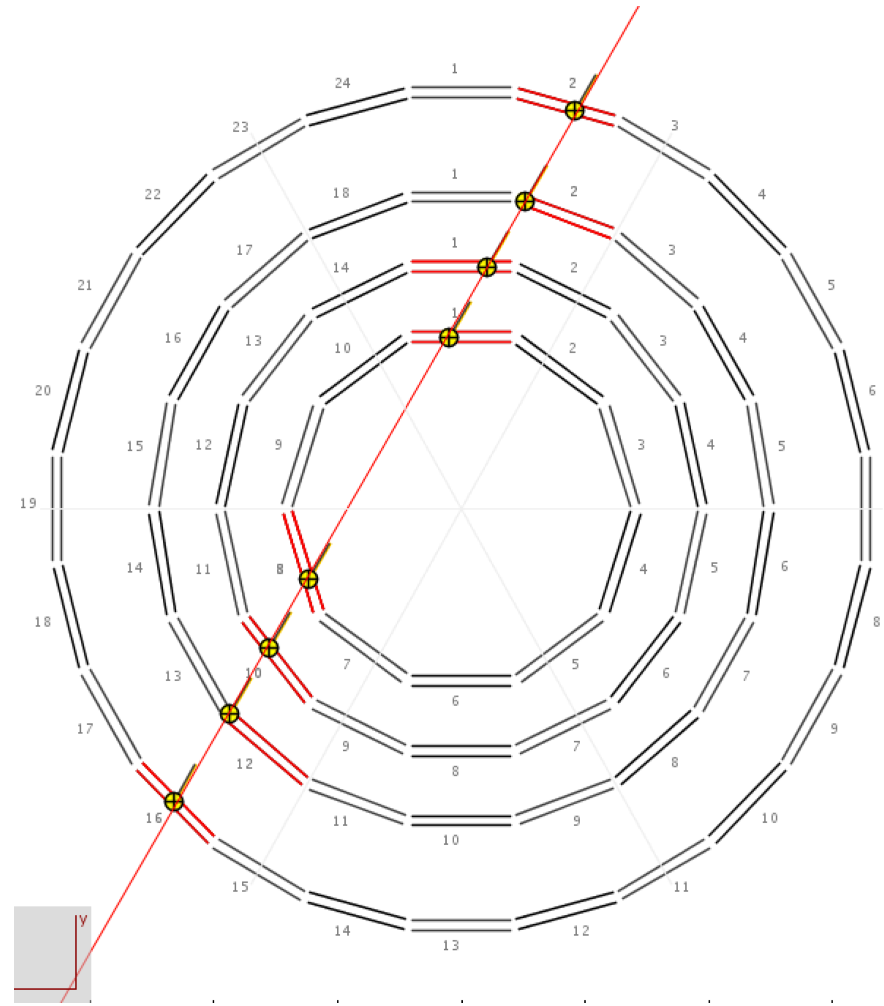




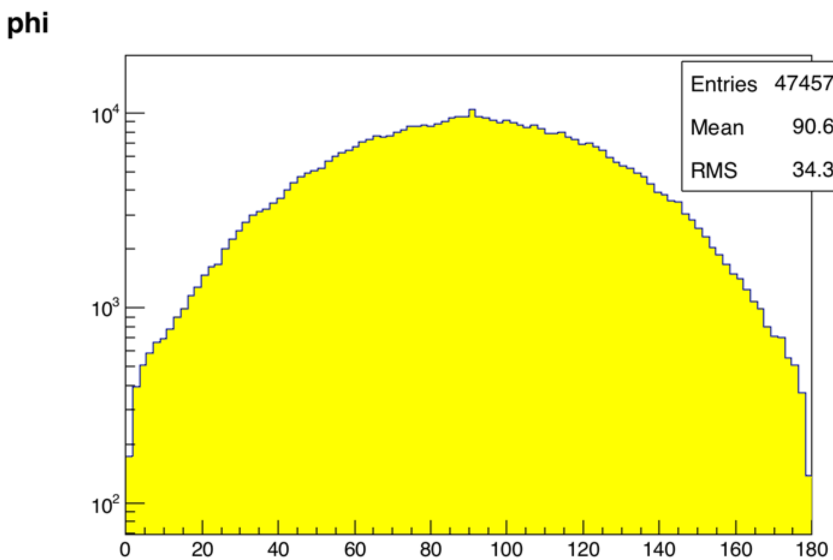
Sector Occupancy for Events with 8 OnTrack Crosses

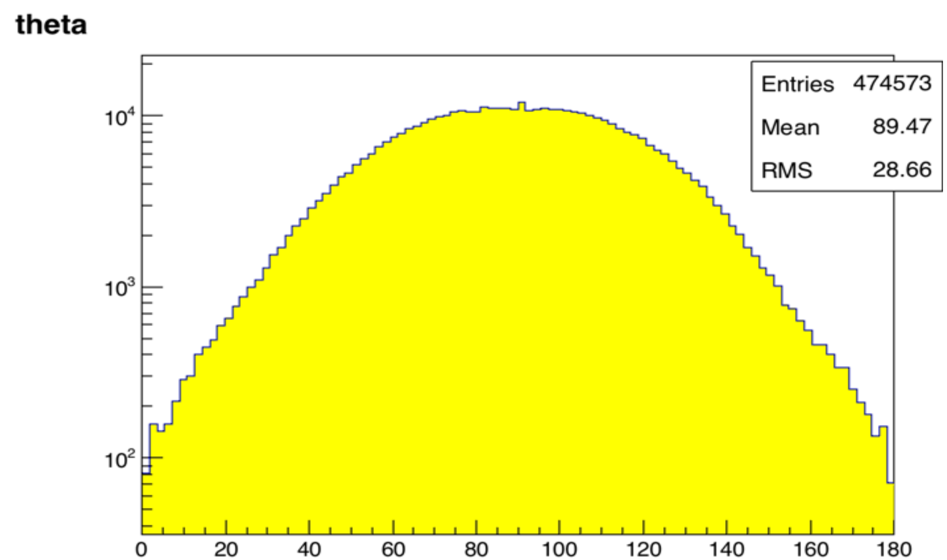


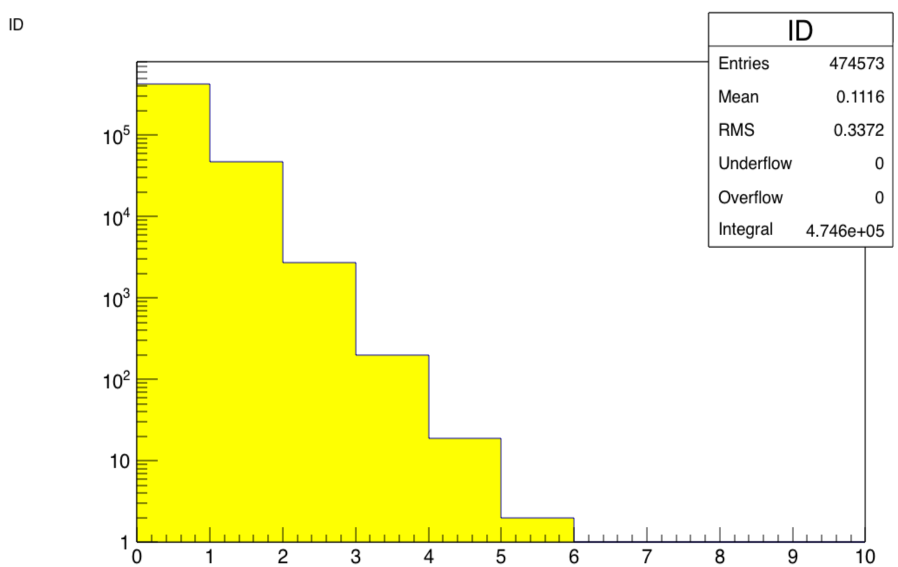


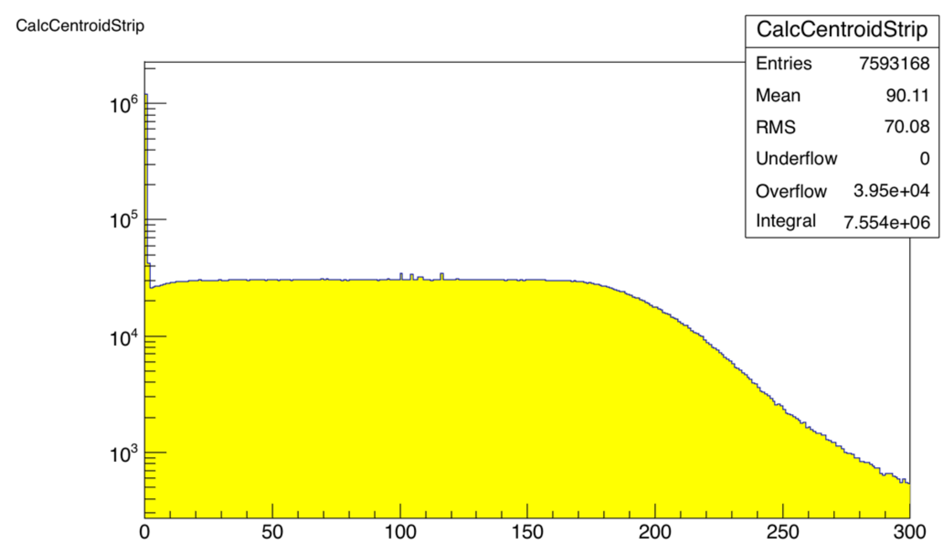


SVT Cosmic Data Validation

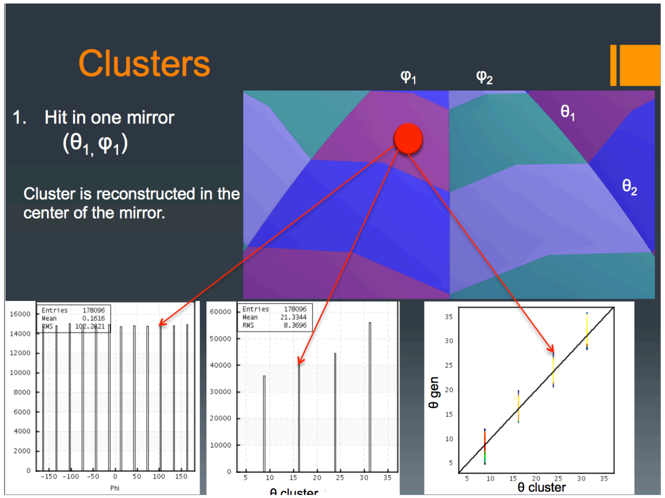


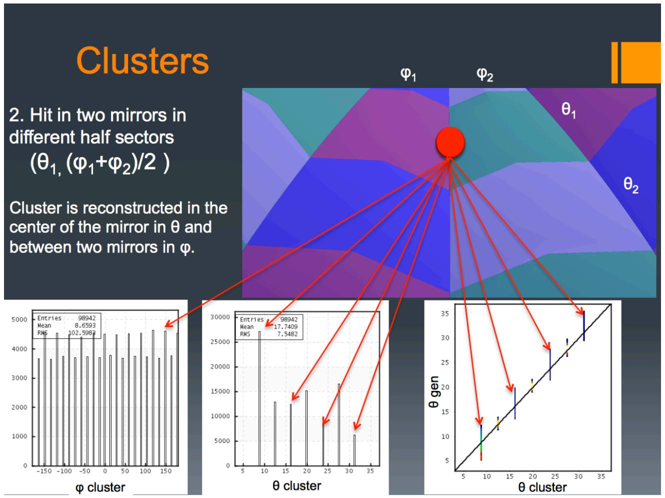


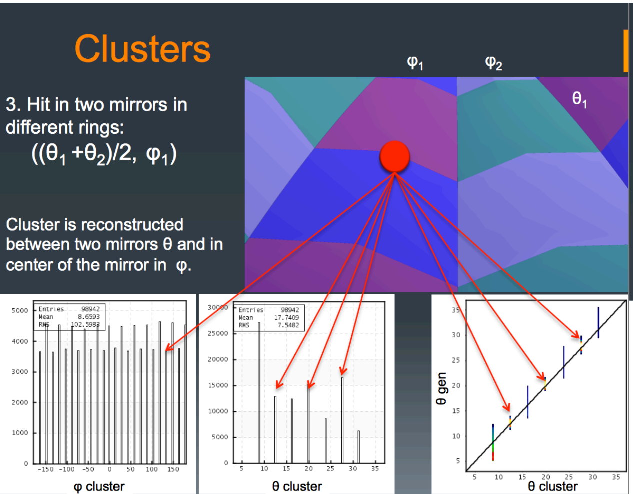


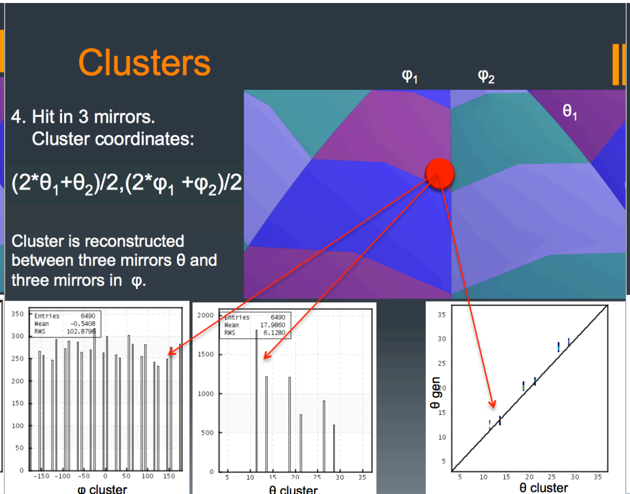


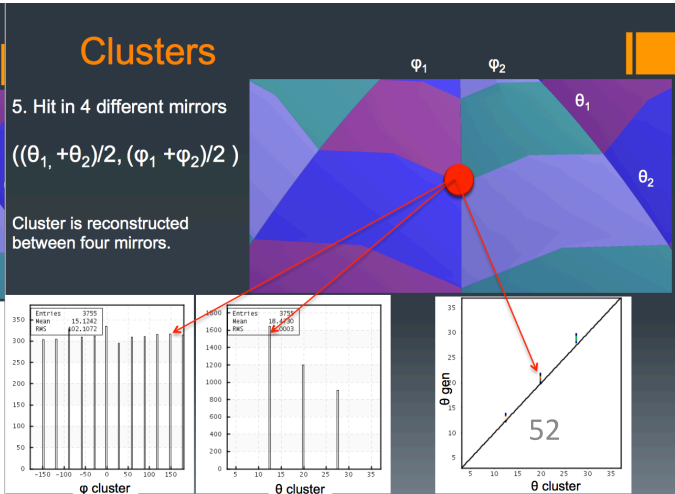
HTCC Clustering algorithms in place and being validated





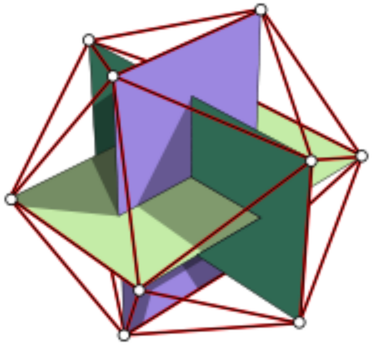


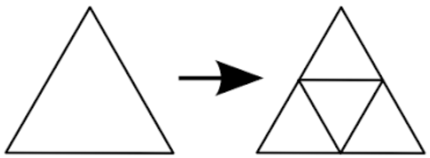


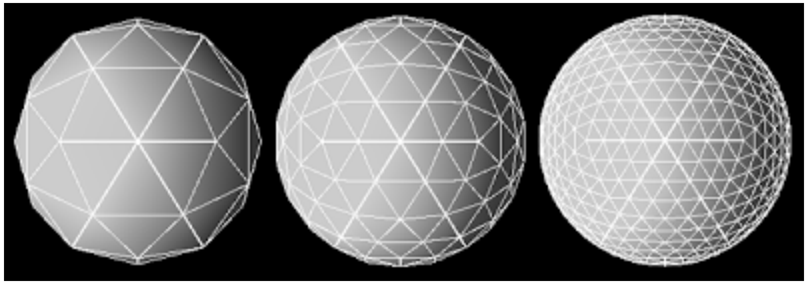


Programming of the spherical mirror is underway:

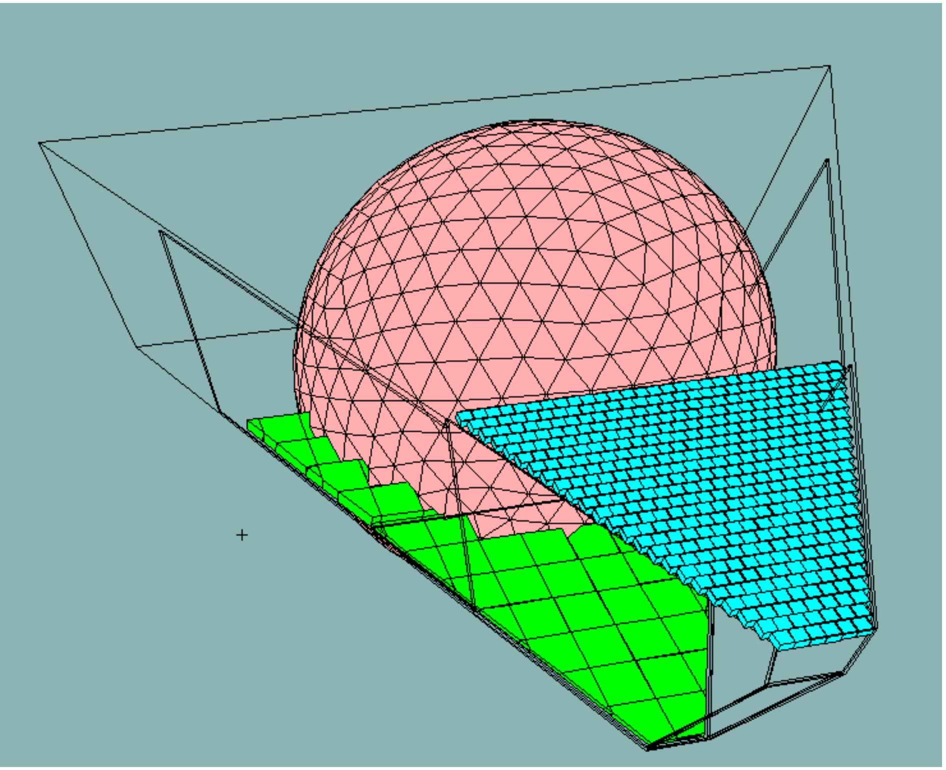
Idea:







Results:



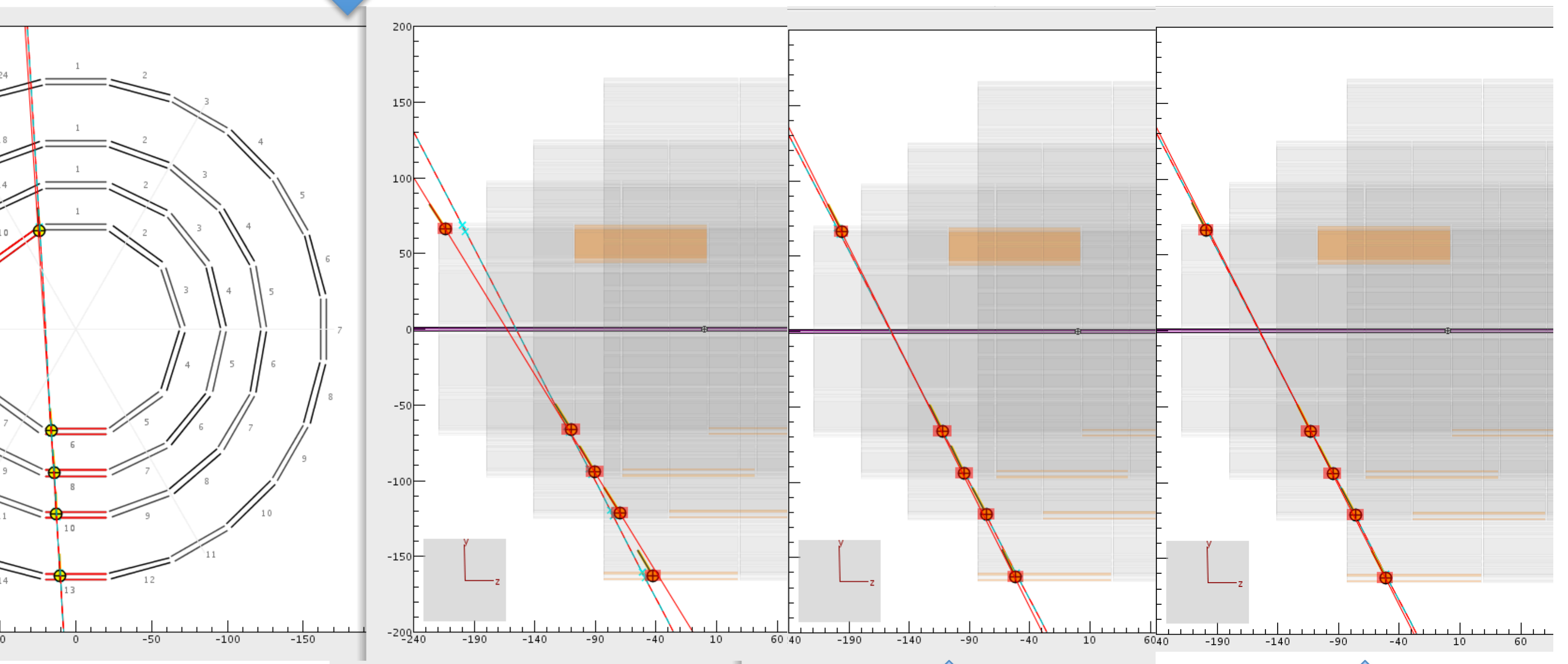
Succeeded to add the Sphere class to CLAS12 geometry, we can make it big or small, move it around, but didn't succeed to cut it yet.

Work done by F. Benmokhtar
and Duquesne's students

Reconstruction Algorithms Refinements

issues with corner clippers resulting in poor z resolution

Ongoing code validation, debugging, algorithm improvements using cosmic data simulations



remove corner clipper from global fit

Include in fit to strip indexes using a KF