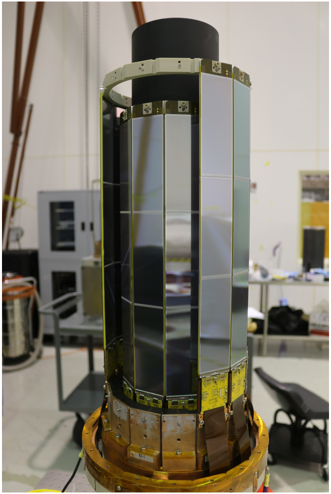
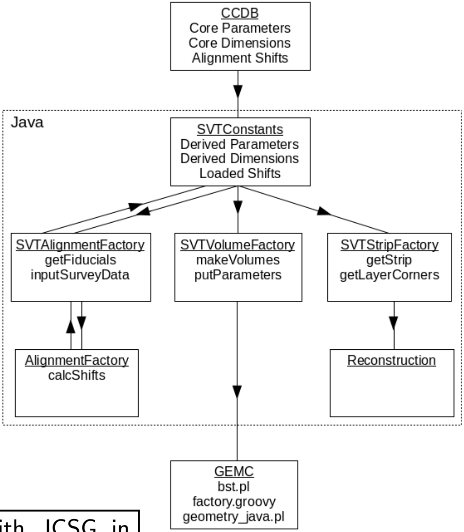
SVT Track-Based Alignment

1. Goal: Align the SVT to reach the resolution design specification of $\approx 65\ \mu m$.
2. Build accurate and complete representation of the SVT geometry and materials as part of the CLAS12 Common Tools.
3. Provide the geometry for the *gemc* simulation and the CLAS12 reconstruction from a common set of parameters.
4. Develop algorithms to measure and correct misalignments in the SVT.
5. Document it.



SVT Geometry Package Classes

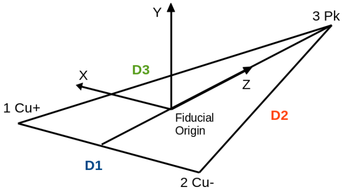
1. Main author: Peter Davies, University of Surrey
2. SVTConstants
   - Connects to CCDB.
   - Loads core parameters.
   - Loads alignment shift data from file.
3. SVTStripFactory
4. SVTVolumeFactory
5. SVTAlignmentFactory
   - Fiducial points.
   - I/O for alignment data.
6. AlignmentFactory
   - Applies alignment shifts to points and volumes.
7. Util
8. Matrix

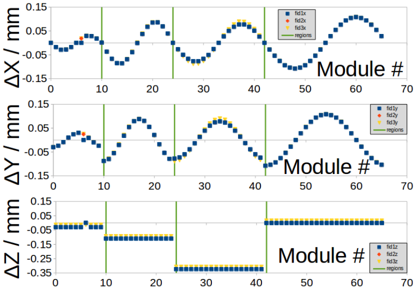


Merged with JCSG in Common Tools

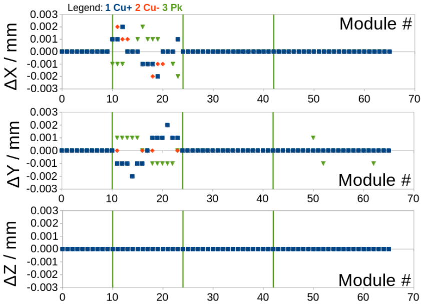
Assessment and Validation

Surveyors measured three fiducial points on each module - CU+. CU-. and Pk.





Comparison of fiducial data: `Factory Ideal` from `Survey Ideal` before corrections.



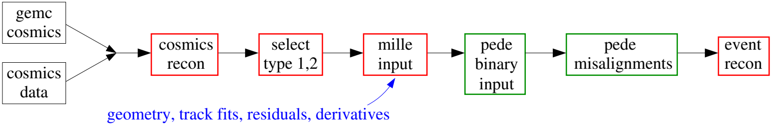
Comparison of fiducial data: `Factory Ideal` from `Survey Ideal` after corrections.

Next Steps for SVT Geometry Package

1. Finish the CLAS-NOTE.
2. Make the code thread safe.
3. Tune the geometry to eliminate Geant4 overlap warnings for small (10's of microns) overlaps.
4. Combine many files into a few.
5. Replace redundant utilities.
6. Test with cosmics (simulated and measured).
7. Test with events from the target (simulated).
8. Write code to correct misalignments.

SVT Track-Based Alignment

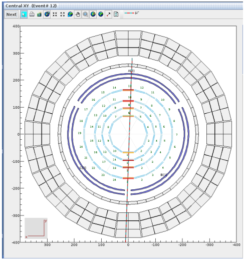
1. Track-based alignment of SVT requires fitting many parameters - up to 792 here.
2. Program `millepede` does linear least squares with many parameters.
   - Matrix form of least squares method.
   - Global parameters the geometry misalignments. Same in all events.
   - Local individual track fit parameters. Change event-to-event.
   - Requires first partial derivatives of residuals with respect to the local (fit) parameters and global parameters (geometry misalignments).
3. Analysis chain: red boxes - Java; green boxes - C^{++}.

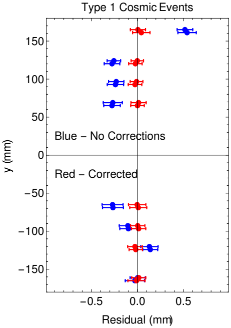


4. Full chain has been tested and validated using *gemc* simulation and cosmic data for simplified case (Type 1 events).

Some Results and Next Steps

1. Algorithm applied successfully to measured Type 1 cosmics.
2. Works on *gemc* cosmics with shifted regions.
3. Status:
    - Type 2 events selected. Algorithm for Type 2's has problems.
    - *gemc* version 4a.0.2 in use.
    - Java/Groovy scripts updated to coatjava 4a.3.0.
4. Next steps:
    - `millepede` codes are built for Centos 6.5 - need to upgrade to Centos 7.
    - Get Type 2 algorithm working.
    - Test with cosmics (simulated and measured).
    - Test with *gemc* events from the target.


Central XY (Event# 12)
Next
400
300
200
100
0
-100
-200
-300
-400


Type 1 Cosmic Events
y (mm)
Residual (mm)
Blue – No Corrections
Red – Corrected