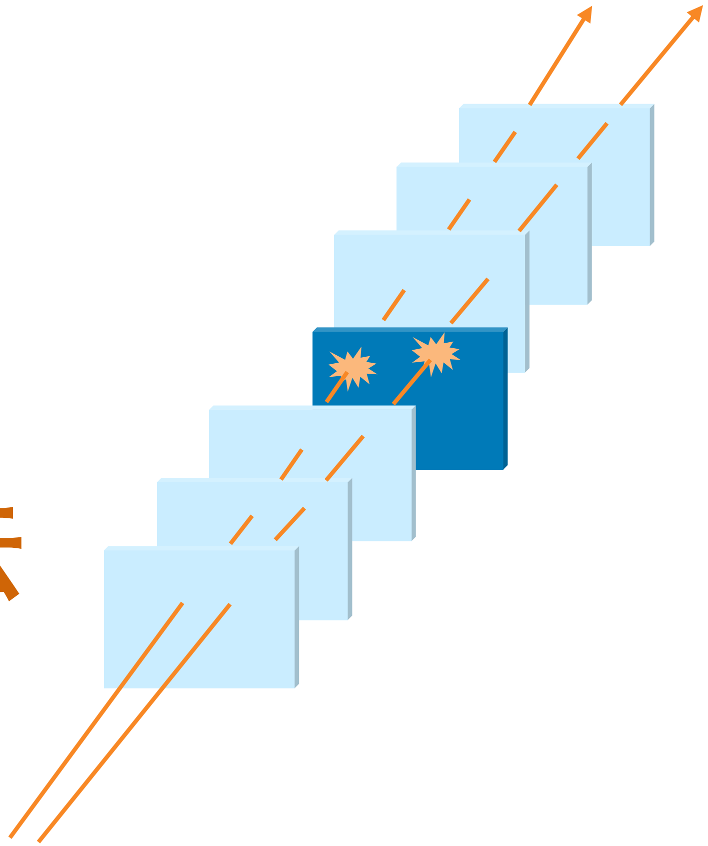
束流测试中的数据分析方法



艾小聪 (ZZU)

测试束流与先进探测技术研讨会 (TB&D), 郑州·登封, 2025-08-15

Test beam is about detector prototyping

- High-energy particle beam with known type, energy, direction, and intensity is widely used for characterizing particle detectors in realistic environment
  - Hit efficiency
  - Position resolution
  - Energy resolution
  - …

Will focus on test beam with particle telescope in this talk

Particle beam telescope is a tracker

More in Y. Liu's talk

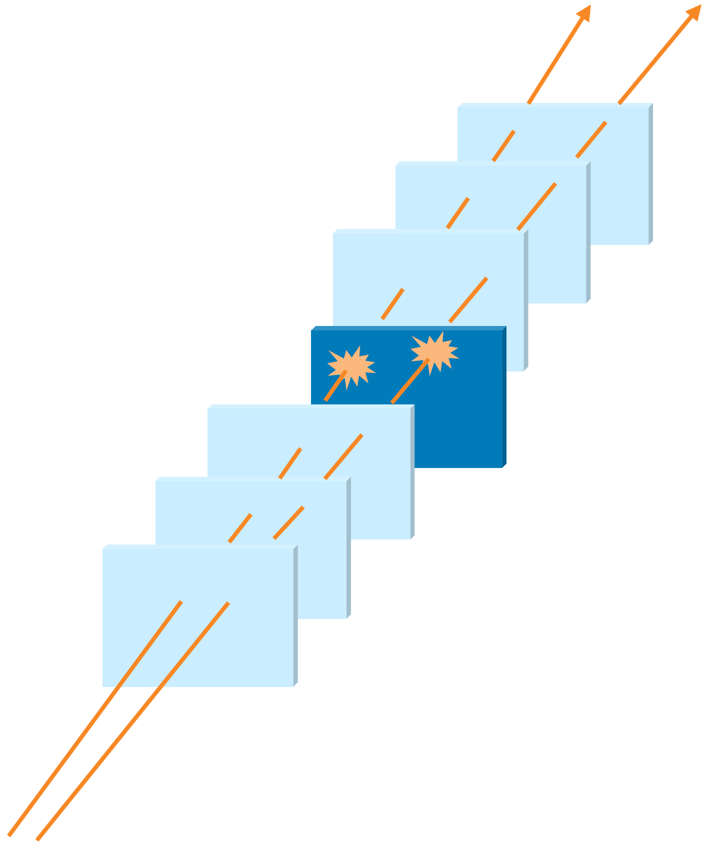
ALPIDE telescope at DESY

Yi Liu *et al* 2023 *JINST* **18** P06025

- Basically a tracking system composed of a few silicon pixel detector planes
- To reconstruct trajectories of beam particles with high spatial precision
    - Usually no magnetic field
- The position/counting of beam particles can be used to characterize the spatial resolution/hit efficiency of a tracker module (and beyond)

Telescope beam test is about tracking

- Tracking with particle beam telescope (usually a simple detector geometry) with low track multiplicity seems trivial
- But things can be more tricky than it sounds like
  - The device being tested is not always a silicon pixel detector
    - Though the telescope planes are usually pixel
  - Multiple scattering is not negligible
  - The detectors can be poorly aligned (i.e. misaligned geometry)

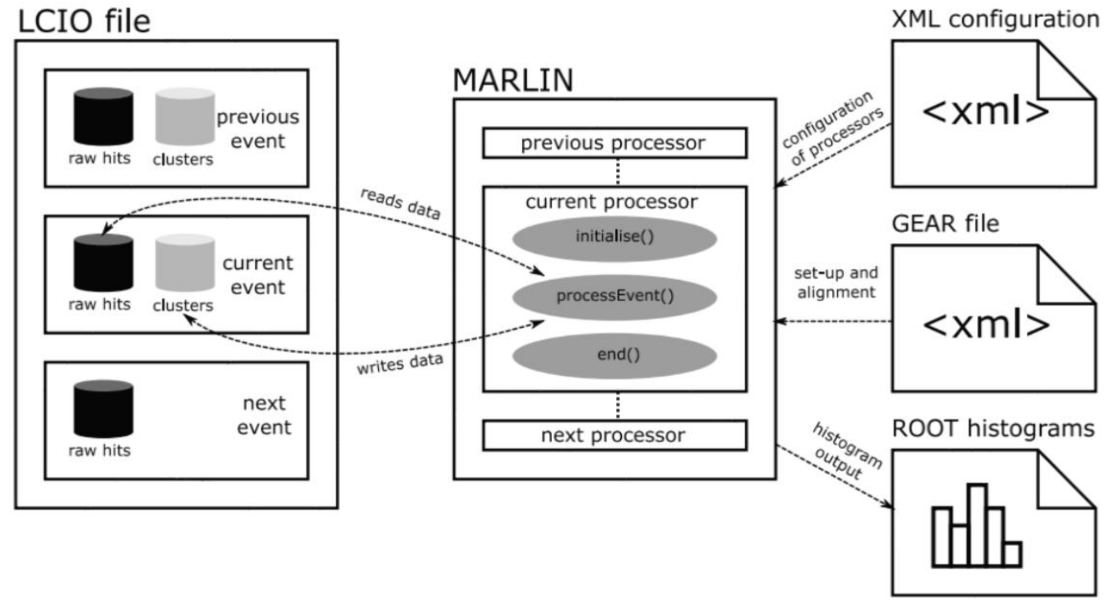


Earlier experience with EUTelescope

About EUTelescope

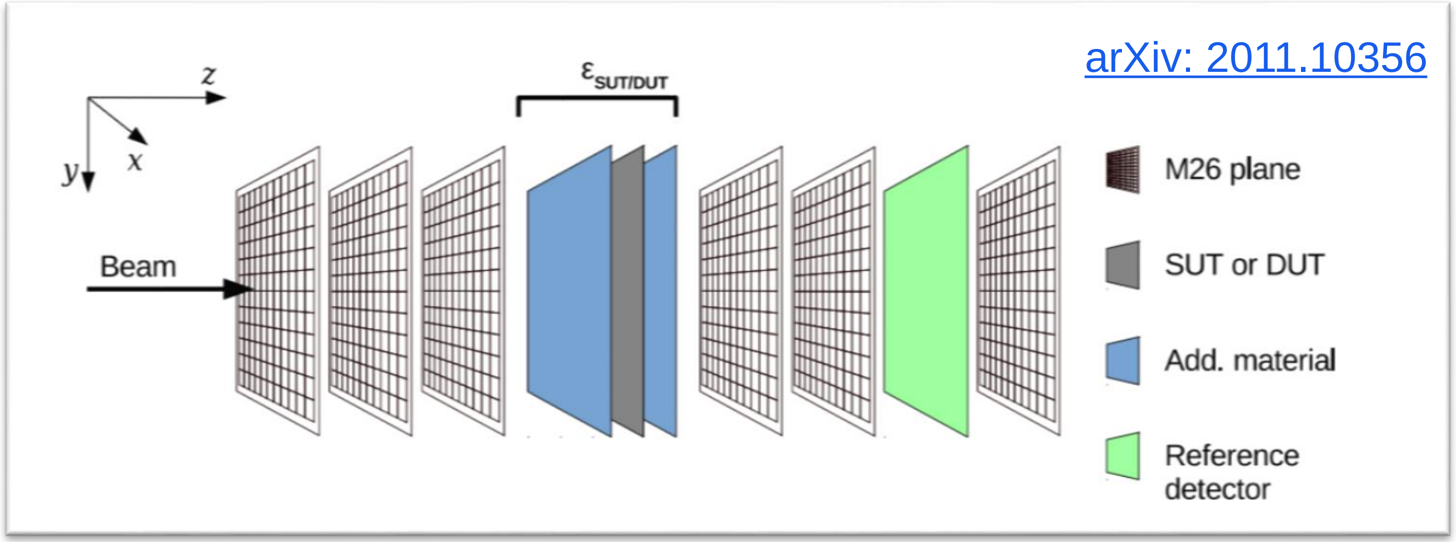
- A legacy software for beam test data reconstruction developed at DESY
  - Heavy external dependency (LCIO, MARLIN, GEAR, ROOT, EIGEN, GBL …)
    - As far as I know, not so many people managed to install it successfully…
- Obsolete. Not maintained any more
  - Last update was 5 years ago

[arXiv: 2011.10356](https://arxiv.org/abs/2011.10356)



EUTelescope workflow

- Duplet finding → triplet finding for left and right arms → left triplet and right triplet match → track fitting (General Broken Lines) → alignment with MillePedeII → iteration of above steps
- For MIMOSA telescope, additional Time reference detector (FEI4) is needed to select tracks matched with FEI4 → match track with Device Under Test (DUT)

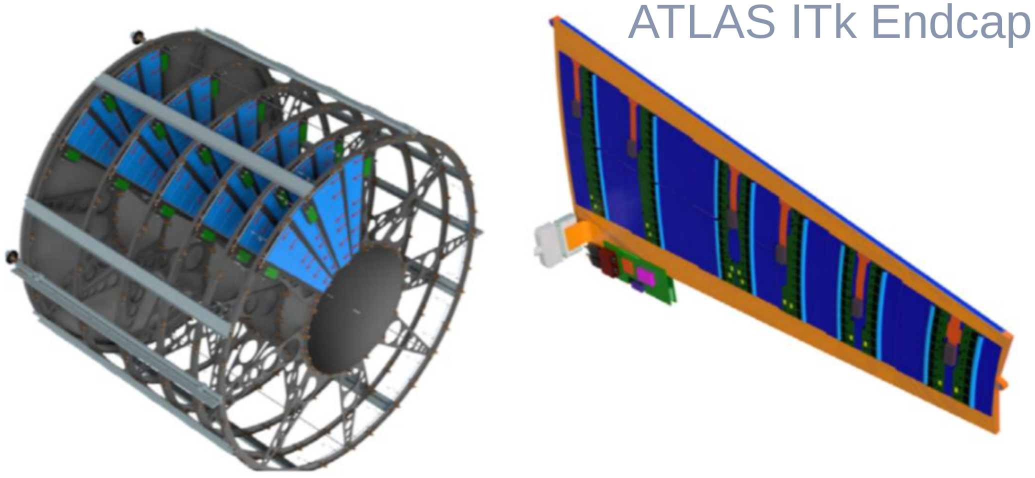


EUTelescope for ATLAS ITk Endcap Strip

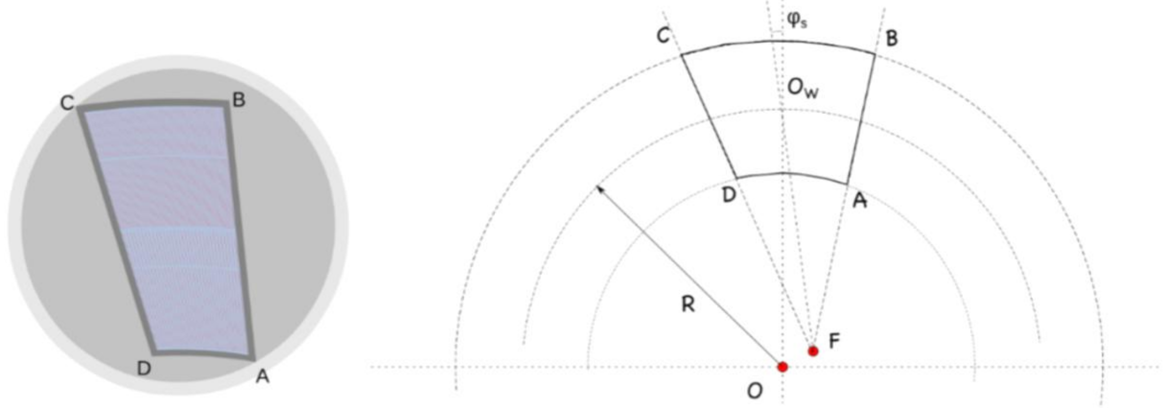
- Hardcoded geometry: 6 pixel detector planes + 1 FEI4 (pixel) + 1 DUT (pixel or strip)
  - Only supports Cartesian coordinates for all detector planes
- This resulted in very awkward experience with ATLAS ITk Endcap Strip beam test
  - Since ITk Endcap Module sensor has the Annulus shape with built-in stereo angle (i.e. measurement is not in Cartesian frame)

NIMA 924, 137 (2019)

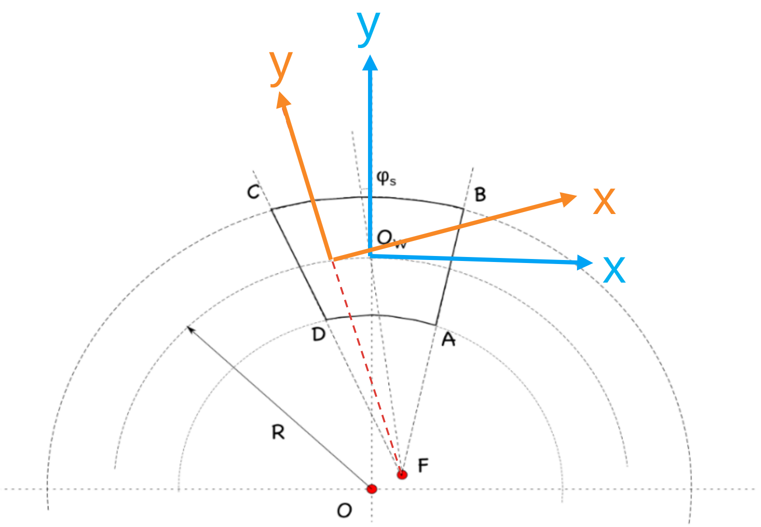
ATLAS ITk Endcap

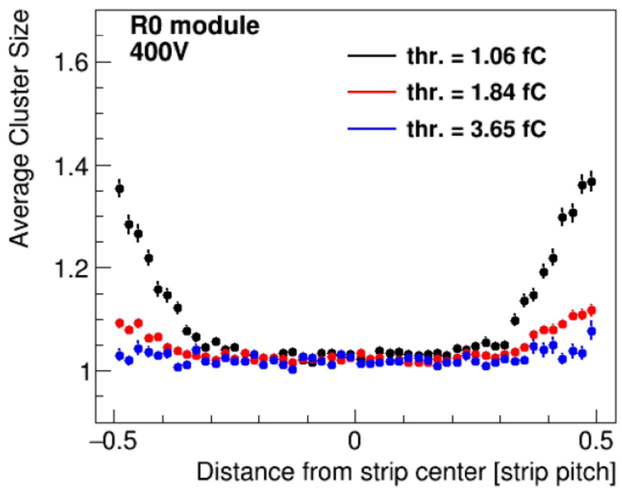


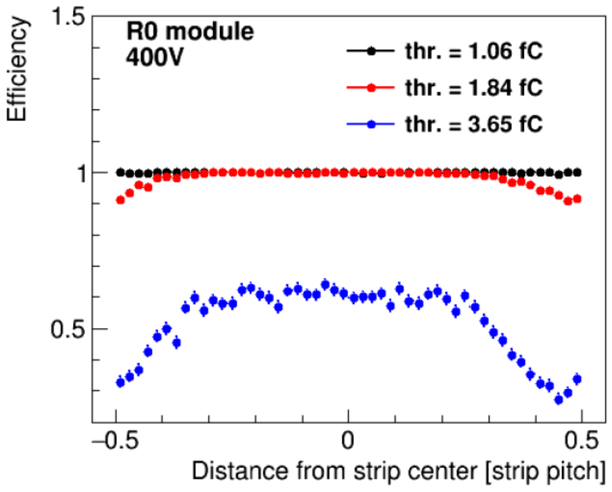
ATLAS ITk Endcap Module geometry



Extended EUTelescope for radial strips



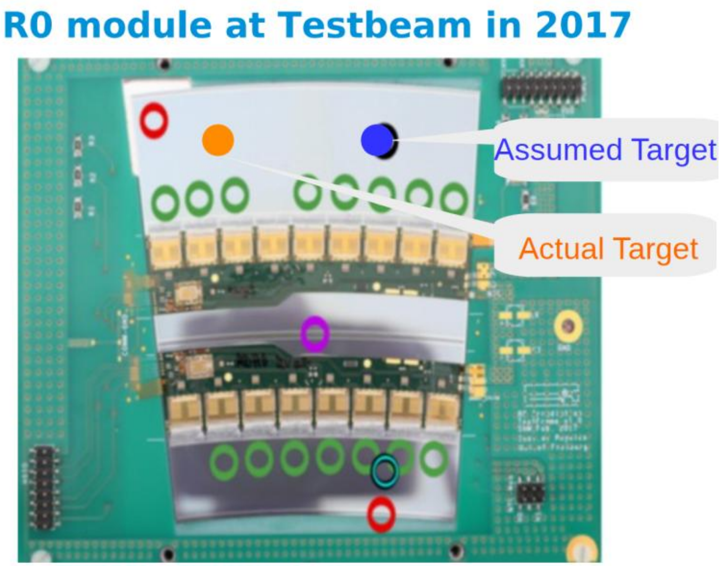


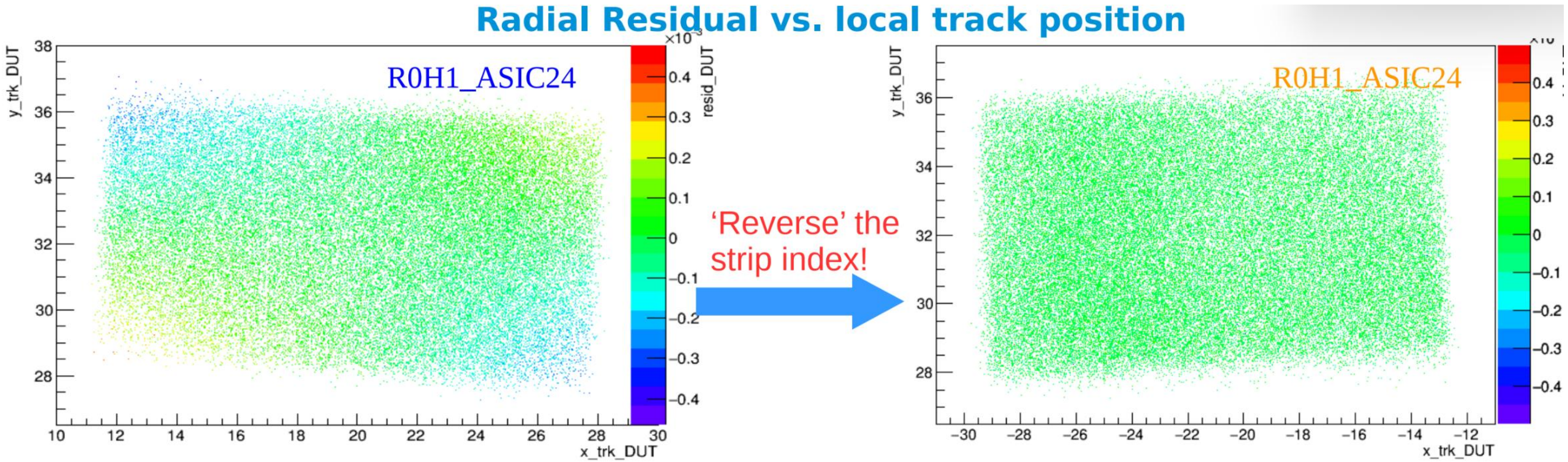


- For radial strips, the measurement frame varies with each strip, which is different from the module Cartesian frame (i.e. frame to be aligned)
- Extended tracking and alignment implemented to allow characterization of first ITK Endcap module in 2017
  - Development still in my personal github repo (Radial-recon. Radial-DSides-recon). Never merged into EUTelescope master unfortunately.

Lesson learned about alignment

- There is strong interplay between tracking and alignment
  - Tracking can be deteriorated by misalignment.
  - Misalignment can be detected from tracking performance

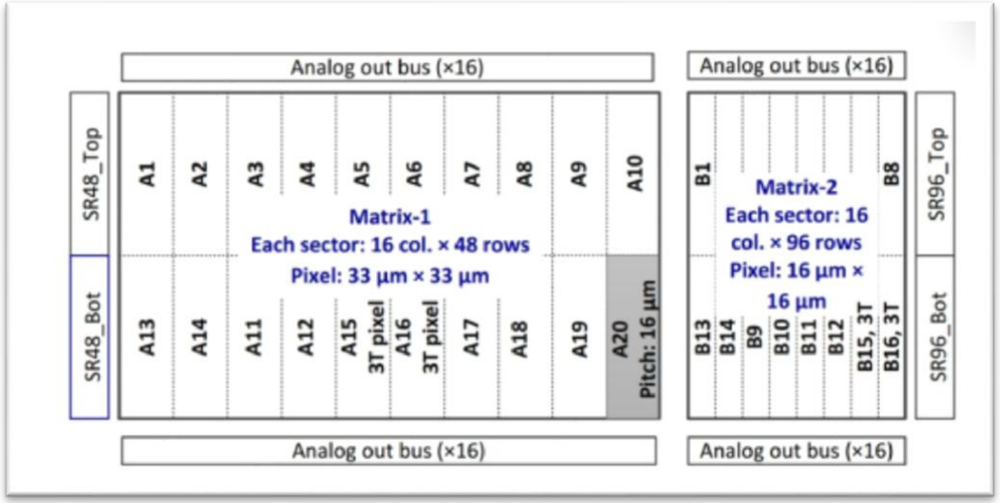


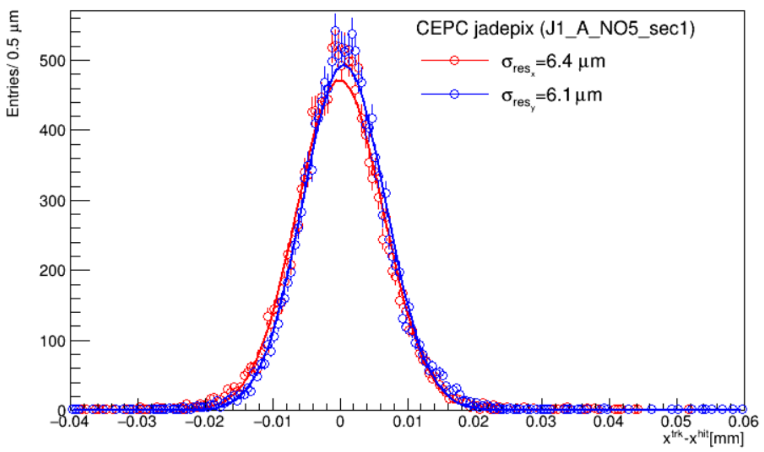


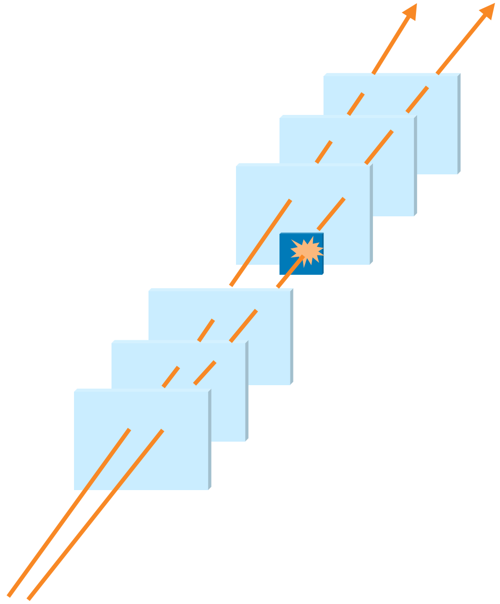
Extended EUTelescope for CEPC JadePix-I

- JadePix-1 has the pixel sensor, but the size is much smaller than MIMOSA
  - Trivial extension (code in my personal github repo: nonRadial-reco-jadePix) for JadePix-1 beam test in 2018
    - Geometry implementation
    - Need to mask out tracks out of JadePixe acceptance

NIMA 977, 164267, 2020





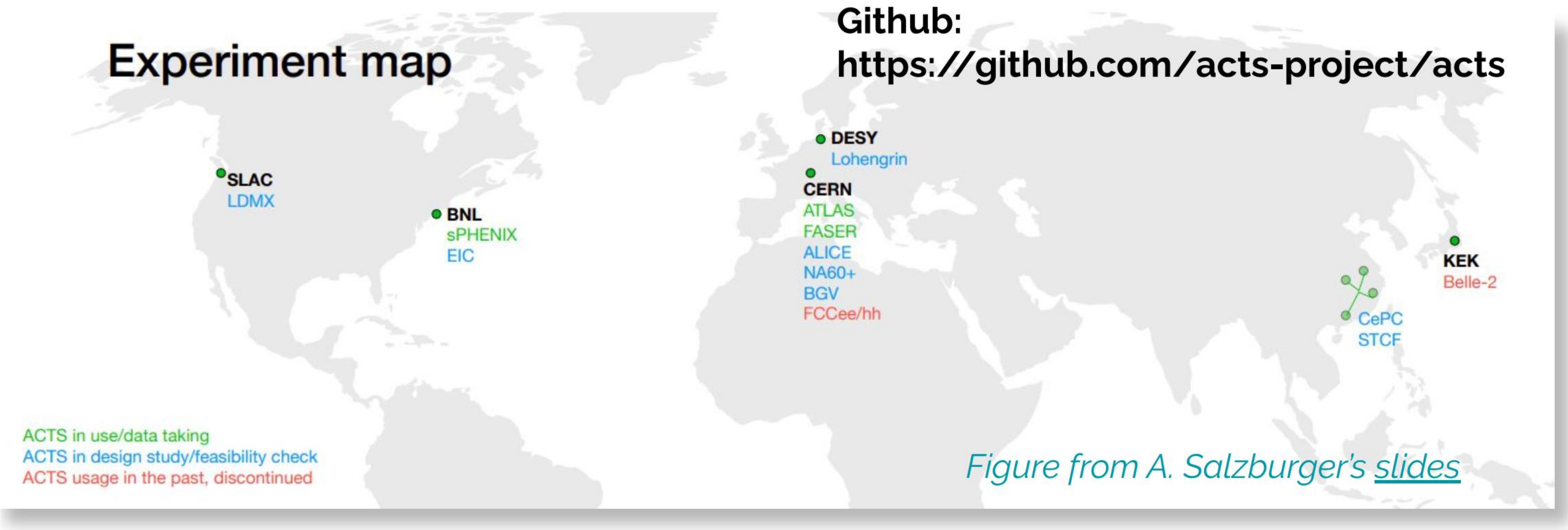


Is there a generic software for beam test data reconstruction?

So that we don’t need all those nontrivial and trivial extensions.

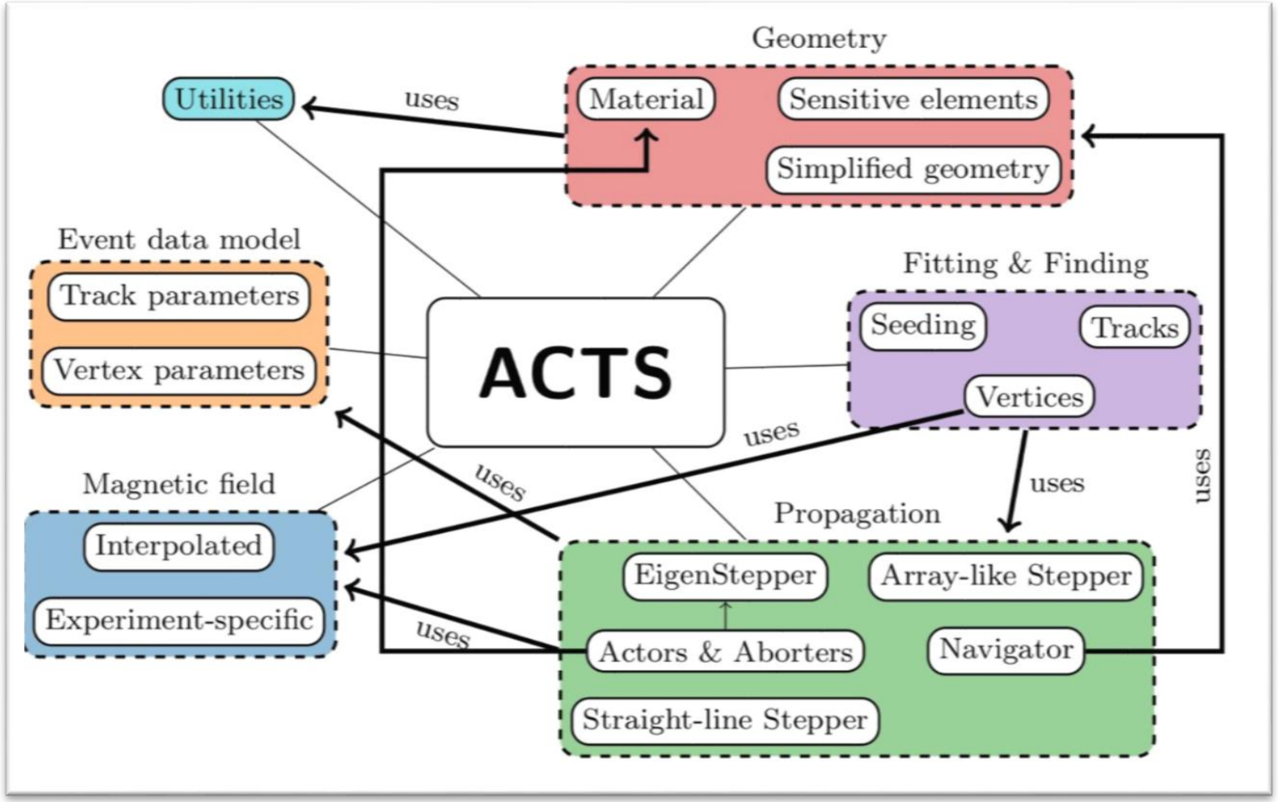
A Common Tracking Software (ACTS) project

- A modern **open-source detector-independent** tracking toolkit for current&future HEP experiments based on LHC (and beyond) tracking experience
  - Data production by **ATLAS, FASER, sPHENIX**
  - Detector R&D by **CEPC, STCF, EIC, ePIC, LDMX**…
- A R&D platform for ML-based tracking, heterogeneous computing and 4D tracking



ACTS design and functionalities

- Fully C++20 compliant, template design, strict thread-safety, contextual condition data …
  - More in *A. Salzburger's slides*

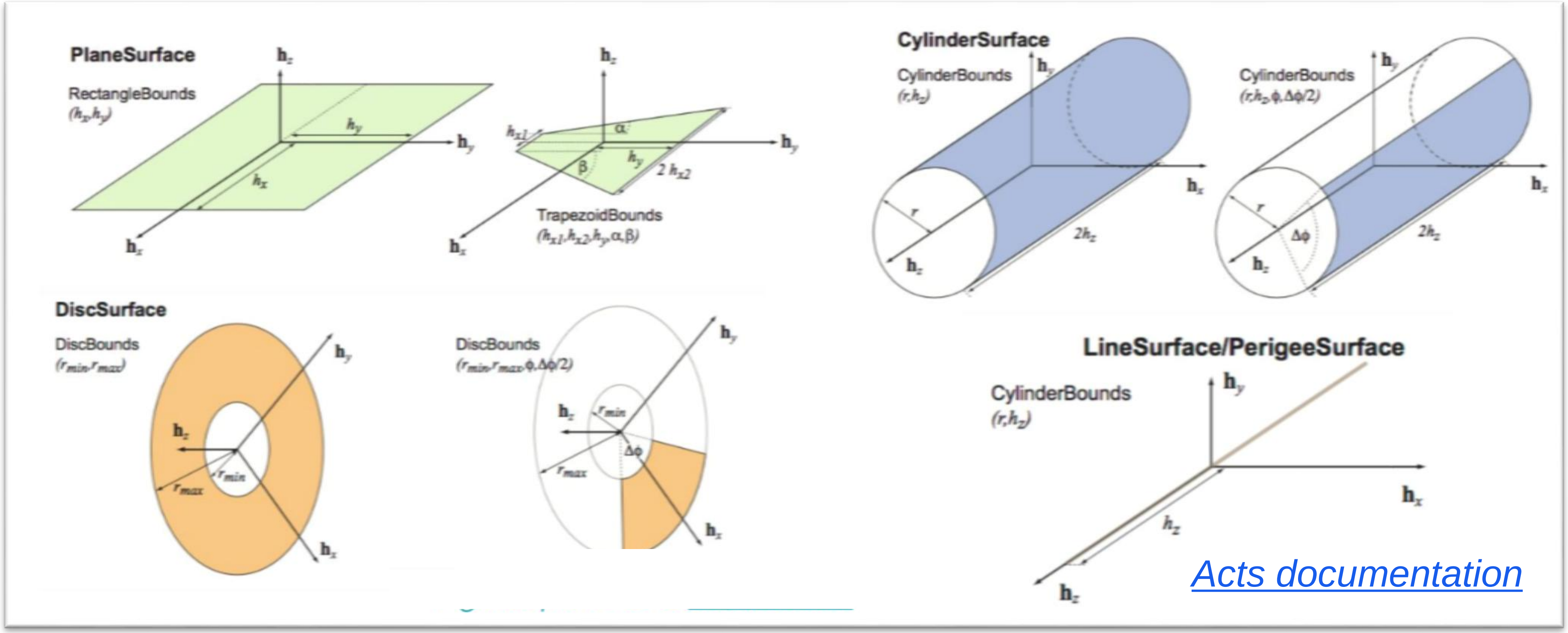


CSBS (2022) 6, 8

- **Track fitting**
  - (Extended) Kalman Filter (KF), Gaussian Sum Filter, Non-linear KF
  - Global chisq fitter
- **Track finding**
  - Seeding, Combinatorial Kalman Filter (CKF), Graph Neural Networks, Hough Transform
- **Vertex finding&fitting**
  - Primary vertex: AMVF, IVF
- **KF-based Alignment** (prototype developed in 2022 and being slowly validated and optimized)

ACTS tracking geometry

- Tracking geometry is simplified from detailed full simulation geometry for fast navigation, but with material effects well taken into account
- Different concrete surfaces for different tracking detectors (pixel/strip silicon, gaseous)
  - A surface has shape, bounds, **rotation+translation**, local coordinates and unique identifier…



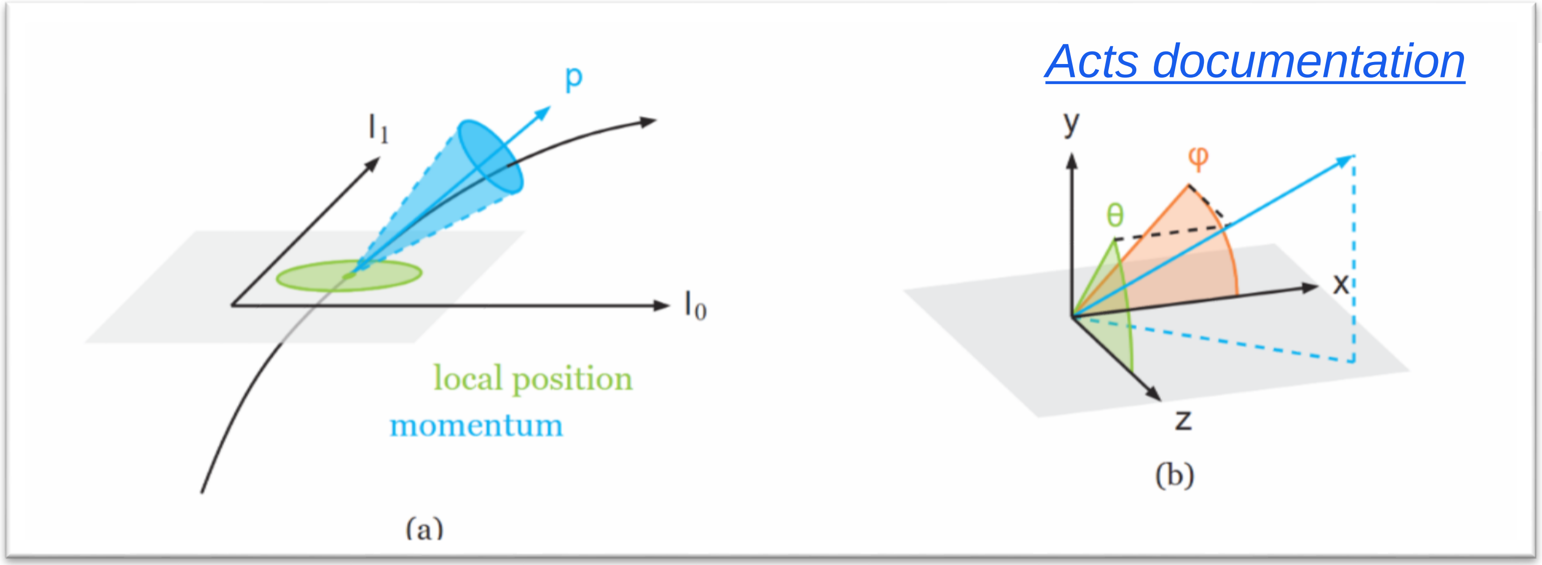
Acts documentation

ACTS track parameterization

- 6-dimensional bound/local track parameters (integration of particle flight time in track propagation, i.e. inherent support for 4D tracking)

$$\vec{x} = (l_0, l_1, \phi, \theta, q/p, t)^T$$

- Measurement (1, 2, or 3 dimension) is a subset of the 6 parameters: $\vec{m} = H \cdot \vec{x}$



e.g. $H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$

Track-based alignment

Ideas of track-based alignment

- Tracks share the same detector geometry (a.k.a. global alignment parameters n-dim $\vec{\alpha}$) while they have their own track parameters (a.k.a. local track parameters $\vec{x_i}$)
- The global track parameters can be estimated by minimizing the χ^2 sum of a set of good quality tracks:

$$\chi^2 = \sum_i {\chi_i}^2 = \sum_i [\vec{m}_i - \vec{h}_i(\vec{x}_i(\vec{\alpha}), \vec{\alpha})]^T V^{-1} [\vec{m}_i - \vec{h}_i(\vec{x}_i(\vec{\alpha}), \vec{\alpha})]$$

- This involves solving the non-linear equation iteratively, i.e. $\vec{\alpha}$ is updated iteratively to approach its optimal value:

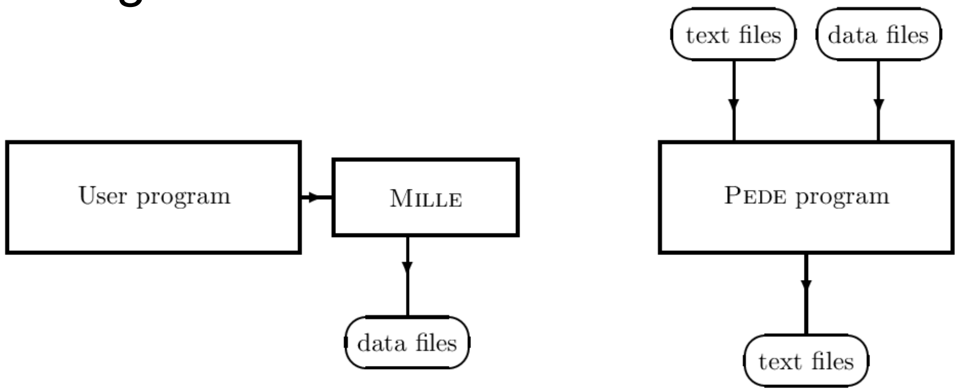
$$\frac{d^2\chi^2}{d^2\vec{\alpha}} |_{\vec{\alpha_0}} \Delta\vec{\alpha} = -\frac{d\chi^2}{d\vec{\alpha}} |_{\vec{\alpha_0}}$$

We can use either Global least-square fitter or Kalman Filter fitter to fit each track in each iteration

Alignment with MillePede

A traditional alignment tool used and validated by BESIII, CMS …

- MillePede = Mille + Pede
  - **Mille**: write residual, derivatives of residual w.r.t. alignment parameters and track parameters into binary file
  - **Pede**: solve equation with dimension n to get solution for delta of track parameters :
    - Solver: Inversion, Cholesky decomposition, diagonalization …



Called for each dimension of a residual of each track

◆ mille()

```
void Mille::mille ( int            NLC,
                    const float *  derLc,
                    int            NGL,
                    const float *  derGl,
                    const int *    label,
                    float          rMeas,
                    float          sigma
                  )
```

Add measurement to buffer.

Parameters

[in]	**NLC**	number of local derivatives
[in]	**derLc**	local derivatives
[in]	**NGL**	number of global derivatives
[in]	**derGl**	global derivatives
[in]	**label**	global labels
[in]	**rMeas**	measurement (residuum)
[in]	**sigma**	error

Alignment in ACTS

More in [my slides at tracking2025]

A common alignment algorithm being validated and optimized

$r = m - h(x, \alpha)$	The track residual
V	The measurement covariance (usually diagonal)
$H = \left.\frac{\partial h(x)}{\partial x}\right\|_{x_0}$	The projection matrix from (bound) track parameters to measurement
C	The covariance of track parameters at different measurements: Straightforward with global chi2 fitter. Can provided by Kalman Filter as well
$A_{k\ell} \equiv \frac{\partial r_k}{\partial \alpha_\ell}$	The derivative of residual w.r.t. alignment parameters

The first and second derivatives for a single track and then summed over tracks

$$\frac{\mathrm{d}\chi^2}{\mathrm{d}\alpha} = 2A^T V^{-1} \left(V - HCH^T\right) V^{-1} r,$$

$$\frac{\mathrm{d}^2\chi^2}{\mathrm{d}\alpha^2} = 2A^T V^{-1} \left(V - HCH^T\right) V^{-1} A.$$

A nxn matrix

A n-dim vector

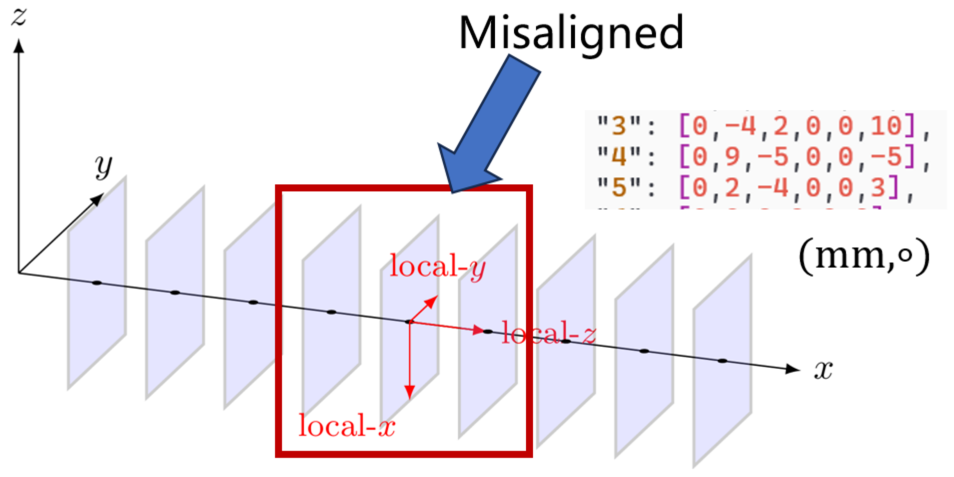
$$\left.\frac{\mathrm{d}^2\chi^2}{\mathrm{d}\alpha^2}\right|_{\alpha_0} \Delta\alpha = - \left.\frac{\mathrm{d}\chi^2}{\mathrm{d}\alpha}\right|_{\alpha_0}$$

Solved with Eigen LU decomposition

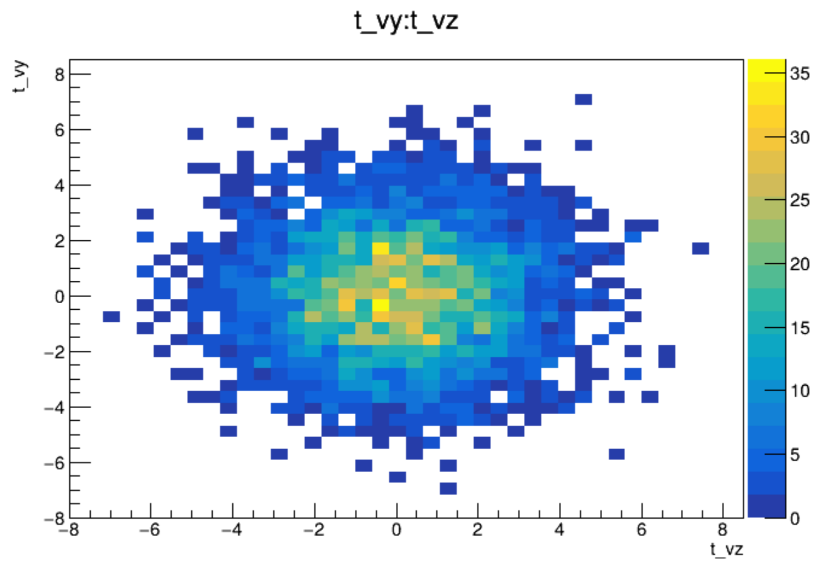
(claimed stable and well tested with large matrices)

ACTS alignment for a telescope-like detector

- Particle Gun:
  - 10k muon per event, $p = 10\ \mathrm{GeV}$, direction of momentum along x-axis
  - Vertex: x, t=0, y and z~2D Gaussian with $\sigma_y = \sigma_z = 2\ \mathrm{mm}$
- Alignment degree of freedom: Ty+Tz+Rz
- Only align the misaligned layers

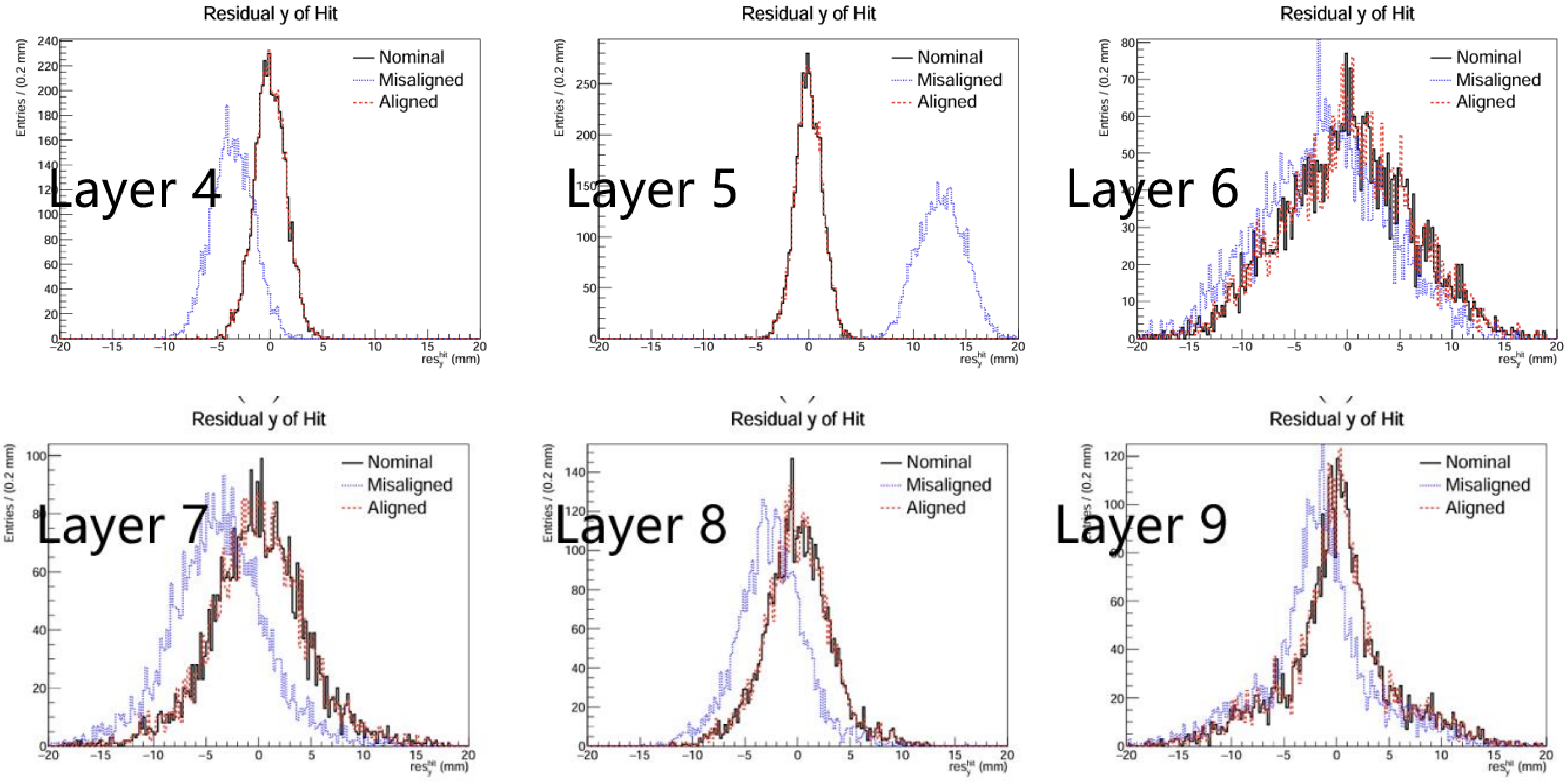


25 x 25 um2 resolution



Residual with misalignment + aligned

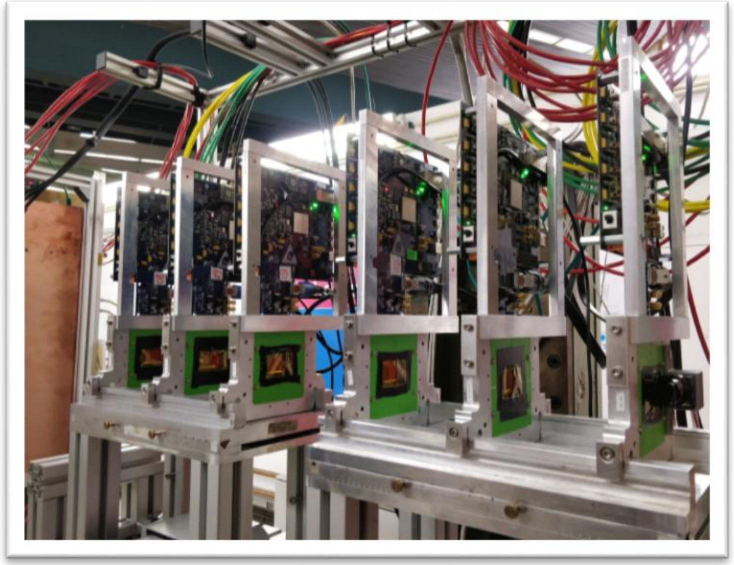
Credit to Z. Sun for the plots



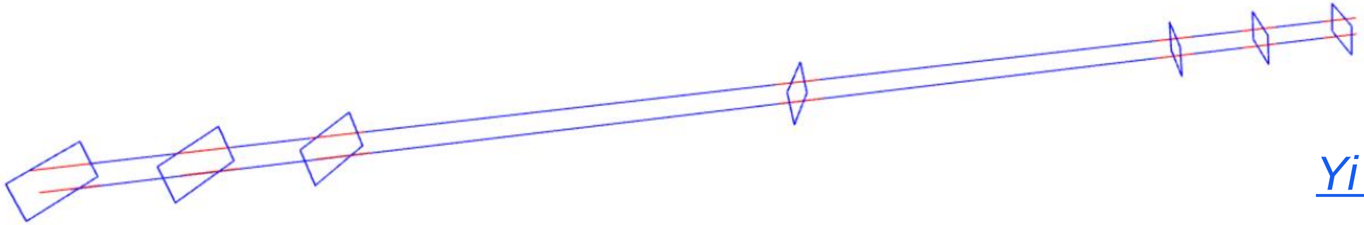
ACTS tracking for Telescope-like detector

ACTS for AlPIDE telescope at DESY

- ALPIDE telescope was the first ACTS client which has a telescope geometry
  - No constraint on the number and geometry of detector planes
  - Tracks are reconstructed using ACTS seeding + Combinatorial Kalman Filter
  - Intra-tracking matching of track with DUT hit
  - Alignment using MillePedeII



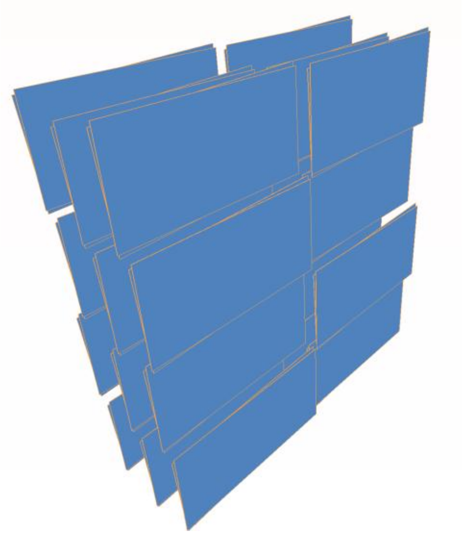
- Minimal runtime overhead of ACTS (<1 ms/event) allowed online track visualization



*Yi Liu et al 2023 JINST **18** P06025*

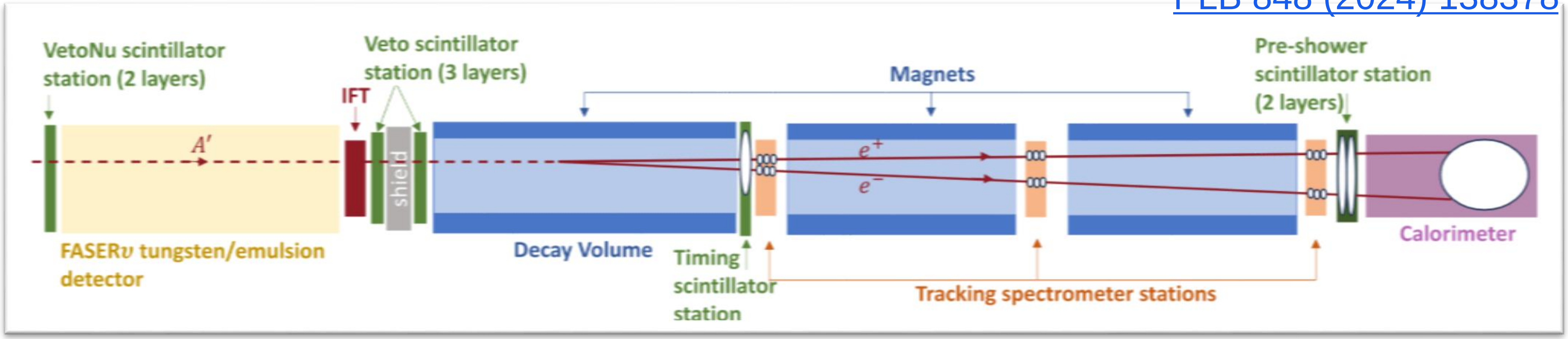
ACTS for FASER

- FASER tracking system has a telescope-like geometry with 4 tracking stations
  - Each tracking station is made of 3 layers of double-sided ATLAS SCT modules
- Adopted ACTS for tracking from very beginning (facilitating direct collider neutrino observation, dark photon search…)
- Alignment currently done with MillePedeII.



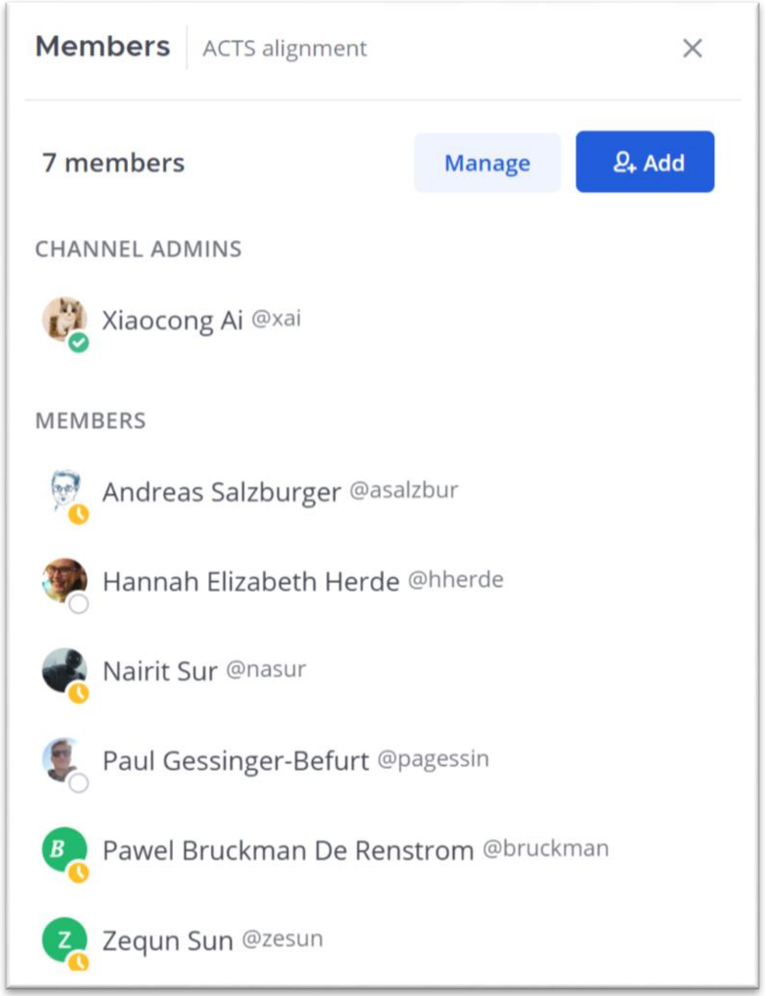
8 modules/layer
(coverage 24cm x 24cm)

PLB 848 (2024) 138378



Summary

- Tracking and alignment is pivotal to particle beam telescope data reconstruction and performance evaluation
- Highly recommend ACTS for flexible and fast track reconstruction
- MillePede is well validated for detector alignment
- ACTS alignment is coming soon.
- Join us if you are interested:
  - https://mattermost.web.cern.ch/acts
    - Alignment channel : https://mattermost.web.cern.ch/acts/channels/acts-alignment



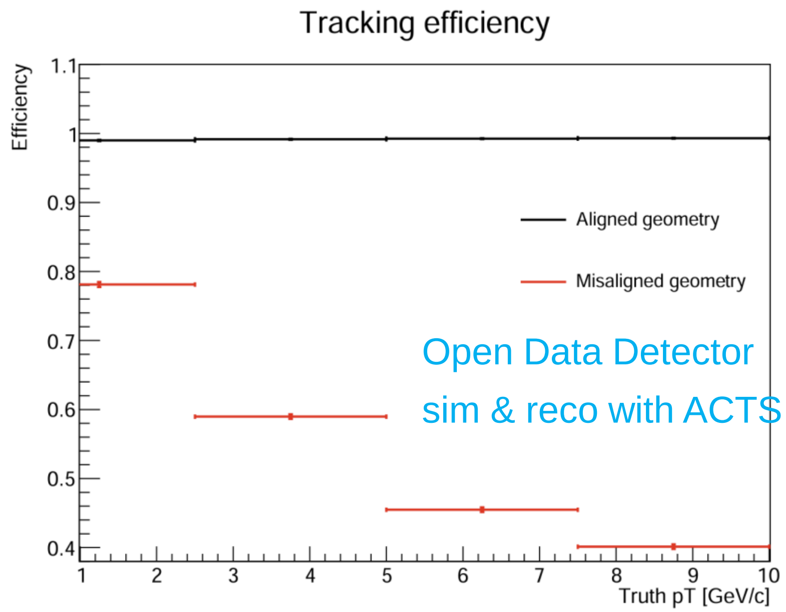
Back up

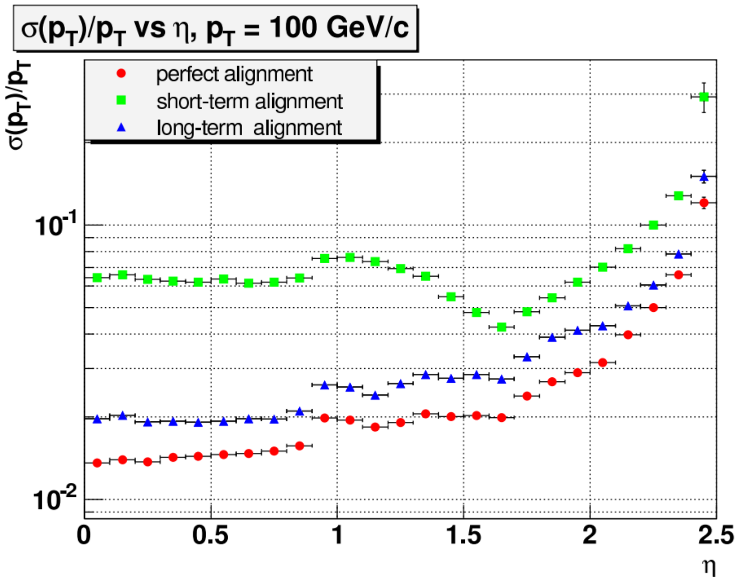
Why alignment matters?

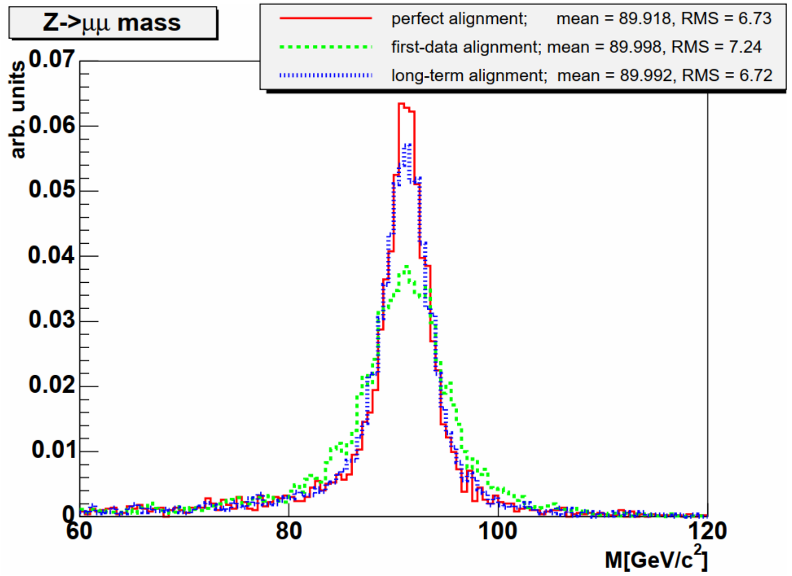
See more in P. Bruckman's talk

- Limited detector placement precision upon installation
- Misalignment is the dominant source of measurement resolution degradation

➡ Degradation of tracking precision and efficiency, and eventually **physics precision**!







From G. Steinbruck

Alignment Parameters

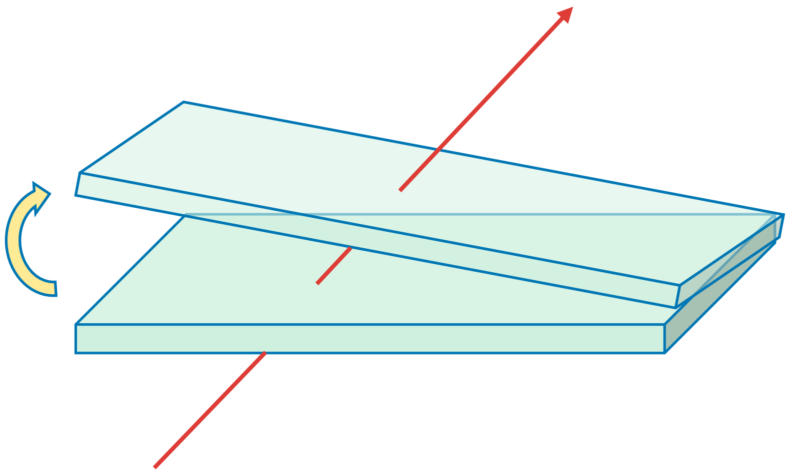
- Detector element placement description:
  - Translation (3 parameters) + Rotation (3x3 rotation matrix)
- Alignment parameters (6 parameters for now):
  - Translation + Rotation about original local axes ($\vec{x}_L$, $\vec{y}_L$, $\vec{z}_L$) using Euler angles
    - Suppose rotation in the order: 1) around $\vec{x}_L$ about α → 2) around $\vec{y}_L$ about β → 3) around $\vec{z}_L$ about γ , then the new local axes become

$$(\vec{x}_L''', \vec{y}_L''', \vec{z}_L''') = (\vec{x}_L, \vec{y}_L, \vec{z}_L) \begin{pmatrix} cos\beta cos\gamma & sin\alpha sin\beta cos\gamma - cos\alpha sin\gamma & cos\alpha sin\beta cos\gamma + sin\alpha sin\gamma \\ cos\beta sin\gamma & sin\alpha sin\beta sin\gamma + cos\alpha cos\gamma & cos\alpha sin\beta sin\gamma - sin\alpha cos\gamma \\ -sin\beta & sin\alpha cos\beta & cos\alpha cos\beta \end{pmatrix}$$

Caveat: a rotation can be expressed in 24 equivalent sequence of Euler angles

Residual derivative

- Suppose the rotation around the original local axes (α, β, γ) is small, then



$$\frac{\partial \vec{x}_L'''}{\partial(\alpha,\beta,\gamma)} = \begin{pmatrix}\vec{x}_L & \vec{y}_L & \vec{z}_L\end{pmatrix}\begin{pmatrix}0 & 0 & 0\\ 0 & 0 & 1\\ 0 & -1 & 0\end{pmatrix} = R\begin{pmatrix}0 & 0 & 0\\ 0 & 0 & 1\\ 0 & -1 & 0\end{pmatrix}$$

$$\frac{\partial \vec{y}_L'''}{\partial(\alpha,\beta,\gamma)} = \begin{pmatrix}\vec{x}_L & \vec{y}_L & \vec{z}_L\end{pmatrix}\begin{pmatrix}0 & 0 & -1\\ 0 & 0 & 0\\ 1 & 0 & 0\end{pmatrix} = R\begin{pmatrix}0 & 0 & -1\\ 0 & 0 & 0\\ 1 & 0 & 0\end{pmatrix}$$

$$\frac{\partial \vec{z}_L'''}{\partial(\alpha,\beta,\gamma)} = \begin{pmatrix}\vec{x}_L & \vec{y}_L & \vec{z}_L\end{pmatrix}\begin{pmatrix}0 & 1 & 0\\ -1 & 0 & 0\\ 0 & 0 & 0\end{pmatrix} = R\begin{pmatrix}0 & 1 & 0\\ -1 & 0 & 0\\ 0 & 0 & 0\end{pmatrix}$$

Residual $\vec{r} = (r_x, r_y)$:

$r_x = (\vec{x}_{track} - \vec{o}_{module}) \cdot \vec{x}_L - x_{hit}$

$r_y = (\vec{x}_{track} - \vec{o}_{module}) \cdot \vec{y}_L - y_{hit}$

- $\vec{x}_{track}$ is the intersection of track with detector module
  - Changed with $\vec{o}_{module}$ and $\vec{x}_L, \vec{y}_L, \vec{z}_L$
- $\vec{o}_{module}$ is the center of the detector module
- x_{hit} and y_{hit} are hit local position on module

An alignment prototype

- It takes a set of tracks and sort out the sets of detector elements which can be and requested to be aligned => the initial value of $\vec{\alpha}$
- Perform the fit for each track using $\vec{\alpha}$, and estimate the χ^2 derivatives w.r.t. $\vec{\alpha}$
- Solve the equation to obtain $\Delta\vec{\alpha}$
- Update $\vec{\alpha}$ to become $\vec{\alpha}'$ using provided alignment parameter updater
- Stop iteration of the above three steps when provided converging criteria is met

```
template <typename trajectory_container_t,
          typename start_parameters_container_t, typename fit_options_t>
Acts::Result<AlignmentResult> align(
    const trajectory_container_t& trajectoryCollection,
    const start_parameters_container_t& startParametersCollection,
    const AlignmentOptions<fit_options_t>& alignOptions) const;
```