Running HipPy for CMS Tracker Alignment

Tracker Training Days

March 02, 2023 - 354/1-019 CERN (hybrid via Zoom)

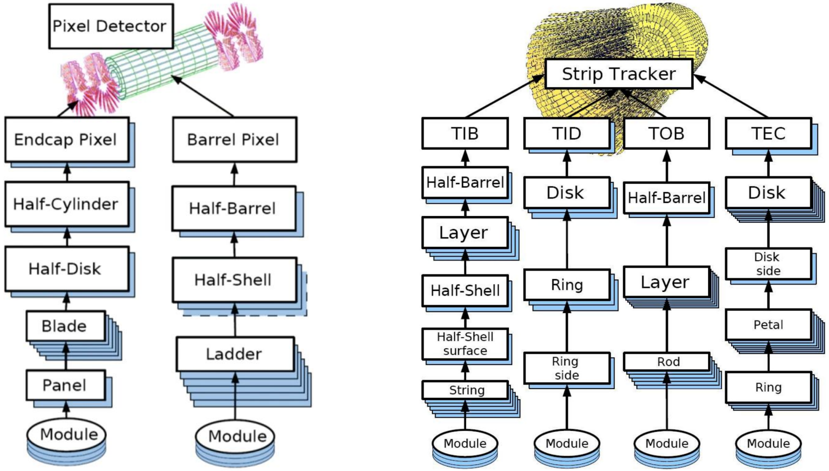
Lucas Kang, Johns Hopkins University

Outline

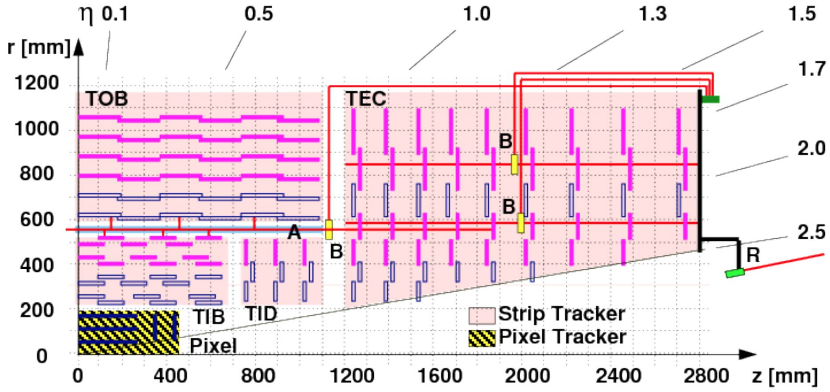
Tracker Alignment

High-level Structures

- 2 half barrels in the barrel pixel tracker (BPIX)
- 4 half cylinders in the 2 forward pixel tracker regions (FPIX)
- 2 half barrels in the strip tracker inner barrel (TIB) and in the strip tracker outer barrel (TOB)
- 2 endcaps in the tracker inner disks (TID) and in the tracker endcaps (TEC)



Diagrams of the hierarchical structures of the Pixel Detector and Strip Tracker [3]

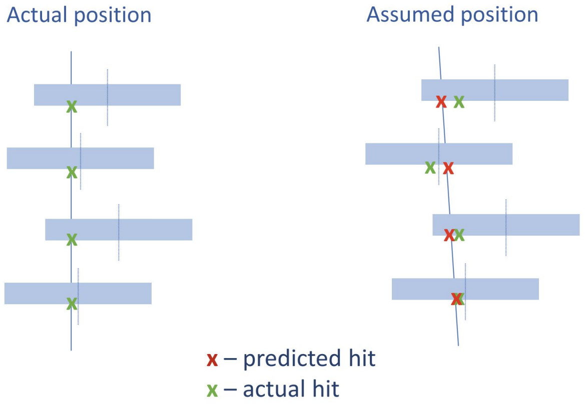


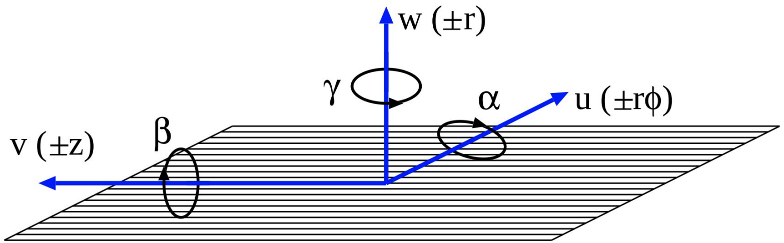
Scales of the Pixel Detector and Strip Tracker which surrounds it [3]

Basics of Alignment

- Hit positions and impact points of a track are systematically displaced if the module position is not known correctly
- The difference in local module coordinates between these two quantities are the "**track-hit residuals**" [7]
- Modules are parameterized as seen below
  - **w** axis is normal to the module
  - **u** and **v** axes are in the plane
  - **u** axis more sensitive to direction of measurement
  - Angles **α**, **β**, and **γ** describe rotations around **u**, **v**, **w** respectively
- Hierarchical systems
  - {**r**, **g**, **q**} = coordinates in {global, composite, local} system
  - {**R**, **G**} = rotations between global and {local, composite} system

Simple illustration of a track through misaligned layers [5]





Example local coordinates of a module. Global parameters are shown in parentheses for modules in the TIB and TOB. [3]

Example Procedure

- Outline of an iterative alignment procedure:
  - Load track data and hits
    - The tracks are fit using the current estimate of the alignment parameters
  - Hit χ^2 are computed for the selected hits
  - Then minimize each sensor's χ^2
    - w.r.t. a change in only that sensor's local alignment
    - Hold the parameters of every other sensor fixed
  - Calculate for every sensor
  - Update the alignment parameters for all sensors with the computed change to the original estimate
  - Use the updated local alignment to fit the tracks for the next iteration
  - Repeat the process until the alignment converges

$$\chi^2 = \sum_i^{\text{hits}} \epsilon_i^T(\mathbf{p})\mathbf{V}_i^{-1}\epsilon_i(\mathbf{p})$$

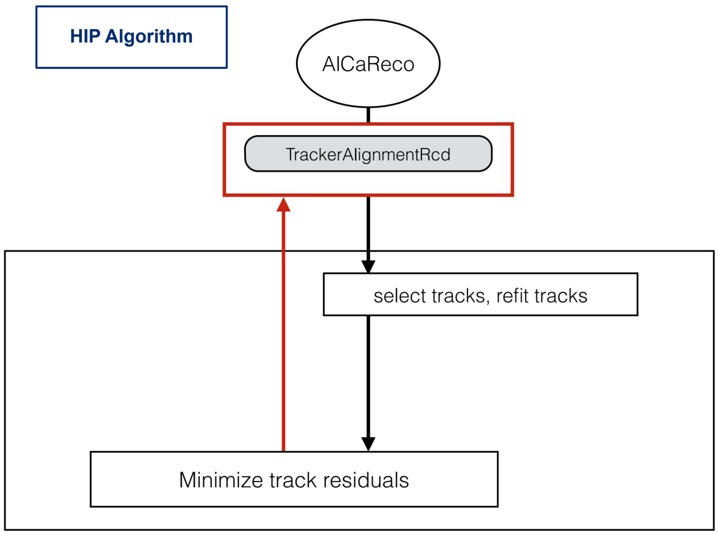
χ^2 function for a single iteration [4]

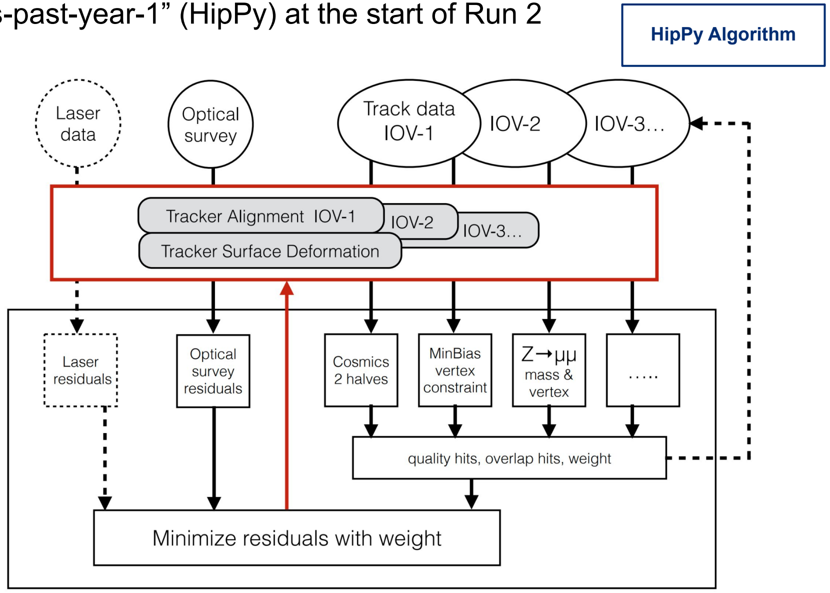
- This is the basic concept of the hits-and-impact-points (HIP) algorithm [10]

The HipPy Algorithm

HIP vs HipPy

- The “hits-and-impact-points” (HIP) algorithm [6][7] precedes what we now call HipPy
  - Utilized during commissioning [3] of the CMS tracker and for Run 1 [8][9]
- Several notable features were added over time [10], as illustrated below
- The improved algorithm was renamed to “hits-and-impact-points-past-year-1” (HipPy) at the start of Run 2





Process diagrams of the HIP and HipPy algorithms [5]

HipPy Algorithm

- Improvements to the alignment algorithm since Run 1
  - The inclusion of 3 alignment parameters to describe sensor curvature (beyond the 6 position/orientation coordinates)
  - The ability to weight certain types of input
  - The option to perform sequential, hierarchical alignment over multiple time periods (multi-IOV)
  - Optional mass and/or vertex constraints in certain types of events with known physics process
- These features proved invaluable during Run 2, with successful HipPy-based alignment constants used in CMS data reconstruction (along with MP objects) in the prompt and EOY alignments [5]

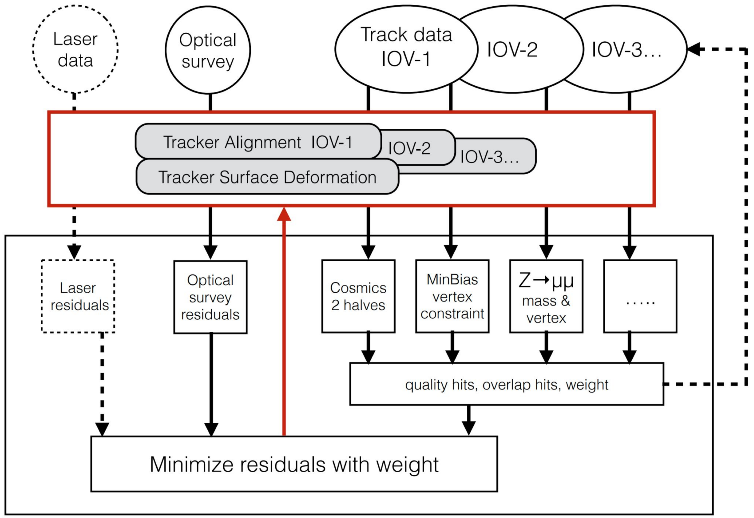


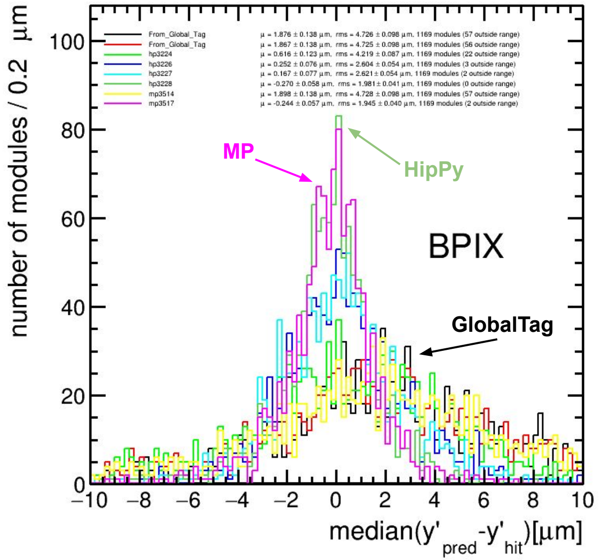
Diagram of the HipPy algorithm [5]

HipPy Algorithm cont.

- HipPy leverages full integration of CMS software, which is helpful especially for track reconstruction
    - Can make use of any constraint (e.g. mass, vertex) defined in that software
    - Adds flexibility and simplifies development (e.g. Kalman filter)
- Position and orientation of each sensor are determined independently of other sensors in every iteration
    - Multiple iterations are required to solve correlations between sensor parameters (requires multiple track fits)
    - However, each iteration is a straightforward application of a small matrix inversion
- The alternative alignment algorithm MILLEPEDE-II (MP) opts to use a global matrix and forgoes the iterative approach
- In the past HipPy and MP alignment algorithms have been run independently to cross-check each other
- The two algorithms have also been run successively to refine the alignment

Use and prospects

- HipPy was utilized extensively throughout Run 1 and Run 2 of the LHC
  - Strip tracker commissioning and for continued tracker alignment since
- Contributed an additional point of comparison in past alignment campaigns
  - Illustrated in the example validation plot shown to the right
  - Plan to continue doing so (Run 3)
- The HipPy algorithm is relatively simple in its design, but this (along with its native full access to CMS software) can be advantageous
  - Can also be seen as complementary to MP
- Currently working on running HipPy with the latest datasets for alignment
  - Aim to generate a contemporary alignment in a way analogous to MP, with different weights assigned to different datasets



Example DMR validation comparing HipPy and MP (05/31/2022) [11]

Example Alignment

HipPy campaign hp3252

124X_dataRun3_Prompt_v4

Run2022F

mp3588

- CMSSW: CMSSW_12_4_9
- GlobalTag: 124X_dataRun3_Prompt_v4
- Alignables:
  - PIXEL at Module Level
  - Strips fixed
- Dataset: MinBias [361579, 361580] → 23 M
- Referenced from: https://indico.cern.ch/event/1199476/#11-pixel-ml-alignment-after-cp

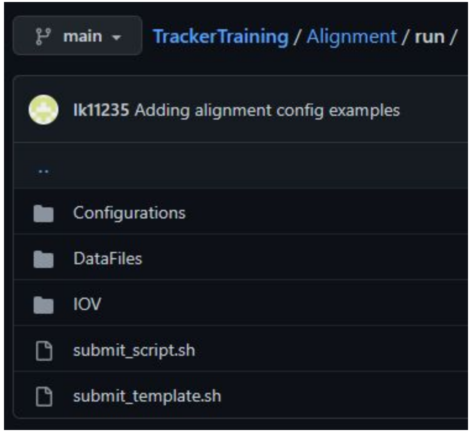
hp3252

- CMSSW: CMSSW_12_4_9
- GlobalTag: 124X_dataRun3_Prompt_v4
- Alignables:
  - PIXEL at Module Level
  - Strips fixed
- Dataset: /StreamExpress/Run2022F-TkAlMinBias-Express-v1/ALCARECO
- Run range: 361579

Example alignment uploaded to Git: [lk11235/TrackerTraining]

Running HipPy

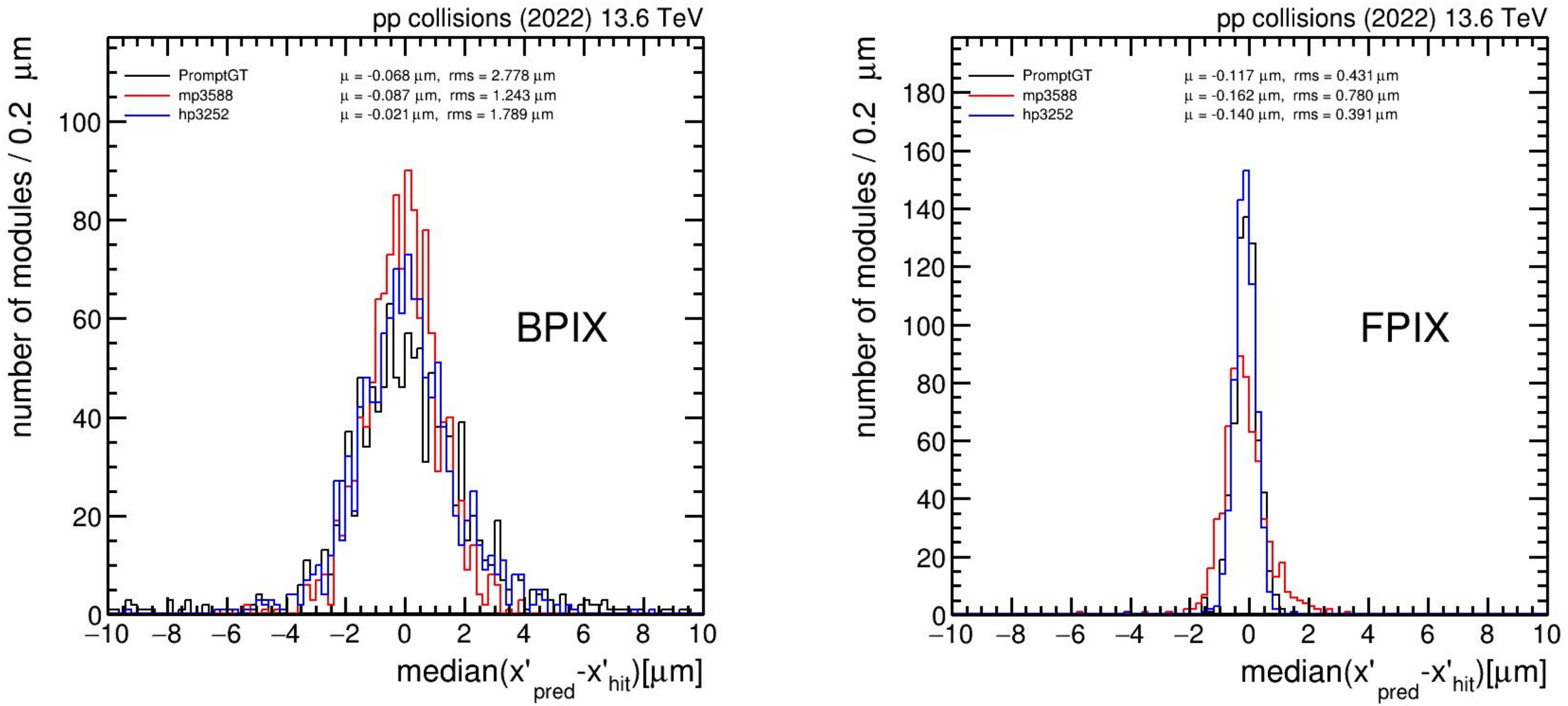
- Updated twiki with instructions to compile and run the HipPy algorithm
  - https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideHipPyAlgorithm
- Legacy page with further information about HipPy options
  - https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideHIPAlgorithm
- OOTB working example
  1. First, compile HipPy with the appropriate CMSSW release
  2. Second, copy the sample configuration files into your run directory
  3. Third, edit the run number and paths in ‘**submit_template.sh**’
     a. Previous runs are listed in ‘/afs/cern.ch/cms/CAF/CMSALCA/ALCA_TRACKERALIGN2/HipPy/alignments’
  4. Commit your changes and run ‘**./submit_script.sh submit_template.sh**’



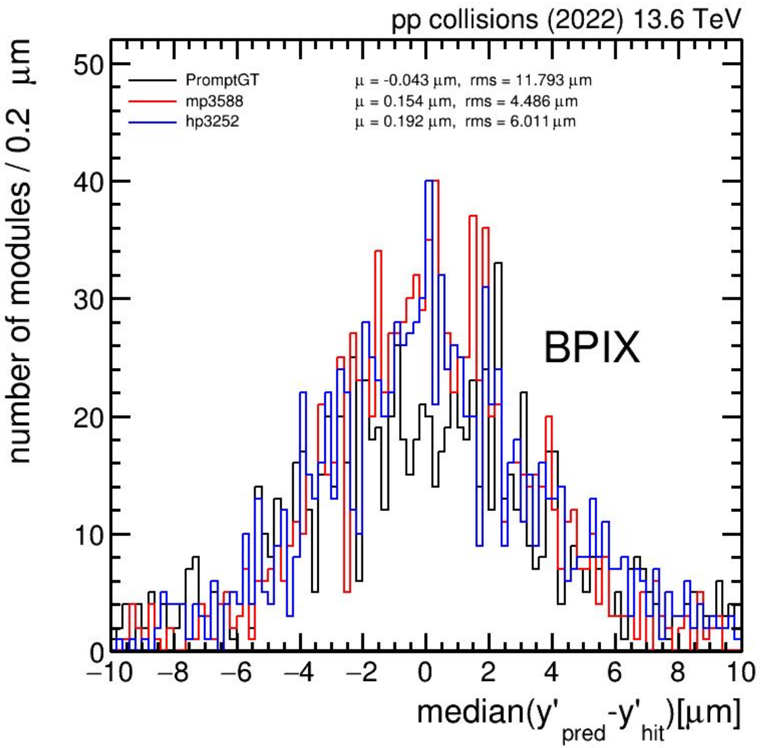
Configuration files here:
TrackerTraining/Alignment/run

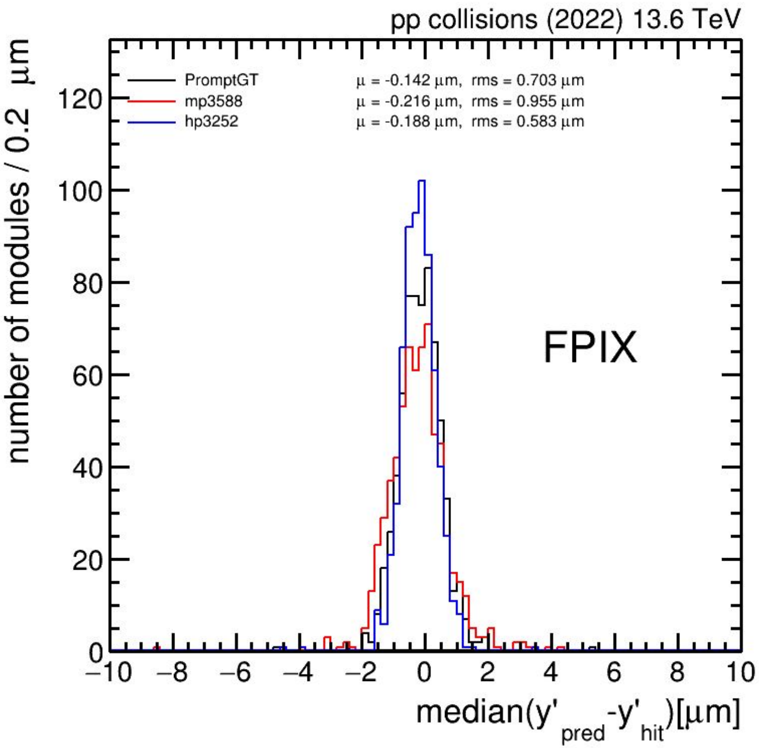
DMR validation: x residual in PIXEL

https://kang.docs.cern.ch/lkang/Alignment/hp3252_DMR_new/hp3252_DMR_new/ExtendedOfflineValidation_Images/



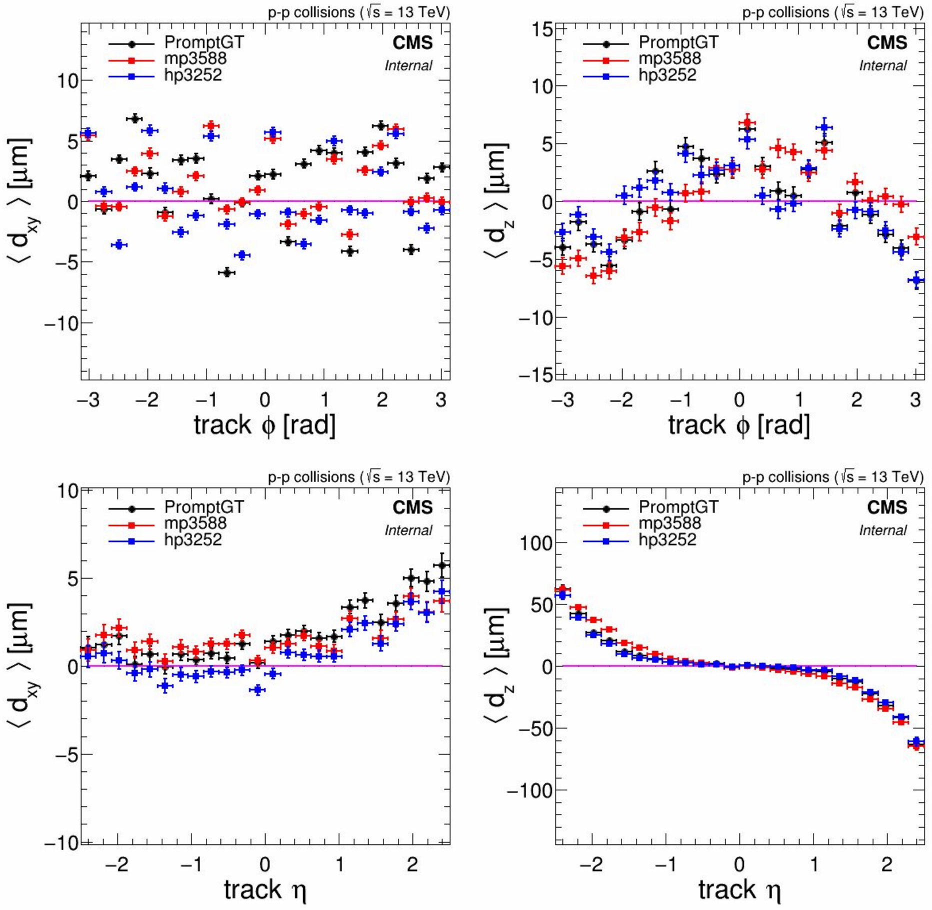
DMR validation: y residual in PIXEL



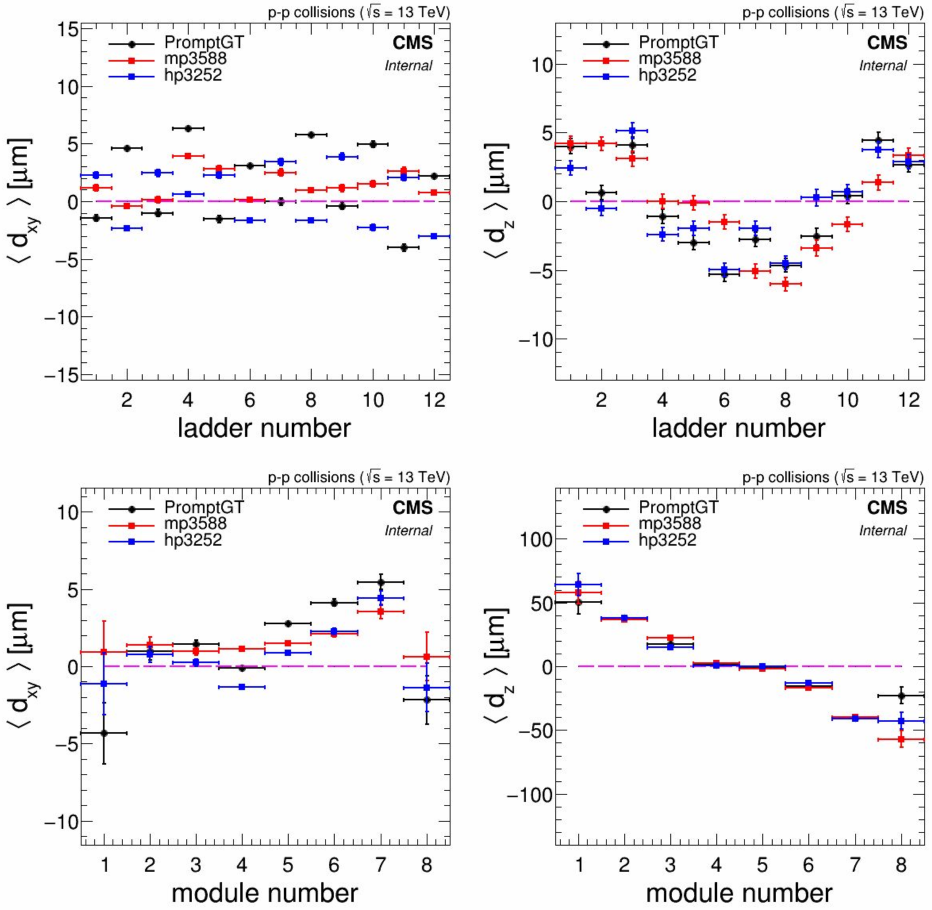


PV validation: PIX

https://kang.docs.cern.ch/lkang/Alignment/hp3252_PV/PrimaryVertexValidation/



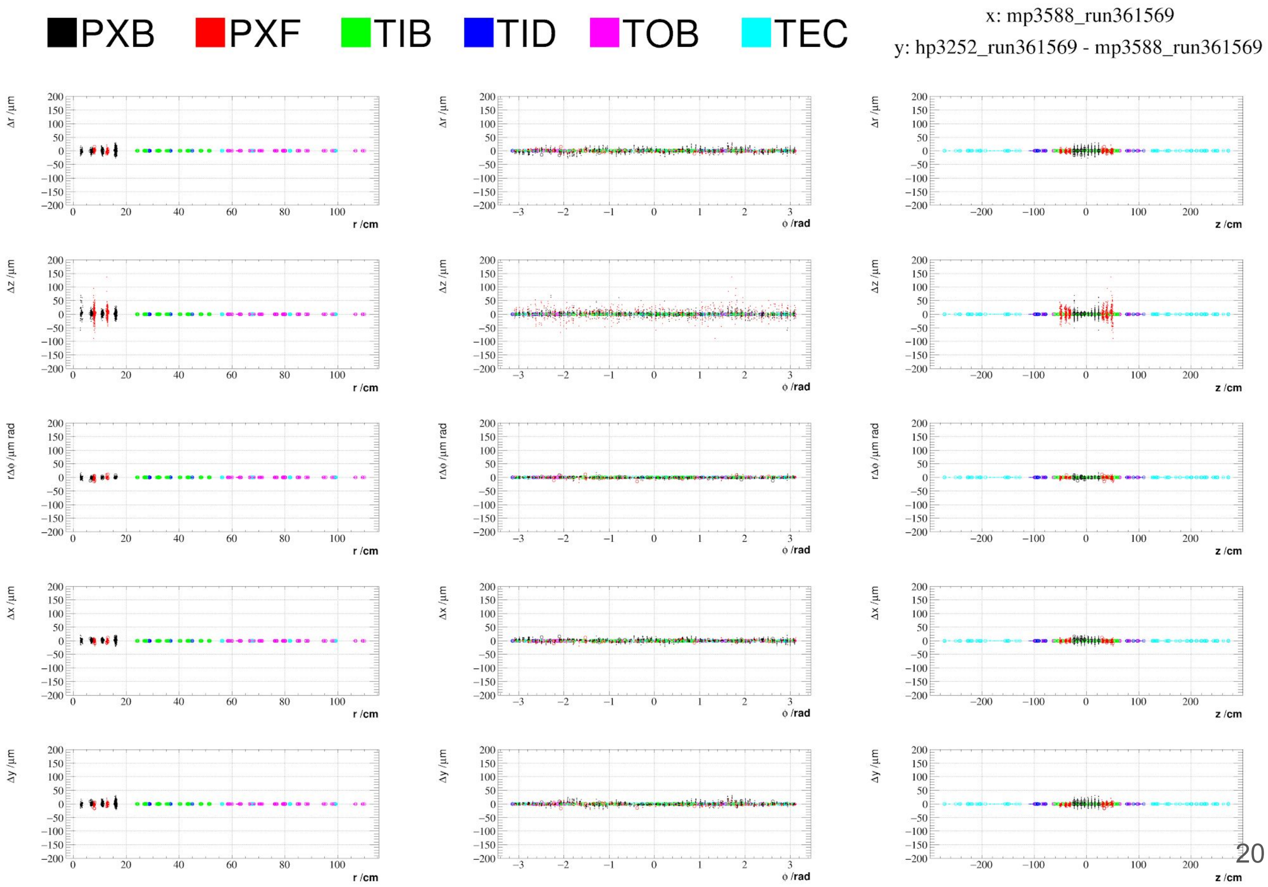
PV validation: BPIX Layer 1



GC validation: hp3252 - mp3588

mp3588 as described in: https://indico.cern.ch/event/1199476/#11-pixel-ml-alignment-after-cp

https://kang.docs.cern.ch/lkang/Alignment/hp3252_GC/



Thank you
for your attention!

References

[1] Mousa, J., Romaniuk, R., Pozniak, K., Zabolotny, W., Wrochna, G., Królikowski, J., Collaboration, C., & Warsaw, C. (2012). A New Boson with a Mass of 125 GeV Observed with the CMS Experiment at the Large Hadron Collider. Science, 338, 1569-1575.

[2] CMS Outreach, http://cmsinfo.cern.ch/outreach/.

[3] CMS Collaboration (2009). Alignment of the CMS silicon strip tracker during stand-alone commissioning. Journal of Instrumentation, 4(07), T07001–T07001.

[4] A. Bonato et al., "Application of Survey Measurements in Tracker Alignment", CMS IN-2009/027.

[5] CMS Collaboration (2022). Strategies and performance of the CMS silicon tracker alignment during LHC Run 2. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 1037, 166795.

[6] Karimäki, V., Heikkinen, A., Lampen, T., & Linden, T.. (2003). Sensor Alignment by Tracks.

[7] Karimäki, V., Lampen, T., & Schilling, F.P. (2006). The HIP Algorithm for Track Based Alignment and its Application to the CMS Pixel Detector [White paper]. CERN.

[8] CMS Collaboration (2010). Alignment of the CMS silicon tracker during commissioning with cosmic rays. Journal of Instrumentation, 5(03), T03009–T03009.

[9] The CMS collaboration (2014). Alignment of the CMS tracker with LHC and cosmic ray data. Journal of Instrumentation, 9(06), P06009–P06009.

[10] D.N. Brown, A.V. Gritsan, Z.J. Guo, & D. Roberts (2009). Local alignment of the BaBar Silicon Vertex Tracking detector. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 603(3), 467–484.

[11] J. Davis (2022). CRAFT alignment with HipPy algorithm. Weekly Tracker DPG Meetings.

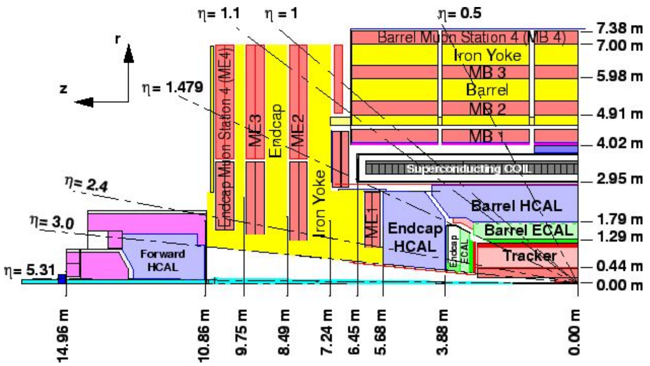
[12] L. Kang (2022). HipPy Alignment Algorithm. CMS Tracker Alignment Workshop: second edition.

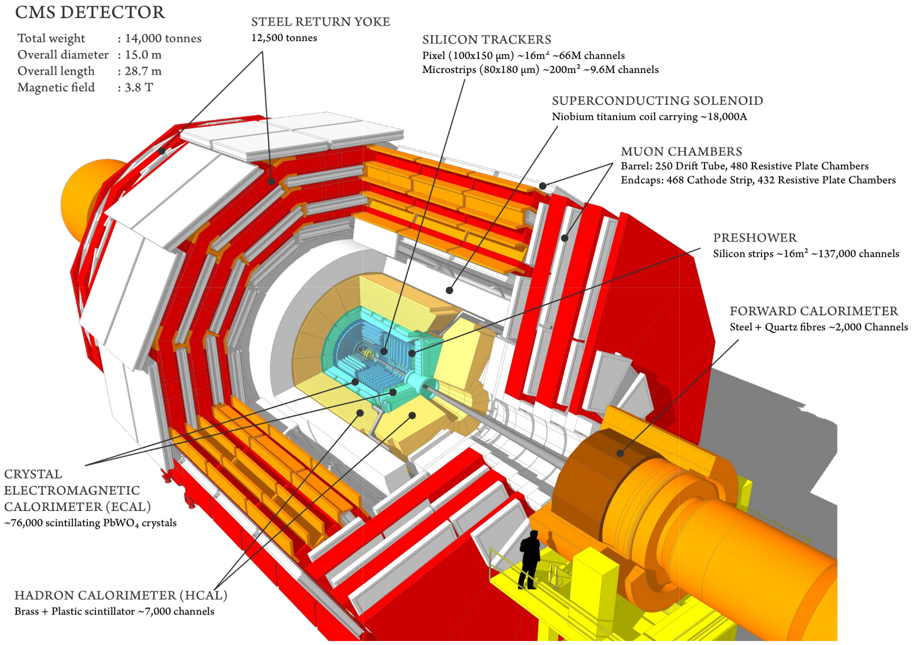
Backup Slides

The CMS Tracker

The Full Detector

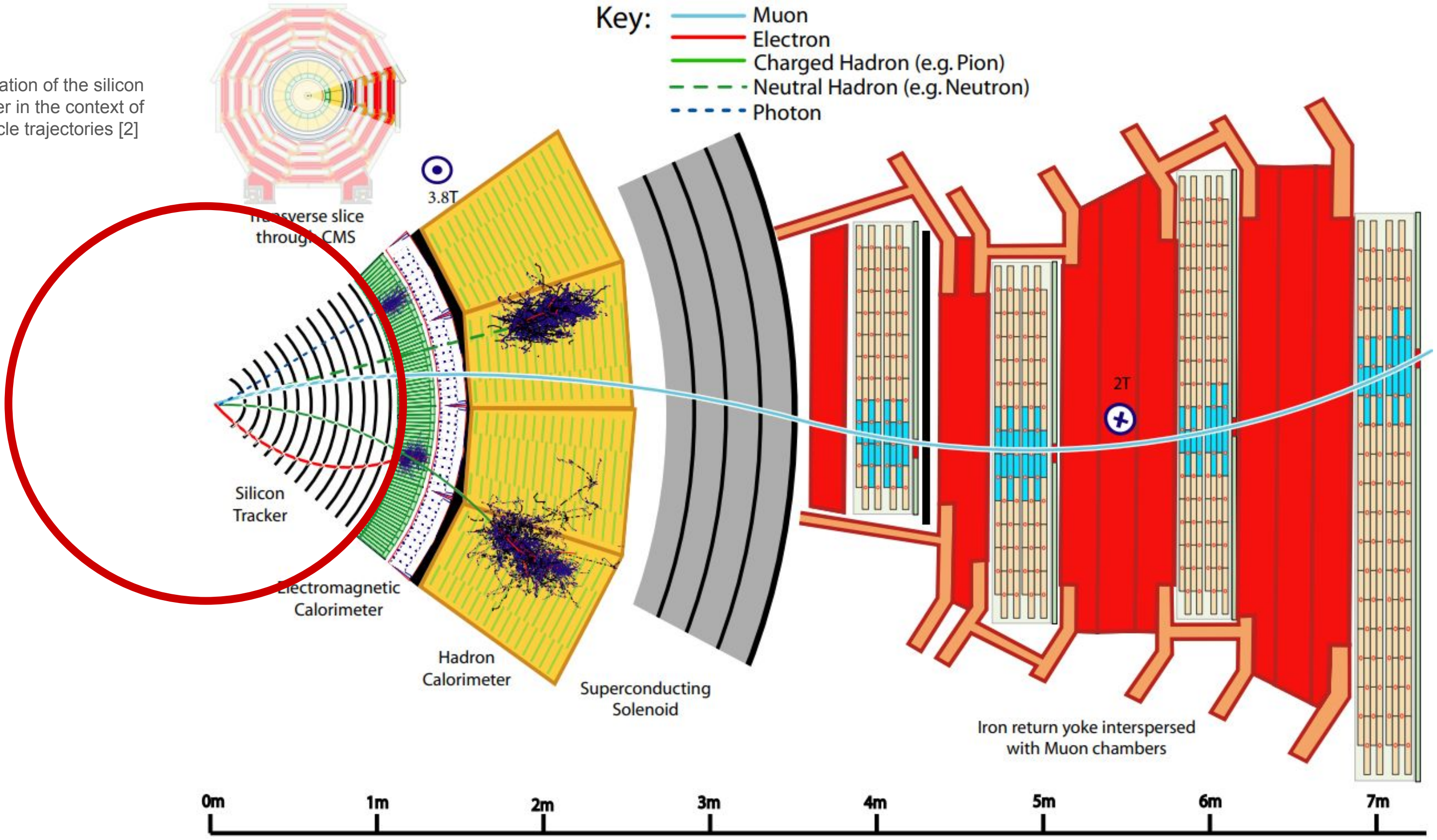
- Superconducting solenoid
  - 6 meter internal diameter, field of 3.8 T
- Inside solenoid
  - **Silicon trackers**
  - Calorimeters (ECAL and HCAL)
- Outside solenoid
  - Muon detection





Cross-sectional views of CMS for illustration of the tracker w.r.t. the full detector [1][2]

Illustration of the silicon tracker in the context of particle trajectories [2]



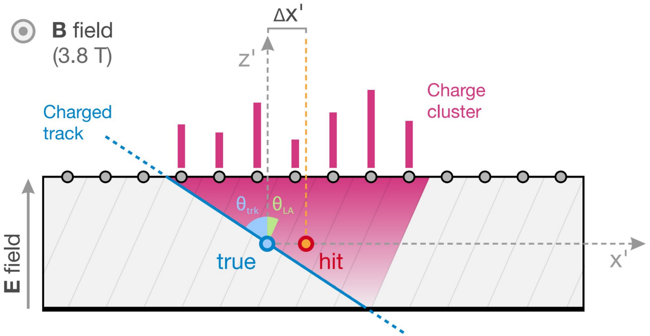
HipPy algorithm

- **p** represents the alignment parameters (also called alignables)
- **q** represents the track parameters (e.g. parameters related to the track curvature and the deflection by multiple scattering)
- $\mathbf{r}_i$ represents the track-hit residuals (1-dimensional vectors in the case of a single-sided module)
- $\mathbf{V}_i$ is a covariance matrix of the measurement uncertainties
- Jacobian $\mathbf{J}_i$ is defined as the derivative of the residual with respect to the sensor position parameters
  - Found analytically via small angle approximation
- Correlations between different modules and effects on the track parameters are accounted for by iterating the minimisation process and by refitting the tracks with new alignment constants after each iteration

$$\chi^2 = \sum_i^{\text{hits}} \mathbf{r}_i^T(\mathbf{p}, \cancel{\mathbf{q}})\mathbf{V}_i^{-1}\mathbf{r}_i(\mathbf{p}, \cancel{\mathbf{q}})$$

$$\mathbf{p}_m = \left[\sum_i^{\text{hits}} \mathbf{J}_i^T \mathbf{V}_i^{-1} \mathbf{J}_i\right]^{-1} \left[\sum_i^{\text{hits}} \mathbf{J}_i^T \mathbf{V}_i^{-1} \mathbf{r}_i\right]$$

χ^2 function for HipPy iteration [3]



Transverse view of a silicon module [5]

MILLEPEDE algorithm

- **p** represents the global alignment parameters (also called alignables)
- **q** represents the local track parameters (e.g. parameters related to the track curvature and the deflection by multiple scattering)
- $\mathbf{y}_{ji}$ represents the uncorrelated hit measurements (e.g. hits) and $\mathbf{f}_{ji}$ the predictions
- $\boldsymbol{\sigma}^{m}$ represents the uncertainty in the measurements (e.g. local hit resolution, alignment uncertainty)

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \frac{\left(y_{ji} - f_{ji}(\mathbf{p},\mathbf{q}_j)\right)^2}{\sigma_{ji}^2}$$

χ^2 function for MILLEPEDE-II global fit [3]

Types of Alignment

- Alignment during data taking
  - The tracker is realigned several times during the year
  - Especially when restarting the detector after a technical shutdown
  - Prompt calibration loop (PCL) for high-level structures in the pixel detector
- Alignment for physics analysis
  - EOY reconstruction
  - Legacy reprocessing
- Alignment in simulation
  - MC events are processed through the same reconstruction chain used for data