CBM Interactive Alignment Display

TRD Retreat 2023

Axel Puntke

Universität Münster

Purpose of the Tool in one Sentence

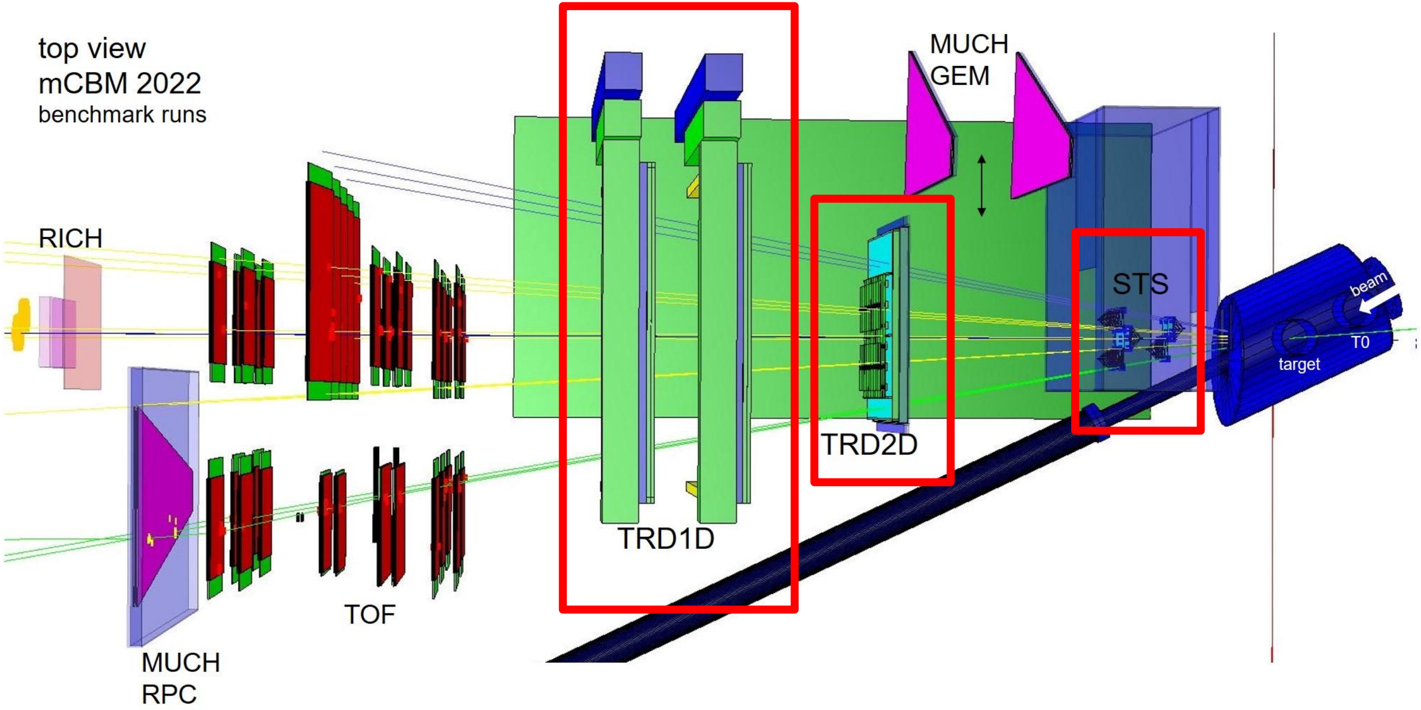
*User should be able to **align a setup of detector modules quickly** by looking at **correlations and residuals of straight tracks and hits** within the setup using **previously collected beamtime data***

(like aligning an optical system)

Technical Information

- Developed in C++
- QT 6 for the basic GUI
- Qwt - Qt Widgets for Technical Applications for histograms
- GPU code written in OpenCL

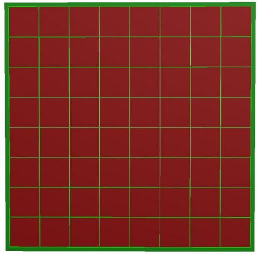
Tool Test Environment: mCBM 2022



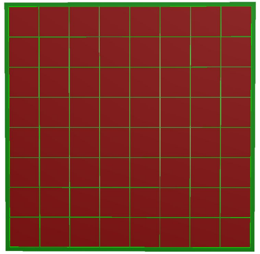
Goal: Spatially align TRD1D, TRD2D and STS

Degrees of Freedom

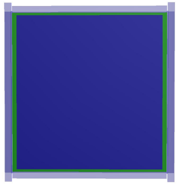
- 6 degrees of freedom per module: x, y, z, rot-x, rot-y, rot-z
- Modules in the 5 involved layers:



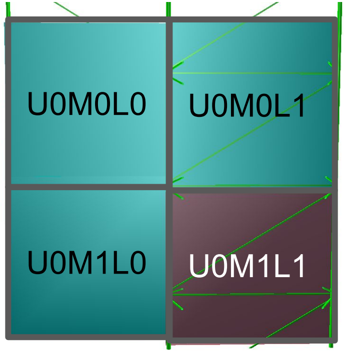
TRD1D-Y



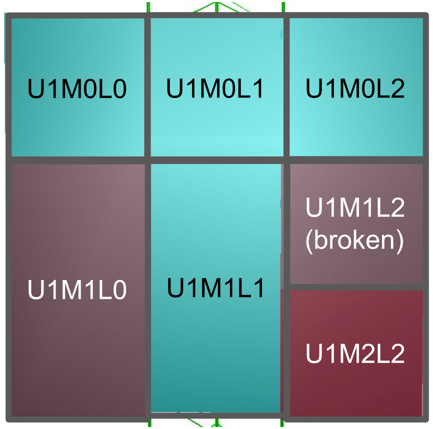
TRD1D-X



TRD2D-XY



STS-Unit0



STS-Unit1

	TRD1D-Y	TRD1D-X	TRD2D-XY	STS-Unit0	STS-Unit1
#Modules:	1	1	1	4	6 (7)

➔ Total degrees of freedom: 78

Data Preparation

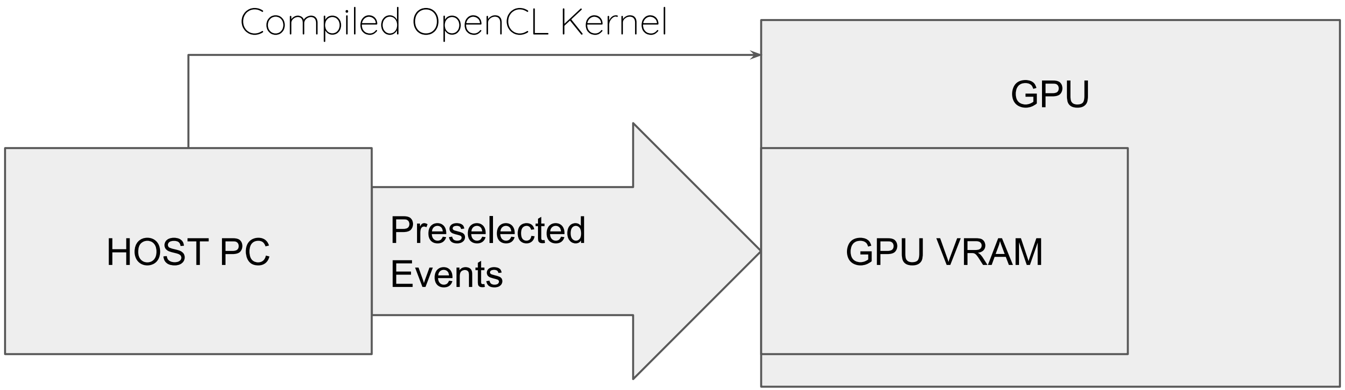
- Suitable Events are selected on Virgo Cluster
  - Exactly one hit per STS station to build unambiguous tracks
  - 9600 TimeSlices a 128 ms (20.5 min), from NiNi Run 2391 (May 26 2022)
  - Event count: 1,948,235 (761.5 MB)
- Saved in custom data format (GPU ready):
  - STS base hit 1&2 position + originating module ID
  - 20x TRD reference hit position + originating module ID

```
struct AlignEvent
{
    int32_t baseHit1Pos[3];
    uint32_t baseHit1AlignModuleId;
    int32_t baseHit2Pos[3];
    uint32_t baseHit2AlignModuleId;
    int32_t refHitPos[20 * 3];
    uint32_t refHitAlignModuleId[20];
} __attribute__((packed));
```

- FairTask available for selection

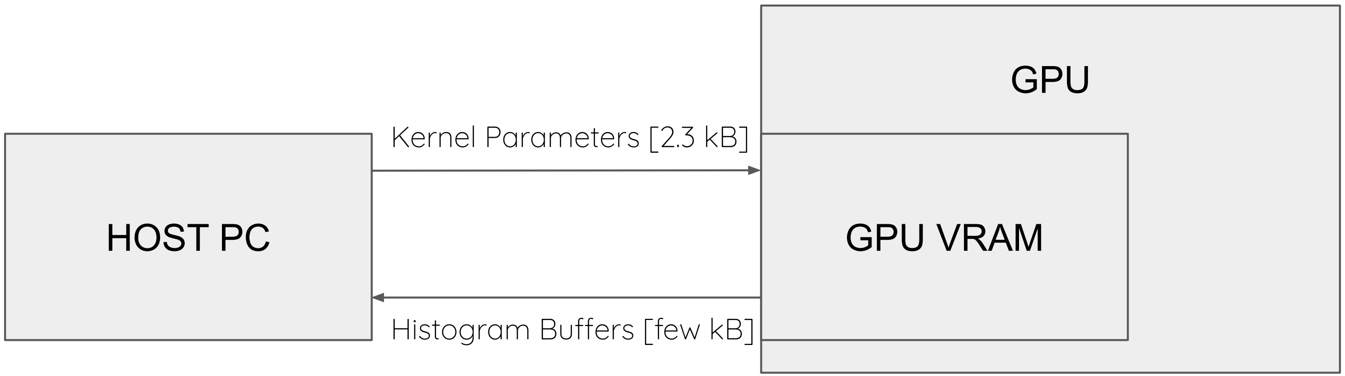
Tool Workflow

- Happens only once at program startup time:
  - Compilation of OpenCL kernel and upload to GPU
  - Upload of preselected events into GPU VRAM



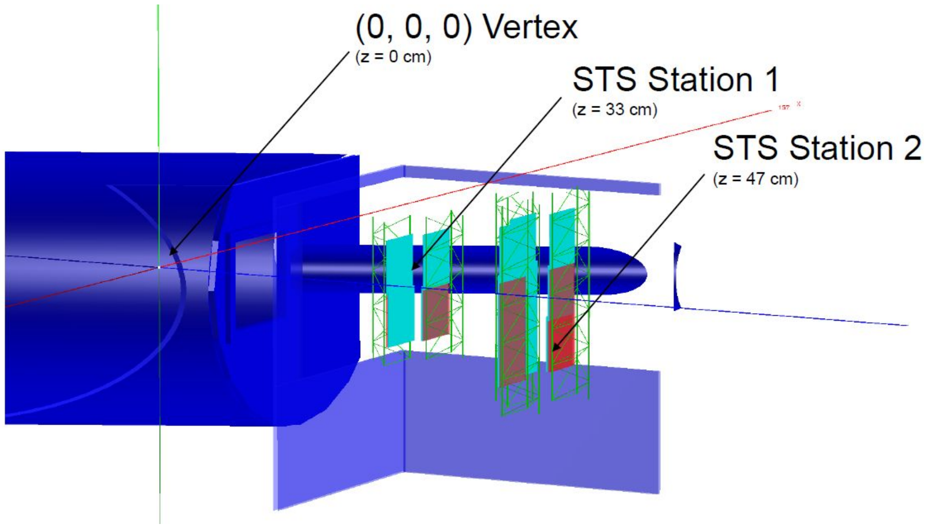
Tool Workflow

- Happens whenever a value is changed by the user in the UI (“every new frame”):
  - Upload of kernel parameters (alignment matrices, histogram borders/binnings etc.)
  - Re-Initialization of histogram buffers
  - Computation of correlation histograms on the GPU
  - Download of computed histograms from GPU



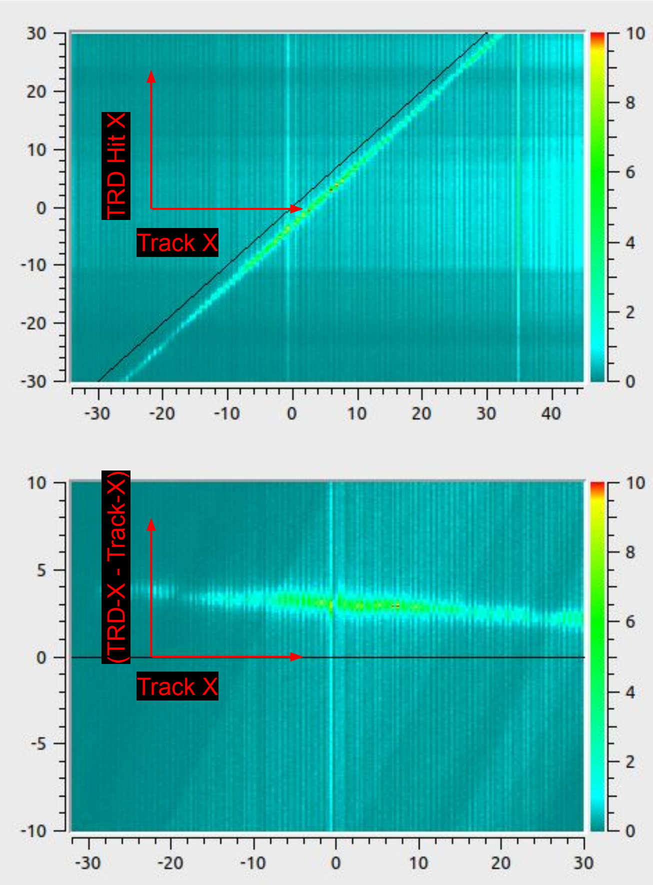
Track Building on GPU

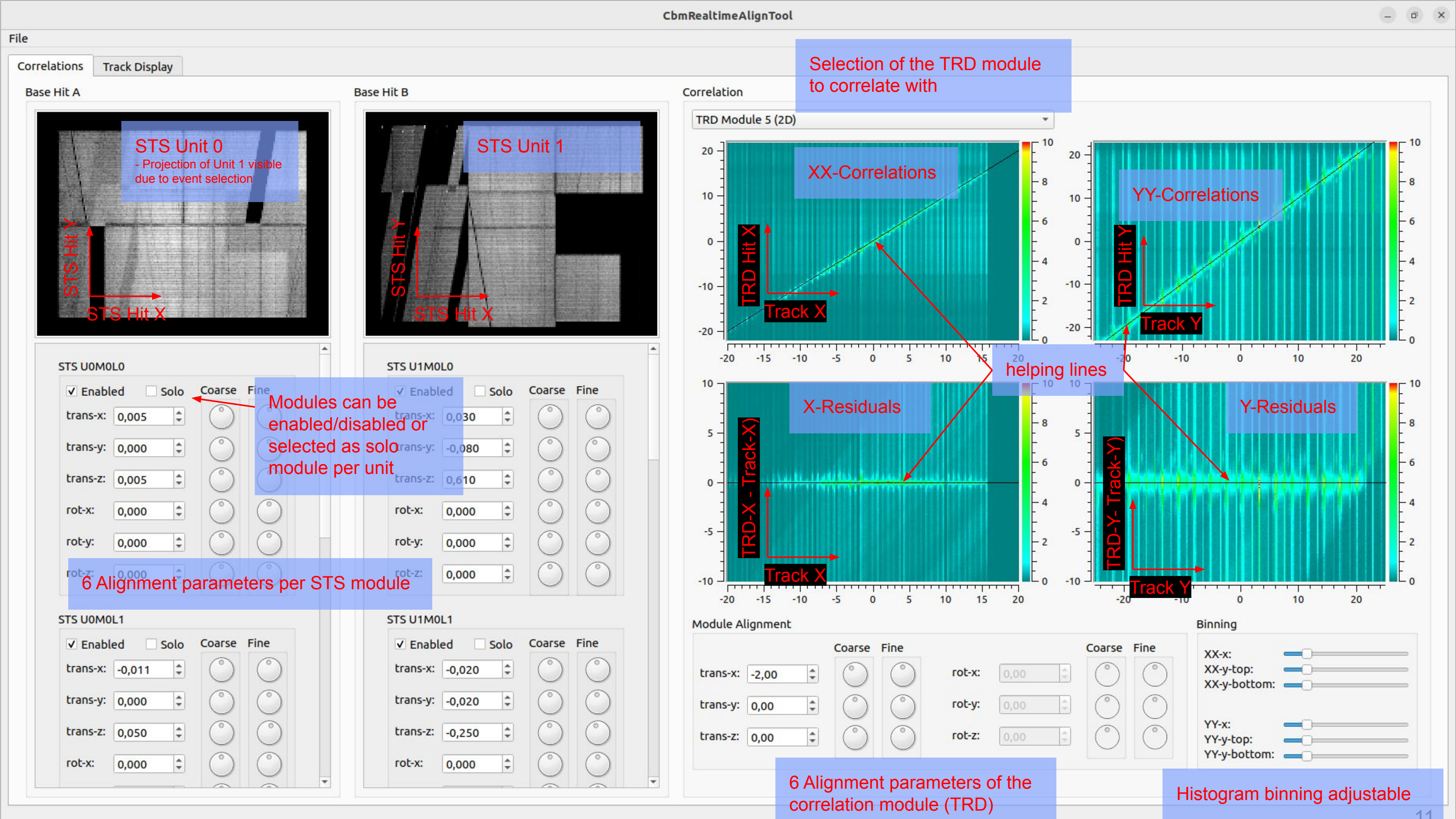
- Track is built using the two STS base hits, connected by straight line (no fit)
- Computation of these tracks happens on GPU
- Alignment matrix (module translation & rotation) is applied to the hit positions in advance



Correlation Plots

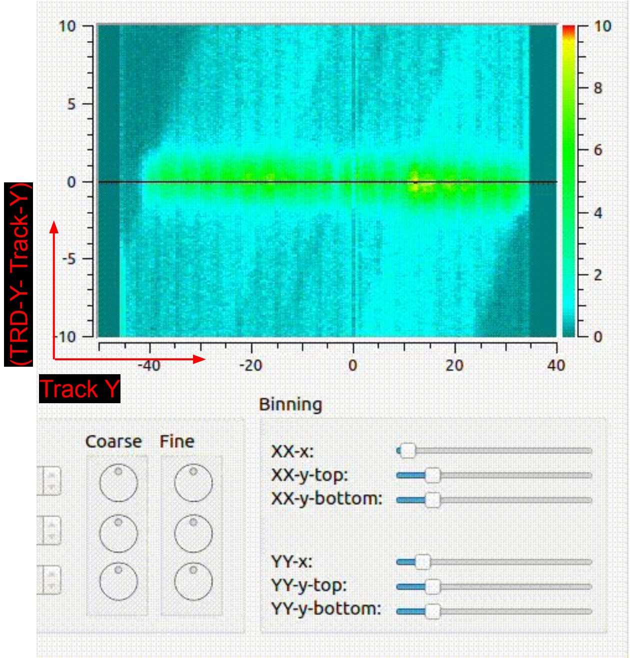
- Intersection point I of track with plane z = d is computed
    - In this case: d = distance of TRD padplane to origin
- Top plot shows the track I.x position vs. TRD x position
    - Should ideally be on the angle bisector (plotted as black helping line)
- Bottom plot shows track I.x position vs. residual (x_{TRD} - x_{Track})
    - Should ideally be always zero (plotted as black helping line)




CbmRealtimeAlignTool
File
Correlations
Track Display
Base Hit A
Base Hit B
Correlation
Selection of the TRD module to correlate with
TRD Module 5 (2D)
STS Unit 0
- Projection of Unit 1 visible due to event selection
STS Hit Y
STS Hit X
STS Unit 1
XX-Correlations
YY-Correlations
TRD Hit X
Track X
TRD Hit Y
Track Y
helping lines
X-Residuals
Y-Residuals
TRD-X - Track-X)
TRD-Y- Track-Y)
STS U0M0L0
STS U1M0L0
Enabled
Solo
Coarse
Fine
Modules can be enabled/disabled or selected as solo module per unit
trans-x:
trans-y:
trans-z:
rot-x:
rot-y:
rot-z:
6 Alignment parameters per STS module
STS U0M0L1
STS U1M0L1
Module Alignment
Binning
XX-x:
XX-y-top:
XX-y-bottom:
YY-x:
YY-y-top:
YY-y-bottom:
6 Alignment parameters of the correlation module (TRD)
Histogram binning adjustable

Histogram Binning adjustable in Real-Time

- Because histograms are re-computed within few ms, also their binning is adjustable on the fly in real-time
  - Good demonstration of GPU computing capabilities



Y-Residuals of last TRD module, ~2M events, GIF plays in real-time, x-binning gets adjusted

Config File

Base Hit Module Definition (STS)


```
"base_hit_1_modules": {

    "group_name": "STS Unit 0",
    "hist_xy": {"xmin" : -7, "xmax" : 7, "ymin" : -7, "ymax" : 7, "nbinsx" : 200, "nbinsy" : 200},
    "module_list": [
        {
            "id" : 0,
            "name" : "STS U0M0L0",
            "geo_node_path" : "/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_1/HalfLadder09d_2/HalfLadder09d_Module03_1",
            "center" : [-2.979600, 0.000000, 32.435000],
            "rot_axis_x" : [-1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [-0.000000, -1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, 1.000000]
        },
        {
            "id" : 1,
            "name" : "STS U0M0L1",
            "geo_node_path" : "/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_2/HalfLadder09d_2/HalfLadder09d_Module03_1",
            "center" : [2.979600, 0.000000, 34.065000],
            "rot_axis_x" : [1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [0.000000, -1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, -1.000000]


        },
        {
            "id" : 10,
            "name" : "STS U0M1L0",
            "geo_node_path" : "/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_1/HalfLadder09d_2/HalfLadder09d_Module03_2",
            "center" : [-2.979600, -2.870000, 32.635000],
            "rot_axis_x" : [-1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [-0.000000, -1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, 1.000000]
        },
        {
            "id" : 11,
            "name" : "STS U0M1L1",
            "geo_node_path" : "/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_2/HalfLadder09d_2/HalfLadder09d_Module03_2",
            "center" : [2.979600, -2.870000, 33.865000],
            "rot_axis_x" : [1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [0.000000, -1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, -1.000000]
        }
    ]
},
"base_hit_2_modules": {

    "group_name": "STS Unit 1",
    "hist_xy" : {"xmin" : -10, "xmax" : 10, "ymin" : -10, "ymax" : 10, "nbinsx" : 200, "nbinsy" : 200},
    "module_list": [
        {
            "id" : 100,
            "name" : "STS U1M0L0",
            "geo_node_path" : "/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder10_1/HalfLadder10d_2/HalfLadder10d_Module03_1",
            "center" : [-5.959200, 0.000000, 47.765000],
```


Correlation Module Definition (TRD)


```
"correlation_modules": {

    "group_name": "Correlation Modules",
    "module_list": [
        {
            "id" : 501,
            "name" : "TRD Module 5 (2D)",
            "hist_corr_xx" : {"xmin" : -20, "xmax" : 20, "ymin" : -22, "ymax" : 22, "nbinsx" : 200, "nbinsy" : 200},
            "hist_corr_yy" : {"xmin" : -25, "xmax" : 25, "ymin" : -23, "ymax" : 23, "nbinsx" : 200, "nbinsy" : 200},
            "hist_res_x" : {"xmin" : -20, "xmax" : 20, "ymin" : -10, "ymax" : 10, "nbinsx" : 200, "nbinsy" : 200},
            "hist_res_y" : {"xmin" : -25, "xmax" : 25, "ymin" : -10, "ymax" : 10, "nbinsx" : 200, "nbinsy" : 200},
            "geo_node_path" : "/cave_1/trd_v22h_mcbm_0/layer01_20101/module9_101001001",
            "center" : [-3.000000, 0.000000, 128.700000],
            "rot_axis_x" : [1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [0.000000, 1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, 1.000000],
            "corr_plane_z_offset" : -11.93
        },
        {
            "id" : 502,
            "name" : "TRD Module 21 (1D-X)",
            "hist_corr_xx" : {"xmin" : -34, "xmax" : 45, "ymin" : -30, "ymax" : 30, "nbinsx" : 200, "nbinsy" : 200},
            "hist_corr_yy" : {"xmin" : -40, "xmax" : 40, "ymin" : -32, "ymax" : 32, "nbinsx" : 70, "nbinsy" : 200},
            "hist_res_x" : {"xmin" : -34, "xmax" : 45, "ymin" : -10, "ymax" : 10, "nbinsx" : 200, "nbinsy" : 200},
            "hist_res_y" : {"xmin" : -40, "xmax" : 40, "ymin" : -10, "ymax" : 10, "nbinsx" : 70, "nbinsy" : 200},
            "geo_node_path" : "/cave_1/trd_v22h_mcbm_0/layer02_10202/module8_101002001",
            "center" : [0.000000, 0.000000, 175.700000],
            "rot_axis_x" : [1.000000, 0.000000, 0.000000],
            "rot_axis_y" : [0.000000, 1.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, 1.000000],
            "corr_plane_z_offset" : -11.9
        },
        {
            "id" : 503,
            "name" : "TRD Module 37 (1D-Y)",
            "hist_corr_xx" : {"xmin" : -50, "xmax" : 40, "ymin" : -40, "ymax" : 50, "nbinsx" : 70, "nbinsy" : 200},
            "hist_corr_yy" : {"xmin" : -50, "xmax" : 40, "ymin" : -40, "ymax" : 50, "nbinsx" : 200, "nbinsy" : 200},
            "hist_res_x" : {"xmin" : -50, "xmax" : 40, "ymin" : -10, "ymax" : 10, "nbinsx" : 70, "nbinsy" : 200},
            "hist_res_y" : {"xmin" : -50, "xmax" : 40, "ymin" : -10, "ymax" : 10, "nbinsx" : 200, "nbinsy" : 200},
            "geo_node_path" : "/cave_1/trd_v22h_mcbm_0/layer03_11303/module8_101303001",
            "center" : [0.000000, 0.000000, 202.700000],
            "rot_axis_x" : [-0.000000, -1.000000, 0.000000],
            "rot_axis_y" : [1.000000, -0.000000, 0.000000],
            "rot_axis_z" : [0.000000, 0.000000, 1.000000],
            "corr_plane_z_offset" : -11.9
        }
    ]
},
```

Config File

- Contains all modules, identified by unique id (user-definable), grouped in
  - 1. Modules used for Base Hit A (STS Unit 0)
  - 2. Modules used for Base Hit B (STS Unit 1)
  - 3. Modules to correlate with (TRD Modules)
- Allows to set all histogram axis limits and binnings individually per module
- Contains root geo node paths
  - Used to export alignment matrices to CbmRoot compatible macro
- Contains module center positions
  - Used for determining the plane of the modules to correlate with
  - Used for module rotation
- Contains rotation axes (x, y, z)

- Geo node paths, center positions and rotation axes can be extracted out of existing root geometries using `GetGeometryModulePositions.C` macro

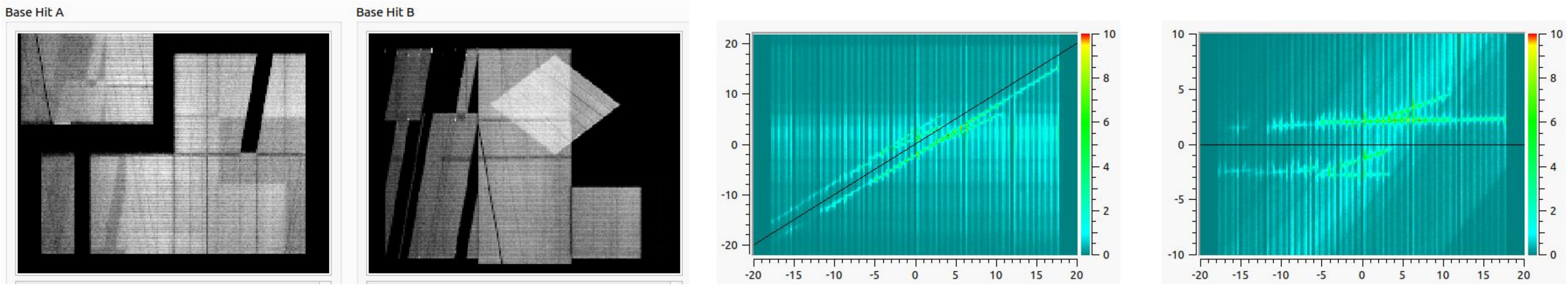
Import/Export of Alignment Matrices

- Alignment matrices for each TRD and STS module can be loaded from and saved to a ROOT macro
    - Produces a root file directly compatible with CBM reconstruction chain

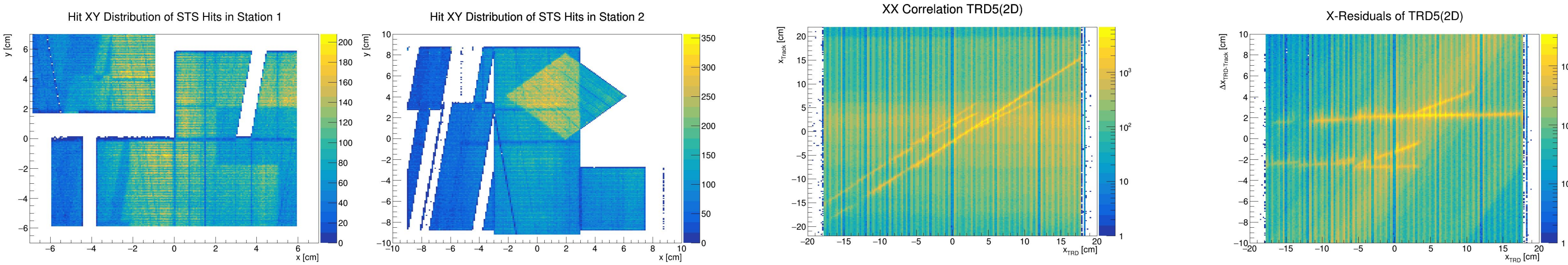
```cpp
int create_alignment_mcbm_beam_2022_05_23_nickel()
{
  // Define the basic structure which needs to be filled with information
  // This structure is stored in the output file and later passed to the
  // FairRoot framework to do the (miss)alignment
  std::map<std::string, TGeoHMatrix> matrices;

  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_1/HalfLadder09d_2/HalfLadder09d_Module03_1", 0, -0.03, 1.05, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_2/HalfLadder09d_2/HalfLadder09d_Module03_1", 0, -0.03, 0.72, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_1/HalfLadder09d_2/HalfLadder09d_Module03_2", -0.03, -0.03, 0.99, 0, -3, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station01_1/Ladder09_2/HalfLadder09d_2/HalfLadder09d_Module03_2", 0, -0.015, 0.51, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder10_1/HalfLadder10d_2/HalfLadder10d_Module03_1", 0.39, 0, 0.32, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder12_2/HalfLadder12d_2/HalfLadder12d_Module03_1", 0.33, -0.06, 0.87, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder11_3/HalfLadder11d_2/HalfLadder11d_Module03_1", 0.36, -0.03, 0.45, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder10_1/HalfLadder10d_2/HalfLadder10d_Module04_2", 0.36, 0.03, 0.28, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder12_2/HalfLadder12d_2/HalfLadder12d_Module04_2", 0.3, -0.03, 0.48, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder11_3/HalfLadder11d_2/HalfLadder11d_Module03_2", 0, 0, 0, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/sts_v22f_mcbm_0/Station02_2/Ladder11_3/HalfLadder11d_2/HalfLadder11d_Module03_3", 0.3, 0, 0.05, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/trd_v22h_mcbm_0/layer01_20101/module9_101001001", -0.03, -0.21, 0.63, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/trd_v22h_mcbm_0/layer02_10202/module8_101002001", 0.3, 0, 0, 0, 0, 0));
  matrices.insert(AlignNode("/cave_1/trd_v22h_mcbm_0/layer03_11303/module8_101303001", 0, -0.6, 0, 0, 0, 0));


  // save matrices to disk
  TFile* misalignmentMatrixRootfile = new TFile("AlignmentMatrices_mcbm_beam_2022_05_23_nickel.root", "RECREATE");
  if (misalignmentMatrixRootfile->IsOpen()) {
    gDirectory->WriteObject(&matrices, "MisalignMatrices");
    misalignmentMatrixRootfile->Write();
    misalignmentMatrixRootfile->Close();
  }

  return 0;
}
```

Test of Alignment Matrices in CbmRoot

- Control plots of the tool are re-plotted in CbmRoot using exported alignment matrices
  - Plots match, so alignment tool correctly reproduces CbmRoot geometry transformations

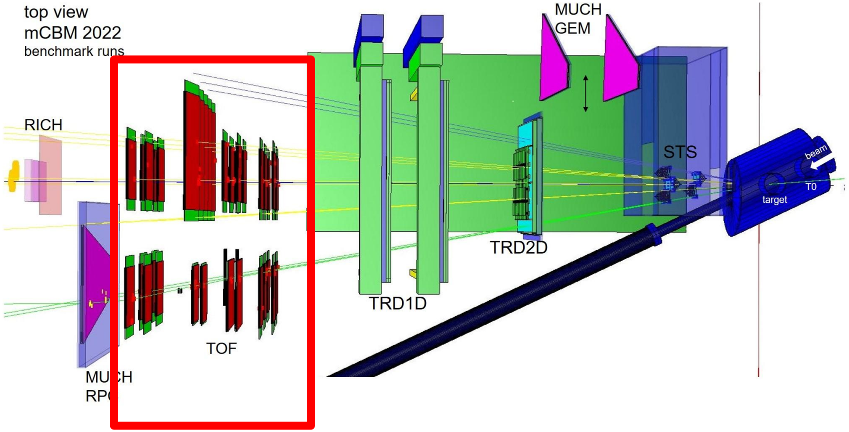


Top: Control plots computed on GPU, Bottom: CbmRoot Reco plots, modules misaligned on purpose



Outlook: Inclusion of TOF

- After TRD and STS are aligned well, one could replace the second base hit with e.g. TRD2D or one of the TRD1Ds
  - Larger lever arm
  - Would take the alignment of the other modules take into account indirectly
- One could also try to implement a real fit routine on GPU
  - Is already used in diffusion Magnetic Resonance Imaging (dMRI) [https://github.com/robbert-harms/MOT]



Source Code Download

- Source code is available on Git:
  https://git.cbm.gsi.de/trd/software-extra/CbmRealtimeAlignTool
- Readme provides instructions how to compile and run the tool and how to produce the necessary input data set using CbmRoot
  - Input data is also available for download for use without CbmRoot

trd > software-extra > CbmRealtimeAlignTool

CbmRealtimeAlignTool

General Information

This tool was developed to adjust and try out alignment matrices for CBM detectors and see their effect on spatial correlations. The tool accepts events stored in a structure `AlignEvent` which is saved sequetially in an input file `data/2391.alignev.data` and loaded into the GPU VRAM. Using these events, correlation plots are calculated directly on the GPU with the user-specified alignment matrices and displayed within milliseconds. This enables the user to interactively find a suitable alignment. See the talk "Interactive (semi-automatic) alignment display" (https://indico.gsi.de/event/17503/) for an overview of the tool.

Pre-Requirements

- Qt6 or Qt5
- Qwt 6.2.0 (https://qwt.sourceforge.io/qwtinstall.html)
- For development it is recommended to download and install QtCreator (https://www.qt.io/download).

Compilation

```
git clone https://git.cbm.gsi.de/trd/software-extra/CbmRealtimeAlignTool.git src
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX=../inst ../src
make -j install
```

Getting Alignment Events

Alignment events are currently events which have exactly 2 STS hits in two different stations (to have a clean sample). They can be produced by running this FairTask with your CbmRoot over your data: https://git.cbm.gsi.de/apuntke/cbmroot/-/blob/master/analysis/detectors/trd/CbmTrdAlignmentEventSelector.h Example initialization:

```
        CbmTrdAlignmentEventSelector* trdAliEventSelTask = new CbmTrdAlignmentEventSelector();
        trdAliEventSelTask->SetOutfile(Form("%s.alignev.data", sOutFileBase.Data()));
        run->AddTask(trdAliEventSelTask);
```

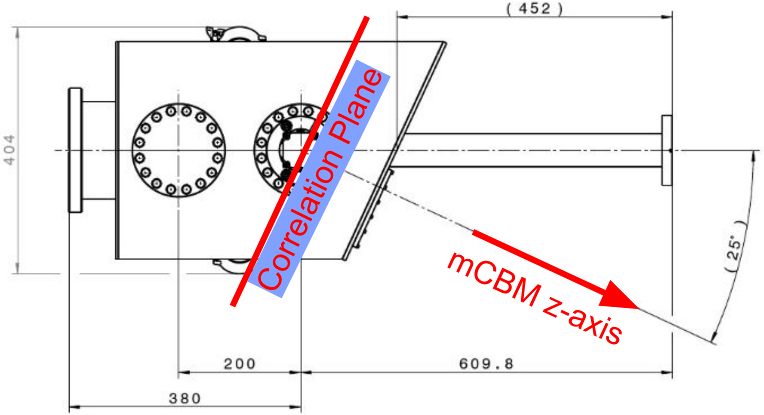
If you distribute the analysis to multiple processes and therefore have multiple `*.alignev.data` files, you can just

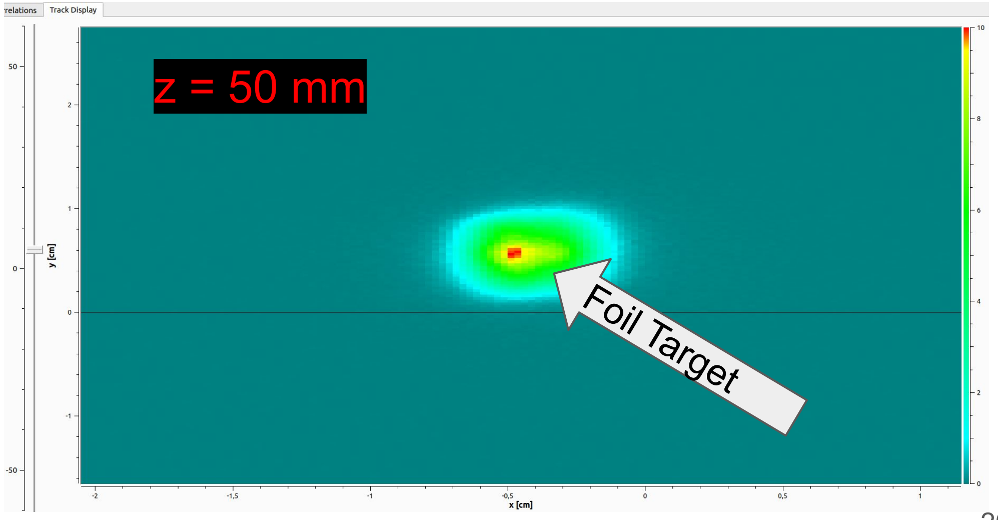
Possible Application in upcoming mCBM Beamtime

- Collect data in NiNi run (low multiplicity) for ~ 20 min
  - No L1 tracking needed, we are not dependent on L1 developers (in case L1 tracking does not run from day 1 out of the box)
- Extract AlignEvents and load into alignment display
  - As seen in previous slides
- Zero all residuals by adjusting alignment parameters
- Apply alignment and extract second set of AlignEvents, this time including TOF hits as reference hits (instead TRD)
- Also align TOF using the same procedure as for TRD

Currently in Development: Track Display

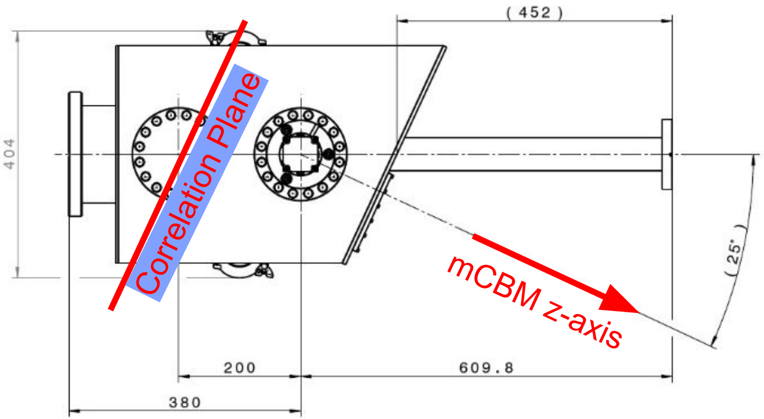
- Displays the distribution of intersection points of a plane perpendicular to z-axis and reference tracks (alignment matrices applied)
- Allows to see structures inside of the target chamber (target tomography)
- Available in git branch "trackdisplay"

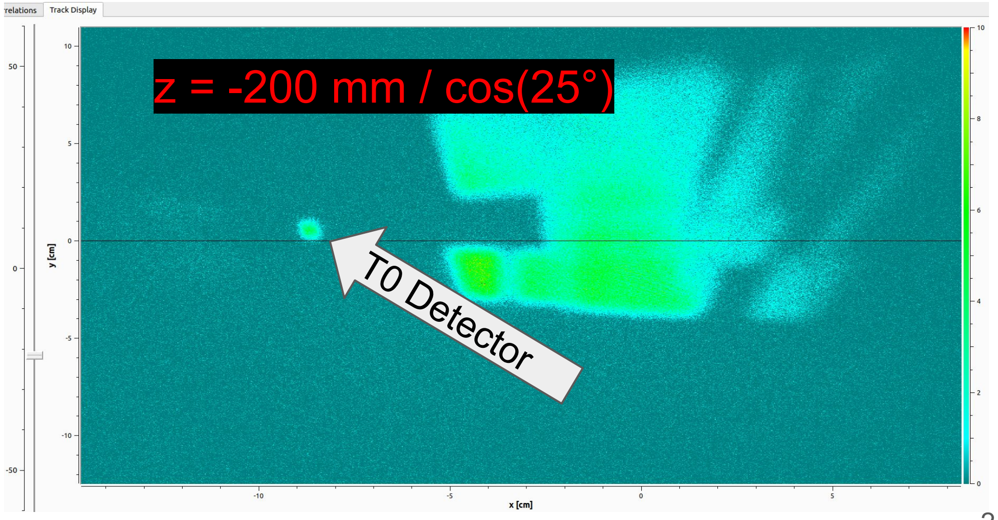




Currently in Development: Track Display

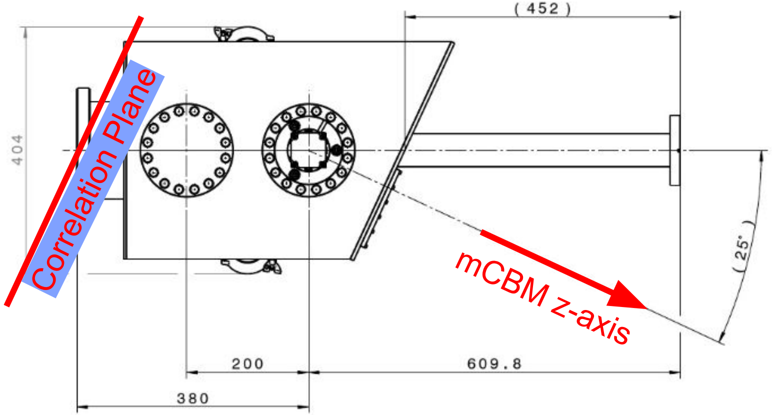
- Displays the distribution of intersection points of a plane perpendicular to z-axis and reference tracks (alignment matrices applied)
- Allows to see structures inside of the target chamber (target tomography)
- Available in git branch "trackdisplay"

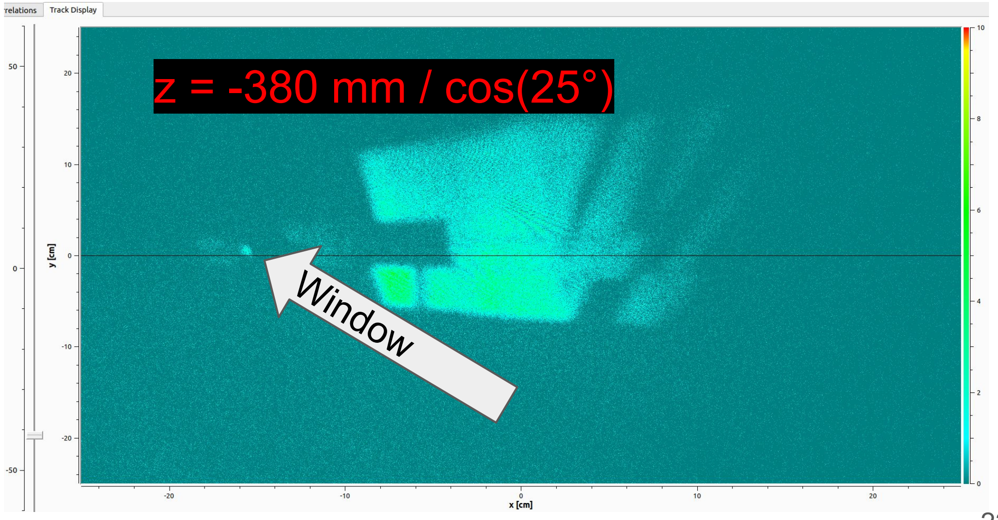




Currently in Development: Track Display

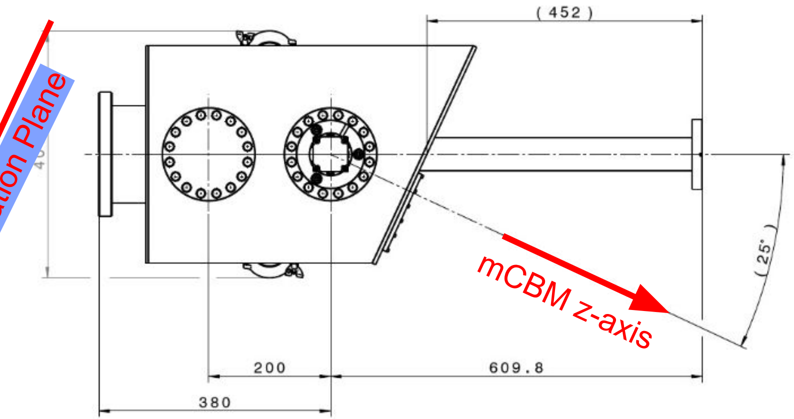
- Displays the distribution of intersection points of a plane perpendicular to z-axis and reference tracks (alignment matrices applied)
- Allows to see structures inside of the target chamber (target tomography)
- Available in git branch "trackdisplay"

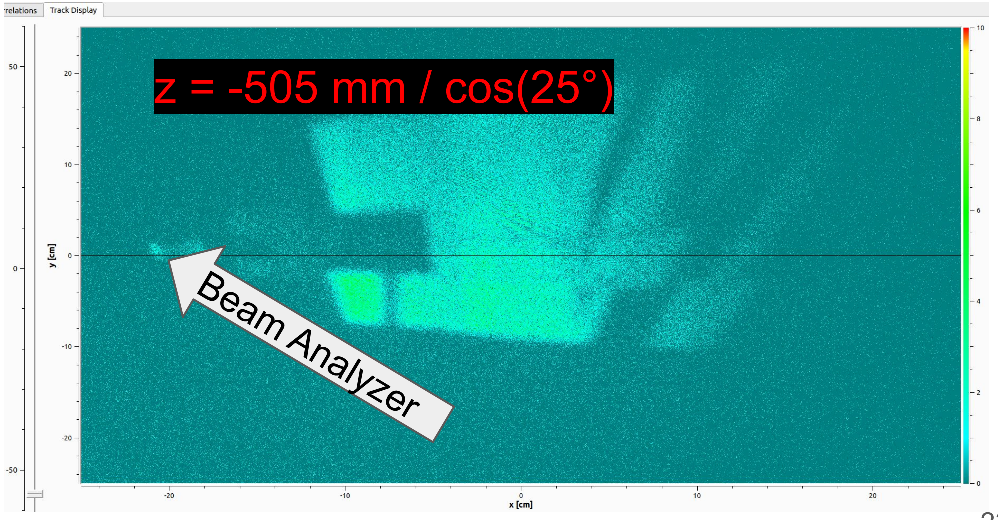




Currently in Development: Track Display

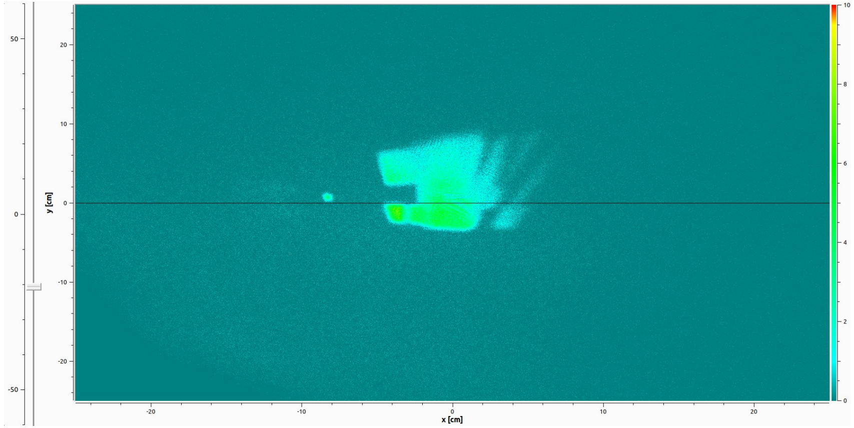
- Displays the distribution of intersection points of a plane perpendicular to z-axis and reference tracks (alignment matrices applied)
- Allows to see structures inside of the target chamber (target tomography)
- Available in git branch "trackdisplay"





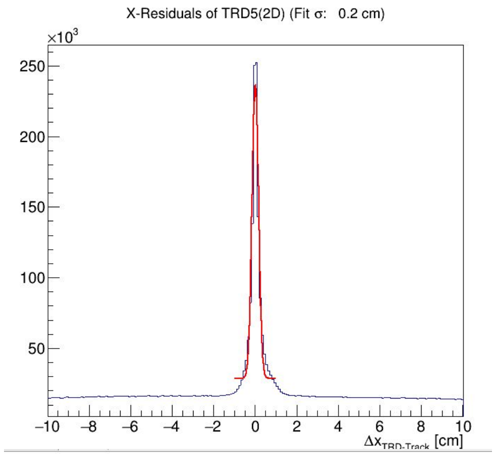
How to evaluate the Quality of the Alignment?

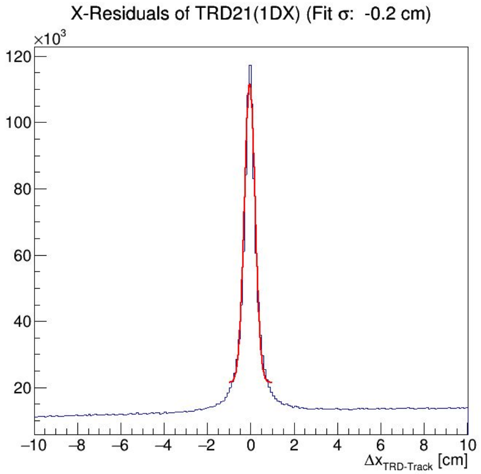
- Residuals and HitX-TrackX correlations are 0 or on angle bisector by definition
- Structures of target chamber (previous slide) are also visible without applying any alignment
  - Not suitable for quality evaluation
- Other plots would be nice to evaluate the alignment quality

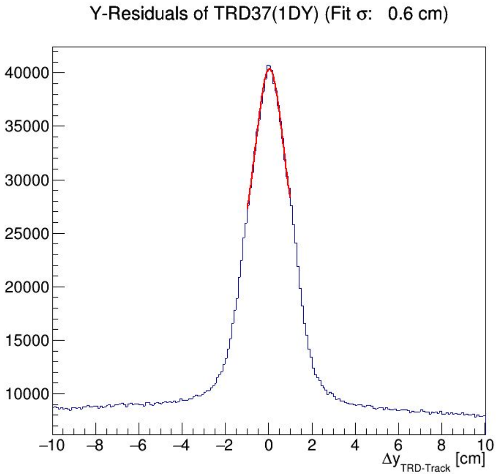


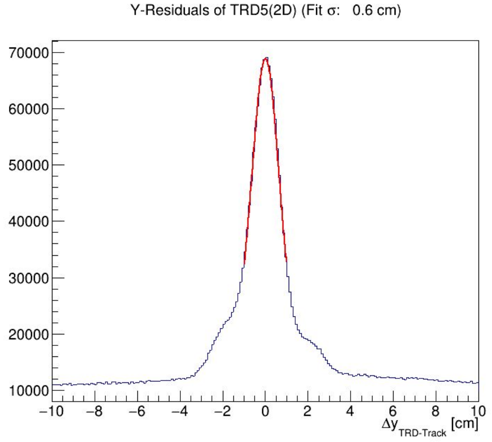
Reference track distribution without applying any alignment

Resolution Determination using Reference Tracks?





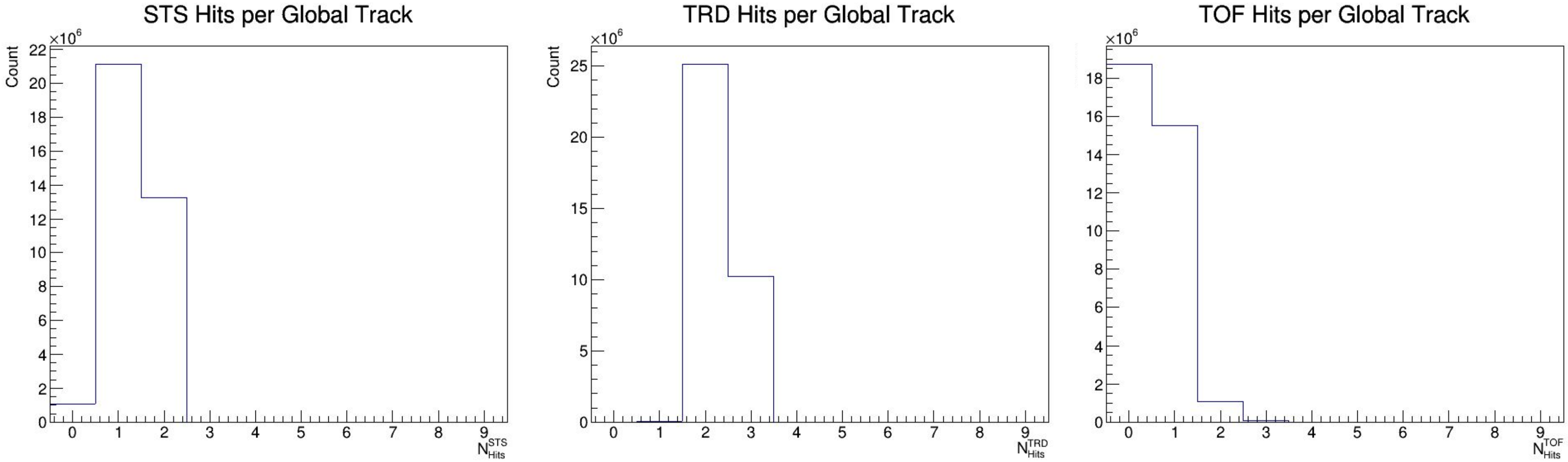




- Using **official alignment** from CbmRoot computing master
- Does not archive the resolutions obtained by Alexandru @ CbmWeek
- **L1 Global Tracks needed** with more than 2 hits

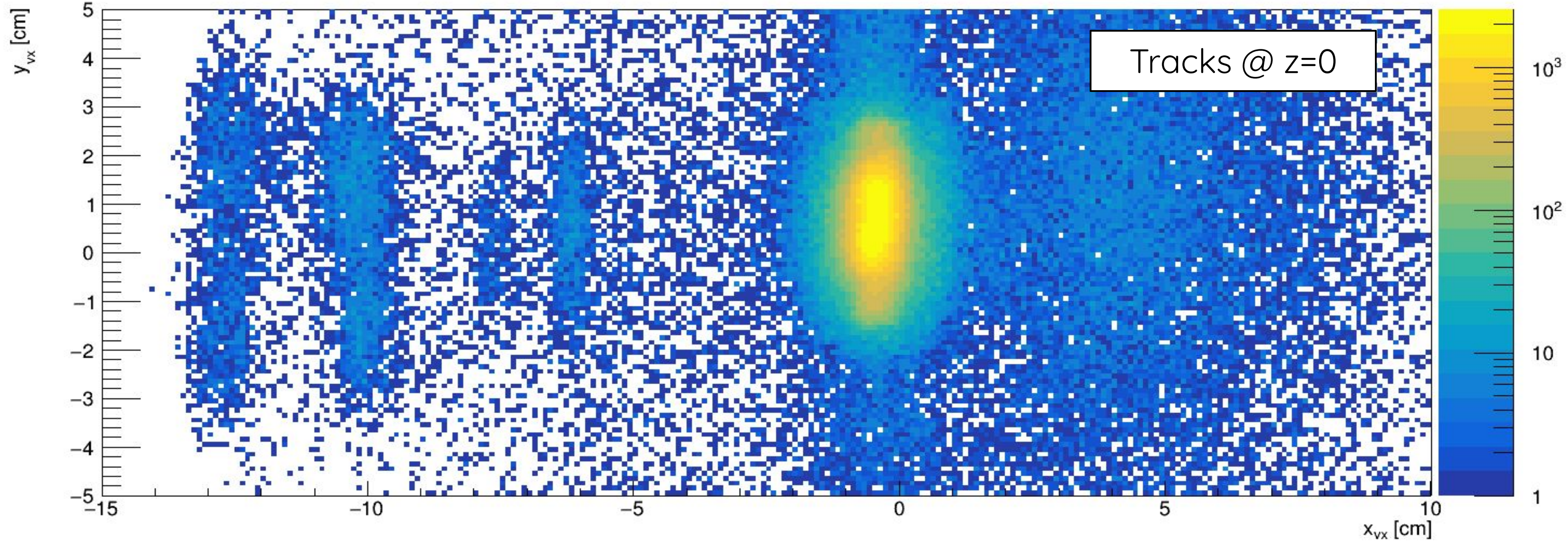
WIP: Alignment QA using L1 Tracks

- L1 Track Selection: 2 STS Hits, ≥ 1 TRD Hits, ≥ 1 TOF Hits
- Fit straight line through STS hits and TRD1D hit
- Using **official alignment** from CbmRoot computing master



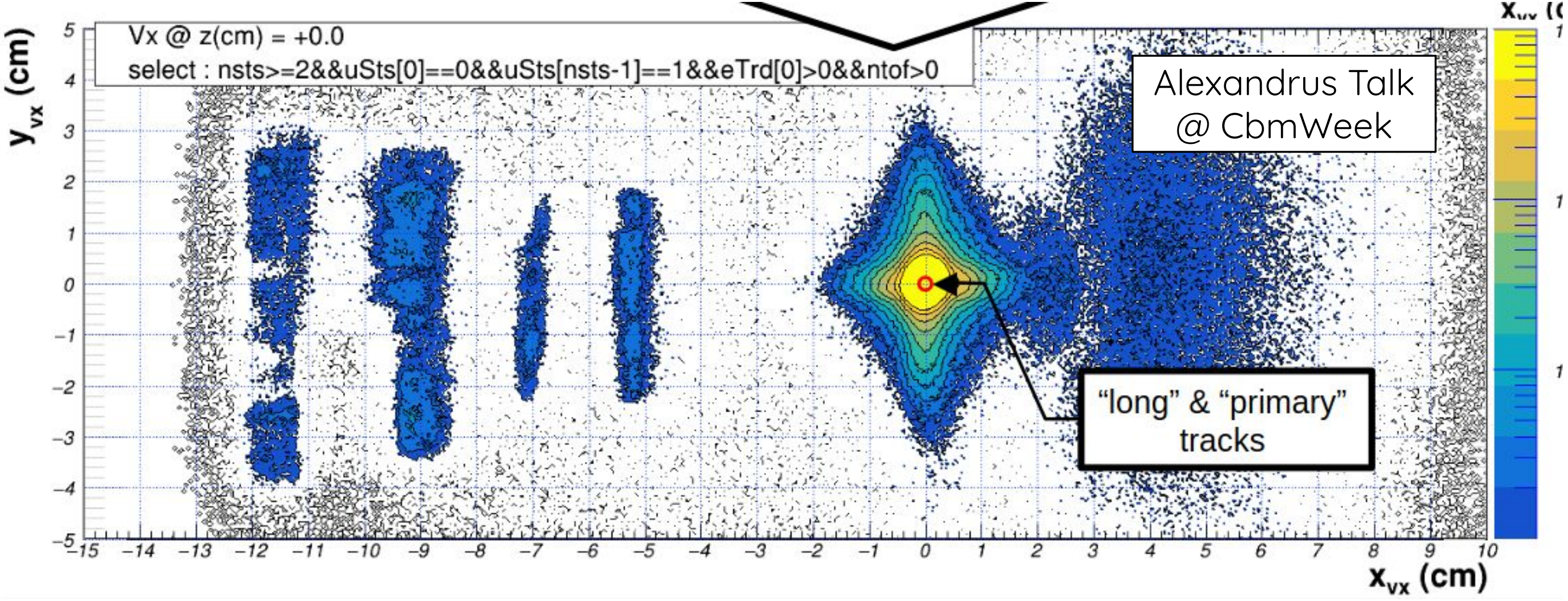
WIP: Alignment QA using L1 Tracks

- L1 Track Selection: 2 STS Hits, ≥ 1 TRD Hits, ≥ 1 TOF Hits
- Fit straight line through STS hits and TRD1D hit
- Using **official alignment** from CbmRoot computing master



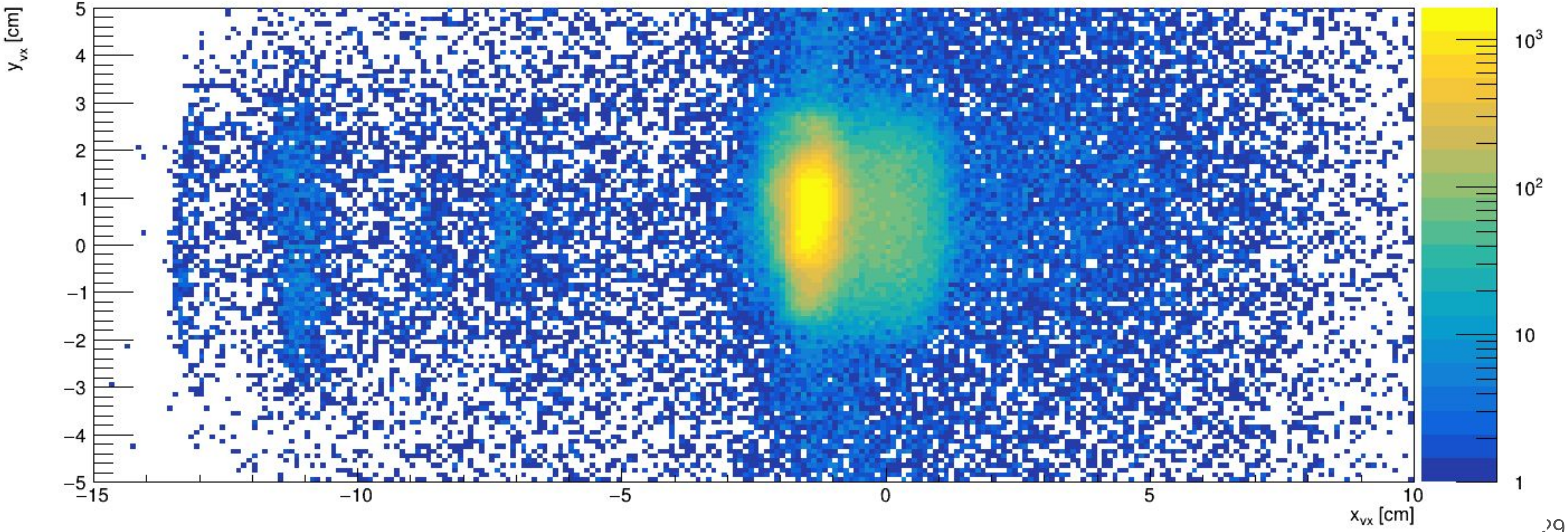
WIP: Alignment QA using L1 Tracks

- Comparable to Alexandrus results
  - But his target spot is more central at (0,0), so probably he uses different alignment than CbmRoot master



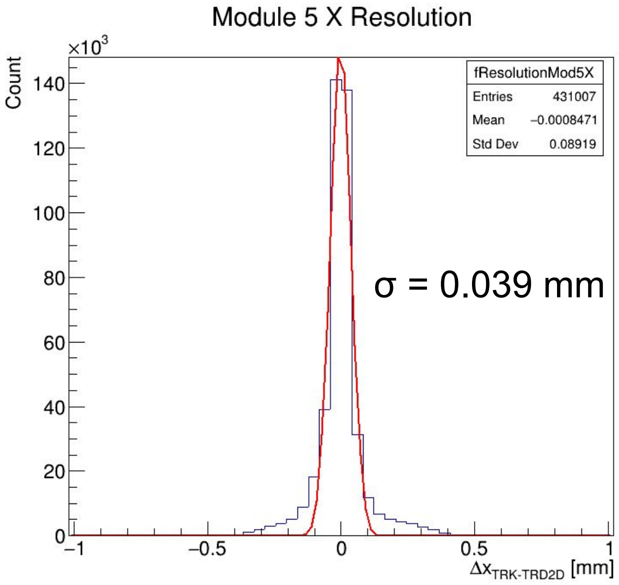
WIP: Alignment QA using L1 Tracks

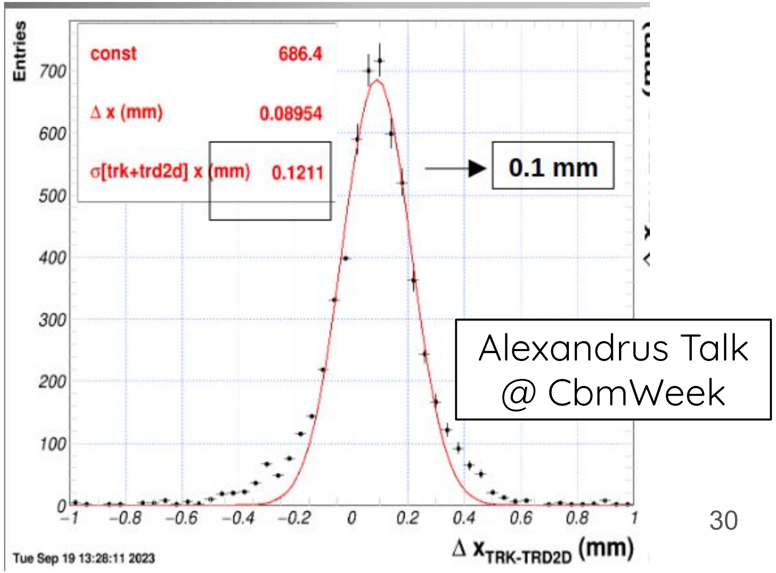
- Cross-check: Without any alignment



WIP: Alignment QA using L1 Tracks

- Calculating module resolutions:
  - Plots do not match yet with Alexandrus ones
  - Resolution too small, probably because fit is done without respecting uncertainties
- Need to ask Alexandru for exact procedure he uses to obtain these plots





Summary

- Tool allows to produce alignment matrices with the input of the user
- Alignment matrices are correctly applied (cross-checked with CbmRoot)
- Tool can help to produce an alignment for next mCBM beamtime faster than in the past
- Alignment quality evaluation is in progress
    - Outlook: Sergey will provide a fitting routine for the L1 tracks in the next days
    - Then we will have a common base for the reference tracks to work with

Backup

Alignment Strategy (Draft)

- For each STS module from Unit 0
  - Set this module as solo module for this unit
  - For each STS module from Unit 1
    - Set this module as solo module for this unit
    - Adjust translation parameters such that histograms match with helping lines
- If all modules of a station show e.g. the same non-zero z-shift, reset it to 0 and try to compensate it by moving the TRD correlation module in z

- Method has currently to be done by hand
  - Automation possible by eg. providing a script interface

Performance & Optimization Possibilities

- Computation of Histograms on GPU in 10-50 ms (NVIDIA GeForce GTX 1080, 2560 cores @ 1733 MHz, no performance optimization done yet)
- OpenCL work group size set automatically, maybe performance can be higher if set manually
- Rotation matrices could get pre-computed per module on CPU
  - Currently rotation is done around every axis separately and separately every processed event
  - Lots of computation time could be saved here
- Reference hits could be selected more strict, by e.g. applying already a fit in the FairTask and reject very incompatible hits
  - Reduction of AlignEvent structure size possible, therefore more statistics possible

Scalability

- Since parallelization happens at the level of events, and each event is processed independently, the problem is highly scalable
    - Currently only up to 20 different module IDs are supported, but this is extendable
- Different event sets could be loaded into different GPUs
    - Resulting histograms are then merged in the end
    - No limitation in event count if you have enough GPUs
    - GPUs must not necessarily be in same machine, multiple machines could be connected as slaves to a master server which distributes the alignment matrices to apply and collects & merges the histograms in the end