The Alignment and Calibration of the Belle-II Central Drift Chamber with first cosmic data

Egor Sedov, Novosibirsk state university, Russia

September 2, 2016

Abstract

This document reports on one of the alignment and calibration procedures of the Belle-II Central Drift Chamber. The study is performed on cosmic data collected with the CDC. General Broken Lines algorithm and Millepede II are used as the main instruments in this procedure. ROOT and Shell scripts have been implemented for experimental data processing and analysis of the distributions. Corrections of Central Drift Chamber parameters were found. Distributions of resolution for each layer are fitted with theoretical function. The best resolution is about 120 μm for outer layers and 140 μm for inner layers.

Contents

1 Introduction

Belle II is an experiment under construction in Tsukuba, Japan, and it will continue and extend the physics program of Belle experiment. By November 2009, the Belle detector [1], operating at the asymmetric electron positron collider KEKB [2], had accumulated a data sample with an integrated luminosity of 1000 fb^{-1}. With the much larger data sample that will become available at SuperKEKB, a new panorama of measurements in heavy flavor physics will be possible. These studies will provide an important and unique source of information on the details of new physics processes that are expected to be uncovered at hadron colliders in the coming years. Therefore, tracking measurements with very high precision are important. It is achieved using several detectors, one of them is a central drift chamber.

So alignment and calibration of the central drift chamber are important procedure. For this purpose, set of Shell and ROOT scripts, which is described below,are used. This scripts use General Broken Lines (GBL) algorythm and Millepede II. In this study, cosmic data are used to find resolution and accuracy of CDC.

2 The Belle II Central Drift Chamber (CDC)

In the Belle II detector, the central drift chamber (CDC) [3], which is illustrated in Figure 1, plays three important roles. First, it reconstructs charged tracks and measures their momenta precisely. Second, it provides particle identification information using measurements of energy loss within its gas volume. Low momentum tracks, which do not reach the particle identification device, can be identified using the CDC alone. Finally, it provides efficient and reliable trigger signals for charged particles.

2.1 Construction

It is designed as concentric cylinders with the inner radius of 160 mm and the outer radius of 1130 mm. The length of CDC is 2.4 m. In CDC, there are 14 336 sense wires in 56 layers. The diameter of wires is 30 μm and the gas mixture is He-C_2H_6. CDC is divided into layers and super-layers. There are six layers in each super-layer to make track segment finding easier. The innermost super-layer has two additional layers that contain active guard wires. Each layer is divided into cells with wire in the center.

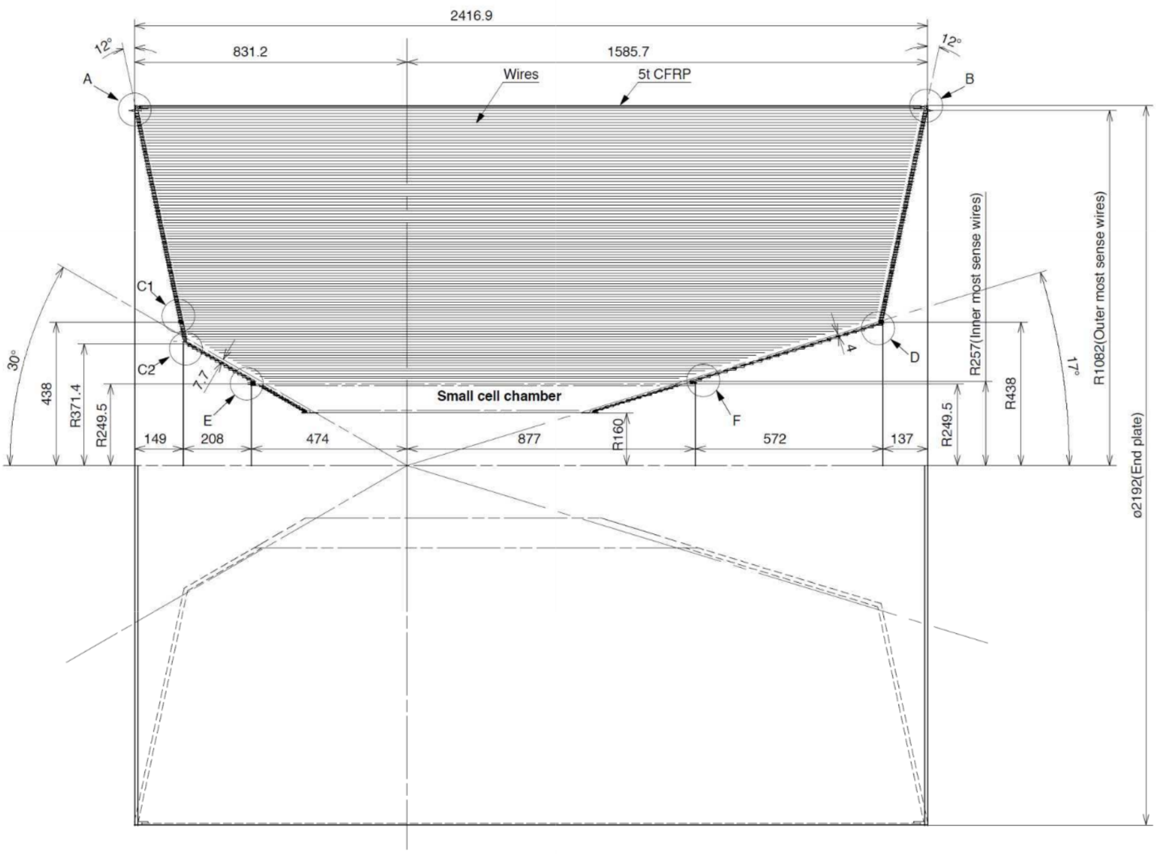


Figure 1: Central drift chamber (CDC).

The radial cell size is 10mm for the innermost super-layer and 18.2mm for the other super-layers.

2.2 Drift time measurement

The x–t (time-to-distance) relation is extracted as polynomial function or linear function, depending on the drift time region. Initially, the x-t relation is taken from the GarField simulation, which is used to find out initial correction for drift velocity.

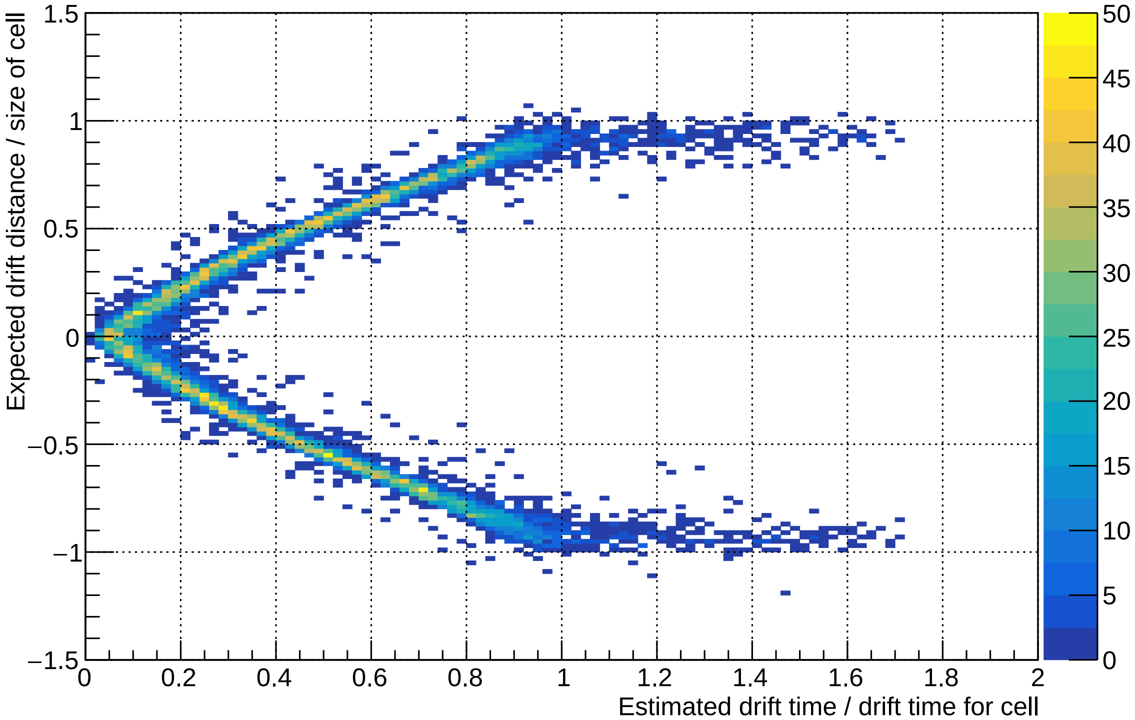


Figure 2: Typical normalized x-t relation for super-layers.

3 Theory

3.1 GBL

In HEP experiments the description of the trajectory of a charged particle is obtained from a fit to measurements in tracking detectors. The parametrization of the trajectory has to account for bending in the magnetic field, energy loss and multiple scattering in the detector material. General broken lines define a track model with proper description of multiple scattering leading to linear equations with a special structure of the corresponding matrix allowing for a fast solution with the computing time depending linearly on the number of measurements. The calculation of the full covariance matrix along the trajectory enables the application to track based alignment and calibration of large detectors with global methods. General Broken Lines algorithm is described in [4].
At the first stage, the possible parts of the track are being found, based on information received from the detector (Figure 3). After that, possible parts combine into two parts

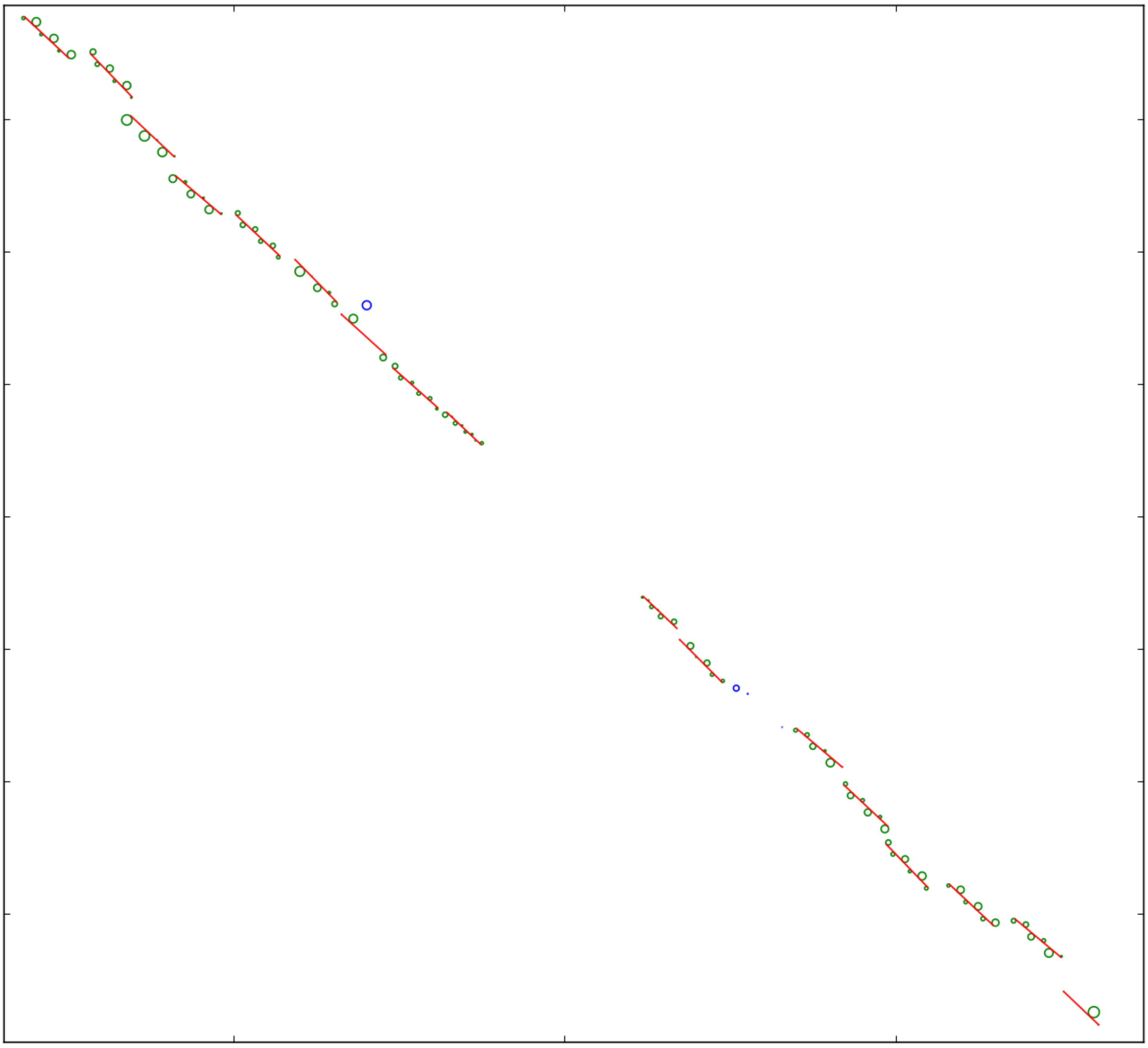


Figure 3: First stage of track fitting algorithm (Possible parts of the particle track).

of the whole track. There are a lead plate and scintillator in the center of the drift chamber, so it is necessary to take into account the scattering of particles. Finally, two parts combine into whole track, which is shown in Figure 4. Track parameters are calculated and written to the binary file.

3.2 Millepede II

The straightforward application of the least squares method for the simultaneous fit of e.g. 1000 global (alignment) parameters and of e.g. 1 Million tracks, each parametrized

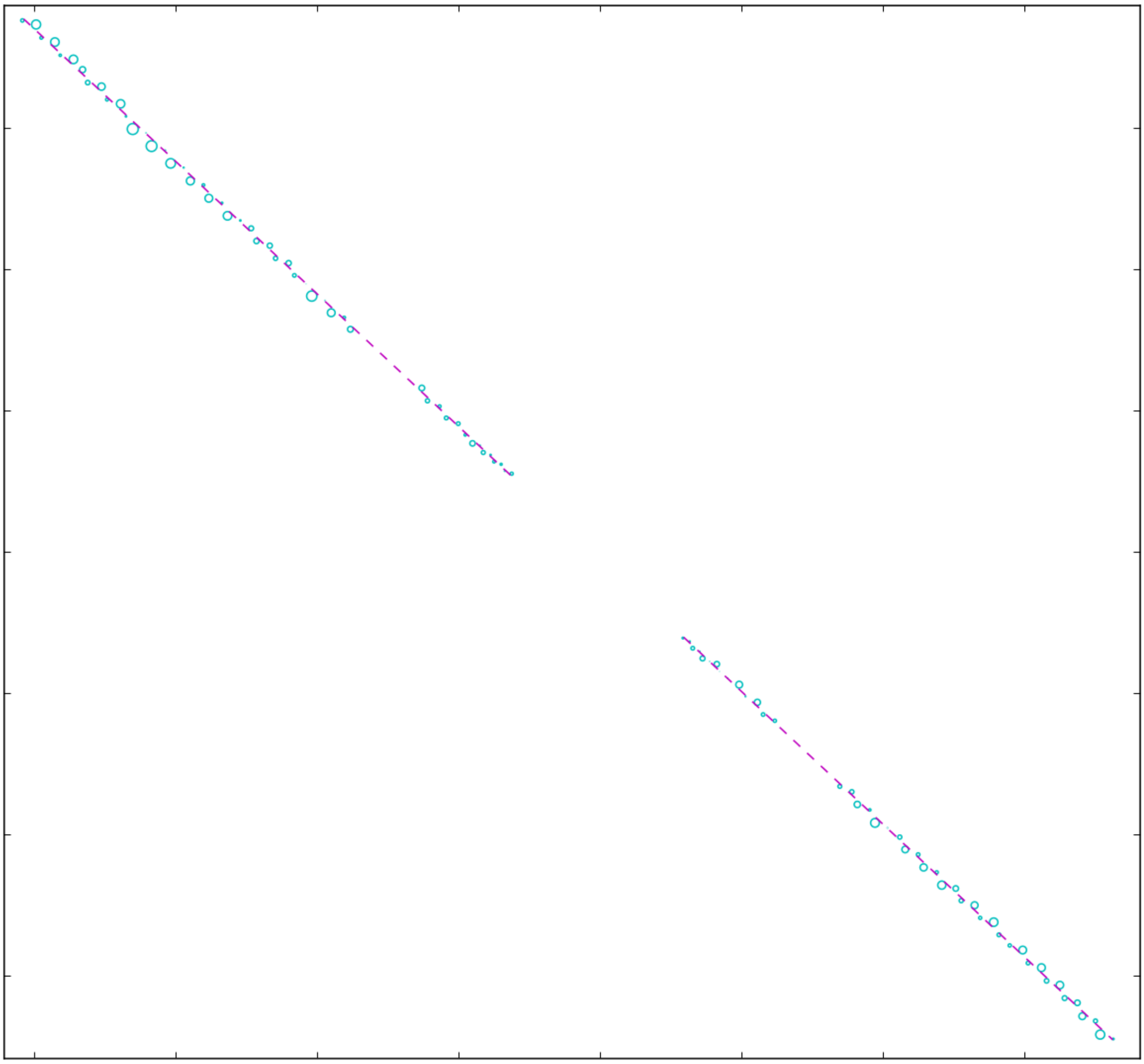


Figure 4: Third stage of track fitting algorithm (Fitting of the whole track).

with 5 local (track) parameters, would result in normal equations with a 5 001 000 x 5 001 000 matrix. Present day computers are not able to solve such large systems. However the large matrix has a special structure. In the Millepede Principle a partitioning of submatrices, based on this structure, is used to include all information from all local (track) fits in the 1 000 x 1 000 matrix of global parameters. The Millepede Principle allows to solve a huge problem in one step, without iterations (unless there are other reasons to perform iterations). A second program version, Millepede II, has been developed since 2005, which allows to have more than 5 000 parameters; it has already been used for up to 50 000 (CMS — 200 000) parameters. It includes several different methods for the solution of large systems of equations. To find more information about Millepede II, read [5].

3.3 Corrections

It is necessary to use certain corrections for each layer and super-layer for a more precise track fitting. In this study, the following corrections are used: $\mathbf{T}_0$ correction includes all time-dependence parameters: time of flight, propagation time, offset time. **Drift velocity** correction shows the adjustment on the real rate of ionization propagation to the wire. **Inhomogeneity** corrections contain the coefficients of the second and third order Legendre polynomials which describe change of drift velocity. **Corner** correction includes relative changing of drift velocity in the cell corner for each layer. **Time-walk** correction is due to the finite rise time of the analog pulse reaching the threshold relative to a reference time. **Alignment** correction depends on the spatial arrangement of the wire and the angle between track and wire.

3.4 Resolution

Resolution of each layer of CDC depends on drift distance. This distribution can be fitted by the following functions:

$$\sigma(x) = [0] \cdot \sin \arctan \frac{[1]}{x} + [2] \cdot \sqrt{x} + [3] + [4] \cdot e^{[5]\cdot(x-[6])^2}, \quad (1)$$

$$\sigma(x) = \sqrt{\frac{[0]}{x^2 + [1]} + [2] \cdot x + [3] + [4] \cdot e^{[5]\cdot(x-[6])^2}}. \quad (2)$$

The second function is preferable, because it describes “real” situation (the square root of the sum of squared errors). The first term describes primary ion statistics, the second — longitudinal diffusion of drift electron, the third is responsible for fluctuations due to electronics and the last describes distortion of electric field.
But it is possible to use first function in some cases to find possible parametrisation and corrections.

4 Manual

4.1 Overview

The following procedure describes the process of finding alignment and calibration of CDC. It consist of few steps. The first step is obtain raw data from detector. In this work, cosmic data from Belle II CDC was provided by Oliver Frost. Data samples with run numbers 784, 787, 788, 803 is used. Special scripts convert data to text file which is used as input for track finding algorithm.

The second step is track finding and fitting, as described in **3.1**. GBL algorithm takes as input initial corrections, which are described above, and raw data from detector. Initial corrections are provided by GarField simulation. The result of the GBL algorithm is binary file, which contains all information about tracks parameters, and ROOT Tree with measured values.

The third step is using binary file to finding new parameters corrections. Millepede II takes as input binary file and solve the linear equations system. Output is file with new parameters, which can be used as input for new iteration of track finding and fitting.

4.2 Implementation

The result of this work is set of scripts which produce connections of different algorithm steps. The main script is responsible for the main loop. The sequence of actions is as follows:

1. Run GBL algorithm with corrections
2. Run Millepede II with the input data, which is the GBL output binary file
3. Calculate new corrections from result of Millepede II
4. Go to the first step with new corrections or go to the next step
5. Find resolution per each layer and super layer for different drift distance
6. Fit resolution distribution with the theoretical function and get parameters
7. Go to the first step and use new parameters or finish

The general script is called "full_script.sh" and it provides all steps. It takes as input data from detector and initial corrections. To run general script you should type next in command line:
`./full_script.sh -KEY PARAMETER`
The list of required parameters with key:

`-n NUMBER` — Unique number for each data set.

`-i FILE` — Input data file in text format.

`-d DIR` — Directory which contains initial parameters.

`-o DIR` — Output directory.

`-s FILE` — Steer file for Millepede II.

The list of optional parameters:

`-h` — Print help.

`-l NUM` — Number of main loops. Default value is 2.

`-t NUM` — Number of events which program will use from data sets. Default value is 50000.

`-m KEY` — Which variable program will use to calculate final resolution. Default value is 1 — dmeas, 0 — distxt, 2 — dexpected.

`-p` — Key for printing output files.

`-f KEY` — Which fitting function will be used to fit final distribution. Default value is 0 — (1), 1 — (2).

`-r NUM` — Range of plotting. Default value is 0 — plot data from 0 to size of cell for layer, 1 — from one cell size to two cell sizes, 2 — from two cell sizes to three cell sizes.

`-c NUM` — Continue from defined loop number. Number should be less than total number of main loops (option `-l NUM`) and all previous corrections should exist.

Example of running script:

```
./full_script.sh -d /home/input/ -o /home/output/ -s /home/steer.txt -n 123
-i /home/input/data123.txt
```

You can also use several input data files:

```
./full_script.sh -d input/ -o output/ -s steer.txt -n 123 -i input/data123.txt
-n 234 -i input/data234.txt -n 345 -i input/data345.txt
```

Result of running this script is the ROOT Trees with values of resolution and parameters of fit function per each layer. List of variables which are available in the final ROOT Tree:

- `lyrnum` — Layer number. It is calculated as "super layer number"·10 + "layer number"
- `slyrnum` — Super-layer number
- `ndistxt` — Coarse drift distance estimation
- `ndmeas` — Measured drift distance divided by the size of cell
- `ndexpected` — Expected drift distance divided by the size of cell
- `cellsize` — The size of cell
- `mean, mean_err` — The centrality of the Gaussian function with error
- `sigma, sigma_err` — Sigma of the Gaussian function with error
- `constant, constant_err` — The constant, which multiply Gaussian function with error
- `chi2` — Value of chi-square for fitting
- `slmean ... slchi2` — Same values for superlayers
- `par[0] ... par[6], err[0] ... err[6]` — Calculated parameters of fitting function for resolution distribution with errors.

These values can be used to precise estimation of resolution of the Central Drift Chamber or calculating new corrections.

General script consist of subscripts which you can also use:

`b2crec.py` — This python script is responsible for track finding and fitting with GBL algorithm.

`fit_data.sh` — Shell script, which provides analyzing of the resolution per each layer and fitting of the resolution distribution.

`change_start.sh` — This script calculates new corrections from Millepede II result file.

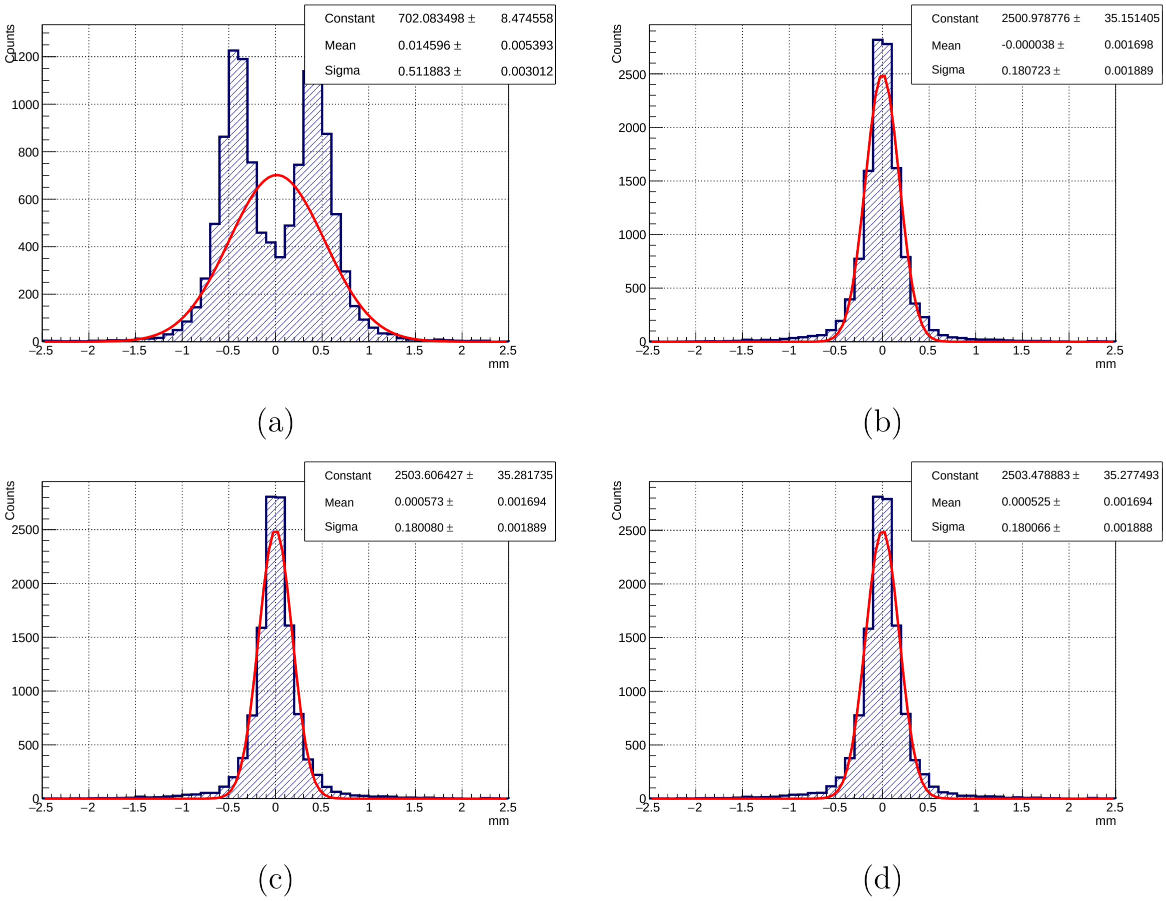
`change_align.sh` — This script calculates new alignment corrections.

5 Results

Distribution of difference between measured and expected drift distances describes resolution of central drift chamber. Figure 5 shows this distribution after each iteration. Distributions are fitted by Gaussian function, σ describes resolution in mm. At the first step, T_0 correction is equal to zero, so two peaks are appeared in final distribution. After second and third iteration, resolution changed very little.
Figure 6 shows final distribution of the resolution for one of layers. At the beginning of the cell, there is worse resolution because there is inaccuracy of track fitting near the wire. There is a rise in resolution and then worsening at the cell border, it is typical for almost all layers. Figures 7, 8, 9, 10 show the distribution of the resolution for middle layer in each super-layer.

6 Conclusion

The drift chamber alignment and calibration algorithm has been implemented. It is based on the General Broken Lines algorithm and Millepede II, which are used to track fitting and solving linear equation systems of drift chamber parameters. Shell scripts have been implemented to connect all parts. ROOT scripts are used to the analysis of the experimental data distributions. Difference between expected and measured drift distance is fitted by Gaussian function, distributions of resolution for each layer are fitted with theoretical function (2). The best resolution is about 120 μm for layer number 2 for super-layer 6 and 140 μm for inner layers.



(a) (b) (c) (d)

Figure 5: Difference between measured and expected drift distances for one layer after (a) first, (b) second, (c) third, (d) forth iteration. Two peaks in (a) are appeared due to absence of T_0 correction.

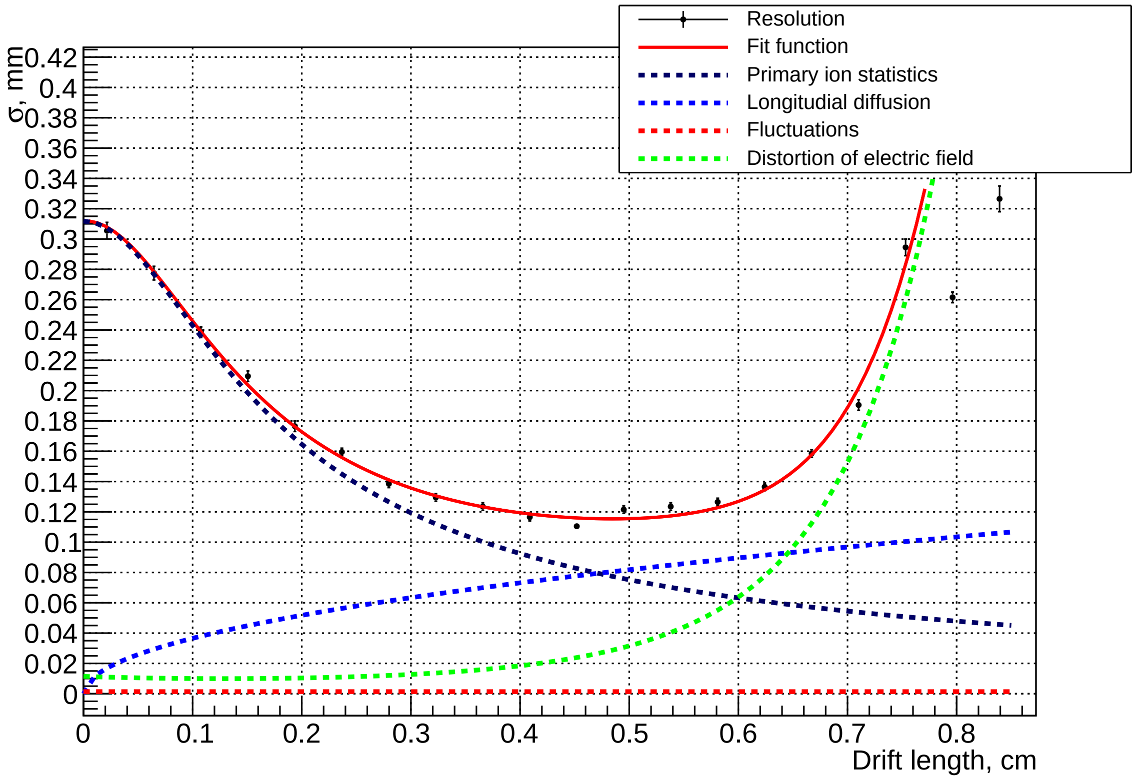
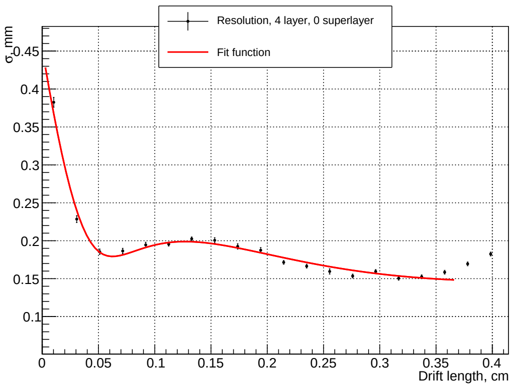
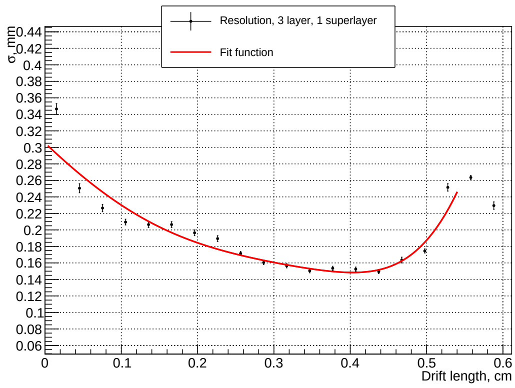


Figure 6: Final distribution of the resolution for layer number 3 in super-layer 7.

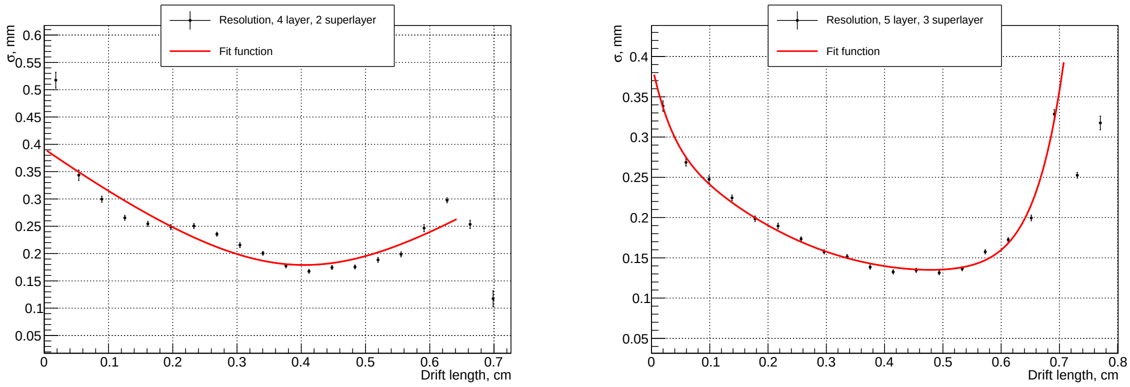


(a)



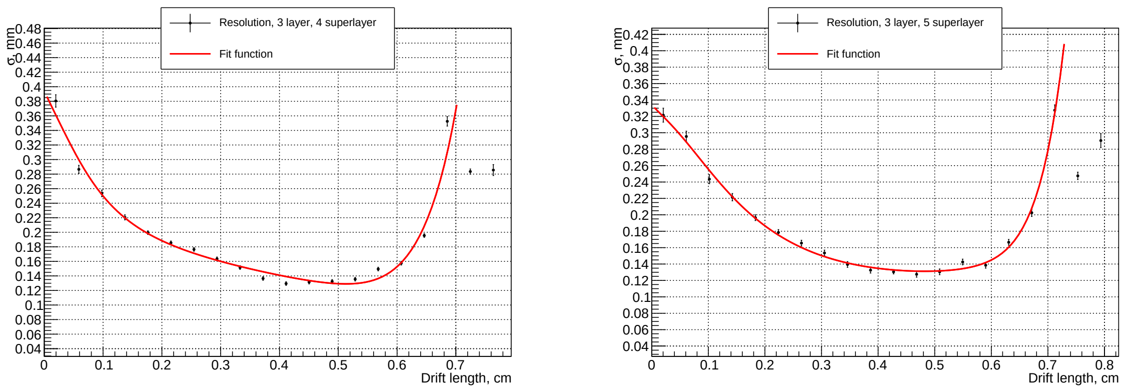
(b)

Figure 7: Final distribution of the resolution for central layer in (a) super-layer 0, (b) super-layer 1.



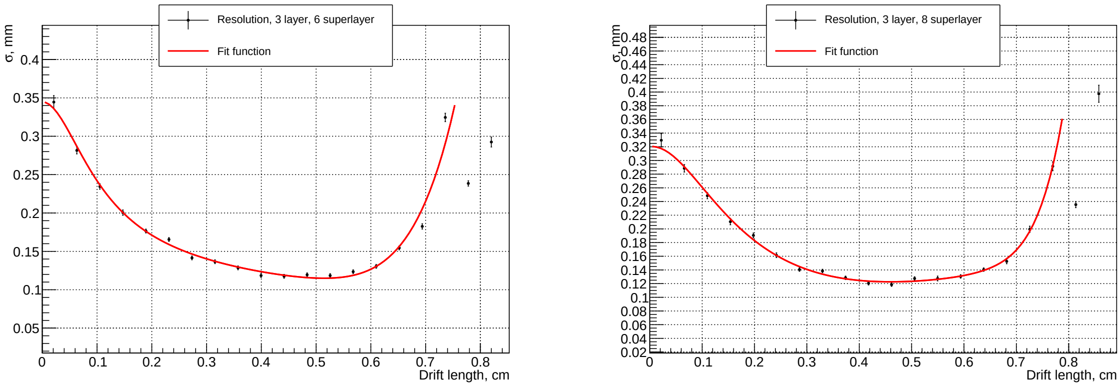
(a) (b)

Figure 8: Final distribution of the resolution for central layer in (a) super-layer 2, (b) super-layer 3.



(a) (b)

Figure 9: Final distribution of the resolution for central layer in (a) super-layer 4, (b) super-layer 5.



(a) (b)

Figure 10: Final distribution of the resolution for central layer in (a) super-layer 6, (b) super-layer 8.

References

[1] *Belle, A. Abashian et al.*, Nucl. Instrum. Meth. A479, 117 (2002), 10.1016/S0168-9002(01)02013-7.

[2] *S. Kurokawa et al.*, Nucl. Instrum. Meth. A499, 1 (2003), 10.1016/S0168-9002(02)01771-0.

[3] *T. Abe et al.*, Belle II Technical Design Report.

[4] *Claus Kleinwort*, General Broken Lines as advanced track model.

[5] A program description for Millepede II and the code is available via `http://www.desy.de/~kleinwrt/MP2/doc/refman.pdf`.