



Research Internship Report

presented by

Jean Declerck

Internship from 01/07/2026 until 25/08/2026 in :

Laboratoire de Physique des Hautes Énergies (EPFL), CERN

Subject :

**DUBT/DSBT Test Beam at CERN: characterisation of
SHiP's Upstream Background Tagger**

Director of the laboratory: **Prof. Lesya Shchutska**

Supervisors: **Ettore Zaffaroni, Valerii Kholoimov, Vasilisa Guiliaeva**

In the scope of: *Laidlaw Scholars Leadership and Research Programme*



Abstract

The SHiP (Search for Hidden Particles) experiment, to be hosted at CERN’s High-Intensity ECN3 Facility, is designed to search for Feebly Interacting Particles (FIPs) predicted by several extensions of the Standard Model. As a zero-background experiment, SHiP relies on dedicated veto systems, including the Upstream Background Tagger (UBT), to reject residual charged particles entering the decay volume. This report presents the characterization of a prototype UBT tracking system, based on the EPFL SciFi (Scintillating Fiber) telescope, using data collected during a dedicated testbeam campaign at the CERN Proton Synchrotron.

A complete offline reconstruction chain, implemented in a software developed by LPHE, was used during the internship, comprising clustering, straight-line track fitting and geometric alignment. I wrote the code to measure the occupancy and implement a tag-and-probe efficiency measurement. Data collected under two beam configurations, a low-intensity 5 GeV muon beam and a high-intensity 15 GeV pion beam, were compared to check both the intrinsic performance of the telescope and its behaviour under realistic occupancy conditions. The analysis identified two distinct detector features: a fixed hot channel, present across all runs, and a periodic pattern of efficiency dips spaced at approximately 15–16 mm, attributed to the physical segmentation of the SiPM readout devices. A beam-dependent degradation was also observed on one sensor under the pion beam, consistent with hadronic interactions. Despite successive methodological improvements, the measured efficiency did not fully reach the expected $\sim 99\%$ efficiency on every sensor, motivating further investigation as this validation effort continues ahead of SHiP’s Technical Design Report submission.

Contents

Abstract	2
1 Theory introduction	4
1.1 First steps in particle physics: the Standard Model	4
1.2 Beyond the Standard Model	5
1.3 Dark matter as a concept	6
2 The SHiP experiment	8
2.1 Presentation, motivations and goals	8
2.2 The Scattering Neutrino Detector (SND)	10
2.3 The Hidden Sector Decay Spectrometer (HSDS)	11
2.4 The background issue: UBT and SBT	13
3 Core of my internship: the DUBT/DSBT testbeam at CERN	14
3.1 Presentation and goals of the testbeam	14
3.2 Methods and goals of the data analysis	15
3.2.1 Offline reconstruction chain	15
3.2.2 Occupancy analysis	15
3.2.3 Tag-and-probe efficiency method	15
4 Results	17
4.1 Event display	17
4.2 Occupancy plot	17
4.3 Efficiency	19
5 Discussion	21
5.1 Improving the efficiency	21
5.2 Effect of the beam type (muons vs pions)	24
5.3 Periodic efficiency dips	25
6 Conclusion	26
7 Next Steps	27
Glossary	28
Appendix	29

Theory introduction

1.1 First steps in particle physics: the Standard Model

The idea that matter is made of indivisible atoms dates back to Ancient Greek philosophy. Experiments at the end of the 19th and over the 20th century later revealed that atoms are themselves made of a nucleus (protons and neutrons) surrounded by electrons, and that protons and neutrons are built from smaller particles called quarks.

Our Universe is built from twelve fundamental particles: six quarks and six leptons (including the electron and the neutrino), organized into three "generations" that are identical except for mass (see Figure 1.1). Only the lightest, first-generation particles make up everyday matter, as heavier ones decay rapidly.

These particles interact through four fundamental interactions: gravity, electromagnetism, the strong nuclear interaction (binding quarks into protons and neutrons), and the weak nuclear interactions (responsible for radioactive decay). A fifth interaction, mediated by the Higgs boson, gives mass to the other particles. Each force has a corresponding carrier particle: the photon, gluons, and the W/Z bosons.

The theory combining all these particles and forces (excluding gravity) is the Standard Model of particle physics. It was formulated in the early 1970s, and the most recent addition to it was the discovery of the Higgs boson at CERN in 2012. It remains one of the most rigorously tested theories in science. [12] [14]

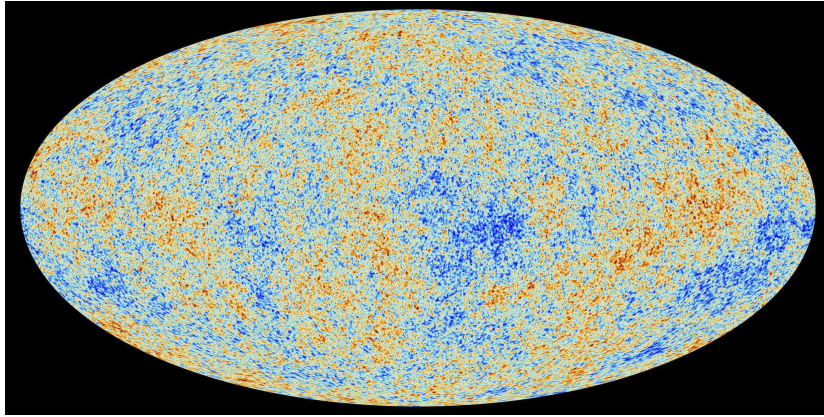


Fig. 1.3 The Cosmic Microwave Background (CMB).

The SHiP experiment

2.1 Presentation, motivations and goals

SHiP (Search for Hidden Particles) is a new experiment in preparation at the SPS (see Glossary, chapter 7) to search for “hidden” particles as predicted by models of the Hidden Sector, which could explain dark matter (hence the theory introduction on dark matter) and neutrino oscillations for example. In a more pictorial way, SHiP will go beyond the Standard Model, "explore the confines of the ocean of particle physics and try to reach the unreachable seas of physics." [3]

SHiP is designed to detect any kind of particle that could explain these phenomena: Heavy Neutral Leptons (HNL), dark photons and different types of Light Dark Matter (LDM) [13].

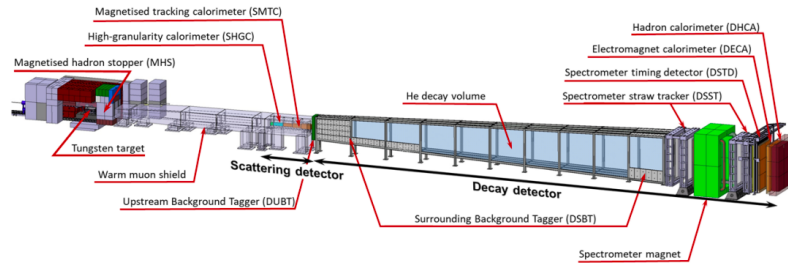


Fig. 2.1 Detailed representation of SHiP [13]

These particles, which interact very weakly with the SM particles, are called Feebly Interacting Particles (FIPs). Thus, SHiP will search for FIPs through scattering and decay signatures: these particles are predicted to interact with SM particles, which are detectable. We could then conclude if the detected SM particles correspond to the model, through a method called vertexing (more detailed in the part on the detectors). For a generic FIP model in the mass range (100 MeV) - few-GeV, SHiP is expected to provide world-leading or record-leading sensitivity.

The second goal of SHiP is to make progress in neutrino physics. Indeed, SHiP will benefit from an enriched sample of tau neutrinos and electron neutrinos (compared to the typically dominant muon neutrino) enabling a rich study of their interaction dynamics. So, one of SHiP’s key goals is the first detection of tau anti-neutrinos. [5]

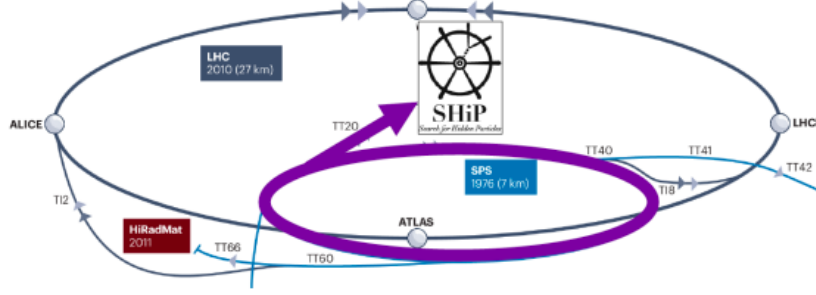


Fig. 2.2 Map of the CERN particle accelerator complex: SHiP will use the SPS. [3]

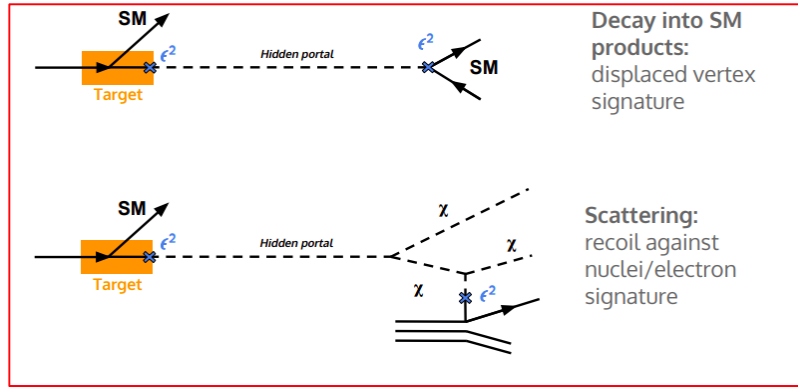


Fig. 2.3 Beyond the Standard Model: how to look for FIPs? [3]

The experiment will be hosted at the High-Intensity ECN3 Facility (Experimental Cavern North 3, still under Technical Design Report), in the North Area of CERN. SHiP will use the high intensity proton beams (400 GeV) from SPS, in order to search for rare events. This beam is sent onto a dense target (the "beam dump"), which absorbs almost all conventional particles produced in the collisions. A muon shield then stops the remaining muons, leaving mainly neutrinos and, potentially, new feebly interacting particles to travel further downstream, where SHiP's detectors are placed to catch them.

The total number of protons directed by SPS onto the target per year (PoT, protons-on-target) is evaluated at 4×10^{19} , so for 15 years of data taking, it will be 6×10^{20} PoT. It will produce an unprecedented sample of tau neutrinos and antineutrinos. Plus, to give enough space for FIPs to decay (and thus potentially detect them), an important decay volume is needed. Finally, to observe faint events, the presence of high-precision scattering detectors and decay spectrometers are mandatory. [5] [13]

The SHiP device is made up of many elements, and I will not detail each one in this report. However, we will focus on the most important: the detectors. There are two detectors: the Scattering Neutrino Detector (SND) and the Hidden Sector Decay Spectrometer (HSDS). [13]

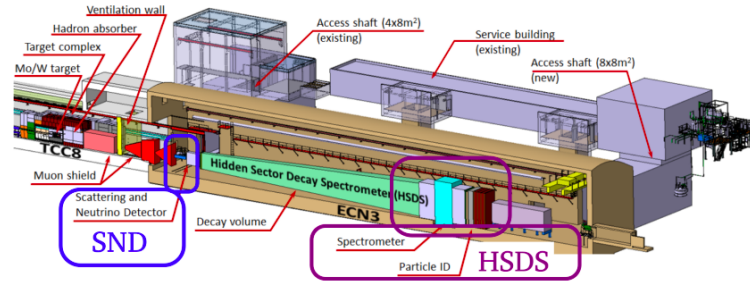


Fig. 2.4 Location of the SND and HSDS in the SHiP installation.[11]

2.2 The Scattering Neutrino Detector (SND)

The Scattering Neutrino Detector (SND) is one of the two main detectors of SHiP. It is situated upstream of the SHiP device and exploits the intense neutrino flux produced in the dump. The detector is embedded within the downstream part of the muon shield (a device used to magnetically deflect the muons out of the acceptance of the spectrometer). It is composed of alternating layers of tungsten and silicon sensors for precise vertex reconstruction in tau neutrino interactions and electromagnetic showers from electron neutrinos. A magnetized tracking calorimeter (MTC, it is a specialized particle detector that measures both the trajectory and the energy of charged particles passing through a magnetic field) follows the silicon tracker. The MTC is optimized for flavour identification of neutrinos, including tau neutrinos, through kinematic reconstruction. (In particle physics, flavour is a characteristic that helps distinguish different kinds of quarks, leptons and neutrinos) [5]

The role of the SND is to detect LDM and to carry out the measurements of neutrinos.

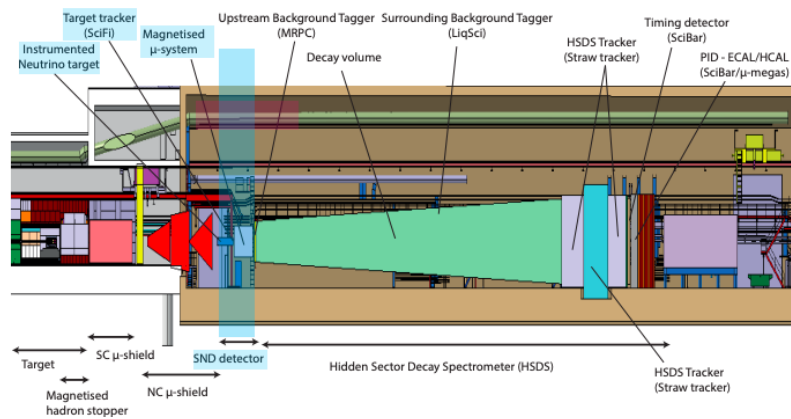


Fig. 2.5 Detailed representation of the SHiP device. The SND is highlighted in blue.[8]

As explained in section 2.1, this detector (and the HSDS) has the objective to detect “beyond Standard Model” particles, by observing a decay giving SM particles and checking if the decayed particles are in adequacy with the model.

	Physics model	Final state
	SUSY neutralino	$\ell^\pm \pi^\mp, \ell^\pm K^\mp, \ell^\pm \rho^\mp, \ell^+ \ell^- \nu$
	Dark photons	$\ell^+ \ell^-, 2\pi, 3\pi, 4\pi, KK, q\bar{q}, D\bar{D}$
	Dark scalars	$\ell\ell, \pi\pi, KK, q\bar{q}, D\bar{D}, GG$
	ALP (fermion coupling)	$\ell^+ \ell^-, 3\pi, \eta\pi\pi, q\bar{q}$
HSDS	ALP (gluon coupling)	$\pi\pi\gamma, 3\pi, \eta\pi\pi, \gamma\gamma$
	HNL	$\ell^+ \ell^- \nu, \pi l, \rho l, \pi^0 \nu, q\bar{q}' l$
	Axino	$\ell^+ \ell^- \nu$
	ALP (photon coupling)	$\gamma\gamma$
	SUSY sgoldstino	$\gamma\gamma, \ell^+ \ell^-, 2\pi, 2K$
	LDM	electron, proton, hadronic shower
SND	$\nu_\tau, \bar{\nu}_\tau$ measurements	$\tau^\pm$
	Neutrino-induced charm production (ν_e, ν_μ, ν_τ)	$D_s^\pm, D^\pm, D^0, \bar{D}^0, \Lambda_c^+, \bar{\Lambda}_c^-$

Fig. 2.6 Table of the predicted physics model and what final state they would decay into. For example, for tau neutrino and antineutrino detection, we should observe positive and negative-charged tau particles. (See 7) for detailed particle symbols) [8]

2.3 The Hidden Sector Decay Spectrometer (HSDS)

The Hidden Sector Decay Spectrometer is the second main detector of SHiP, and it is located downstream of the SHiP device. It is made of two elements: the Hidden Sector Decay Volume and the HSDS Tracker. [8]

Let's talk about the Decay Volume first: it is a 50-meter long pyramid-shaped space filled with helium (He) at atmospheric pressure. Rather than air or a hard vacuum, helium was chosen because it is light enough to minimise unwanted interactions with the neutrino flux and any FIPs travelling through, while remaining far more practical and cost-effective to maintain over such a large volume than a true vacuum [2]. Its dimensions are $1.0 \times 2.7 \text{ m}^2$ at the upstream end and $4 \times 6 \text{ m}^2$ at the downstream end [8]. It acts as the search region for long-lived FIP decays: it is in this volume that the FIPs will decay into SM particles, and these particles will hit the Decay Spectrometer, thus permitting a good reconstruction of the vertex and comparison to the model (cf table of predicted physics). The Decay Volume is also surrounded by background tagger systems designed to veto incoming particles or interactions occurring near the detector walls, in order to avoid stray particles and assuring that the produced particles effectively come from the upstream target.[5] [13]

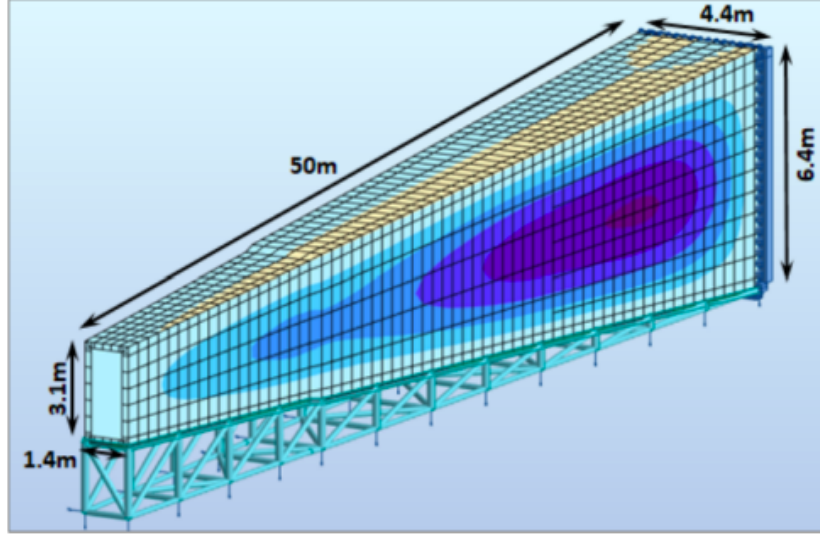


Fig. 2.7 Schematic representation of the Decay Volume. [11]

The second element of the HSDS is the tracker itself. It is made of multiple elements: a spectrometer magnet, a straw tracker, and two detectors (a timing detector and a Particle IDentification detector). The spectrometer magnet has the same aperture as the Decay Volume's downstream end, $4 \times 6 \text{ m}^2$ (indeed the magnet is located at the output of the Decay Volume) and provides a field integral of $0.6\text{--}0.8 \text{ T}\cdot\text{m}$. Its role is to ensure precise momentum reconstruction. The magnetic field geometry and the arrangement of the four straw-based drift tracking stations are carefully optimized to ensure excellent angular resolution within the decay volume. [5]

The straw tracker is made of 4 layers of straw. Its goal is to measure trajectories and momenta of particles, reconstruct decay vertices and impact parameters at proton target. Finally, the timing detector is made of scintillators and SiPMs (Silicon PhotoMultipliers), and its role is to measure coincidence in order to reject combinatorial background. The PID system is designed to discriminate between hadrons and muons, and to ensure good electromagnetic showers reconstructions (e and $\gamma\gamma$) [11]

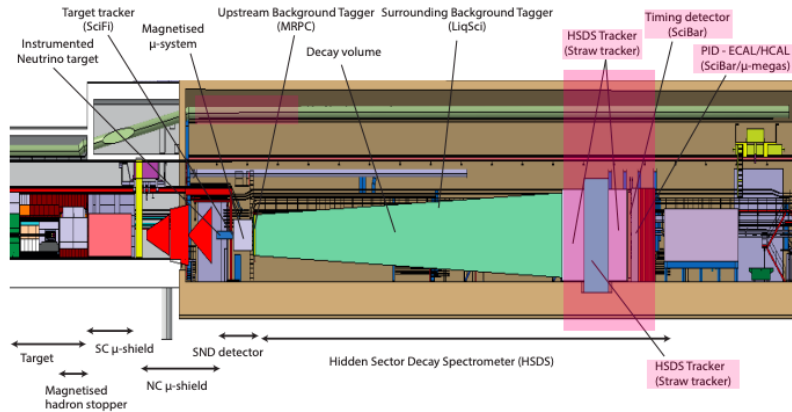


Fig. 2.8 Detailed representation of the SHiP device. The HSDS is highlighted in pink, the decay volume is in green. [8]

2.4 The background issue: UBT and SBT

Throughout this report, the naming conventions UBT and DUBT (as well as SBT and DSBT) are used interchangeably: the latter are the names used in the official SHiP detector schematics and in the naming of the testbeam campaign itself, while the former are used in the general description of the detector system.

The SHiP experiment is a zero-background experiment: this means that to ensure the detection of extremely rare FIPs, the decay volume remains clean from any uncounted events, but also that background events need to be flagged to ensure a good detection. This is the role of UBT and SBT.

The Upstream Background Tagger (UBT) is placed between the muon shield and the decay volume (see figure 2.1). It should provide position and time information on the residual charged particles that enter the decay volume. These charged particles, mainly residual muons, may lead to background in the form of fake vertices formed from accidental combinations of charged tracks in the decay volume. Studies show that the UBT detector should have lateral coverage of up to ± 2 m to be able to detect all the background muons reaching the decay spectrometer. The choice for the UBT consists of a combination of the scintillating tile timing detector and a dedicated tracker based on straw tubes, to ensure an excellent time and spatial resolution and guarantee the required tagging efficiency. [13]

The Surrounding Background Tagger (SBT) is the other part of SHiP's background tagger system. Its main purposes are:

- Tagging inelastic interactions of muons and in particular of neutrinos on helium inside the decay volume by detecting the outgoing particles from the inelastic interaction.
- Tagging of muons entering the decay volume, complementing the UBT background rejection power. [10]

As the SBT is not the system I studied during the testbeam, I will not detail how it is made.

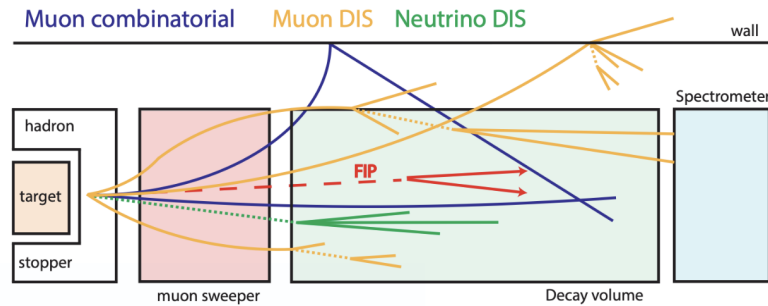


Fig. 2.9 Different interactions with the SHiP device that could disrupt the identification of FIPs: this is why the UBT and the SBT are needed. [8]

Core of my internship: the DUBT/DSBT testbeam at CERN

3.1 Presentation and goals of the testbeam

The DUBT/DSBT testbeam at CERN took place at the Proton Synchrotron, in CERN East Area, on July 7-15. I will only focus on the DUBT part, as I only worked on the Upstream Background Tagger. The goal of the testbeam was to compare the edge efficiency of two types of tiles, one with straight edges, one with a slot, in order to characterise the UBT performance in a real beam environment.

The detector used in this testbeam is the EPFL SciFi (Scintillating Fiber) telescope. It consists of four tracking stations, each composed of two orthogonal fibre mats measuring the X and Y coordinates, with 512 channels per mat and an active acceptance of approximately $13 \times 13 \text{ cm}^2$. Two PhotoMultiplier Tubes (PMT) and a scintillator were used to provide a trigger. [9]

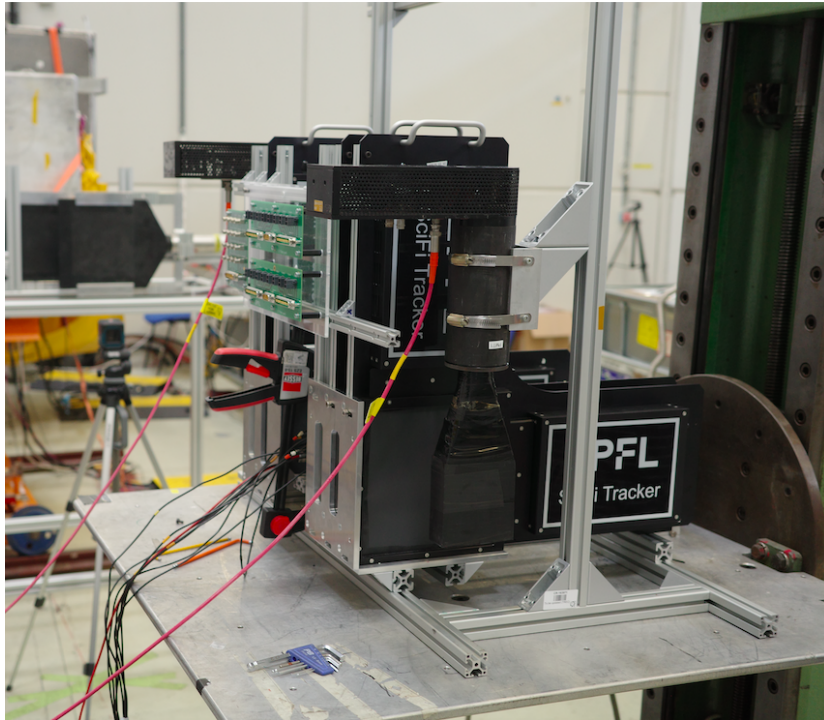


Fig. 3.1 Picture of the testbeam setup at CERN. We used the EPFL SciFi telescope for tracking.

Two beam configurations were used during the testbeam: a low-intensity muon beam at 5 GeV, providing clean single-track events well suited for calibration, and a high-intensity pion beam at 15 GeV, allowing the study of detector performance under denser, more realistic occupancy conditions closer to what the UBT would encounter during SHiP operation, and

most importantly acquire enough events to have a good statistics in the time at our disposal. Data acquisition and reconstruction relied on the CuboDAQ software, developed at LPHE (mostly working in C++, with in control scripts in Python). The full offline reconstruction chain and the efficiency measurement method are described in the following section.

3.2 Methods and goals of the data analysis

3.2.1 Offline reconstruction chain

The raw data recorded during the testbeam consist of individual channel hits, each characterised by a channel number and a timestamp. These hits are processed offline through a four-step reconstruction chain.

1. Hits are grouped into clusters on each sensor plane: hits on adjacent channels within a fixed gap tolerance are merged into a single cluster, representing the estimated crossing point of a particle on that plane.
2. Clusters from the different stations are combined into tracks through a straight-line fit across the telescope; for a track to be built, a minimum number of aligned clusters is required, and a maximum number of tracks per event is imposed to limit combinatorics.
3. A dedicated alignment procedure corrects residual geometrical offsets between planes. Rather than relying solely on mechanically measured z-positions, this step iteratively adjusts the x, y, and t offsets of each sensor by minimising the residuals between the reconstructed tracks and the measured cluster positions, run by run.
4. Finally, the aligned geometry and reconstruction chain are used to compute per-plane efficiencies, described below.

3.2.2 Occupancy analysis

In addition to the efficiency measurement, the spatial distribution of hits and clusters across each sensor was studied through occupancy plots, both as a function of raw channel number and of reconstructed local position (clusters). This analysis was used to characterise the beam profile on each plane, and to identify localised detector anomalies such as malfunctioning channels (hot or dead channels).

3.2.3 Tag-and-probe efficiency method

A naive efficiency estimate (checking whether a cluster is found near the position predicted by a track built using all eight planes, including the one being tested) would be biased: the tested plane could have contributed to its own track fit, artificially inflating the apparent match rate. To avoid this bias, I measured efficiency using an unbiased tag-and-probe method. For each plane under test, a track is reconstructed using only the remaining seven sensors ("tag"), then extrapolated to the position where it should cross the excluded plane ("probe"). This predicted position is then compared to the clusters actually measured on that plane within the same event. If a cluster is found within a fixed matching window around the predicted position, the probe is counted as matched; otherwise, it is counted as missed. The efficiency of a plane is then defined as the fraction of matched probes over the total number of probes: [6]

$$\varepsilon = \frac{N_{\text{matched}}}{N_{\text{total}}} \quad (3.1)$$

The size of the matching window was chosen empirically, by comparing the efficiency computed from real data to the efficiency obtained by an event-mixing control sample — pairing each predicted position with clusters from a randomly chosen, unrelated event. This control sample provides an estimate of the purely combinatorial (random) matching rate as a function of window size; the working point was chosen where the real efficiency curve clearly separates from this combinatorial background, avoiding windows too narrow (missing genuine matches due to the limited precision of a seven-plane track extrapolation) or too wide (admitting random coincidences). Predictions extrapolated outside the active area of the sensor were also excluded from the calculation, as these do not correspond to genuine detection failures.

Results

4.1 Event display

The first step was to display a reconstructed event, and visualise where particles crossed the SciFi telescope's mats. This was done using a dedicated Python script, `evt_display.py`

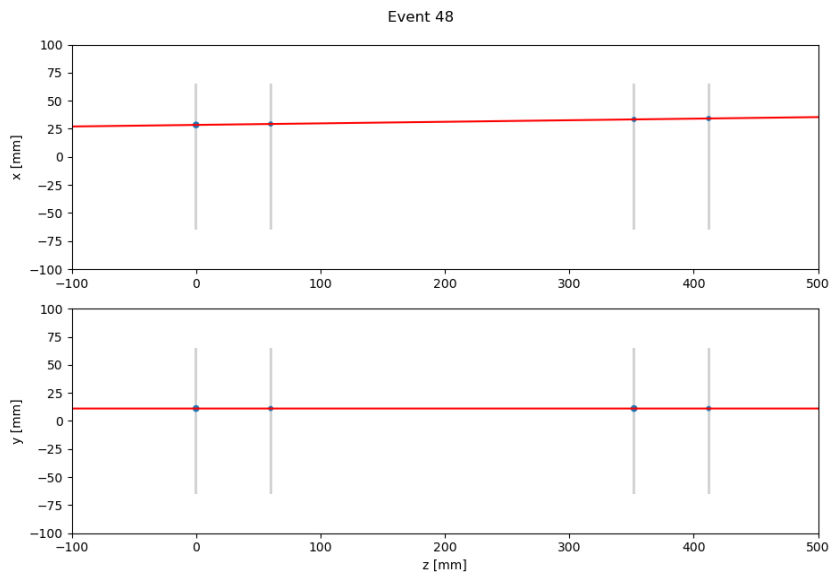


Fig. 4.1 Display of event number 48 of run 110 (muons, 440k events in total!), in the x,y plane. In gray, the 4 planes of the telescope, in blue where the particles hit the planes, in red the reconstructed track.

The event display is simply a visualisation of a reconstructed particle track (there is nothing further to analyse from it directly) but it was an important first step to understand what the reconstruction chain produces, and what subsequent analyses would build upon.

4.2 Occupancy plot

A more informative step is the occupancy plot: the distribution of hits as a function of channel (or, equivalently, of cluster position) on each sensor. This distribution reflects the beam profile: for a wide, uniform beam, occupancy should be approximately flat across the active area of each sensor. I plotted the occupancy for two runs: run 110 (muons, 440k events) and run 137 (pions, 2.6M events), to compare the two beam configurations (5 GeV muons vs. 15 GeV pions).

Run	Particle	Number of events
110	Muons, low intensity	440k
137	Pions, high intensity	2.6M

Tab. 4.1 Runs analysed during the testbeam.

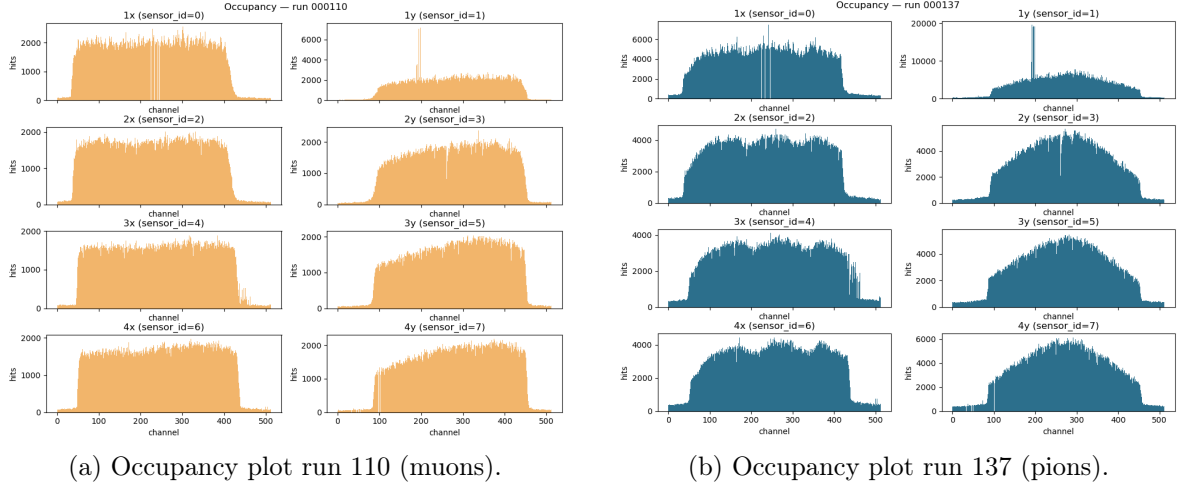


Fig. 4.2 Occupancy plots of runs 110 (muons, orange) and 137 (pions, blue). Both confirm the expected wide-beam profile. A "hot" channel is visible on sensors 1y — this channel was found to be present consistently across all five runs examined, confirming it as a fixed hardware defect rather than a transient or beam-dependent artefact.

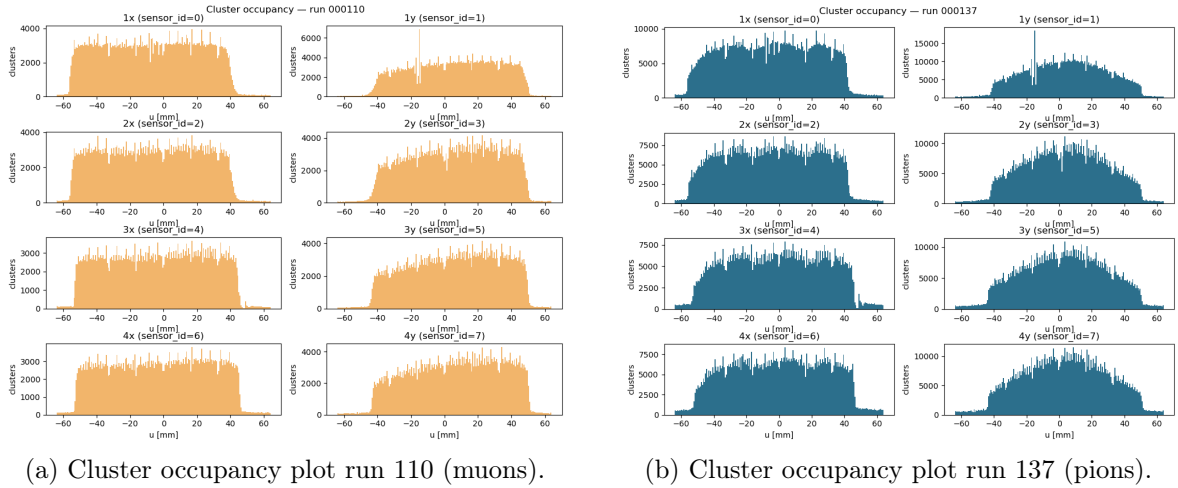


Fig. 4.3 Cluster occupancy, closely following the hit occupancy shown above. A localised dip appears at the position of the hot channel identified on sensors 1x/1y, rather than an excess: the defective channel appears to merge with, or shift, neighbouring clusters during the clustering step, producing a deficit at that position instead of an excess of counts. A separate, periodic pattern of dips, spaced at approximately 15 mm intervals, is also visible across several sensors; this is attributed to the physical construction of the SiPM readout devices, a known feature of the SiPM arrays used in scintillating fibre trackers such as the LHCb SciFi tracker.

4.3 Efficiency

Having characterised the occupancy, the next step of the analysis was to measure the detection efficiency of the telescope. As with the occupancy study, I focus here on run 110 (muons) and run 137 (pions), to compare efficiency under the two beam conditions.

Efficiency was computed using the unbiased tag-and-probe method described in Section 3.2.3. The matching window size was chosen empirically, following the procedure described in the "Methods and goals of the data analysis" section: a window of 5 mm was found to capture the large majority of genuine matches while keeping the combinatorial background low, based on a comparison with an event-mixing control sample. An edge margin of 5 mm was also applied, to exclude predictions falling outside the active area of the sensor from the efficiency calculation. Increasing this margin further, up to 15 mm, was found to have little effect on the overall result (less than 0.1 percentage points), indicating that edge effects are not the dominant source of inefficiency.

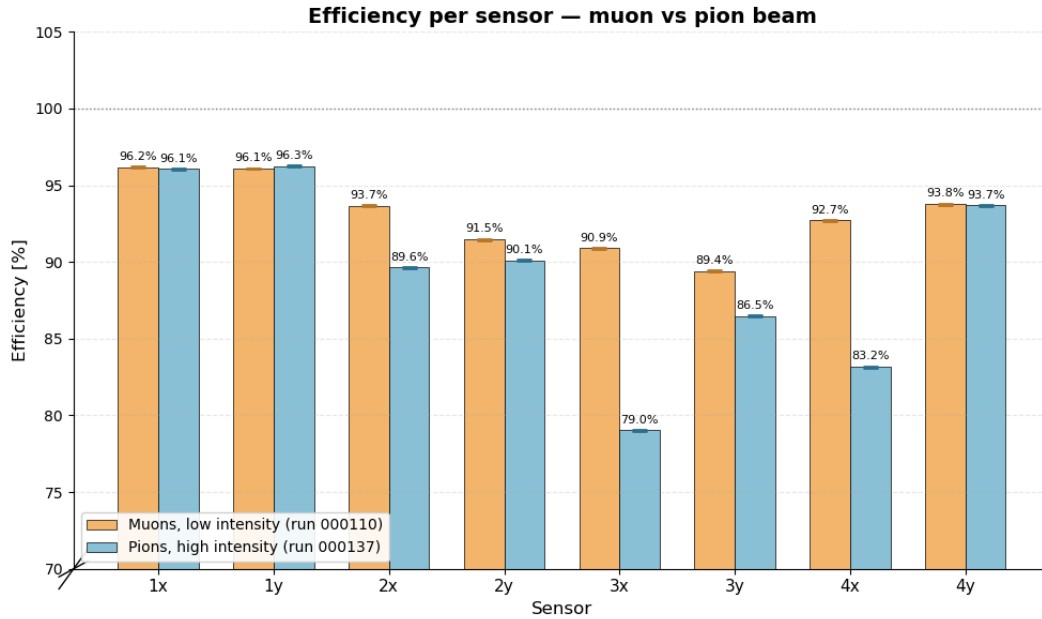


Fig. 4.4 Efficiency per sensor, compared between the muon and pion beams. The telescope is clearly less efficient under the 15 GeV pion beam, particularly on sensor 3x.

To investigate this further, I also plotted efficiency as a function of position across each sensor.

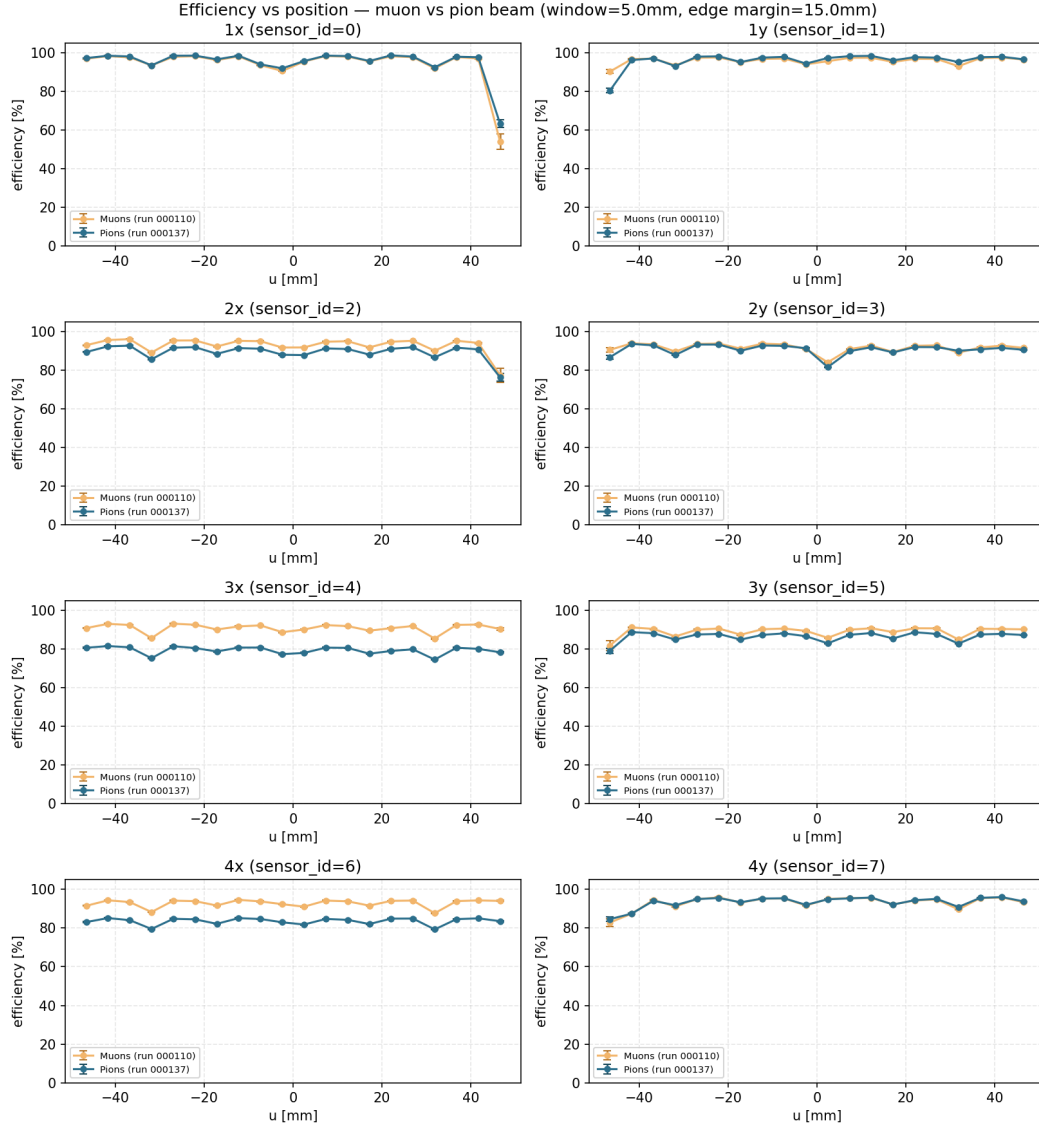


Fig. 4.5 Efficiency as a function of position, compared between the muon and pion beams.

This spatial analysis revealed two distinct effects. First, sensor 3x shows a markedly lower efficiency under the pion beam across essentially its entire active area, consistent with the sensor-averaged result above. The explanation of this phenomenon is not clear for the moment, further studies are necessary.

Independently of the pion effect, several sensors show a periodic pattern of local efficiency dips, recurring at approximately 15 mm intervals across the active area, most clearly visible on sensors 2x and 4x. These dips are statistically significant (typically 4-6 percentage points below neighbouring bins, over samples of 15,000-20,000 events each) and appear at the same positions in both the muon and pion runs. They coincide with the periodic structure already observed in the cluster occupancy, and are attributed to the same cause: the physical construction of the SiPM readout devices. This periodic structure, rather than edge effects, appears to be the main factor keeping the overall efficiency below the expected 99% benchmark for most sensors, alongside the additional pion-specific degradation observed on sensor 3x.

Discussion

5.1 Improving the efficiency

The first results of the efficiency were not satisfactory enough (sometimes 80% for the pion run), the efficiency calculation was reimplemented in a new script, `efficiency_new.py`, replacing the initial manual matching-window approach with a method based on the `match_idx` field computed internally by the reconstruction software. For each sensor under test, only events containing exactly one reconstructed track were retained, to ensure an unambiguous one-to-one correspondence between each track and its corresponding cluster. The efficiency was then simply defined as the ratio between the number of events with a matched cluster (`match_idx` pointing to a valid cluster) and the total number of single-track events, avoiding the need of a manual matching window or an edge margin, as in the earlier approach: this new method is simpler.

Before applying the aligned geometry, the raw signed residual distributions (intercept minus matched cluster position) revealed a clear systematic bias on two sensors: 3y showed a residual centered around -0.35 mm, and 4y around $+0.45$ mm, rather than being centered on zero like the other six sensors. This confirms that these two planes were affected by a residual misalignment not yet corrected by the nominal geometry file. After running the alignment procedure (Section 3.2.1) and using the resulting geometry as input to `efficiency_new.py`, the residual distributions of all eight sensors, including 3y and 4y, became properly centered on zero, with a width comparable to that of the other planes. This directly explains part of the efficiency gain observed after alignment: with a non-aligned geometry, the matching window is effectively searched around the wrong position, so target clusters can fall just outside the window even when the detector itself is working correctly. Once the systematic offset is removed (thanks to the alignment script), the same fixed matching window captures the true signal peak far more efficiently, without requiring any change to the window size itself.

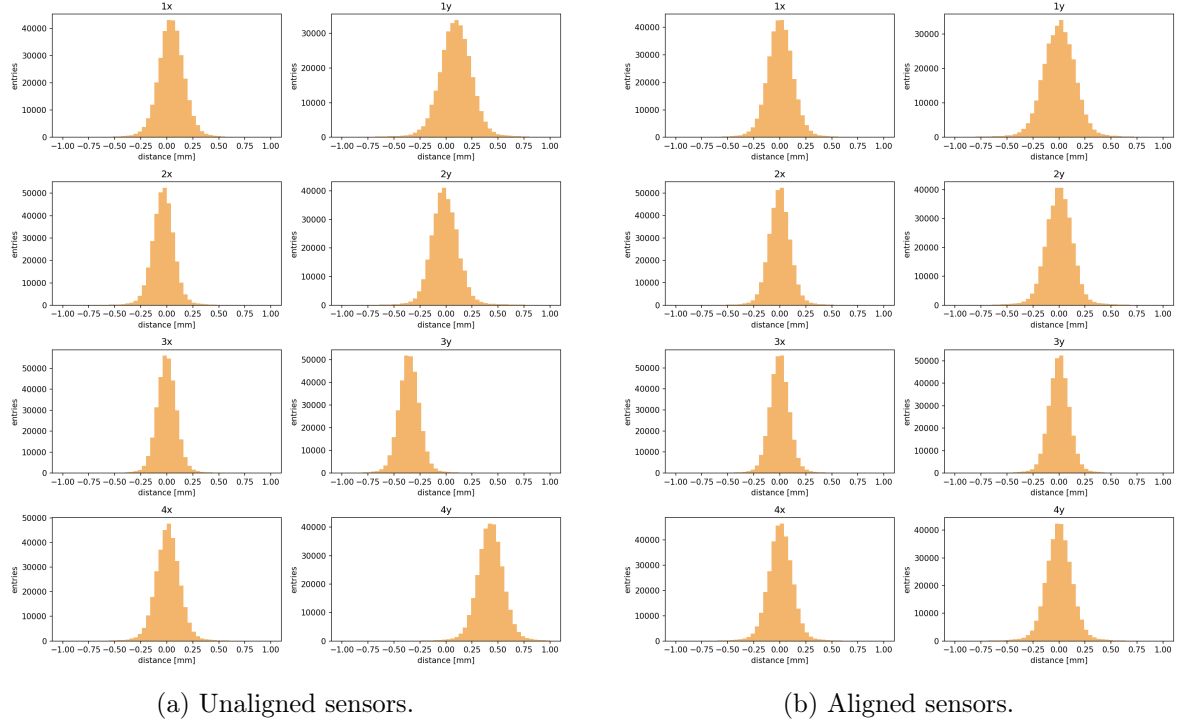


Fig. 5.1 Signed residual distributions (intercept - matched cluster position) for all eight sensors, before (left) and after (right) applying the aligned geometry. Sensors 3y and 4y show a clear systematic offset before alignment (-0.35 mm and $+0.45$ mm respectively), which is corrected after alignment, with all sensors then centred on zero.

A cut on the track quality ($\chi^2/dof < 10$) was also tested in combination with the single-track selection, to check if poorly-fitting tracks lowered the measured efficiency. Comparing the results with and without this cut showed a negligible difference (well below 0.2 percentage points on every sensor), indicating that the internal `match_idx`-based matching is already largely insensitive to track quality, and that this cut is not a significant lever for improving the measured efficiency.

To obtain statistically rigorous uncertainties, the efficiency was computed using Clopper-Pearson confidence [15] rather than a standard binomial distribution (it works in first approach but it not precise enough). This choice is particularly important near the edges of the sensors, where the number of events per position bin is low and the efficiency approaches 100%.

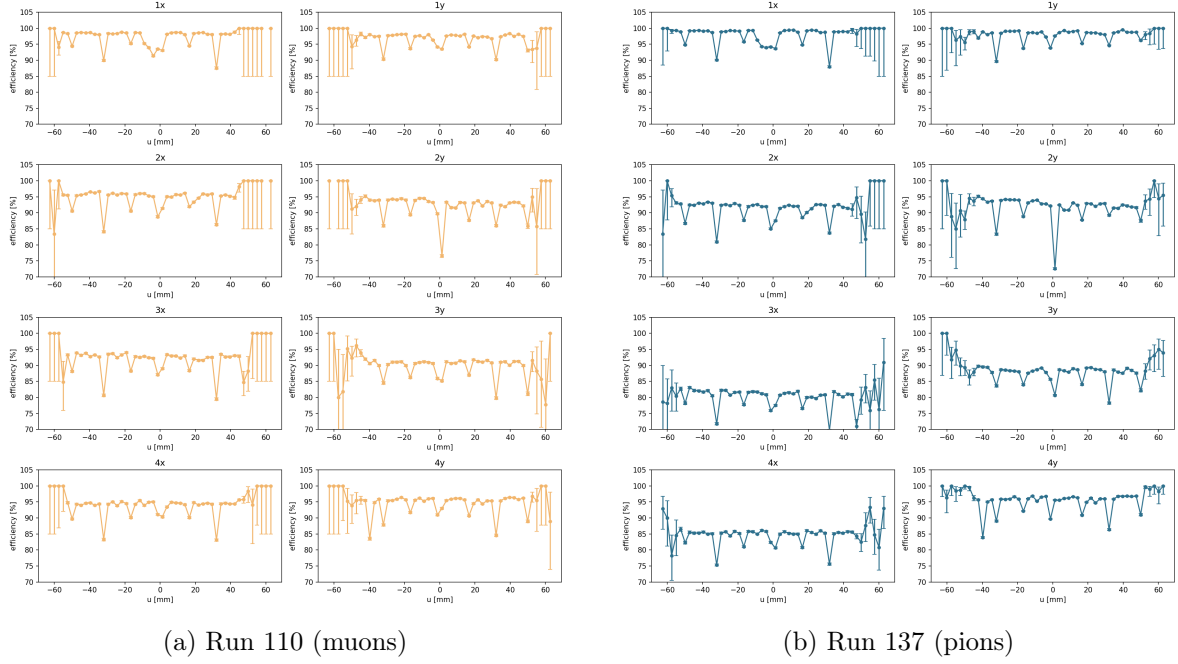


Fig. 5.2 Efficiency as a function of position for run 110 (muons) and run 137 (pions). The periodic dip pattern attributed to the SiPM readout segmentation is present in both runs at consistent positions, while the plateau efficiency is systematically lower under the pion beam, most notably on sensor 3x (75–85%) and 3y (80–90%), consistent with the beam-dependent degradation discussed in Section 5.2

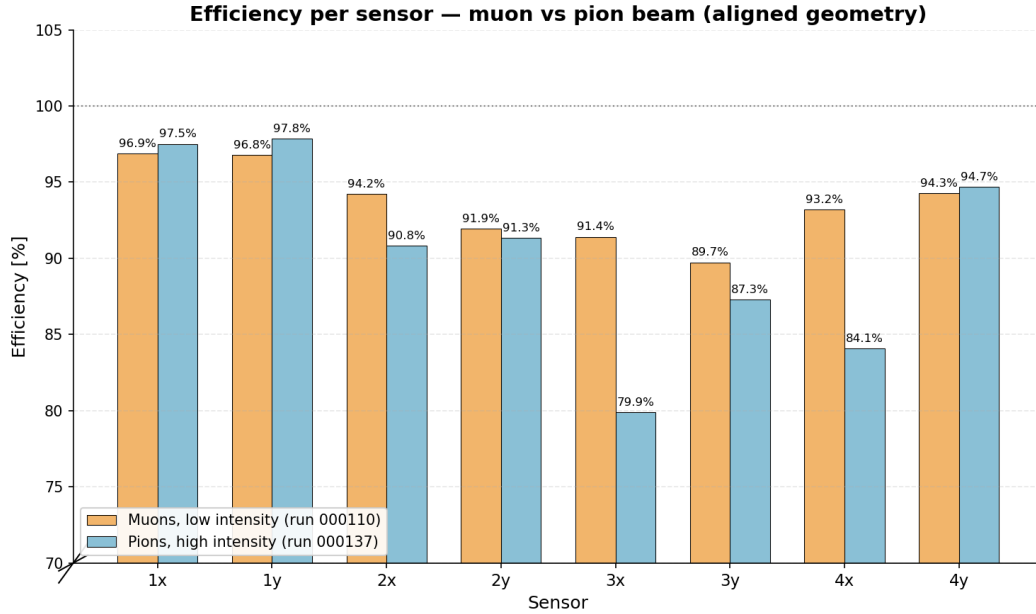


Fig. 5.3 Efficiency per sensor (improved method), compared between the muon and pion beams. The telescope is clearly less efficient under the 15 GeV pion beam, particularly on sensor 3x.

To quantify the overall impact of the methodological improvements described above, Table 5.1 compares the per-sensor efficiency obtained with the original manual matching-window

method against the improved approach, combining the internal `match_idx`-based matching with the aligned run geometry. For the muon beam, the improvement is modest but systematic, typically below one percentage point on every sensor, consistent with the fact that the unaligned geometry already provided a reasonably good description of this cleaner, lower-occupancy dataset. The pion beam, however, shows a substantially larger gain, reaching up to 4.35 percentage points on sensor 1y, reflecting the greater sensitivity of the original fixed-window method to hit ambiguity in the denser occupancy environment produced by the high-intensity beam. Despite this improvement, sensor 3x remains the weakest-performing plane under both beam conditions, confirming that the residual effect discussed before.

Table 5.1: Comparison of the per-sensor efficiency obtained with the original manual matching-window method (fixed 5 mm window, unaligned geometry) and the improved method (internal `match_idx`-based matching, aligned run geometry), for the muon beam (run 110) and the pion beam (run 137).

(a) Muon beam (run 110)

Sensor	Original method [%]	Improved method [%]	Difference [pp]
1x	96.12	96.88	+0.76
1y	95.94	96.78	+0.84
2x	93.73	94.22	+0.49
2y	91.42	91.94	+0.52
3x	90.89	91.40	+0.51
3y	89.27	89.73	+0.46
4x	92.57	93.20	+0.63
4y	93.61	94.26	+0.65

(b) Pion beam (run 137)

Sensor	Original method [%]	Improved method [%]	Difference [pp]
1x	94.10	97.50	+3.40
1y	93.50	97.85	+4.35
2x	88.30	90.83	+2.53
2y	87.80	91.35	+3.55
3x	77.90	79.90	+2.00
3y	85.10	87.30	+2.20
4x	81.50	84.08	+2.58
4y	91.70	94.69	+2.99

5.2 Effect of the beam type (muons vs pions)

To check whether the telescope’s performance depends on the type of incident particle, the efficiency measured under the low-intensity 5 GeV muon beam (run 110) was compared to the one obtained under the high-intensity 15 GeV pion beam (run 137), using the same tag-and-probe method for both. While most sensors showed comparable, high efficiency under both beam conditions, sensor 3x stood out as a clear exception: its efficiency dropped significantly under the pion beam relative compared to the muon beam, a degradation not observed on the other seven sensors.

This different behaviour points to a beam-dependent effect rather than a hardware defect. Muons are minimum-ionising particles that traverse matter with minimal interaction, produc-

ing clean, well-localised hits. This hypothesis is still under investigation and further studies are needed, but pions could undergo hadronic interactions within the detector material or its surroundings, potentially producing secondary showers. In a high-occupancy environment such as the pion run, this can lead to hit ambiguity: when several clusters are present near the position predicted by the reconstructed track, the tag-and-probe matching becomes less reliable, artificially lowering the apparent efficiency even if the sensor itself is functioning correctly. The fact that this effect was localised to a single sensor, and not observed uniformly across the telescope, is consistent with a hadronic interaction rather than a hardware default of sensor 3x itself. This point is important to emphasize, as it could impact the UBT performance once SHiP will be built: it is necessary to anticipate these behaviours to avoid bad surprises when the data taking of SHiP will start.

5.3 Periodic efficiency dips

Applying the geometry obtained from the alignment procedure (Section 3.2.1) as an input to `efficiency_new.py` improved the measured efficiency, bringing the plateau between the periodic dips close to the expected $\sim 99\%$ efficiency on most sensors. The periodic dip pattern identified in Section 3.2.2 was also confirmed in the "efficiency as a function of the position" distributions: using a peak-finding algorithm on the inverted efficiency curve, the spacing between consecutive dips was measured to be approximately 15/16 mm on all eight sensors, consistent with the physical segmentation of the SiPM readout devices.

Table 5.2: Number of detected efficiency dips and their spacing for each sensor, compared between the muon (run 110) and pion (run 137) beams. The mean spacing is consistent with the expected $\sim 15\text{--}16$ mm periodicity attributed to the SiPM readout segmentation.

Sensor	Muons (run 110)			Pions (run 137)		
	Dips	Mean [mm]	Std. dev. [mm]	Dips	Mean [mm]	Std. dev. [mm]
1x	7	14.93	4.03	6	16.38	1.25
1y	7	16.64	1.28	7	16.64	1.28
2x	7	15.36	2.56	7	17.07	1.91
2y	8	14.99	4.20	7	17.49	2.73
3x	8	14.63	4.04	9	13.76	4.42
3y	8	16.82	4.30	7	16.21	1.21
4x	7	17.07	1.91	9	14.72	3.56
4y	7	14.93	3.44	8	15.73	3.73

Conclusion

This work presented the analysis of the data collected by the EPFL SciFi telescope during the DUBT/DSBT testbeam at the CERN Proton Synchrotron, within the framework of the characterisation of SHiP’s UBT. Starting from raw hit data, a full reconstruction chain was used and improved during this internship: clustering, straight-line track fitting, geometric alignment, and a tag-and-probe efficiency measurement. Two complementary beam configurations were studied (a low-intensity 5 GeV muon beam and a high-intensity 15 GeV pion beam) checking both the performance of the telescope and its behaviour under dense occupancy conditions.

The occupancy analysis confirmed the expected wide, uniform beam profile on all eight sensors, and identified two distinct detector features: a single hot channel, present consistently across every run and beam type tested, attributed to a fixed electronics defect; and a periodic pattern of local dips, spaced at approximately 15/16 mm, attributed to the physical segmentation of the SiPM readout devices, a feature also documented for the SiPM used in the LHCb SciFi tracker. The efficiency measurement was improved through the use of the internal `match_idx`-based matching, Clopper-Pearson confidence intervals, and the aligned run geometry, which together brought the plateau efficiency of most sensors closer to the expected 99%. However, despite these successive improvements, the measured efficiency did not reach the 99% expected for this type of detector on every sensor: the periodic SiPM-related dips, together with the beam-dependent degradation observed on sensor 3x under the pion beam, account for a large part of this residual gap, but do not fully explain it. Further work would be required to close this discrepancy entirely, as discussed below.

Overall, this study demonstrates that the SciFi telescope, once properly calibrated and aligned, performs reliably as a tracking reference, with its main sources of inefficiency now identified and understood, even though the target efficiency was not fully achieved within the scope of this internship.

Next Steps

Several points identified during this analysis would benefit from further investigation :

- The pion-induced degradation observed on sensor 3x should be studied in details: comparing the efficiency-versus-position profile of this sensor across multiple pion runs of varying intensity could help determine whether the effect scales with occupancy, supporting the hadronic interaction hypothesis, or whether it points to a more localised, sensor-specific issue, and more generally help identify why the 99% efficiency value was not reached despite the corrections applied in this work.
- This characterisation effort fits within the ongoing validation of SHiP's background veto detectors, as the collaboration continues to study the different sub-systems of the experiment ahead of the submission of the Technical Design Report, a phase now nearing completion. Continued testbeam campaigns and analyses of this kind remain essential to ensure that each detector, including the Upstream Background Tagger, meets the performance requirements needed for reliable background rejection once SHiP is constructed.

Glossary

- SPS** Super Proton Synchrotron. Located at CERN, it is a 7 km circumference particle accelerator switched on in 1976.
- Hidden Sector** An hypothetical collection of particles and forces that interacts weakly with the Standard Model particles.
- SM** Standard Model. The standard theory of particle physics describing the fundamental constituents of matter and their interactions.
- HNL** Heavy Neutral Leptons. Hypothetical “sterile” neutrinos, theoretical candidates to explain the nature of dark matter.
- LDM** Light Dark Matter. Hypothetical dark matter particles with masses significantly lower than a proton.
- FIP** Feebly Interacting Particle. A hypothetical particle that interacts very weakly with Standard Model particles, predicted by several Beyond-the-Standard-Model theories.
- BSM** Beyond the Standard Model. Refers to any theoretical framework or particle not accounted for by the Standard Model.
- UBT / DUBT** Upstream Background Tagger. Detector system placed between the muon shield and the decay volume, designed to tag residual background muons entering the decay volume. “DUBT” is the naming convention used for the corresponding testbeam campaign and detector schematics; both terms refer to the same system.
- SBT / DSBT** Surrounding Background Tagger. Detector system surrounding the decay volume, designed to tag inelastic interactions and background muons. “DSBT” is the naming convention used for the corresponding testbeam campaign; both terms refer to the same system.
- SiPM** Silicon PhotoMultiplier. A solid-state photodetector used for precise timing measurements, employed here as an external timing reference detector.
- PID** Particle IDentification. A detector sub-system designed to distinguish between different particle types, such as hadrons and muons.
- CMB** Cosmic Microwave Background. The oldest observable light in the Universe, emitted approximately 380,000 years after the Big Bang.
- PoT** Protons on Target. The number of protons directed onto the target, used as a measure of accumulated beam exposure.
- TDR** Technical Design Report. A formal document detailing the final design of an experiment ahead of its construction phase.

Appendix A

Python code used for the analysis

The scripts `align.py` and `evt_display.py` were provided as part of the CuboDAQ software examples by the LPHE team, and were adapted for the four-station telescope configuration used in this testbeam. The script `efficiency_new.py` was developed during this internship to implement the unbiased tag-and-probe efficiency measurement described in Chapter 5.

Additional scripts used for occupancy analysis, window-size calibration, and residual studies are not reproduced here for conciseness, but are available upon request.

```
evt_display.py

#!/usr/bin/env python3

import argparse
import awkward as ak
import matplotlib.pyplot as plt
import numpy as np

from read_files import *
from conf_utils import RecoConfiguration

def track(x, a, b):
    return b * x + a

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('run', type=int, help='run to be processed')
    parser.add_argument('-g', '--geo-run', type=int, help='geometry run',
                        , default=None)
    args = parser.parse_args()

    rc = RecoConfiguration(geometry=f'geo_files/geometry_{args.geo_run
:06d}.toml' if args.geo_run is not None else None, configuration=
'configuration_4planes.toml')

    events = eventData(args.run, branches=['number'])
    clusters = clusterData(args.run)
    # hits = hitData(args.run)
    tracks = trackData(args.run)

    # plt.hist(ak.flatten(cluEcalX['t']), bins=50)
    # plt.show()

    SCIFI_X_SEL = (clusters['sensor_id'] % 2 == 1) & (clusters['
sensor_id'] < 12)
    SCIFI_Y_SEL = (clusters['sensor_id'] % 2 == 0) & (clusters['
sensor_id'] < 12)
    cluScifiX = clusters[SCIFI_X_SEL]
    cluScifiY = clusters[SCIFI_Y_SEL]

    TRK_SEL = (tracks['chi2']/tracks['dof'] < 30) & ((tracks['slope_x
']**2 + tracks['slope_y']**2) < 0.05**2)
```

```

tracks = tracks[TRK_SEL]

TRK_EVT_SEL = (ak.num(tracks['x'], axis=-1) >= 1) # & (ak.num(
    cluEcalX['x'], axis=-1) + ak.num(cluEcalY['x'], axis=-1) >= 16) #
    & (ak.num(calorTracks['x'], axis=-1) == 1)
cluScifiX = cluScifiX[TRK_EVT_SEL]
cluScifiY = cluScifiY[TRK_EVT_SEL]
tracks = tracks[TRK_EVT_SEL]
events = events[TRK_EVT_SEL]

X_LIMITS = (-100, 500)
Y_LIMITS = (-100, 100)

# X_LIMITS = (-900, -500)
# Y_LIMITS = (-80, 80)

xrange = np.array(X_LIMITS)

for i in range(0, 200):
    try:
        csx = cluScifiX[i]
        csy = cluScifiY[i]
        trk = tracks[i]
        evtn = events[i]['number']
    except IndexError as e:
        print(e)
        continue

    fig, ax = plt.subplots(2, 1)
    fig.suptitle(f'Event_{evtn}')

    for idx, sensor in enumerate(rc.sensors()):
        geo = rc.sensorGeometry(idx)
        offs = geo['offset']
        if sensor['subsystem'] == 'scifi':
            width = 2
            thickness = 130
            posX = (offs[2], offs[0])
            rectangleX = plt.Rectangle((posX[0]-width/2, posX[1]-thickness
                /2), width, thickness, fc='lightgray', ec='k', lw=0)
            posY = (offs[2], offs[1])
            rectangleY = plt.Rectangle((posY[0]-width/2, posY[1]-thickness
                /2), width, thickness, fc='lightgray', ec='k', lw=0)
            ax[0].add_patch(rectangleX)
            ax[1].add_patch(rectangleY)
        elif sensor['subsystem'] == 'ecal':
            width = 4
            thickness = 384
            direction = np.cross(geo['unit_u'], geo['unit_v'])[2]
            for i in range(8):
                posX = (offs[2] + direction * (i * 22.6 + 2.0), offs[0]) #
                    TODO should use thickness
                rectangleX = plt.Rectangle((posX[0]-width/2, posX[1]-
                    thickness/2), width, thickness, fc='lightgray', ec='k',
                    lw=0)
                posY = (offs[2] + direction * (i * 22.6 + 2.0), offs[1])
                rectangleY = plt.Rectangle((posY[0]-width/2, posY[1]-

```

```

        thickness/2), width, thickness, fc='lightgray', ec='k',
        lw=0)
    ax[0].add_patch(rectangleX)
    ax[1].add_patch(rectangleY)

width = 5.6
thickness = 384
for i in range(8):
    posX = (offs[2] + direction * (i * 22.6 + 7.8), offs[0]) #
        TODO should use thickness
    rectangleX = plt.Rectangle((posX[0]-width/2, posX[1]-
        thickness/2), width, thickness, fc='gray', ec='k', lw=0)
    posY = (offs[2] + direction * (i * 22.6 + 7.8), offs[1])
    rectangleY = plt.Rectangle((posY[0]-width/2, posY[1]-
        thickness/2), width, thickness, fc='gray', ec='k', lw=0)
    ax[0].add_patch(rectangleX)
    ax[1].add_patch(rectangleY)

scx = ax[0].scatter(csx['z'], csx['x'], s=10*csx['size'])
# plt.colorbar(scx, ax=ax[0])
ax[0].set_ylabel('x□[mm]')
ax[0].set_ylim(Y_LIMITS)
ax[0].set_xlim(X_LIMITS)

scy = ax[1].scatter(csy['z'], csy['y'], s=10*csy['size'])
# plt.colorbar(scy, ax=ax[1])
ax[1].set_ylabel('y□[mm]')
ax[1].set_xlabel('z□[mm]')
ax[1].set_ylim(Y_LIMITS)
ax[1].set_xlim(X_LIMITS)

for it in range(len(trk['x'])):
    ax[0].plot(xrange, track(xrange, trk['x'][it], trk['slope_x'][it
    ]), c='r')
    ax[1].plot(xrange, track(xrange, trk['y'][it], trk['slope_y'][it
    ]), c='r')

fig.set_tight_layout(True)
plt.show()

if __name__ == '__main__':
    main()

```

```

align.py

#!/usr/bin/env python3

import os
import shutil
import os.path as op
import argparse
import awkward as ak
import matplotlib.pyplot as plt

```

```

import numpy as np
import toml
from datetime import datetime

from read_files import *
from conf_utils import RecoConfiguration
from reco import process_run

def track(x, a, b):
    return b * x + a

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('run', type=int, help='run to be processed')
    args = parser.parse_args()

    startTime = datetime.now()

    geoFilename = op.join('geo_files', f'geometry_{args.run:06d}.toml')

    if op.exists(geoFilename):
        shutil.copy2(geoFilename, geoFilename + '.bak')
        os.remove(geoFilename)

    plotsPath = 'plots_align'
    shutil.rmtree(op.join(plotsPath, f'run_{args.run:06d}'),
                  ignore_errors=True)
    os.makedirs(op.join(plotsPath, f'run_{args.run:06d}'))

    align(args.run, [0, 1, 2, 3, 4, 5, 6, 7], 20, 'u', geoFilename,
           plotsPath=plotsPath, initialPrecision=10., damping=0.8)
    # align(args.run, [0, 1], 20, 'u', geoFilename, plotsPath=plotsPath,
    #        initialPrecision=10., damping=0.8)
    align(args.run, [2, 3, 4, 5, 6, 7], 10, 's', geoFilename, plotsPath=
           plotsPath, initialPrecision=5., damping=0.5)
    # align(args.run, [0, 1, 11], 10, 's', geoFilename, plotsPath=
    #        plotsPath, initialPrecision=5., damping=0.5)

    print('Elapsed time:', datetime.now() - startTime)

def align(run: int, sensors: list, steps: int, coordinate: str,
          geoFileName: str, initialPrecision: float = 30., plotsPath: str = '
plots_align', initialChi2Cut: float = 30., damping=0.8):

    offsets = dict()
    for s in sensors:
        offsets[s] = []

    for it in range(steps):
        print(f'{coordinate}_alignment_for_sensors_{sensors}:_step_{it}.')

        if op.exists(geoFileName):
            geofile = geoFileName
        else:
            geofile = None

```

```

rc = RecoConfiguration(geometry=geofile, configuration='
    configuration_4planes.toml')

addargs = ['-n', '10000', '-l', 'e', '-u', 'align_scifi']
if geofile is not None:
    addargs.extend(['-g', geofile])

print('Running reconstruction...')
process_run(run, 'data_raw', 'data_align', stdout=True,
    additional_args=addargs)

print('Reading data...')
clusters = clusterData(run, branches=['sensor_id', 'track_idx',
    coordinate], dataPath='data_align')
intercepts = interceptData(run, branches=['sensor_id', 'track_idx',
    coordinate], dataPath='data_align')
tracks = trackData(run, branches=['chi2', 'dof'], dataPath='
    data_align')

# we want single tracks events since the beginning
EVT_TRK_CUT = (ak.num(tracks['chi2'], axis=-1) == 1)
tracks = tracks[EVT_TRK_CUT]
clusters = clusters[EVT_TRK_CUT]
intercepts = intercepts[EVT_TRK_CUT]

TRK_SEL = (tracks['chi2']/tracks['dof'] < initialChi2Cut) # & ((
    tracks['slope_x']**2 + tracks['slope_y']** 2) < 0.05**2)
tracks = tracks[TRK_SEL]

EVT_TRK_CUT = (ak.num(tracks['chi2'], axis=-1) == 1)
tracks = tracks[EVT_TRK_CUT]
clusters = clusters[EVT_TRK_CUT]
intercepts = intercepts[EVT_TRK_CUT]

oldGeometry = rc.geo

maxDelta = 0
for i in sensors:

    SENSOR_SEL_CLU = (clusters['sensor_id'] == i) & (clusters['
        track_idx'] == 0)
    clu = clusters[SENSOR_SEL_CLU]

    SENSOR_SEL_INT = (intercepts['sensor_id'] == i) & (intercepts['
        track_idx'] == 0)
    trk = intercepts[SENSOR_SEL_INT]

    SEL = (ak.num(clu['sensor_id'], axis=-1) == 1) & (ak.num(trk['
        sensor_id'], axis=-1) == 1)
    clu = clu[SEL]
    trk = trk[SEL]

    res = ak.flatten(trk[coordinate] - clu[coordinate])
    res = res[~np.isnan(res)]
    res = res[(-initialPrecision < res) & (res < initialPrecision)]
    # center = np.median(resU)
    # resU = resU[(-5 < (resU - center)) & ((resU - center) < 5.)]

```

```

h, be = np.histogram(res, bins=200)
x = (be[1:] + be[:-1]) / 2
xmax = x[np.argmax(h)]

res = res[(xmax-initialPrecision/10 < res) & (res < xmax+
initialPrecision/10)]
delta = np.mean(res)
maxDelta = max(maxDelta, abs(delta))

print(f'{rc.sensorName(i)}_{d{coordinate}}:{delta:.3f}_{{"mm" if _
coordinate_=="u" else "ns"}}')

currentGeo = rc.sensorGeometry(i)
if coordinate == 's':
    offsets[i].append(currentGeo['offset'][3])
    currentGeo['offset'][3] = float(round(currentGeo['offset'][3]
+ damping*delta, 5))
else:
    if abs(currentGeo['unit_u'][0]) > 0.5:
        offsets[i].append(currentGeo['offset'][0])
        currentGeo['offset'][0] = float(round(currentGeo['offset'
][0] + damping*delta, 5))
    else:
        offsets[i].append(currentGeo['offset'][1])
        currentGeo['offset'][1] = float(round(currentGeo['offset'
][1] + damping*delta, 5))

oldGeometry['sensors'][i] = currentGeo

# print(oldGeometry)
with open(geoFileName, 'w') as f:
    toml.dump(oldGeometry, f)
# plt.hist(resU, bins=200)
# plt.show()

if maxDelta <= 0.001:
    print('Detector is aligned!')
    break

offsetsArr = []
legendsArr = []
for s, o in offsets.items():
    offsetsArr.append(o)
    legendsArr.append(rc.sensorName(s))
offsetsArr = np.array(offsetsArr)
sensorsStr = [str(s) for s in sensors]
ss = '-'.join(sensorsStr)

plt.clf()
plt.plot(offsetsArr.T, marker='.')
plt.grid(alpha=0.3)
plt.xlabel('step')
plt.ylabel(f'offset_{{"mm" if _coordinate_=="u" else "ns"}}')
plt.legend(legendsArr)

plt.savefig(op.join(plotsPath, f'run_{run:06d}', f'offsets_{

```

```

        coordinate}_{ss}.png'))
    # plt.show()

if __name__ == '__main__':
    main()

```

efficiency_new.py

```

#!/usr/bin/env python3
import argparse
import os.path as op
import subprocess
import csv
import toml
import uproot
import numpy as np
import awkward as ak
import matplotlib
import matplotlib.pyplot as plt
from scipy.signal import find_peaks
from scipy.stats import beta
#matplotlib.use('TkAgg')

def make_config(base_conf_path, excluded_sensor, out_path,
               all_sensor_ids):
    conf = toml.load(base_conf_path)
    remaining = [s for s in all_sensor_ids if s != excluded_sensor]
    subsection_name = f'eff_test_{excluded_sensor}'
    conf[subsection_name] = {
        'event_processors': {
            'processors': ['scifi_tracker'],
            'scifi_tracker': {
                'clusters_min': 3, 'tracks_max': 25,
                'tracking_sensor_ids': remaining,
                'matching_sensor_ids': [excluded_sensor],
            }
        }
    }
    with open(out_path, 'w') as f:
        toml.dump(conf, f)
    return subsection_name

def clopper_pearson(n_matched, n_total, confidence=0.68): #to compute
the uncertainty with the Clopper-Pearson interval
    alpha = 1 - confidence
    if n_total == 0:
        return np.nan, np.nan, np.nan

    e = n_matched / n_total
    lower = beta.ppf(alpha / 2, n_matched, n_total - n_matched + 1) if
        n_matched > 0 else 0.0
    upper = beta.ppf(1 - alpha / 2, n_matched + 1, n_total - n_matched
        ) if n_matched < n_total else 1.0

    return e, lower, upper

```

```

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('run', type=int)
    parser.add_argument('--base-conf', default='configuration_4planes.
        toml')
    parser.add_argument('--data-raw', default='data_raw')
    parser.add_argument('--data-out', default='data_eff')
    parser.add_argument('--geo', default=None)
    parser.add_argument('--device', default='device_4planes.toml')
    parser.add_argument('--total-sensors', type=int, default=8)
    parser.add_argument('--sensors', type=int, nargs='*', default=None
        )
    parser.add_argument('--window', type=float, default=2.0)
    parser.add_argument('--edge-margin', type=float, default=15.0)
    parser.add_argument('--csv-out', default=None)
    parser.add_argument('--save', action='store_true')
    args = parser.parse_args()

    all_sensor_ids = list(range(args.total_sensors))
    sensors_to_test = args.sensors if args.sensors is not None else
        all_sensor_ids
    tmp_conf = 'configuration_eff_tmp.toml'

    names = [s['name'] for s in toml.load(args.device)['sensors']]

    results = []
    print(f"===Efficiency-Run{args.run:06d}===\n")

    reduced_chi2_sensor = {}
    dist_by_sensor = {}
    eff_vs_pos_by_sensor = {}

    for sid in sensors_to_test:
        subsection = make_config(args.base_conf, sid, tmp_conf,
            all_sensor_ids)
        out_dir = op.join(args.data_out, f'sensor_{sid}', f'run_{args.
            run:06d}')
        cmd = ['reco', '-c', tmp_conf, '-d', args.device, '-u',
            subsection, '-l', 'e', '-k']
        if args.geo:
            cmd += ['-g', args.geo]
        cmd += [op.join(args.data_raw, f'run_{args.run:06d}'), out_dir
            ]
        subprocess.run(cmd, check=True)

        f = uproot.open(op.join(out_dir, 'data_0000.root'))
        intercepts = f['intercepts'].arrays(['sensor_id', 'u'],
            library='ak')
        clusters = f['clusters'].arrays(['sensor_id', 'u', 'match_idx'
            ], library='ak')
        events = f['events'].arrays(library='ak')
        tracks = f['tracks'].arrays(['chi2', 'dof'], library='ak')

        ic_u = intercepts['u'][intercepts['sensor_id'] == sid]

```

```

clu_u = clusters['u'][clusters['sensor_id'] == sid]

print(sid)

# global efficiency

clu_sid = clusters[clusters['sensor_id'] == sid] # choosing
the clusters corresponding to the current sensor
ic_sid = intercepts[intercepts['sensor_id'] == sid] # same for
intercepts

red_chi2 = tracks['chi2'] / tracks['dof'] # computing reduced
chi2
select_tracks = tracks[red_chi2 < 10] # applying a cut on
tracks
tracks_nb = ak.num(select_tracks['chi2'], axis = 1) # to get
the number of tracks by events
single_tracks = tracks_nb == 1 # select only the events with
one track

select_clu = clu_sid[single_tracks] # select clusters of the 1
track events
select_ic = ic_sid[single_tracks] # same for intercepts

n_ic = ak.num(select_ic['u'], axis = 1)
one_ic = n_ic == 1
select_clu_f = select_clu[one_ic]
select_ic_f = select_ic[one_ic]

clu_match = select_clu_f[select_clu_f['match_idx'] == 0] # to
match the events with the corresponding clusters
ic_flat = ak.flatten(select_ic_f['u'], axis = 1) # flattens
the intercepts to convert to a scalar

# to compute
the
distance

dist = np.abs(clu_match['u'] - ic_flat) # distance
min_dist = ak.fill_none(ak.min(dist, axis=1), np.inf) #
minimum distance: if there is an empty array (no cluster e.
g.)

# it
fills
it
with
infinite
(so
could
not
be
the
min
dist)

window = 10.0
matched = min_dist < window # if the minimum distance is in
the window, it is a match
efficiency = ak.sum(matched) / len(select_ic_f)
print(efficiency * 100)

```

```

# efficiency as a function of position

stype = toml.load('device_4planes.toml')['sensor_types'][toml.
    load('device_4planes.toml')['sensors'][sid]['type']]
half_width = stype['channels'] * stype['channel_size_u'] / 2

n_bins = 50
bins = np.linspace(-half_width, half_width, n_bins + 1) # cuts
    the detector into n_bins slices
bin_centers = (bins[:-1] + bins[1:]) / 2 # to find the center
    of the bins

has_match = ak.num(dist, axis=1) > 0 # same logic as before
has_match_np = ak.to_numpy(has_match)
ic_flat_np = ak.to_numpy(ic_flat)

bin_idx = np.digitize(ic_flat_np, bins) - 1 # takes each
    predicted position and allocates the slice number where it
    # lands
eff_per_bin = np.full(n_bins, np.nan) # creates a array of
    size n_bins
err_lower_per_bin = np.full(n_bins, np.nan)
err_upper_per_bin = np.full(n_bins, np.nan)

for b in range(n_bins):
    mask = bin_idx == b
    n = mask.sum()
    if n > 0:
        n_matched_bin = has_match_np[mask].sum()
        e, lower, upper = clopper_pearson(n_matched_bin, n)
        eff_per_bin[b] = e * 100
        err_lower_per_bin[b] = (e - lower) * 100
        err_upper_per_bin[b] = (upper - e) * 100

    eff_vs_pos_by_sensor[sid] = (bin_centers, eff_per_bin,
        err_lower_per_bin, err_upper_per_bin) # gets the
        bins,

bc, eff, err_lower, err_upper = eff_vs_pos_by_sensor[sid]

```

```

# distance between the dips

valid = ~np.isnan(eff)
eff_valid = eff[valid]
bc_valid = bc[valid]

dips_idx, _ = find_peaks(-eff_valid, prominence=3) # -
# eff_per_bin beacuse find_peaks searches for the maximum
# looking for 3
# points
# efficiency
# difference

dip_positions = bc_valid[dips_idx] # to link the positions to
# the list of dips index
if len(dip_positions) > 1:
    spacing = np.diff(dip_positions) # computes the difference
    # between dips
    print(f"{names[sid]}: {len(dip_positions)} dip, {mean_spacing} mm")
    print(f"spacing between consecutive dips: {spacing}")
else:
    print(f"{names[sid]}: not enough dips detected to compute a spacing")

# plot of number of entries as a function of the distance

dist_flat = ak.flatten(dist, axis=1) # flaattens the disatnce
# array
dist_np = ak.to_numpy(dist_flat) # converts to numpy for the
# plot
dist_by_sensor[sid] = dist_np # stocking in a dictionnary for
# plotting

fig, axes = plt.subplots(4, 2, figsize=(10, 12))
axes = np.array(axes).reshape(-1)

for sid in range(8):
    ax = axes[sid]
    ax.hist(dist_by_sensor[sid], bins=50, range=(-1, 1), color = '#2c6f8c')
    ax.set_title(names[sid])
    ax.set_xlabel('distance [mm]')
    ax.set_ylabel('entries')

fig.tight_layout()

if args.save:
    fig.savefig('entries_vs_distance.png', dpi=150)
else:
    plt.show()

```

```

# plot of efficiency as a function of the position

fig, axes = plt.subplots(4, 2, figsize=(11, 12))
axes = np.array(axes).reshape(-1)

for sid in range(8):
    bc, eff, err_lower, err_upper = eff_vs_pos_by_sensor[sid]
    ax = axes[sid]

    cap = 15.0
    err_lower_display = np.minimum(err_lower, cap)
    err_upper_display = np.minimum(err_upper, cap)

    ax.errorbar(bc, eff, yerr=[err_lower_display,
                              err_upper_display],
                fmt='o-', markersize=4, capsize=3, color = '#2
                              c6f8c')

    ax.set_title(names[sid])
    ax.set_xlabel('u□[mm]')
    ax.set_ylabel('efficiency□[%]')
    ax.set_ylim(70, 105)

fig.tight_layout()
if args.save:
    fig.savefig('efficiency_vs_position_new.png', dpi=150)
    print('Figure□saved')
else:
    plt.show()

if __name__ == '__main__':
    main()

```

Appendix B

Particle Symbols Used in Figure 2.6

Summary of notations for the HSDS and SND models (SHiP experiment, CERN)

1. Particle symbols

Symbol	Name (English)
$\ell^\pm$	Charged lepton (electron, muon, or tau, positive or negative)
$\ell^+\ell^-$	Lepton-antilepton pair
ν	Neutrino
$\bar{\nu}$	Antineutrino
ν_τ	Tau neutrino
$\bar{\nu}_\tau$	Tau antineutrino
ν_e	Electron neutrino
ν_μ	Muon neutrino
$\tau^\pm$	Tau (charged lepton, positive or negative)
$\pi^\pm, \pi^0$	Pion (charged or neutral)
$2\pi, 3\pi, 4\pi$	Two, three, or four pions
K, KK	Kaon, kaon pair
$\rho^\mp$	Rho meson (charged)
γ	Photon
$\gamma\gamma$	Photon pair (diphoton)
η	Eta meson
q	Quark
$\bar{q}$	Antiquark
$q\bar{q}$	Quark-antiquark pair
$q\bar{q}'$	Quark-antiquark pair + lepton (different-flavour quark)
G, GG	Gluon, gluon pair
D	D meson (charm)
$D^\pm, D^0$	Charged D meson, neutral D meson
$D_s^\pm$	Charged strange D meson
D^0	Neutral D antimeson
$D\bar{D}$	D meson - D antimeson pair
Λ_c^+	Charmed lambda baryon (positive)
Λ_c^-	Charmed lambda antibaryon (negative)
$l, \pi l, \rho l$	Lepton (generic notation), pion+lepton and rho+lepton combinations

Note: the superscripts/subscripts $\pm, \mp, +, -, 0$ indicate the particle's electric charge (positive, negative, neutral, or the "opposite" charge for $\mp$ when paired with another symbol already marked $\pm$).

2. Model and detector acronyms (SHiP context)

Acronym	Full name	Description
HSDS	Hidden Sector Decay Spectrometer	SHiP's downstream detector, designed to reconstruct the decay vertex of a hidden-sector particle, measure its invariant mass, and identify the decay products
SND	Scattering and Neutrino Detector	SHiP's upstream detector, designed to search for light dark matter (LDM) scattering and to perform neutrino measurements; it also provides the flux normalisation for the hidden-sector particle search
SUSY	Supersymmetry	Theory postulating a “superpartner” for every Standard Model particle
Neutralino	Neutralino	Neutral supersymmetric particle, often a dark-matter candidate in SUSY models
ALP	Axion-Like Particle	Light pseudoscalar boson generalising the axion
HNL	Heavy Neutral Lepton	Hypothetical sterile neutrino, heavier than the known neutrinos
Axino	Axino	Fermionic superpartner of the axion, predicted in theories combining SUSY and axions
Sgoldstino	Sgoldstino	Scalar superpartner of the goldstino (the Goldstone fermion associated with spontaneous SUSY breaking)
LDM	Light Dark Matter	Dark-matter candidates with sub-GeV mass

Additional notes: for ALPs, the indicated coupling (fermionic, gluonic, or photonic) specifies which type of particle the ALP boson is assumed to couple to preferentially. The “dark photon” and “dark scalar” are hypothetical bosons mediating an interaction within the hidden sector, of spin 1 and spin 0 respectively.

Source: A. Alekou et al., “The SHiP experiment at the proposed CERN SPS Beam Dump Facility”, Eur. Phys. J. C 82, 486 (2022).

Bibliography

- [1] Heinz Andernach and Fritz Zwicky. *English and Spanish Translation of Zwicky's (1933) The Redshift of Extragalactic Nebulae*. 2017. DOI: [10.48550/ARXIV.1711.01693](https://arxiv.org/abs/1711.01693). URL: <https://arxiv.org/abs/1711.01693> (visited on 08/17/2026).
- [2] A. Brignoli et al. *Performance of Prototypes with Different Reflector Materials for the SHiP Liquid Scintillator Surrounding Background Tagger*. arXiv:2503.10250 [physics.ins-det]. June 2025. DOI: [10.48550/arXiv.2503.10250](https://arxiv.org/abs/2503.10250). URL: <http://arxiv.org/abs/2503.10250> (visited on 09/10/2026).
- [3] Matei Climescu. *The SHiP/NA67 experiment: the flagship to probe hidden sectors*. English. PowerPoint presentation. CERN, Jan. 2026.
- [4] *Des grumeaux de matière noire dans la Voie Lactée ?* fr. URL: <https://lejournal.cnrs.fr/nos-blogs/focus-sciences/des-grumeaux-de-matiere-noire-dans-la-voie-lactee> (visited on 08/18/2026).
- [5] A. Di Crescenzo. “The SHiP experiment at CERN”. eng. In: *Il Nuovo Cimento C* 49.1-2 (Jan. 2026), pp. 1–6. ISSN: 03905551, 03905551. DOI: [10.1393/ncc/i2026-26049-8](https://doi.org/10.1393/ncc/i2026-26049-8). URL: <https://doi.org/10.1393/ncc/i2026-26049-8> (visited on 08/18/2026).
- [6] *Efficiency Studies using the Tag and Probe Method*. URL: <https://cms-opendata-workshop.github.io/workshop-lesson-tagandprobe/index.html> (visited on 08/18/2026).
- [7] Sylvia Galli. *Les contraintes du fond diffus cosmologique*. fr. PowerPoint presentation. Collège de France, Dec. 2024. URL: <https://www.college-de-france.fr/fr/agenda/seminaire/le-secteur-sombre-de-univers-matiere-et-energie-noires/les-contraintes-du-fond-diffus-cosmologique>.
- [8] Vasilisa Guliaeva. *The BDF/SHiP facility at the ECN3 high-intensity beam facility at the CERN SPS*. PowerPoint presentation. Kolymbari, Crete, Mar. 2024.
- [9] Guido Haefeli. *SiPM for LHCb SciFi Tracker*. PowerPoint presentation. CERN, Geneva, 2022.
- [10] Heiko Laiker. *The SHiP Surround Background Tagger (SBT)*. PowerPoint presentation. Dec. 2021.
- [11] Anne-Marie Magnan. *The SHiP Experiment*. PowerPoint presentation. Tokyo, Jan. 2024.
- [12] *Particle Physics / David Tong*. URL: <https://davidtong.org/teaching/particle-physics/> (visited on 08/17/2026).
- [13] SHiP Collaboration. *BDF/SHiP Annual Report 2025*. Tech. rep. CERN-SPSC-2025-039 ; SPSC-SR-370. Geneva: CERN, 2025. URL: <https://cds.cern.ch/record/2948477>.
- [14] *Standard model – Home / CERN*. fr-FR. URL: <https://home.cern/fr/science/physics/standard-model/> (visited on 08/18/2026).
- [15] Gabor Szekely. *Statistics for the 21st Century: Methodologies for Applications of the Future*. eng. Statistics: a Series of Textbooks and Monographs v.Vol. 161. Boca Raton: Taylor & Francis Group, 2000. ISBN: 9780824790295 9781482277494.
- [16] *The Standard Model of Particle Physics and Beyond / ATLAS Open Data*. en. URL: https://atlas-outreach-data-tools.gitlab.cern.ch/docs/documentation/introduction/SM_and_beyond (visited on 08/18/2026).