



UNIVERSITÀ DEGLI STUDI DI SIENA

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea Triennale in Fisica e Tecnologie Avanzate

TESI DI LAUREA
18 Aprile 2011

A Large GEM detector prototype

Test-beam results and analysis

Candidato:
Elena Graverini

Relatore:
Prof. Nicola Turini

Abstract

My thesis presents the characterization of a new Gas Electron Multiplier detector prototype. The tests were performed at CERN laboratories, in Genève, and aimed at checking the prototype efficiency when working in high-radiation environments similar to the ones which it was designed to work in, the LHC (Large Hadron Collider).

The manufacture of the detector is described, with its distinctive features; the preliminary tests on the components; the data acquisition electronics; the test beam setup; and lastly the test beam extracted data.

My thesis will also go through the data analysis procedure, presenting some of the routines that were written for this purpose; the achieved results will be shown and commented, focusing on the differences between the prototype and a "common" GEM detector and pointing out opportunities for further improvements.

Sommario

La mia tesi presenta la serie di test eseguiti su un nuovo prototipo di rivelatore di particelle di tipo Gas Electron Multiplier. Si tratta di un rivelatore di grande area realizzato con una tecnica che permette di unire più fogli GEM di dimensioni minori in un solo piano e di evitare problemi di allineamento in fase di foratura.

I test sono stati eseguiti nei laboratori del CERN di Ginevra allo scopo di verificare le prestazioni del prototipo e la sua resistenza in ambienti ad alto contenuto di radiazioni: questa nuova tipologia di rivelatori è stata infatti disegnata per un futuro utilizzo in LHC (Large Hadron Collider).

Esporrò in dettaglio il processo di costruzione del prototipo, i test preliminari di guadagno e l'analisi del rumore; verrà descritta l'elettronica di readout, per poi passare all'allestimento del periodo di test su fascio di particelle; esaminerò i risultati ottenuti nell'ottica di un'integrale caratterizzazione del rivelatore. Mostrerò alcuni degli algoritmi utilizzati per l'analisi dei dati raccolti durante il test beam.

I risultati ottenuti verranno infine esposti illustrando le differenze con le prestazioni dei rivelatori a tecnologia GEM di dimensioni standard, e suggerendo le migliorie che possono ancora essere apportate.

Contents

1	Large GEM prototype and test setup	9
1.1	GEM detectors	9
1.2	A large triple GEM detector	11
1.3	Electronics	14
1.3.1	VFAT readout chip	14
1.3.2	Turbo readout card	14
1.3.3	Chip parameters	14
1.4	Experimental setup	14
1.4.1	Test beam	14
1.4.2	The telescope	15
1.5	Data analysis system	17
1.5.1	Hits, clusters and tracks	17
1.5.2	Beam profile reconstruction	18
2	Off beam analysis	23
2.1	Large GEM characterization	23
2.1.1	Gain curve	23
2.1.2	Gas mixtures	24
2.2	Noise	24
2.2.1	Threshold scan	24
2.2.2	S-curve	24
2.2.3	Detector mapping	24

3	On beam analysis	25
3.1	Tracker efficiency	25
3.2	Prototype efficiency	25
3.2.1	High voltage scan	25
3.2.2	Threshold scan	27
3.2.3	Timing scan	27
3.2.4	Behaviour with hadron beam	30
4	Remarks	35
4.1	(In)homogeneity of the prototype	35
4.1.1	Chamber borders, spacer frame and foils junction zone	35
4.1.2	Effects of the different dimensions of the pads	39
4.2	Data analysis system efficiency	41
4.2.1	Efficiency radius	41
4.2.2	Cuts	43
5	Conclusions	45
5.1	Quality of the large GEM prototype	45
5.2	Large GEM detectors in TOTEM and CMS experiments . . .	45
	Appendix A ROOT analysis routines	49
	Bibliography	55

Large GEM prototype and test setup

1.1 GEM detectors

High-energy physics experiments aim at detecting the presence and characteristics of particles. Different kinds of detectors are used in order to achieve this, which can reveal different features of the particle itself: kinetic energy, momentum, charge, mass and even its internal structure.

Gas ionization detectors can reveal high-rate electromagnetically-interacting particles. If the incident photon or massive charged particle has enough energy to ionize one atom of gas inside the detector, the ion-electron pair flux will generate a current pulse for each event which can be separately revealed by some conducting material structure and then can be sent to any electronic *data acquisition* (DAQ) setup.

Depending on the voltage applied, gaseous detectors can produce an avalanche multiplication of the signal by accelerating the first electrons produced by the incident particle and causing them to create more pairs. Gas electron multipliers (*GEMs*) are based on this principle. *GEM foils* consist of two thin copper layers, with a kapton foil between them. Small and regular holes are produced through copper and kapton foils, via chemical processes such as photolithography and acid etching. High voltage is then

applied between the two copper foils (*anode* and *cathode*) to put the GEM foil in operation; this will create a high electric field through the holes:

$$E = \frac{V}{d}$$

where E is the electric field, V is the voltage and d is the distance between anode and cathode. Since the kapton foil is very thin, but the voltage applied can be high¹, the electric field inside the GEM holes can be as high as

Typically a GEM-based detector consists of one or more GEM foils inserted between a copper foil (*cathode*) and a readout plane (*anode*):

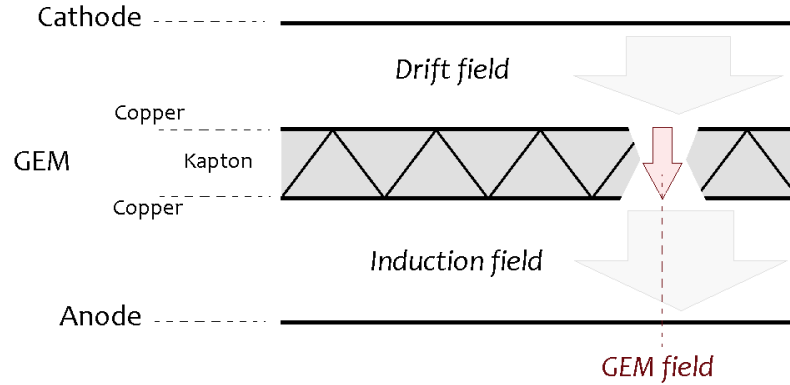


Figure 1.1: Schematics of a GEM-based detector (single GEM foil)

The different detector layers require different voltage settings: a *drift voltage* to guide electrons from the cathode drift foil to the GEM, an *amplification voltage* between the copper layers of every GEM foil, and an *induction voltage* to guide electrons from the GEM bottom layer to the readout plane.

¹Kapton is a plastic material designed so that its molecular structure is not affected by extreme environment conditions

1.2 A large triple GEM detector

Using $66 \times 66 \text{ cm}^2$ GEM foils, a prototype triple GEM detector with an active area of about 2000 cm^2 was built [3]. Two innovative techniques were used to manufacture this detector: *single-mask* etching and *GEM foils splicing*.

GEM foils are produced by the same photolithographic processes as for building common printed circuit boards. Typically, small GEMs are etched on both sides to produce symmetric holes; however, it is very uncomfortable to align masks when the GEM foil dimensions grow. In the case of this large area prototype, it was virtually mandatory to develop a single-mask etching technique. As shown in Figure 1.2, after the top copper layer is etched, the holes in the polyimide material (Kapton) are made by a basic mixture containing potassium hydroxide (KOH, which etches isotropically) and ethylene diamine ($\text{C}_2\text{H}_4(\text{NH}_2)_2$, which etches anisotropically). An anisotropic etcher is needed to keep the holes aspect ratio, defined as $\frac{\text{depth}}{\text{width}}$, high.

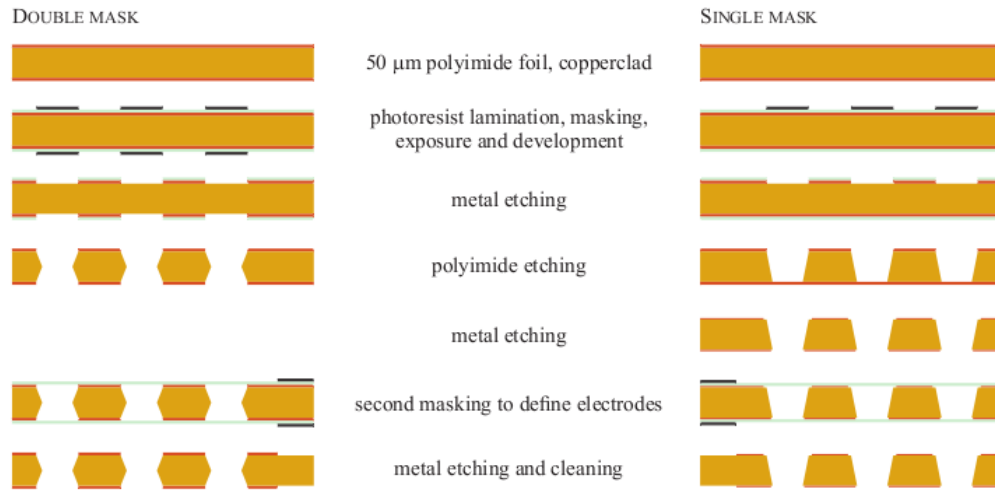
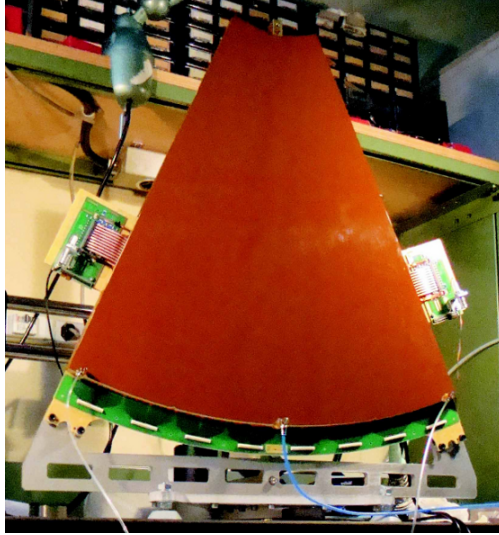


Figure 1.2: Comparison of the standard *double-mask* and the new *single-mask* GEM etching procedures

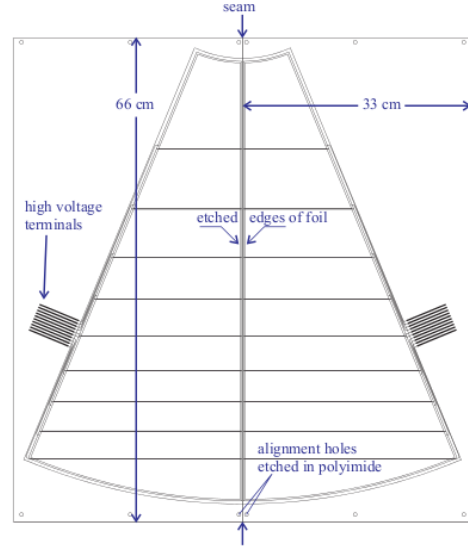
The bottom copper layer is then etched from both sides, using the holes in the kapton as mask, dipping the whole foil in an acidic etchant mixture in

order to finalize the holes and to slim the electrodes thickness.

Suppliers of the base material to produce GEM foils can so far offer only around half a meter wide rolls. Two crosswise halves of a $66 \times 66 \text{ cm}^2$ foil were spliced together by covering the junction with a $25 \mu\text{m}$ Kapton adhesive substrate. The glue polymerized after a baking. The loss of efficiency due to the 2 mm seam will not be gone into in this thesis, as in this triple GEM detector a $0,5 \text{ cm}$ wide plastic spacer placed within the chamber totally covered the junction area.



(a) View of the prototype



(b) Layout of the GEM foils

Figure 1.3: Schematics of the large area triple GEM detector

To reduce the discharge probability the cathode electrodes are segmented and connected to the power supply via $10 \text{ M}\Omega$ resistors. For the high voltage distribution a compact divider board was used, making it easy to eventually debug the circuit. Since it was not planned to handle such high voltages, groups of pins were connected together and alternated with groups of floating strips (see Figure 1.4(a) on the next page). The readout configuration consists of 1024 pads, each with a surface that goes from $0,25 \text{ cm}^2$ (in the

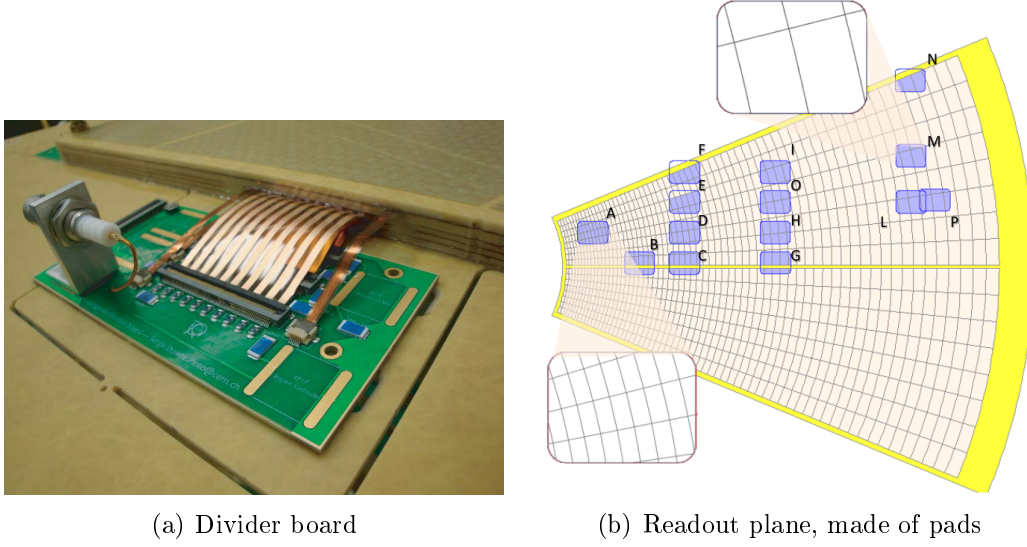


Figure 1.4: A view of the compact divider board, and the schematics of the pad-based readout with indication of the beam-tested regions

narrower part of the detector) to 6cm^2 . Figure 1.4(b) shows the differences in dimension within the readout pads; the image also points out the zones where we directed the beam during the test of the chamber.

The readout VFAT chips were connected to the detector via small hybrid printed circuit boards, which were bound to the detector itself on its larger rounded border.

1.3 Electronics

1.3.1 VFAT readout chip

1.3.2 Turbo readout card

1.3.3 Chip parameters

Threshold

Latency

Monostable pulse lenght

1.4 Experimental setup

1.4.1 Test beam

CERN provides researches with **test beam** facilities to check detectors before letting them down inside the LHC cavern. Our trial period was scheduled between 12th and 22th August, 2010, and it took place at the **H4** beam line in Prévessin.

Bunches of pions are accelerated in the SPS, a circular particle accelerator, and then extracted with a momentum of about $150\text{GeV}/c$ to a straight beam line near the site of Prévessin; the beam is then divided into several lines to provide more than one permanent or temporary test beam facility at a time.

Both hadron and lepton beams are provided, in order to cover all the possible test requests made by the various experiments. Pions have a mean lifetime $\tau = (2,6033 \pm 0,0005) * 10^{-8}s$; their primary decay branch of pions (with *branching ratio* $\Gamma_i/\Gamma = (99,98770 \pm 0,00004)\%$ [1]) is leptonic:

$$\pi^- \longrightarrow \mu^- + \bar{\nu}_\mu$$

It is then obvious that such beams as described above contain normally both pions and their decay products, muons. Switching to a pure lepton beam is possible by moving the beam collimators...

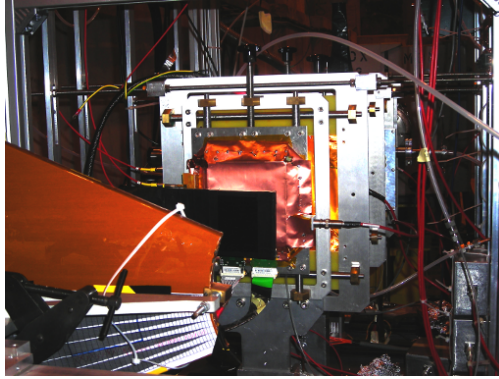
1.4.2 The telescope

The test-beam experimental setup was conceived so that it made use of the RD51-GDD² tracker. The tracker is made of:

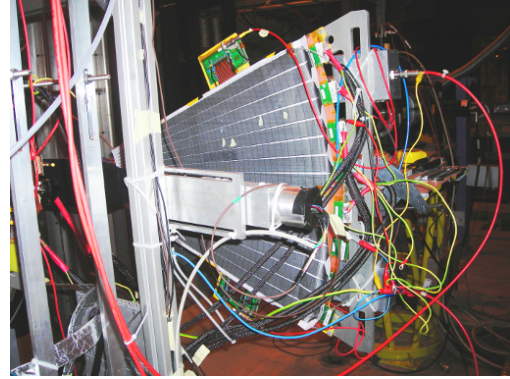
- 3 scintillators, used as a trigger;
- 3 10cmx10cm GEM detectors, used as a track detecting system;
- a metallic frame to support the tracker together with the detector prototypes (one or more) that are to be tested.

All the cables, the dividers and the cards were also fixed to the metallic frame, which was aligned so that the detectors were perpendicular to the beam line. Figure 1.5 on the next page shows the final setup. The signals from the scintillators were sent to three comparators connected to an **AND** port, the output signal of which was used as **trigger**. The trigger signal was sent to the *turbo₀* card, which acted as master and forwarded that signal to itself (on another input pin) and to the slave card *turbo₁*. The two *turbo* cards controlled and received inputs from the *VFAT* chips (both those on the tracking GEM detectors and those on the LG prototype), which collect and preamplify the detected signals.

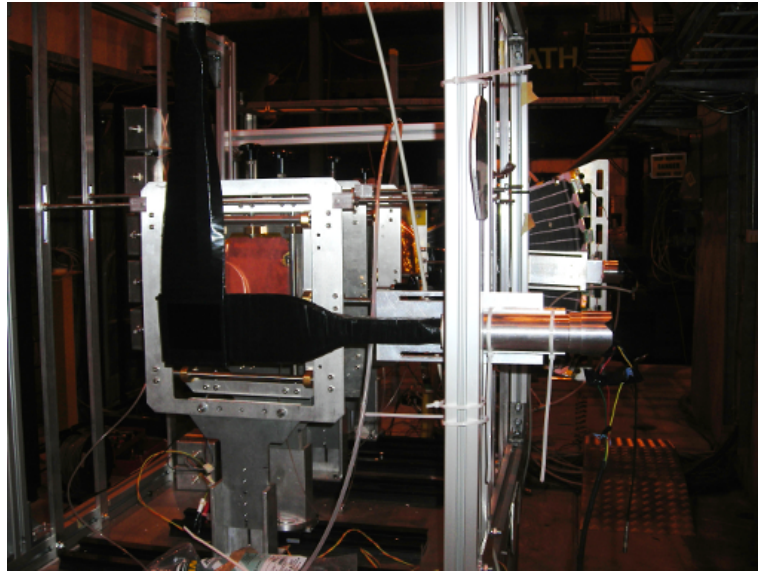
²The abbreviations stand for the CERN *Research and Development* group no. 51, and the *Gas Detector Development* group



(a) Tracker system setup



(b) Large GEM prototype setup



(c) View of the telescope

Figure 1.5: The test-beam experimental setup: a telescope made of scintillators, small GEM tracking chambers, and the Large GEM prototype

DAQ chain

1.5 Data analysis system

1.5.1 Hits, clusters and tracks

A **hit** on a detector occurs when one of its readout channels collects enough charge to exceed the threshold set in the readout chip. This happens when a particle has produced an avalanche in the multiplication area of the detector, but noise and *cross-talk* between adjacent readout channels can also produce a moderate number of hits.

We define **cluster** a set of adjacent hit channels along the x or y axis. A one-channel wide gap is allowed within the set. To make the test simpler, the tracks of the incident particles were only reconstructed if:

1. all of the tracker chambers showed one and only one cluster along the x axis;
2. the hit number in all the tracker chambers was ≤ 120 .

The track reconstructing algorithm exploits ROOT's class *TGraphErrors*. The distance along the z axis between the tracker chambers is measured; the x position of clusters is plotted versus z in a *TGraphErrors* object, and fitted with a first order polynomial via the ROOT *Fit()* function. This way, the fit function itself and its parameters (χ^2 , residuals, q and m) become available into the x -track object itself. The same procedure is used to define y -track objects.

If we define n_{act} as the number of actual hits and n_{exp} as the number of expected hits, then

$$\frac{n_{\text{act}}}{n_{\text{exp}}}$$

is the Large GEM's **efficiency**. n_{act} is incremented, and a simple histogram is updated, every time that the distance between the hit on the LG and the

projection on the LG of the corresponding track is minor than an assigned *efficiency radius*.

1.5.2 Beam profile reconstruction

A first check on the DAQ chain and the data analysis system efficiency was done by reconstructing the beam profile (which was known) from the tracker and LG data. Figure 1.6 on the facing page shows our first attempt at it:

1. we set a cut on the data: we requested that $\chi^2 < 10$ for every track, in both x and y directions, and low residuals for the hit positions ($\Delta x < 10channels$ and $\Delta y < 10channels$);
2. we plotted the position of the x clusters detected by the first tracker chamber (the *zero* is set at the lower left corner of it; since the strip pitch is $0.4mm$ the position is computed by making $0.4 * pos_{CL}$, where pos_{CL} is the position of the cluster expressed in channels number);
3. we plotted the hits on the LG channels to detect the most irradiated ones;
4. we plotted again the position of the x clusters detected by the first tracker chamber, each time requesting that the hit LG channel was one of the previously found ones.

This way we made a sort of puzzle which pieces represent the beam portion seen by one of the LG channels. It is like projecting the shadow of the LG pad corresponding to that channel on the tracker beam profile reconstruction.

We can see on Figure 1.6 on the next page that the LG pads fairly detected the beam profile. The difference in height of the two profiles (that from the tracker and that from the LG) is due to the fact that we did not include in the plot some of the adjacent LG channels, which indeed caught a little part of the beam.

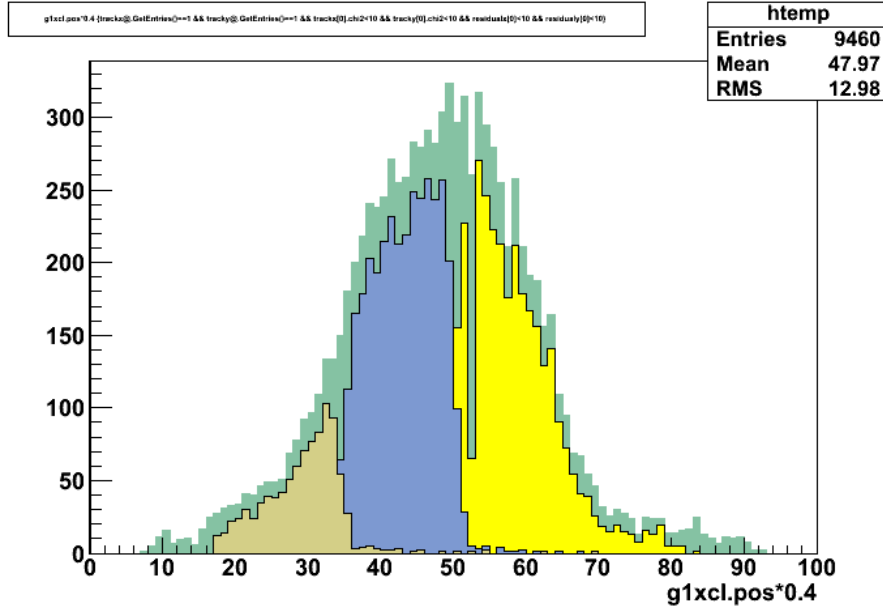


Figure 1.6: LG channels 699, 700 and 701 beam x profile reconstruction. The green background area is the beam x profile as seen by the first tracker chamber.

A two-dimensional plot can be done as well. A `GetX()` and a `GetY()` functions were defined to extract x and y pad position informations from the number of the corresponding LG channel. Figure 1.7 on the following page shows the two-dimensional distribution of the hits gathered by the prototype, in linear and logarithmic scales, as well as the tracker profiles for comparison.

The beam shown in Figure 1.7 on the next page is a muons beam, and thus it is quite large in both directions. When we dealt with a pions beam, it was sharper and 4 adjacent pads on the LG were enough to collect almost all the signal.

The beam profile reconstruction algorithm seems to work well, and so does the track reconstruction one.

Figure 1.7(b) on the following page gives an idea of the noise affecting the prototype: some channels look like hit while the beam was located in another area. Anyway, the scale of the plot is logarithmic, and in this case

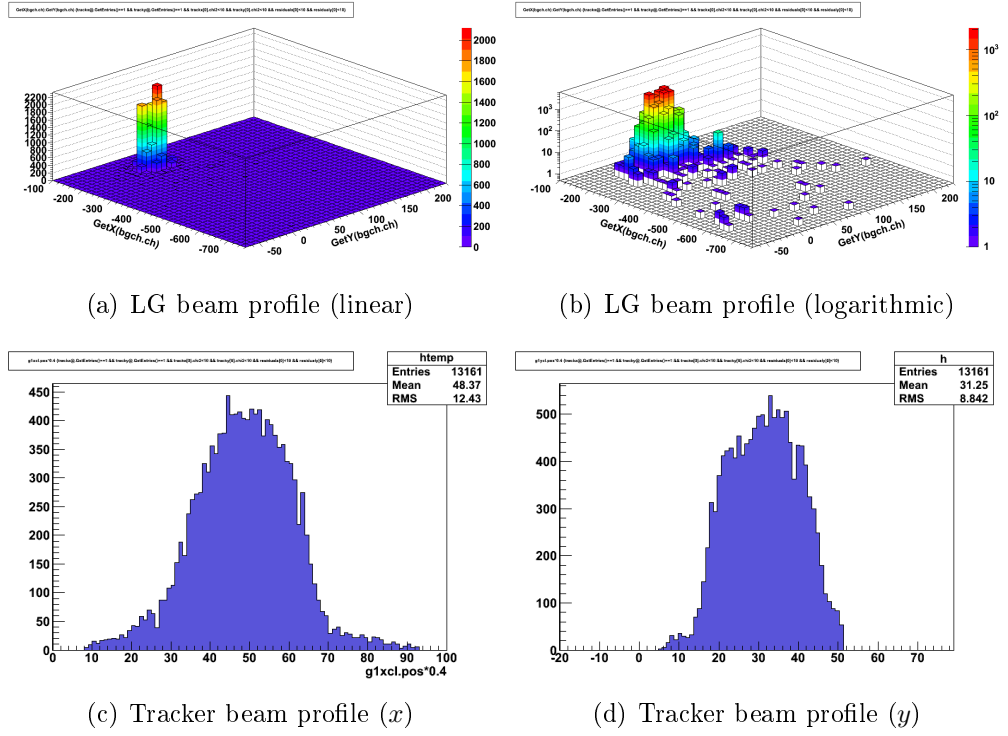


Figure 1.7: Two-dimensional beam profile reconstruction

the noise intensity is really negligible in respect to that of the beam.

Off beam analysis

2.1 Large GEM characterization

2.1.1 Gain curve

In order to measure the gain of the Large GEM detector we use a Cu X-Ray gun. The detector gain is the ratio between its output and input currents, that is:

$$G = \frac{I_{out}}{q * f}$$

where q is the charge collected by the first layer of the detector. In this setup q can be computed as:

$$q = n * e$$

where e is the electron charge and n is the average number of electrons produced in the drift region by the incident photon or particle; f is the interaction rate of the incident particles in the gas. Since the energies of the characteristic X-ray lines of the source, and the ionization energy of the GEM gas mixture are known, n is well determined. In our setup (Ar/CO₂ 70/30) we have:

$$n \simeq 320$$

After q is computed, we have to measure I_{out} and f in order to obtain G . We measure f by sending the LG output signals first to an amplifier, then

to a comparator, and at last to a counter. We connect together all the 128 pads of the readout line, and measure I_{out} by an amperometer.

2.1.2 Gas mixtures

2.2 Noise

2.2.1 Threshold scan

2.2.2 S-curve

2.2.3 Detector mapping

On beam analysis

3.1 Tracker efficiency

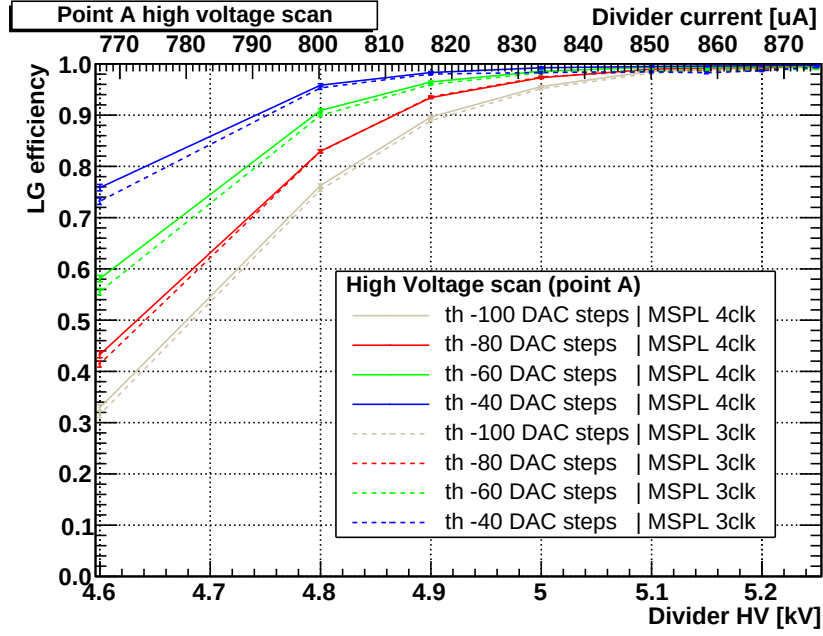
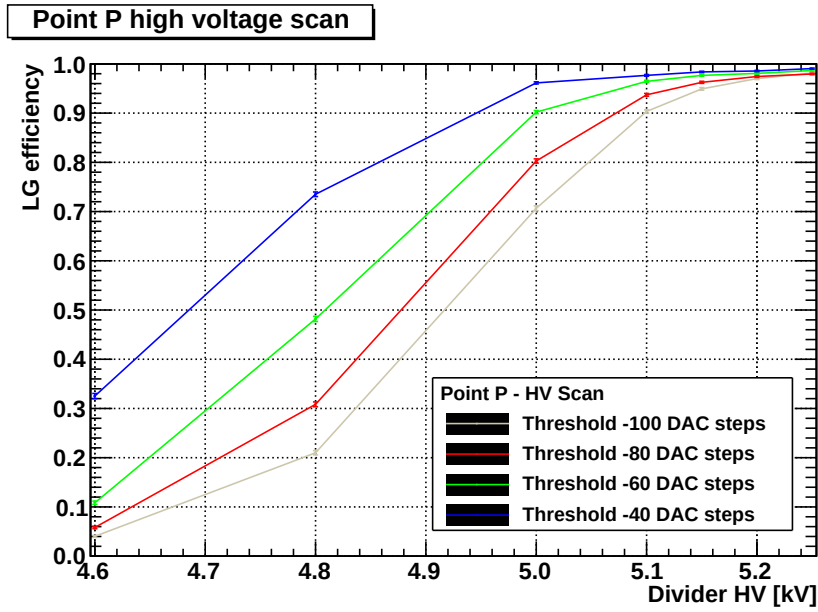
3.2 Prototype efficiency

3.2.1 High voltage scan

Efficiency was first detected as a function of the High-Voltage (subsequently referred to as **HV**) applied on the divider poles. These scans were performed focusing the muon beam on two regions of the chamber (P and A , whose readout plane areas are covered respectively with large and small pads), obtaining slightly different results. This was done to check the LG homogeneity, since it is a large area detector.

The gas mixture inside the detector was Ar/CO₂ in 70/30 proportions.

Four different thresholds were applied, in order to have a complete set of data to be used as reference for future comparison: -40 *DAC steps* (subsequently called *ds*), $-60ds$, $-80ds$ and $-100ds$. The lenght of the monostable pulse (subsequently referred to as **MSPL**) was set at 4 *clock cycles* (*clk*) to detect the signal as well as possible; in *point A* we also repeated the scan setting $MSPL = 3clk$. Figure 3.1 on the next page shows the results of

Figure 3.1: High voltage scan performed with beam on *zone A* padsFigure 3.2: High voltage scan performed with beam on *zone P* pads

the HV scan performed when the beam was centered on A, while Figure 3.2 shows the results for P.

Major efficiency at lower HV values is noticeable on the region made of smaller pads. This effect may be due to the capacitance of the pads themselves, which affects the signal where the pads are large. A study was done about this and will be presented later in this thesis.

In zone *A* we reached an efficiency of about 95% or higher, at thresholds $-40ds$ and $-60ds$, already with a divider current $I_D = 817\mu A$, and at $I_D = 850\mu A$ we got $\varepsilon \simeq 98\%$ for all the four threshold values. $MSPL = 3clk$ graphs do not diverge appreciably from the $MSPL = 4clk$ ones. Instead, in zone *P* the LG prototype approached the full efficiency for all thresholds only at $I_D \geq 866\mu A$, with $\varepsilon > 95\%$ at $I_D = 850\mu A$ only for $th = -40ds$ and $th = -60ds$ data sets.

We also made an HV scan after changing the detector gas mixture, as described in Chapter 3.2.3. We added CF_4 inside the detector, resulting in a lowering of the gain. Figure 3.3 on the next page shows a comparison of the prototype efficiency with its standard gas mixture (Ar/CO₂ 70/30) and with CF_4 (Ar/CO₂/CF₄ 60/20/20). I will come back on the purpose of this scan in Chapter 3.2.3. The loss of efficiency at lower current levels may be reduced by optimizing the divider in order to work with CF_4 , changing the internal electric fields of the detector.

3.2.2 Threshold scan

3.2.3 Timing scan

We worked on some time-performance scans as well, to check how fast was the prototype response, and what could be done to improve the signal timing. First we estimated the latency via the LabVIEW software we were using to interact with the VFAT chips, and it appeared to be around 18 *clock cycles*. Then we investigated all the latencies in the $[10clk, 19clk]$ interval with a set

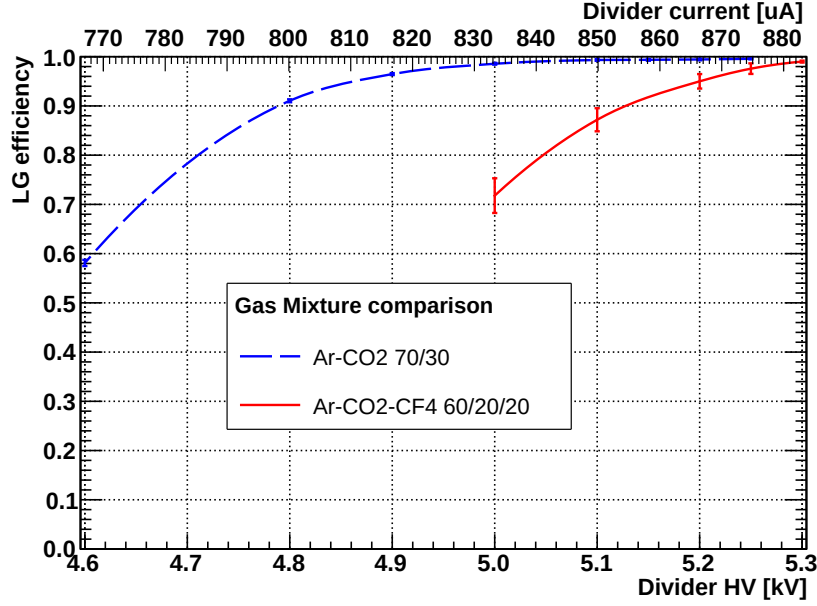


Figure 3.3: High voltage scan performed using an Ar/CO₂/CF₄ 60/20/20 gas mixture (same internal voltages and fields as for Ar/CO₂)

of acquisitions at different thresholds and MSP lengths.

Figure 3.4 on the next page shows all the data sets. Efficiency is plotted versus latency; indeed, the signal appears starting at $18clk$, and the efficiency approaches 90% only at low thresholds or at $MSPL \geq 3clk$.

We clearly get some noise at $th = -40ds$. The dotted lines show that the detector is acquiring signal even out of the right latency range: that signal is not related to the beam and it is caused by noise. This is confirmed by the increase of the hit counts when the MSPL lengthens: by setting a longer MSPL we just integrate the signal (both the "real" one and that due to the noise) over a longer time interval.

Increasing detectors time performance is a priority. The LHC should start working at very high frequency and intensity by the end of 2011. The spot where some TOTEM GEM detectors are inserted is very near to an interaction point, and therefore they run through extreme radiation conditions.

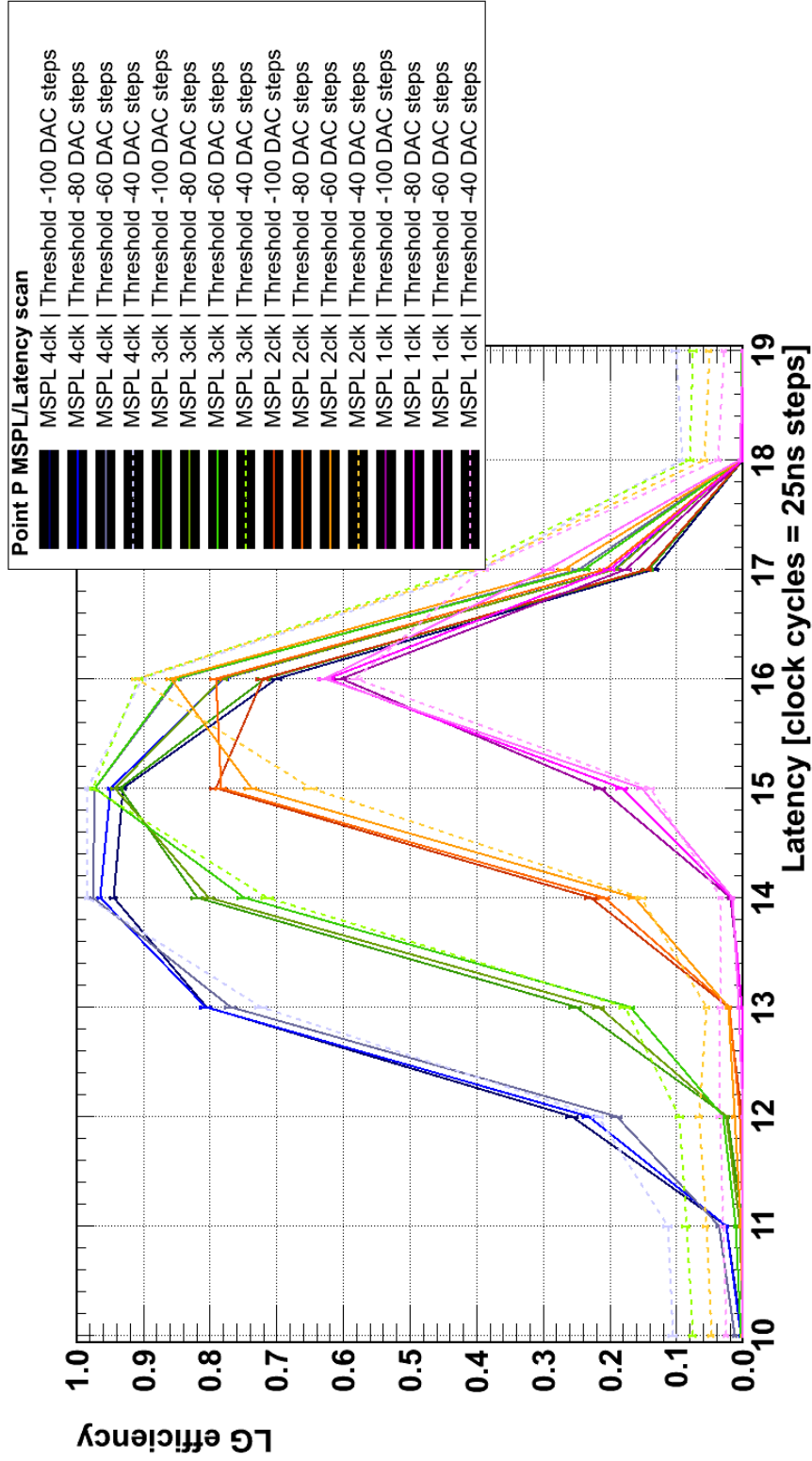


Figure 3.4: Latency scan at $MSPL = 4clk$, $MSPL = 3clk$, $MSPL = 2clk$ and $MSPL = 1clk$. Data sets were taken at four different threshold values, between $-100ds$ and $-40ds$, under a muons beam.

We made a test trying to find some new setup for those TOTEM detectors, which would allow them to have a faster response without changing the divider setup.

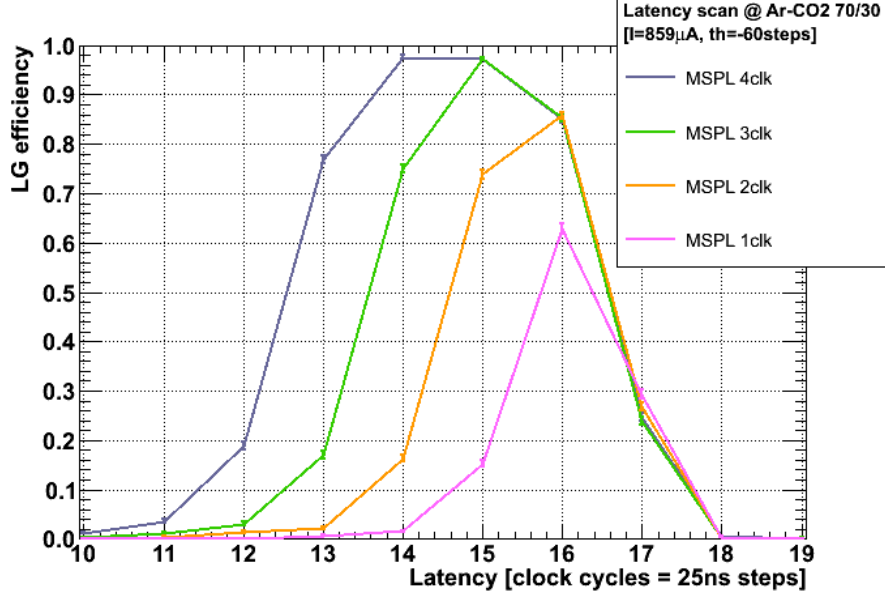
A percentage of CF_4 was added to the gas mixture inside the LG, getting to an $\text{Ar}/\text{CO}_2/\text{CF}_4$ 60/20/20 configuration. After the High-Voltage scan described in we repeated some of the latency scans, the results of which are compared to those we got with the previous gas mixture in Figure 3.5 on the facing page. We focused on $MSPL = 2clk$ tests since a length of $2clk$ would be fitting for TOTEM future runs purposes. Figure 3.5(b) on the next page clearly shows that if CF_4 improves the response speed of this kind of detectors: the new gas mixture allows to approach the full efficiency at $th = -60ds$ and $MSPL = 2clk$. On the other hand, we need to supply a little more current, which however might increase the discharge probability in a high-radiation environment; the LG prototype was not damaged by the HV we applied for these tests, but its tolerance is yet to be tested at those levels.

The fact that the detector was designed to work with Ar/CO_2 70/30 is to be noticed: by optimizing its divider for this new gas mixture we may improve the detector's time resolution without affecting its gain too much.

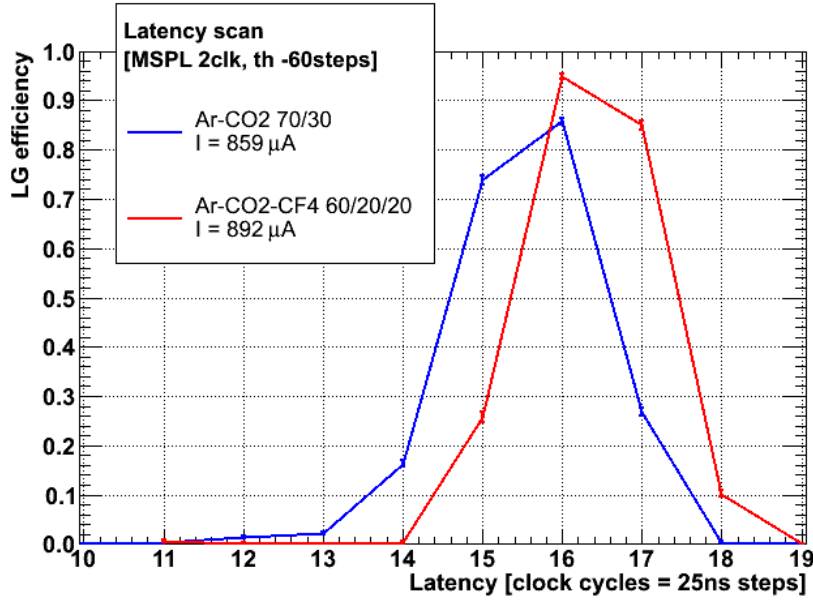
3.2.4 Behaviour with hadron beam

The efficiency of this detector proved to be higher when detecting hadrons, as was demonstrated by tests with pion beams (see Figures 3.6 on page 32 and 3.7).

Figure 3.6 shows that the pions HV scan was performed for different intensities of the beam (however, note that for some intensity-threshold combinations we only got few data). Figure 3.7 on page 32 only shows the data taken when the beam intensity was equal to 380.000 particles per spill and the threshold was set to $-60 DAC\ steps$, in order to compare two scans with the same settings.



(a) Standard gas mixture results

(b) Gas mixtures comparison at $MSPL = 2clk$ Figure 3.5: Data taken adding CF_4 to the standard Ar/ CO_2 70/30 gas mixture (same internal voltages and fields as for Ar/ CO_2)

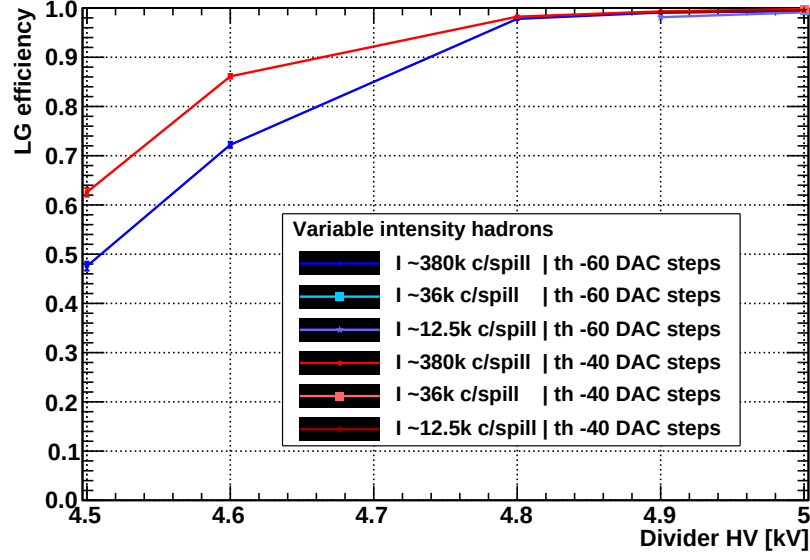


Figure 3.6: High-voltage scan under a beam of pions

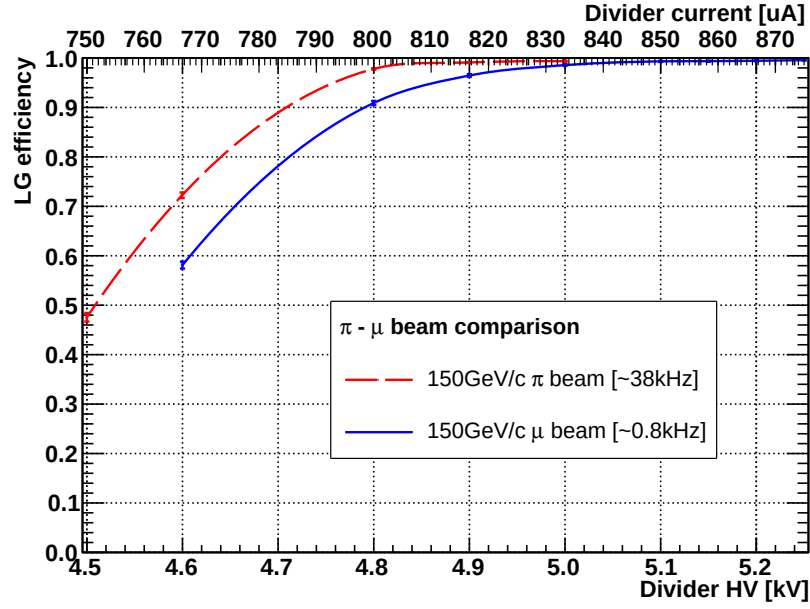


Figure 3.7: Detector behaviour: comparison between pions exposition and muons exposition

An effect of *charging up* occurred in this case: the pion beam was much more intense than the muon beam. Therefore some of the ions produced in the avalanche accumulated on the sides of the insulator holes, which are not perfectly straight, and strengthened the amplification field, raising the GEM foil gain with no need of increasing the external voltage. The high-level visible effect is an apparent raise of efficiency at constant divider HV.

Remarks

4.1 (In)homogeneity of the prototype

4.1.1 Chamber borders, spacer frame and foils junction zone

The availability of such a good tracking system let us highlight some defects of the prototype. When we direct a flux of charged particles over a detector, for some reasons the response may vary spatially. For instance, a pad may be disconnected from the corresponding pin on its VFAT chip; or we may have pointed the beam over a region of the detector containing a piece of the spacer frame.

Figures 4.1 on the following page and 4.2 on page 37 display a radiography of the LG prototype, made by moving the detector around in order to check the response of several areas (see Figure 1.4(b) on page 13). The graphs plot the two-dimensional beam profile as it was detected by the tracker, with the condition that there were hits on the LG in correspondence to those tracks. We spotted:

- *dead* (disconnected) pads: see Figures 4.1(d), 4.1(e) and 4.2(c);
- thin spacers: see Figures 4.1(b), 4.1(h), 4.2(b), 4.2(c), 4.2(d) and 4.2(f).

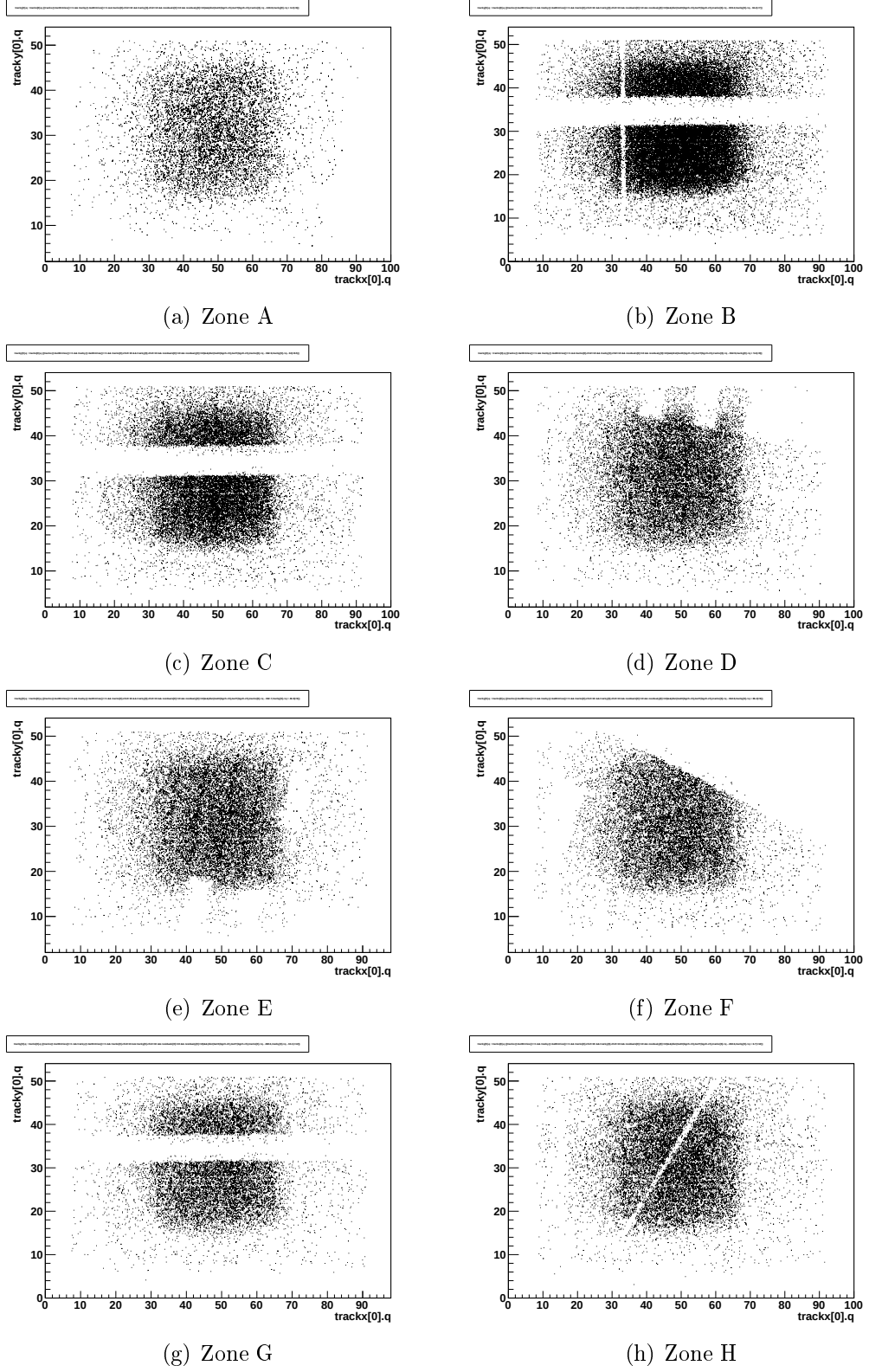


Figure 4.1: A radiography of the prototype triple GEM detector

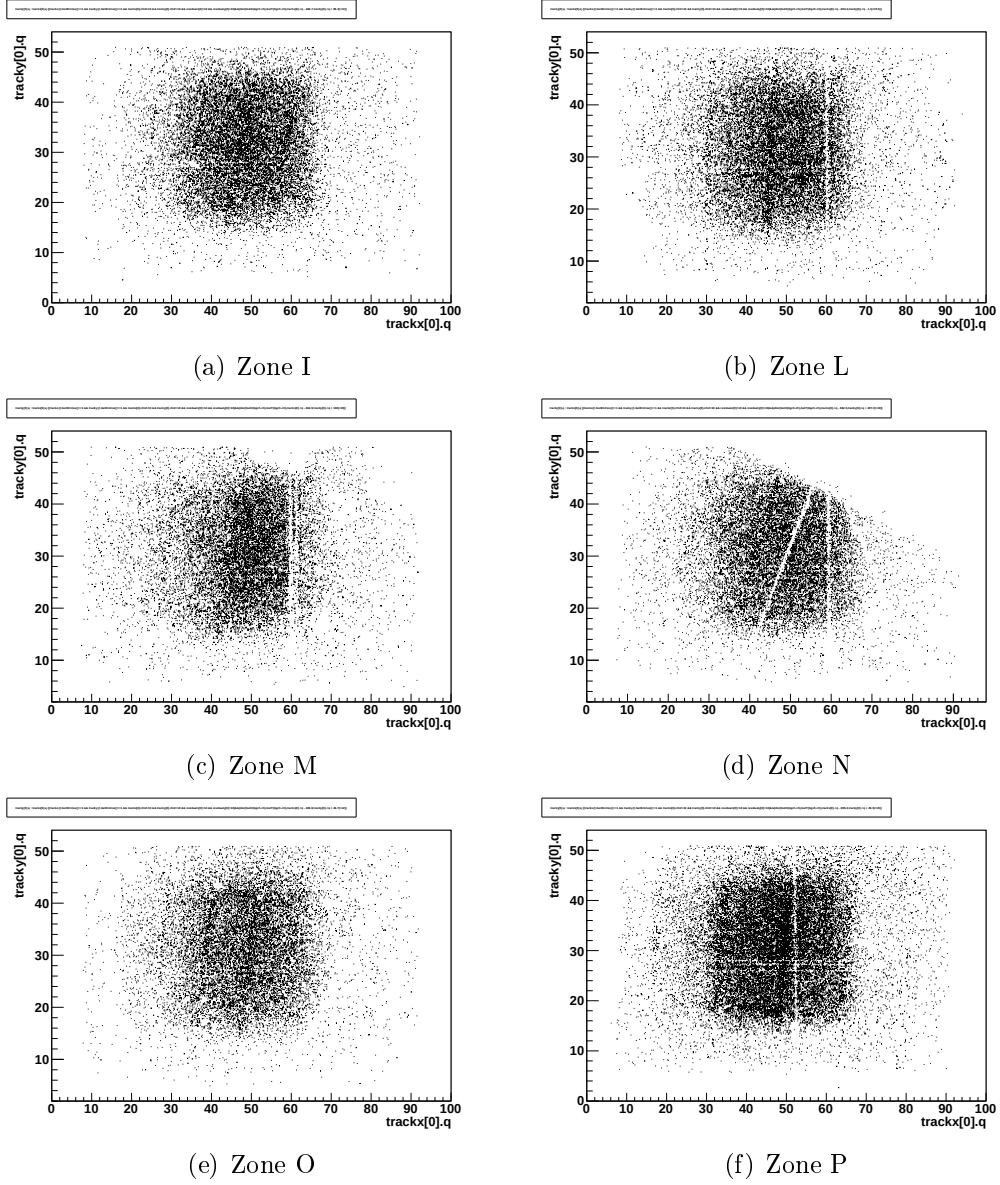


Figure 4.2: A radiography of the prototype triple GEM detector

In particular, Figure 4.2(c) shows a misalignment between a spacer and, probably, the edge of a cathode sector;

- segments of the thick central spacer covering the GEM foils seam: see Figures 4.1(b), 4.1(c) and 4.1(g);
- the edge of the chamber: see Figures 4.1(f) and 4.2(d).

A deeper study of the response of the detector in the junction area (Figure 4.3) shows the steepness of the slope in the curve of the efficiency as a function of y (which means, moving towards and across the spacer). This means that the low-efficiency area around the spacer is narrow, as one would require. No collateral malfunctionings were observed in spacer, border and junction areas.

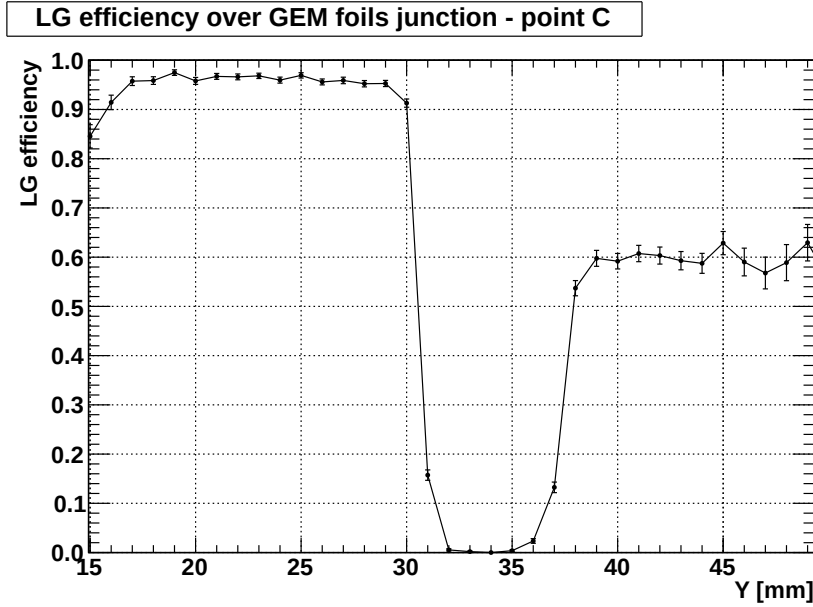


Figure 4.3: Loss of efficiency at the GEM foils junction

However, it also shows a new problem: the side of the detector which wasn't directly involved in the test beam looks less efficient. This effect is too high to be explained in terms of lack of *charging-up*, and it is visible in all the three junction zones we analyzed. We came across this matter after the

test beam period was finished and were not able to setup another irradiation test for it. A new gain curve is needed and will be computed as soon as possible with Cu X-Rays, for comparison with the absolute gain calibration made by S. D. Pinto [3].

4.1.2 Effects of the different dimensions of the pads

Larger copper pads should have bigger capacitance. Analyzing and comparing the two Figures 3.1 and 3.2 on page 26, one can imagine that the loss of efficiency in *zone P* is due to the bigger pads capacitance¹, bringing noise. An off-beam analysis was performed, injecting growing calibration pulses to all of the pads via the VFAT chips, and measuring the noise affecting them.

The **error function**, also known as **Gaussian error function**, is defined as:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

Once we have defined a threshold for the VFAT chips, and we start injecting calibration pulses, as long as they are smaller in ΔV than the threshold value we will not have counts. Then, like an **heaviside step function**, the counts would start as soon as the pulse height will overcome the threshold. Actually, it is not so simple: adding the contribution of the noise in the calibration pulses shapes (see Figure 4.4 on the next page), the certainty of overcoming the threshold at a certain charge value becomes a Gaussian probability, centered around the threshold value.

Evidently, the shape of the counts versus threshold plot becomes a so-called **S-Curve**, that corresponds to the shape of the **erf** function. Indeed, one can obtain the same plot if he integrates the Gaussian distribution of the noise for one VFAT channel, given that he knows the width of that distribution. In our case, we now know the noise distribution's *sigma* since

¹The pads at *point P* are the largest that we tested at this time; HV scans were only performed in *zone P* and *zone A*.

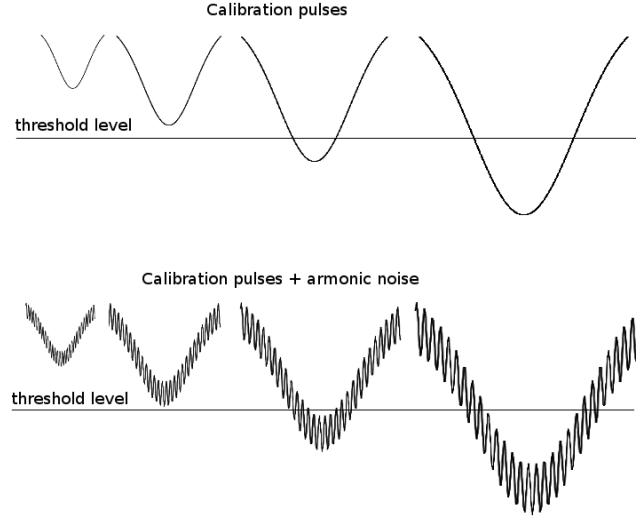


Figure 4.4: How a calibration pulse scan works

it is equal to the S-Curve standard deviation².

Figure 4.5(b) on the next page allows us to state that the lack of efficiency of *zone P* pads is not due to their parasitic capacitance: the bigger pads (those at the right side of the plot) do not show an appreciable increase of the noise. There should be another reason for the difference in response between *point P* and *point A*. Further tests still need to be performed.

The pad-based readout plane appears to work properly, and it could be a proper solution when the need for spatial resolution is moderate. A pad readout simplify the data analysis process, since one does not have to provide coincidence within more readout layers (as it is for strip-based systems) to understand where an event has occurred.

²Actually, we fit the S-Curve with an `erf` function. ROOT stores the fit functions as objects, and makes their parameters (mean, RMS, *sigma*, and so on) available via calls to the objects itself. To compute the *sigmas* of Figure 4.5(b) on the facing page, we fitted all of the S-Curves with `erf` functions and plotted their *sigma* parameters.

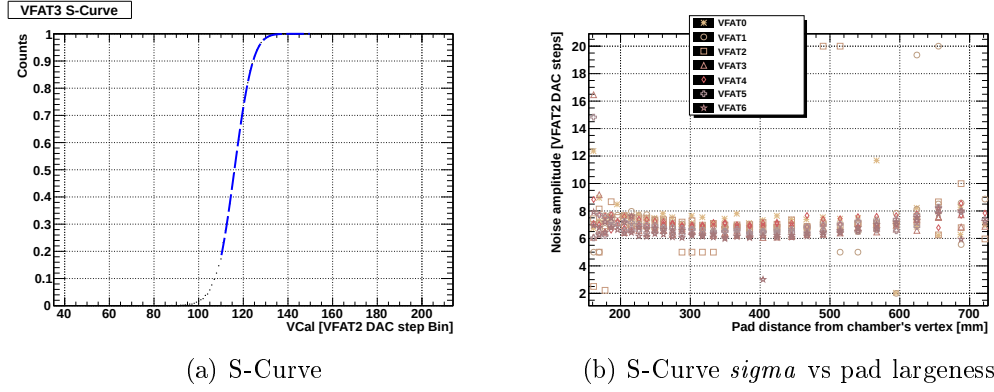


Figure 4.5: An S-Curve fitted by an `erf` function. On the right, the standard deviation (σ) of the pad's S-Curve is plotted as a function of the radial position of the pad in the detector, and thus as a function of its increasing largeness.

4.2 Data analysis system efficiency

4.2.1 Efficiency radius

In Chapter 1.5.1 on page 17 I explained how the efficiency is computed. If an hit occurs in the Large GEM within a fixed radius (an *efficiency radius*) from the projection of the track of the particle, then we say that the chamber has been efficient. If there is not any hit within the efficiency radius, it has been inefficient.

What happens if we set a wrong efficiency radius? There are two scenarios:

1. the radius is too short. In a large-pads area we may not include a whole pad within the radius, and the efficiency-computing algorithm would act wrong.
2. the radius is too long. Some noise hits on adjacent pads may be mistaken as efficient hits.

The average electron cloud produced in the avalanche has about a $3mm$ diameter, so we are sure to include all of the charge if we set a radius as long as it is enough to cover whole pad of the smallest one. Figure 4.6 on the next page shows the fluctuation of the computed efficiency, for every zone

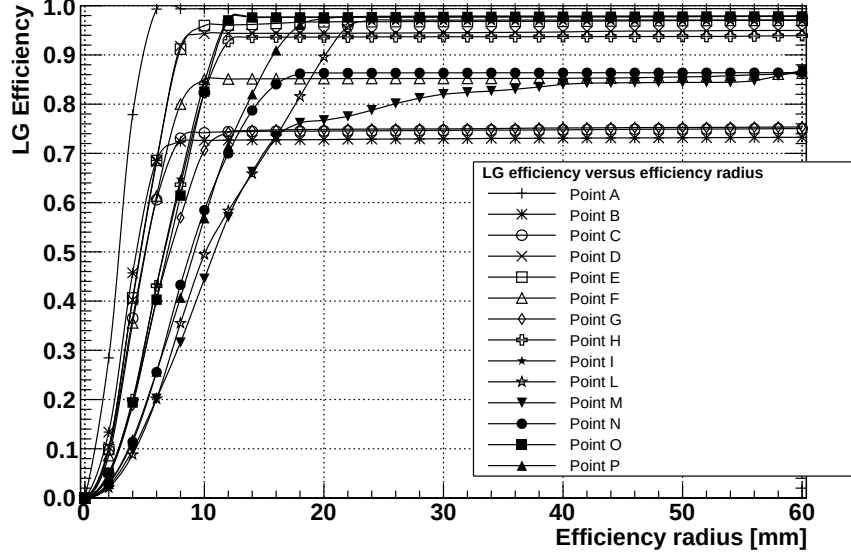


Figure 4.6: Efficiency radius scan: how changing the radius influence the efficiency computing algorithm

we checked, for different efficiency radius values. It appears then clear that $ef_{\text{rad}} \geq 20\text{mm}$ would be fitting for every scan.

Actually, all the result displayed in this text were computed trying to minimize the efficiency radius for each zone of the chamber, just to include the minimum possible amount of noise hits. For each set, the efficiency radius was set equal to the minimum of those for which the efficiency was in the *plateau* of the curve.

The plot of Figure 4.6 is also satisfactory because it shows that the noise level is low for each zone, with the exception of *point M*: the *plateaus* are very flat, which means that enlarging the radius does not mean including many noise hits. *Zone M* was probably adjacent to two noisy channels, one at a distance of about 20mm and the other, less noisy, about 35mm far, as one can see from the plot.

4.2.2 Cuts

Chapter 1.5.1 on page 17 points out that tracks are reconstructed only in some simple cases. In addition, during the analysis process some more conditions were requested for the track to be used. It was asked that:

1. there is exactly one *x-track* and one *y-track* per event;
2. $\chi^2 < 10$ for both the *x-track* and the *y-track*;
3. the distance between the hits on the tracker chambers and their first order polinomial fit defining the track is $< 10mm$ for every hit.

Inside a ROOT framework, this corresponds to declaring a **TCut** object to be used as mandatory option while selecting the data to process:

```
TCut goodtr("goodtr","trackx@.GetEntries()==1 && tracky@.GetEntries()==1 && trackx[0].  
chi2<10 && tracky[0].chi2<10 && residualx[0]<10 && residualy[0]<10")
```

Most of the analysis processes ran through ROOT's **Draw()** command, that accepts **TCut** objects as options. For the previously seen data sets, to acquire which we had to move the chamber 14 times, we needed to declare each time the position of the chamber in respect to the center of the beam. This was done via **TCut** objects, at the same time as setting the efficiency radius. The following declarations were used:

```
TCut Aeff("Aeff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 240.8,tracky[0]->q +  
6.3)<6")  
TCut Beff("Beff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 303.2,tracky[0]->q -  
34.4)<7")  
TCut Ceff("Ceff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 362.9,tracky[0]->q -  
34)<8.5")  
TCut Deff("Deff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 362.9,tracky[0]->q +  
5.6)<9")  
TCut Eeff("Eeff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 362.7,tracky[0]->q +  
46.9)<9")  
TCut Feff("Feff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 362.8,tracky[0]->q +  
86.2)<9")  
TCut Geff("Geff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 482.6,tracky[0]->q -  
34.1)<12")  
TCut Heff("Heff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 482.6,tracky[0]->q +  
6.7)<12")  
TCut Ieff("Ieff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 482.7,tracky[0]->q +  
86.3)<12")  
TCut Leff("Leff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 663.4,tracky[0]->q -  
1.1)<23.5")  
TCut Meff("Meff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 664.9,tracky[0]->q +  
108)<30")
```

```

TCut Neff("Neff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 662.5,tracky[0]->q +
207.3)<18")
TCut Oeff("Oeff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 483.8,tracky[0]->q +
46.7)<12")
TCut Peff("Peff","dist(GetX(bgch.ch),GetY(bgch.ch),trackx[0]->q - 695.2,tracky[0]->q +
49.3)<18")

```

where the first two numbers in each string represent the position of the chamber for that data set, and the last one is the best fitting efficiency radius.

This approach worked fine. The software did not run across any problem when the variables were declared this way, and the many cuts did not affect the analysis process, since the amount of data taken during the test-beam period was huge.

Chapter 5

Conclusions

5.1 Quality of the large GEM prototype

5.2 Large GEM detectors in TOTEM and CMS experiments

Appendices

Appendix A

ROOT analysis routines

Two young researchers¹ from **RD51** built the low-level macros used to convert the binary output of the detectors to much more physicist-friendly ROOT *n-tuples*.

My work only consisted of writing down some routines in order to extract efficiency values from specific runs' *n-tuples*, and make the data accessible via plots. The code I wrote is displayed in the following pages.

Listing A.1: *Builder.C*

```
#include <string>
#include <iostream>
#include <TTree.h>
#include <TFile.h>

int EventBuilderVFAT(const char* rawfilename,
                    const char* rootfilename,
                    const int readmaxevent);

using namespace std;

void Builder(string rawfile_address){

    size_t spos;
    size_t substring_lenght;
    string slash = "/";
    spos = rawfile_address.rfind(slash);
    substring_lenght = 7;

    // Extracts the string "Run###" from the rawfile name
    string rawfile_name = rawfile_address.substr(spos+5,substring_lenght);
    // Sets the output file name
    string rootfile_address = "../RootData/" + rawfile_name + ".root";
```

¹Matteo Alfonsi (CERN) and Gabriele Croci (PhD student at University of Siena)

```

// Builds a .root file from a rawfile
EventBuilderVFAT(rawfile_address.c_str(),rootfile_address.c_str(),100000000);
// Sets the reco file name
string recofile_address = "../RootData/" + rawfile_name + "_reco.root";
// Creates a reco file using Offset_Settings.txt
TFile file0(rootfile_address.c_str());
TTree* t = dynamic_cast<TTree*>(file0.Get("rd51tb"));
t->Process("Reco2d_ps.C",recofile_address.c_str());
}

```

Listing A.2: *effex2.C*

```

#include "TFile.h"
#include "TTree.h"
#include "TH1.h"
#include <iostream>

using std::cout;

double effex2(TFile& file0){

    TH1F* heffbgch = dynamic_cast<TH1F*>(file0.Get("heffbgch"));
    double efficiency;
    efficiency = 1 - heffbgch->GetBinContent(1) / heffbgch->GetEntries();
    return efficiency;
}

```

Listing A.3: *efficiency.cc*

```

#include <string>
#include <iostream>
#include <sstream>
#include <TTree.h>
#include <TFile.h>
#include <cstdio>
#include "TVectorT.h"

using namespace std;

double effex2(TFile& file0);

TVectorD efficiency(long int run_number, long int last_run){

    Int_t size = last_run - run_number;
    TVectorD nRun (size+1);
    TVectorD run_eff (size+1);
    Int_t count = 0;

    for (long int i=run_number;i<last_run+1;i++)
    {
        string s;
        char run_n[30];

        sprintf(run_n,"%ld",i);
        s = string(run_n);

        string rootfile_name;
        string recofile_name;

        if (i < 10)
        {
            rootfile_name = "../RootData/Run000" + s + ".root";

```

```

        recofile_name = "../RootData/Run000" + s + "_reco.root";
    }
    else
    {
        if (i < 100)
        {
            rootfile_name = "../RootData/Run00" + s + ".root";
            recofile_name = "../RootData/Run00" + s + "_reco.root";
        }
        else
        {
            rootfile_name = "../RootData/Run0" + s + ".root";
            recofile_name = "../RootData/Run0" + s + "_reco.root";
        }
    }

    TFile file0(recofile_name.c_str());
    double eff = effex2(file0);
    run_eff[count] = eff;
    nRun[count] = i;
    count++;

}

return run_eff;
}

```

Listing A.4: *EffRadius.C*

```

#include <stdlib.h>
#include <string>
#include "TVectorT.h"
#include <iostream>
#include <sstream>
#include <TTree.h>
#include <TFile.h>
#include <TH1F.h>
#include <TGraph.h>
#include <TAxis.h>
#include <TCut.h>
#include <stdio.h>
#include <cstdlib>
#include "TotemMap.hpp"

using namespace std;

TVectorD EffRadius(long int run_number, double xoffsetmm, double yoffsetmm, long int
    minrad, Int_t nsteps){
    TVectorD rad_eff (nsteps+1);
    string run; //Run number to be converted to string
    char run_n[30];
    string offsetx; //Offsets to be converted to strings
    string offsety;

    sprintf(run_n,"%ld",run_number); //Converting run number to a string
    run = string(run_n);
    { //Converting x offset to string
        ostringstream xx;
        xx << xoffsetmm;
        offsetx = xx.str();
    }
    { //Converting y offset to string
        ostringstream yy;
        yy << yoffsetmm;
        offsety = yy.str();
    }
}

```

```

/*
End converting numbers to strings *****
*/

string rootfile_name;
string recofile_name;

if (run_number < 10)
{
    rootfile_name = "../RootData/Run000" + run + ".root";
    recofile_name = "../RootData/Run000" + run + "_reco.root";
}
else
{
    if (run_number < 100)
    {
        rootfile_name = "../RootData/Run00" + run + ".root";
        recofile_name = "../RootData/Run00" + run + "_reco.root";
    }
    else
    {
        rootfile_name = "../RootData/Run0" + run + ".root";
        recofile_name = "../RootData/Run0" + run + "_reco.root";
    }
}

TFile file0 (rootfile_name.c_str());
TTree* t = dynamic_cast<TTree*>(file0.Get("rd51tb"));
t->AddFriend("recotree",recofile_name.c_str());

//Selecting high quality tracks and events
TCut goodtr("goodtr","trackx@.GetEntries()==1 && tracky@.GetEntries()==1 &&
trackx[0].chi2<10 && tracky[0].chi2<10 && residualx[0]<10 && residually[0]<10");
Int_t count = 0;

//Scans radius from mirad to mazrad every 5 millimeters
for (long int i = minrad; i < 2*nsteps + 1; i = i + 2)
{
    string irad; //Converts current radius to string
    ostringstream rad;
    rad << i;
    irad = rad.str();

    string draw_string = "Sum$((dist(GetX(bgch.ch),GetY(bgch.ch),(trackx[0]->q
- trackx[0]->m*400" + offsetx + "),(tracky[0]->q -tracky[0]->m*400" + " + offsety +
")))<" + irad + ") >>h(5,0,5)";

    t->Draw(draw_string.c_str(),goodtr);

    TH1F *h = (TH1F*)gDirectory->Get("h");

    Double_t ieff = 1. - h->GetBinContent(1)/h->GetEntries();

    rad_eff[count] = ieff;
    count++;
}

return rad_eff;
}

```

Listing A.5: *nevents.cc*

```

#include <string>
#include <iostream>

```

```

#include <sstream>
#include <TTree.h>
#include <TFile.h>
#include <TH1.h>
#include <cstdio>
#include "TVectorT.h"

using namespace std;

TVectorD nevents(long int run_number, long int last_run){

    Int_t size = last_run - run_number;
    TVectorD n_events (size+1);
    Int_t count = 0;

    for (long int i=run_number;i<last_run+1;i++)
    {
        string s;
        char run_n[30];

        sprintf(run_n,"%ld",i);
        s = string(run_n);

        string rootfile_name;
        string recofile_name;

        if (i < 10)
        {
            rootfile_name = "../RootData/Run000" + s + ".root";
            recofile_name = "../RootData/Run000" + s + "_reco.root";
        }
        else
        {
            if (i < 100)
            {
                rootfile_name = "../RootData/Run00" + s + ".root";
                recofile_name = "../RootData/Run00" + s + "_reco.root";
            }
            else
            {
                rootfile_name = "../RootData/Run0" + s + ".root";
                recofile_name = "../RootData/Run0" + s + "_reco.root";
            }
        }

        TFile file0(recofile_name.c_str());

        TH1F* heffbgch = dynamic_cast<TH1F*>(file0.Get("heffbgch"));
        n_events[count] = heffbgch->GetEntries();
        count++;

    }

    return n_events;
}

```

Listing A.6: *Recoizer.C*

```

#include <string>
#include <iostream>
#include <TTree.h>
#include <TFile.h>

using namespace std;

void Recoizer(string rootfile_name){

```

```
string run_number = rootfile_name.substr(15,4);

// Sets the reco file name
string recofile_name = "../RootData/Run" + run_number + "_reco.root";

// Creates a reco file using Offset_Settings.txt
TFile file0(rootfile_name.c_str());
TTree* t = dynamic_cast<TTree*>(file0.Get("rd51tb"));
t->Process("Reco2d_ps.C+", recofile_name.c_str());
}
```

Bibliography

- [1] Review of particle physics, 2008. Particle Data Group.
- [2] R. Bouclier, M. Capeáns, W. Dominik, M. Hoch, J. C. Labbé, G. Million, L. Ropelewski, F. Sauli, and A. Sharma. The gas electron multiplier (GEM).
- [3] S. D. Pinto, M. Alfonsi, I. Brock, G. Croci, E. David, R. de Oliveira, B.-E. Pinchasik, L. Ropelewski, F. Sauli, and M. van Stenis. A large area GEM detector. In *IEEE Nuclear Science Symposium Conference*, 2008.
- [4] F. Sauli. Principles of operation of multiwire proportional and drift chambers. Academic training programme lectures, CERN, 1977.