UNIVERSITÀ DEGLI STUDI DI PISA

DIPARTIMENTO DI FISICA 'E. FERMI'

Tesi di Laurea Magistrale in Scienze Fisiche
Curriculum 'Fisica delle Interazioni Fondamentali'
Ottobre 2013

A GPU-based real time trigger for rare kaon decays at NA62

Candidato:
Elena Graverini

Relatore:
Prof. Marco Sozzi

Anno Accademico 2012–2013

Contents

Abstract

This thesis reports a study for a new real-time trigger for the NA62 experiment based on Graphical Processing Units (GPUs).

The NA62 experiment was devised to study with unprecedented precision the ultra-rare decay $K^+ \to \pi^+\nu\bar{\nu}$, a process mediated by Flavour-Changing Neutral Currents (FCNC) whose exceptional theoretical cleanliness provides a unique probe to test the Standard Model. The use of a high-rate kaon beam will result in an event rate of about 15 MHz, so high that it will be impossible to store data on disk without an efficient selection. The experiment therefore devised three trigger levels, allowing to reduce the data rate fed to the readout PC farm down to $\sim$10 kHz.

For this thesis I developed an online trigger algorithm that uses data fed by the RICH (Ring Imaging CHerenkov counter) detector in real-time to allow a rejection of the dominant background $K^+ \to \pi^+\pi^0$ based on kinematical constraints.

As a starting point for the development of this algorithm, I verified the feasibility of such a trigger through Montecarlo simulations. I measured the reconstruction resolution, achieved by the RICH detector alone, of the kinematical variables used for the event selection. After that, I analysed the background rejection power and the signal efficiency of several kinematical constraints, and I designed an actual trigger algorithm.

The necessity of running the algorithm in real-time, with a maximum latency of 1 ms per event, drove the choice of exploiting the parallel computing power of GPUs. A parallelized algorithm was therefore developed, that can fit up to 4 Cherenkov rings per event. Moreover, a large number of events are processed concurrently. No parallelized and seedless multi-ring fitting algorithm existed before.

The developed algorithm consists of a pattern recognition stage, to assign the hits to up to 4 ring candidates, and of a robust single-ring fit routine. The program was tested on GPUs, and its performance and execution latency proved to be compatible with the requirements.

This work proves that alternative trigger designs are possible for the NA62 experiment, and represents a starting point for the introduction of flexible GPU-based real-time triggers in High Energy Physics.

Sommario

Questa tesi costituisce uno studio per un algoritmo di *trigger* in tempo reale basato su GPU (Graphical Processing Units) per l'esperimento NA62.

NA62 è un esperimento progettato per misurare con precisione il decadimento ultra raro $K^+ \to \pi^+\nu\bar{\nu}$, un canale mediato da correnti neutre *flavour-changing* estremamente sensibile all'eventuale presenza di nuova fisica. L'elevato *rate* di eventi rivelati, dell'ordine di 15 MHz, non permetterà una archiviazione su disco dei dati non moderata da severi criteri di selezione. Sono perciò necessari dei livelli di *trigger* che consentano di ridurre il *rate* di eventi salvati fino a circa una decina di kHz.

L'algoritmo sviluppato si basa sull'uso del rivelatore RICH (*Ring Imaging CHerenkov counter*). Le informazioni primitive inviate dal RICH vengono valutate in tempo reale, per produrre una decisione di *trigger* basata prevalentemente su considerazioni di cinematica.

In una prima fase ho verificato, tramite simulazione Montecarlo, la fattibilità e significatività di tale progetto. Ho dapprima misurato la risoluzione sulla ricostruzione di alcune quantità cinematiche ricavate utilizzando unicamente il rivelatore RICH, poiché per un *trigger* di primo livello in tempo reale non sarà possibile mettere in relazione dati forniti da rivelatori diversi. Ho studiato poi fino a che livello fosse possibile separare il segnale dal fondo, misurando l'efficienza di reiezione e l'accettanza per il segnale al variare di alcuni parametri di selezione.

Data la necessità di eseguire il programma in tempo reale, con una latenza massima di 1 ms per evento, si è deciso di sfruttare il potere computazionale parallelo proprio delle GPU (processori grafici ad elevato parallelismo). E' stato quindi sviluppato un algoritmo in grado di eseguire simultaneamente non solo le istruzioni relative ad eventi diversi, ma anche i *fit* di fino a 4 anelli Cherenkov diversi appartenenti allo stesso evento. Nessun algoritmo parallelo e *seedless* di questo tipo esisteva in letteratura.

L'algoritmo implementato è composto di due parti: una iniziale di riconoscimento di *pattern*, che estrae il numero di anelli presenti nella matrice ed identifica gli *hit* appartenenti a ciascuno di essi, ed una di *fit* dei singoli cerchi. Il programma è stato testato su GPU, ed efficienza e tempi di esecuzione risultano compatibili con le richieste. Questo lavoro apre la possibilità di implementare *trigger* alternativi e flessibili per NA62 e rappresenta un primo esempio prototipale dell'uso di GPU in tempo reale.

Introduction

This thesis deals with two different aspects of the same project. A preliminary stage of validation and characterisation was in fact necessary before starting the development of an actual trigger algorithm. Therefore, this document is organized in parts. My own work is described in Parts II and III.

Part I presents the NA62 experiment, in the contest of which this work was carried out. Chapter 1 outlines the main Physics goal, that is the study of the ultra-rare decay $K^+ \to \pi^+ \nu\bar{\nu}$. The theoretical framework and Physics sensitivities of this process are discussed, and a brief report is given about the previous searches for this decay. Chapter 2 focusses on the NA62 detector, discussing the characteristics of the high-energy beam used to provide kaon decays, and analyses the purpose and layout of the various sub-detectors.

In Part II I describe the results of my feasibility study, with an evaluation of the background rejection efficiency of a trigger based on the identification of Čerenkov rings. Chapter 3 describes the trigger chain for the NA62 experiment, and discusses the benefits of using commercial graphic processors (GPUs) at the earliest trigger stage. In this chapter I also set and describe the objectives of my thesis work. In Chapter 4 I introduce the software framework used for all simulations. The reconstruction procedure I developed to extract physics information from RICH data is discussed in detail. The outcome of this work is reported in Chapter 5, where I explore several possible designs for a direct rejection trigger for $\pi^+\pi^0$ events.

Finally, in Part III I describe how I implemented the actual GPU trigger, and report on its efficiency and timing tests. Chapter 6 presents a general description of the multi-ring algorithm we designed, divided in a pattern recognition step and a single-ring fit, and reports the preliminary tests I did

in order to choose the best single-ring fitting algorithm available. In Chapter 7 I briefly introduce the CUDA toolkit, and I go through the technical details of the GPU algorithm, including some pieces of code. Tests of the program functioning and timing have been performed, whose results are reported and examined in Chapter 8. There I also sum up the present status of this work and discuss the conclusions which may be drawn from this experience.

I thought it useful to devote some pages to the mathematical analysis of the main single-ring fitting methods I used. Appendix A is a brief introduction to the problem of circular regression, and describes two different algebraic algorithms. The code with which I implemented the chosen fitting method in the trigger algorithm is available in Appendix B. Finally, Appendix C lists useful specifications of the specific graphic card used to implement and test the trigger.

Part I

The NA62 experiment

1 The NA62 experiment

Contents

The NA62 experiment was devised to study the ultra-rare decay $K^+ \to \pi^+\nu\bar{\nu}$ at the SPS (Super Proton Synchrotron) at CERN.

This process was first observed at BNL in the dedicated experiments E787 and E949 (1997-2001), and its branching ratio was later measured to be $(1.73^{+1.15}_{-1.05})\,10^{-10}$ [9]. The exceptional theoretical cleanliness of this decay channel makes it extremely attractive to test the Standard Model predictions and probe the existence of new Physics.

NA62 was designed to collect the higher statistics ever obtained for K^+ decay events, allowing for a 10% measurement of the branching ratio of the $K^+ \to \pi^+\nu\bar{\nu}$ decay with a 10:1 signal to background ratio.

1.1 Physics objectives

The decays $K^+ \to \pi^+\nu\bar{\nu}$ and $K_L^0 \to \pi^0\nu\bar{\nu}$, the latter being studied by the KOTO experiment in Japan [37], are unique probes to test the Standard Model.

Both decays are *Flavour-Changing Neutral-Current* (FCNC) processes, a kind of transition that is strongly suppressed in the Standard Model and therefore very sensitive to SM extensions or new Physics scenarios, as summarised in Figure 1.1. This plot reports the branching ratios predicted by the Standard Model and by BSM (Beyond the Standard Model) theories such as the *Constrained Minimal Flavour Violation* effective theory, the *Minimal Supersymmetric Standard Model* and the *Four-Generation* model [28]. A 10% measurement of $B.R.(K^+ \to \pi^+\nu\bar{\nu})$ alone would rule out a large set of models.

Moreover, the above processes feature an exceptional cleanliness and precise theoretical prediction: in the NA62 case, the Standard Model prediction is B.R.$(K^+ \to \pi^+\nu\bar{\nu}) = \left(7.81^{+0.80}_{-0.71\,\mathrm{CKM}} \pm 0.29\right)\ 10^{-11}$ [16], where the first error accounts for the uncertainty on the CKM matrix elements, and the second one is the pure theoretical uncertainty. Any deviation from this expectation would be an evidence for new Physics. These decay channels are thus extremely sensitive, and represent a powerful tool to probe theories beyond the Standard Model (Figure 1.1) [35, 46].

Both processes arise from a $s \to d\nu\bar{\nu}$ transition at quark level, and can be described by "penguin" Feynman diagrams (1-loop diagrams). If a discrepancy were detected between experimental data and SM predictions, this would be a hint for the existence of unknown particles intervening in the loop. We will discuss further in Section 1.2 the theoretical framework behind this kind of transition.

In case of agreement with the Standard Model, instead, a precise measurement of the branching ratio of this process would improve the accuracy of the current experimental determination of the $|V_{td}|$ parameter, which is one of the least precisely known elements of the CKM matrix. This particular measurement would be independent from those already obtained in the analysis of the B system [29, 38, 59].

1.2 Theoretical framework

In this section the CKM matrix is introduced, whose element V_{td} we are going to investigate by measuring the branching ratio of the $K^+ \to \pi^+\nu\bar{\nu}$ process.

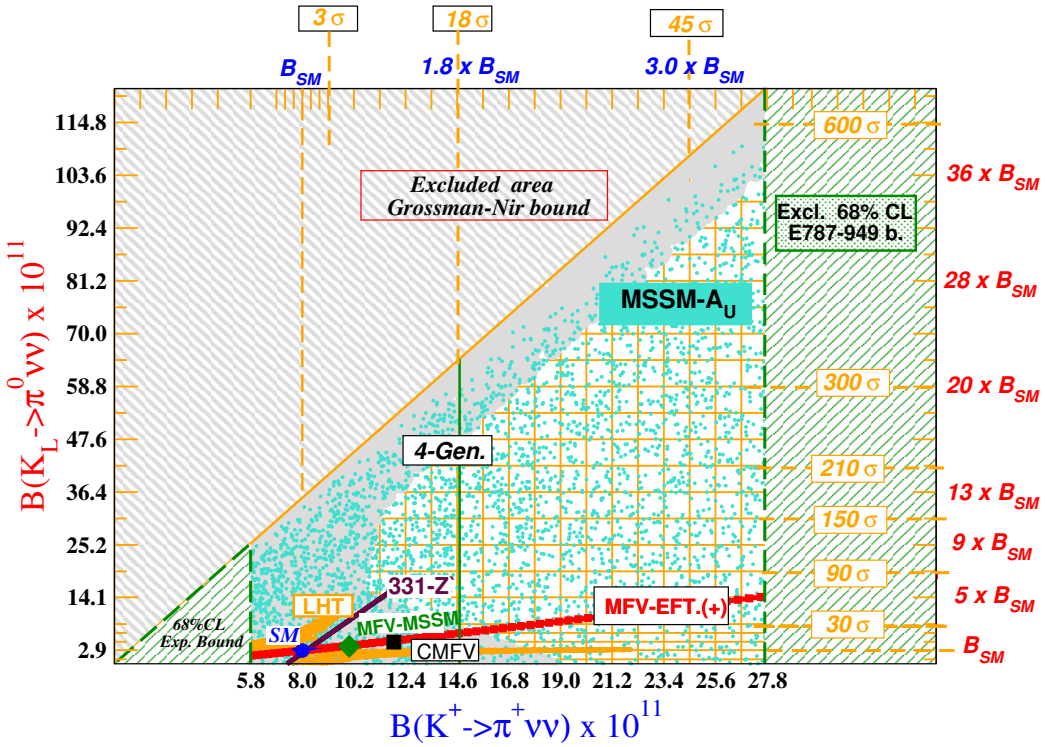


Figure 1.1: The "Mescia-Smith" plot [28] illustrating $K^+ \to \pi^+\nu\bar{\nu}$ and $K_L^0 \to \pi^0\nu\bar{\nu}$ Physics sensitivities. Experimental data on the branching ratios of these ultra-rare decays would allow to put SM and BSM theories at a test.

The CKM framework provides an extension to the Cabibbo 2x2 matrix, that encodes how flavour-changing charged currents ($W^{\pm}$) couple u, c and d, s quark states [20]. The coupling is described by means of the intermediate states d' and s' obtained from the mass eigenstates d and s through a rotation by an angle θ_C:

$$\begin{pmatrix} d' \\ s' \end{pmatrix} = \begin{pmatrix} \cos\theta_C & \sin\theta_C \\ -\sin\theta_C & \cos\theta_C \end{pmatrix} \begin{pmatrix} d \\ s \end{pmatrix} \tag{1.1}$$

This way, Cabibbo described the quark mixing using a single real parameter θ_C, named the Cabibbo angle. Information from the experiments that probed quark flavour transitions yielded the result $\theta_C \simeq 13°$.

The Cabibbo-Kobayashi-Maskawa (CKM) theory generalizes the Cabibbo matrix including also the quark states b, t from the third generation [39]:

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix} \begin{pmatrix} d \\ s \\ b \end{pmatrix} \tag{1.2}$$

The current status of experimental results about quark-mixing processes yields the following evaluations [50]:

$$\begin{pmatrix} |V_{ud}| = 0.97425 \pm 0.00022 & |V_{us}| = 0.2252 \pm 0.0009 & |V_{ub}| = (4.15 \pm 0.49)\,10^{-3} \\ |V_{cd}| = 0.230 \pm 0.011 & |V_{cs}| = 1.006 \pm 0.023 & |V_{cb}| = (40.9 \pm 1.1)\,10^{-3} \\ |V_{td}| = (8.4 \pm 0.6)\,10^{-3} & |V_{ts}| = (42.9 \pm 2.6)\,10^{-3} & |V_{tb}| = 0.89 \pm 0.07 \end{pmatrix} \tag{1.3}$$

From the above values, which were obtained by averaging various measurements, it can be inferred that the diagonal elements are clearly dominant. The most favoured transitions are indeed those between quarks belonging to the same family, i.e. $u \leftrightarrow d$, $c \leftrightarrow s$ and $t \leftrightarrow b$. Transitions between quarks from different families are instead suppressed at various levels.

Unlike the Cabibbo matrix, the CKM matrix does not represent a pure rotation, as it also includes complex parameters. It can be indeed rewritten in the following form, making use of the Wolfenstein parametrisation [71]:

$$V_{CKM} = \begin{pmatrix} 1 - \lambda^2/2 & \lambda & A\lambda^3(\rho - i\eta) \\ -\lambda & 1 - \lambda^2/2 & A\lambda^2 \\ A\lambda^3(1 - \rho - i\eta) & -A\lambda^2 & 1 \end{pmatrix} + \mathcal{O}(\lambda^4) \tag{1.4}$$

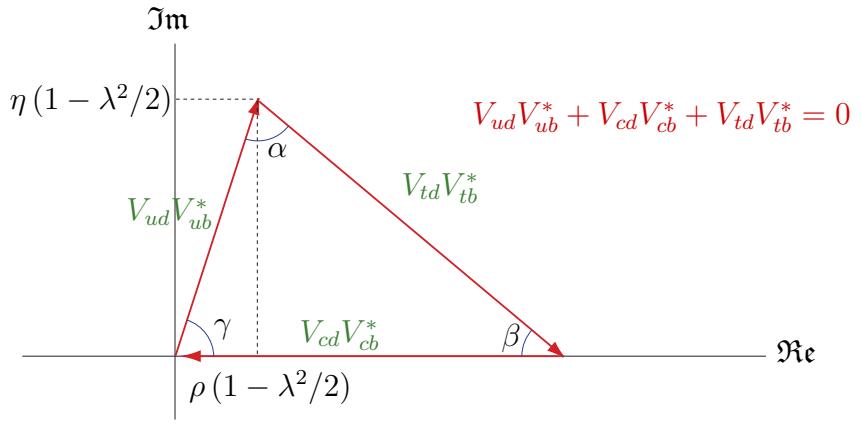


Figure 1.2: Unitarity triangle defined by the relation $V_{ud}V_{ub}^* + V_{cd}V_{cb}^* + V_{td}V_{tb}^* = 0$ in the complex plane.

where

$$A, \lambda > 0 \tag{1.5}$$
$$\lambda = \sin\theta_{12} = \sin\theta_C \tag{1.6}$$
$$A\lambda^2 = \sin\theta_{23} \tag{1.7}$$
$$A\lambda^3(\rho - i\eta) = \sin\theta_{13}e^{-i\varphi} \tag{1.8}$$

In the Wolfenstein parametrisation, θ_{ij} are three real Cabibbo-like angles, while $e^{-i\varphi}$ is a complex phase term encoding CP violation.

Nine conditions arise from the unitarity of the CKM matrix:

$$\sum_{i=u,c,t} V_{ik}^* V_{ij} = \delta_{jk} \qquad j,k = d,s,b \tag{1.9}$$
$$\sum_{j=d,s,b} V_{jk}^* V_{ij} = \delta_{ik} \qquad i = u,c,t;\quad k = d,s,b \tag{1.10}$$

The six vanishing combinations can be represented as triangles in the complex plane (as in Figure 1.2).

According to the Standard Model, the $K^+ \to \pi^+\nu\bar{\nu}$ transition is forbidden at tree-level, and thus it arises from the one-loop contributions shown in Figure 1.3.

Separating the contributions of the quarks u, c and t, intervening as internal lines, at the leading non-trivial order the amplitude of the $s \to d\nu\bar{\nu}$

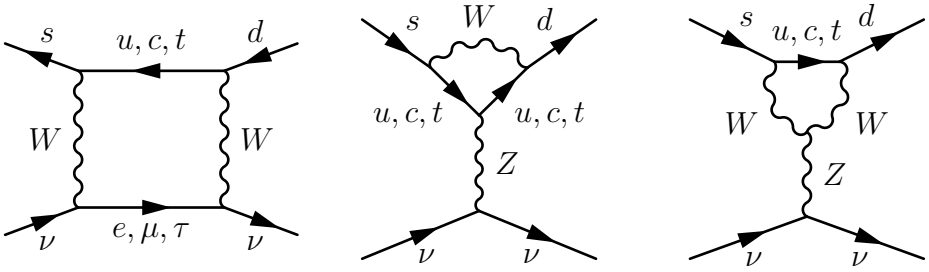


Figure 1.3: A $W-$box and two $Z-$penguin diagrams. These are the one-loop Feynman diagrams contributing to the $s \to d\nu\bar{\nu}$ process.

process may be expressed as

$$A\left(s \to d\nu\bar{\nu}\right) = \sum_{q=u,c,t} V_{qs}^{*} V_{qd}\, A_q \qquad \text{with} \tag{1.11}$$

$$A_q \sim \left(\frac{m_q^2}{m_W^2}\right)^{\delta} \qquad (\delta > 0) \qquad q = u, c, t \tag{1.12}$$

The top quark term dominates, due to its higher mass. As a consequence this process can be well described by short-distance dynamics, with the effective Hamiltonian [47]

$$H_{\text{eff}} = \frac{\alpha G_F}{2\sqrt{2}\pi \sin^2\theta_W} \sum_{l=e,\mu,\tau} \left(V_{cs}^{*} V_{cd} X_l + V_{ts}^{*} V_{td} \Upsilon_t\right) \left(\bar{s}d\right) \left(\bar{\nu}_l \nu_l\right) \tag{1.13}$$

where G_F, α and θ_W are the Fermi and fine-structure constants and the Weinberg angle, respectively. Υ_t is a function representing the dominant top quark contribution, whose associated uncertainty is very small and mainly due to the experimental error on the top quark mass: [58]

$$\Upsilon_t = 1.469 \pm 0.017 \pm 0.002 \tag{1.14}$$

where the two uncertainties correspond to QCD next-to-leading order and two-loops EW corrections respectively. The X_l functions (with $l = e, \mu, \tau$) encode instead the charm quark contributions and can be computed at the next-to-next-to-leading order with an error lower than 4% [36]. Finally, the terms $(\bar{s}d)$ and $(\bar{\nu}_l\nu_l)$ represent $V - A$ neutral weak currents.

Since the coupling amplitude depends on the semi-leptonic operator $(\bar{s}d)\,(\bar{\nu}_l\nu_l)$, the hadronic part of the amplitude of the studied process can be determined from that of the decay $K^+ \to \pi^0 e^+ \nu_e$ by means of isospin

symmetry, leading to [17, 18]

$$\frac{BR(K^+ \to \pi^+ \nu\bar{\nu})}{BR(K^+ \to \pi^0 e^+ \nu_e)} = \frac{r_K}{\lambda^2} \left\{ \left[\mathfrak{Im}(V_{ts}^* V_{td}) \right]^2 \Upsilon_t^2 + \left[\lambda^4 \, \mathfrak{Re}(V_{cs}^* V_{cd}) P_0 + \mathfrak{Re}(V_{ts}^* V_{td}) \Upsilon_t \right]^2 \right\} \tag{1.15}$$

In this equation, P_0 describes the total charm quark contribution

$$P_0 = \frac{1}{\lambda^4} \left(\frac{2}{3} X_e + \frac{1}{3} X_\tau \right) = 0.42 \pm 0.06 \tag{1.16}$$

under the assumption $X_\mu = X_e$ [18], and $r_K = 0.901$ provides the necessary isospin-breaking corrections to be applied in order to relate $BR(K^+ \to \pi^+ \nu\bar{\nu})$ to $BR(K^+ \to \pi^0 e^+ \nu_e)$.

The theoretical expectation is then

$$BR(K^+ \to \pi^+ \nu\bar{\nu}) = \left(7.81^{+0.80}_{-0.71} \pm 0.29\right) \cdot 10^{-11} \tag{1.17}$$

where the uncertainties were separated in order to highlight the first contribution, which is due to the input CKM parameters [16].

In the Wolfenstein parametrisation introduced above, $V_{ts} \simeq -V_{cb}$ at the leading order, and $|V_{cb}|$, $|V_{cs}|$ and $|V_{cd}|$ are currently well known. Therefore we are left with the only free parameter V_{td}, to be experimentally determined with an uncertainty theoretically as low as 5 – 7% [16].

Our current knowledge of $|V_{td}|$ mainly derives from the analysis of the neutral strange B meson system. The most accurate measurement of the mass difference $\Delta m_s = m(B_s^0) - m(\bar{B}_s^0)$ was obtained averaging CDF [2] and LHCb [1] results, yielding

$$\Delta m_s = (17.719 \pm 0.043) \text{ ps}^{-1} \tag{1.18}$$

that, through QCD calculations [40], yields to a combined $|V_{td}|$ / $|V_{ts}|$ measurement:

$$|V_{td}| = (8.4 \pm 0.6) \cdot 10^{-3} \tag{1.19}$$

$$|V_{ts}| = (42.9 \pm 2.6) \cdot 10^{-3} \tag{1.20}$$

$$|V_{td}/V_{ts}| = 0.211 \pm 0.001 \pm 0.006 \tag{1.21}$$

As discussed above, an alternative determination of the $|V_{td}|$ parameter is possible through the branching ratio of $K \to \pi\nu\bar{\nu}$ decays. This kind of processes arising from loop contributions is very sensitive to new Physics, and can be used to over-constrain the CKM matrix elements and check for deviations from the Standard Model.

1.3 Previous searches for $K^+ \to \pi^+\nu\bar{\nu}$

The earliest searches for the $K^+ \to \pi^+\nu\bar{\nu}$ decay date back to 1969, when a bubble chamber experiment at the Argonne National Laboratory of Michigan defined a first upper limit to its branching ratio [22]:

$$BR(K^+ \to \pi^+\nu\bar{\nu}) < 10^{-4} \qquad (1969) \tag{1.22}$$

Four years later, the limit was improved to $BR(K^+ \to \pi^+\nu\bar{\nu}) < 5.6 \cdot 10^{-7}$ by a spark chamber experiment at the Berkeley Bevatron [21], followed by a search at the KEK Proton Synchrotron that yielded [11]

$$BR(K^+ \to \pi^+\nu\bar{\nu}) < 1.4 \cdot 10^{-7} \qquad (1981) \tag{1.23}$$

Since the 80's, a large effort has been devoted at the Brookhaven National Laboratory to the study of rare, ultra-rare and forbidden kaon decays. The E787 collaboration published a first measurement based on 3 events interpreted as $K^+ \to \pi^+\nu\bar{\nu}$ decays [7]:

$$BR(K^+ \to \pi^+\nu\bar{\nu}) = 1.47^{+1.30}_{-0.89} \cdot 10^{-10} \qquad (2004) \tag{1.24}$$

The follow-up experiment E949 was able to collect 7 more candidate events with an estimated background of $0.93^{+0.32}_{-0.24} \pm 0.17$ events, where the first error accounts for the experimental systematics and the second represents the pure statistic uncertainty, leading to a combined result of [9, 10]

$$BR(K^+ \to \pi^+\nu\bar{\nu}) = 1.73^{+1.15}_{-1.05} \cdot 10^{-10} \qquad (2009) \tag{1.25}$$

which is consistent with the Standard Model expectations, within the large statistical errors.

It is important to note that every $K \to \pi\nu\bar{\nu}$ experiment performed up to now has used low-energy stopped-kaon beams. NA62 will instead employ a high-energy beam, thus studying in-flight kaon decays.

1.4 Experimental strategy

The presence of two neutrinos and of a single charged track in the final state makes NA62's goal a challenging precision measurement, requiring hermetic background rejection as well as an excellent detector system for particle identification, tracking, calorimetry and spectrometry. The signature

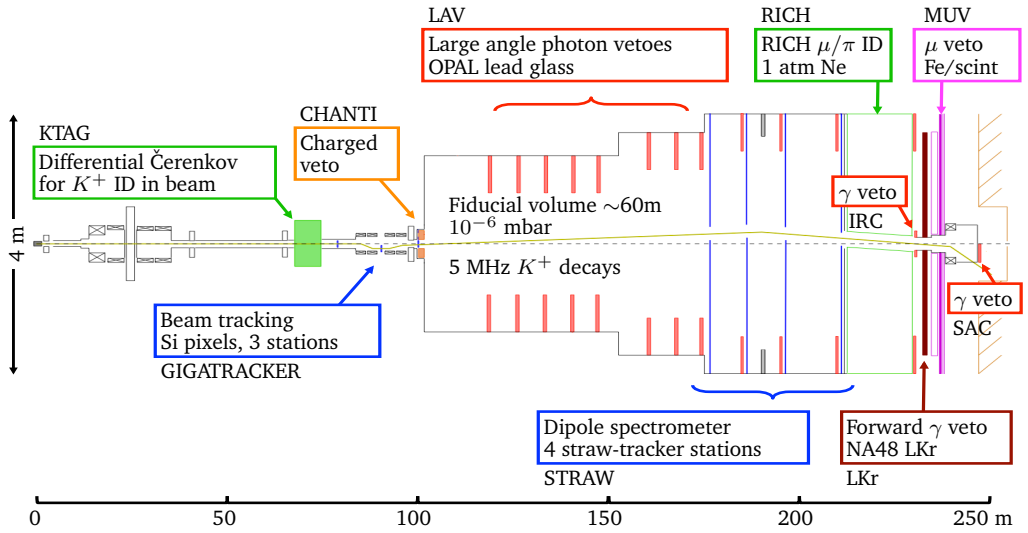


Figure 1.4: Sketch of the full NA62 detector. The CEDAR (KTAG), 8 out of 12 LAV stations, the LKr and SAC calorimeters and the CHOD used for NA48 are already installed on site. The CHANTI, the STRAW spectrometer, the RICH, the IRC calorimeter and the muon vetoes are currently under construction, while the GTK is in a phase of advanced design [51]. See Chapter 2 for a brief description of the above subdetectors.

of a $K^+ \to \pi^+\nu\bar{\nu}$ decay consists in one and only one charged track. Any other event for which only one charged track is detected contributes to the background.

The $K^+ \to \pi^+\nu\bar{\nu}$ process features only three measurable quantities, which are the momenta of the kaon and of the charged decay product, and the laboratory frame angle between the two. It is convenient to construct the squared missing mass to the kaon and the measured decay product, and use it as a discriminating kinematic variable:

$$\begin{aligned} m^2_{\text{miss}} &= (P^\mu_K - P^\mu_\pi)^2 \\ &= (E_K - E_\pi)^2 - \left(P^2_K + P^2_\pi - 2|P_K||P_\pi|\cos\theta_{K\pi}\right) \end{aligned} \tag{1.26}$$

where P_K and P_π are the momenta of the decaying kaon and of the pion, E_K and E_π are their energies and $\theta_{K\pi}$ is the decay angle in the laboratory frame, i.e. the angle between the kaon and pion tracks.

The missing mass is computed under the assumption that the detected decay product is a pion. The m^2_{miss} variable allows to separate the signal from the most important background processes, as shown in the first panel of Figure 1.5. The "fiducial" signal region, i.e. the m^2_{miss} interval allowed for

Decay mode	B. R.	Rejection
$\mu^+\nu_\mu$	63%	Kinematics + μ PID
$\pi^+\pi^0$	21%	Kinematics + γ Veto
$\pi^+\pi^+\pi^-$	6%	Kinematics + $\pi^\pm$ Veto
$\pi^+\pi^0\pi^0$	2%	Kinematics + γ Veto
$\pi^0 e^+\nu_e$	5%	e PID + γ Veto
$\pi^0\mu^+\nu_\mu$	3%	μ PID + γ Veto

Table 1.1: Main background channels for the NA62 experiment.

data analisys, is split into two parts, excluding the region dominated by the $K^+ \to \pi^+\pi^0$ process. In addition, upper and lower limits are set in order to isolate the $\mu^+\nu$ and 3π channels. These three channels account for 92% of the K^+ decay events [50].

The second panel of Figure 1.5 gathers missing mass spectra for background modes which are not kinematically constrained. These modes include radiative versions of the channels described above, and $3-$ and $4-$body semi-leptonic channels. In these cases, the missing mass distributions overlap that of the signal; hence the only way to push background rejection to the needed limits for these processes is to employ reliable particle identification (PID) and VETO systems.

Table 1.1 lists the most frequent K^+ decay modes, together with the respective rejection techniques.

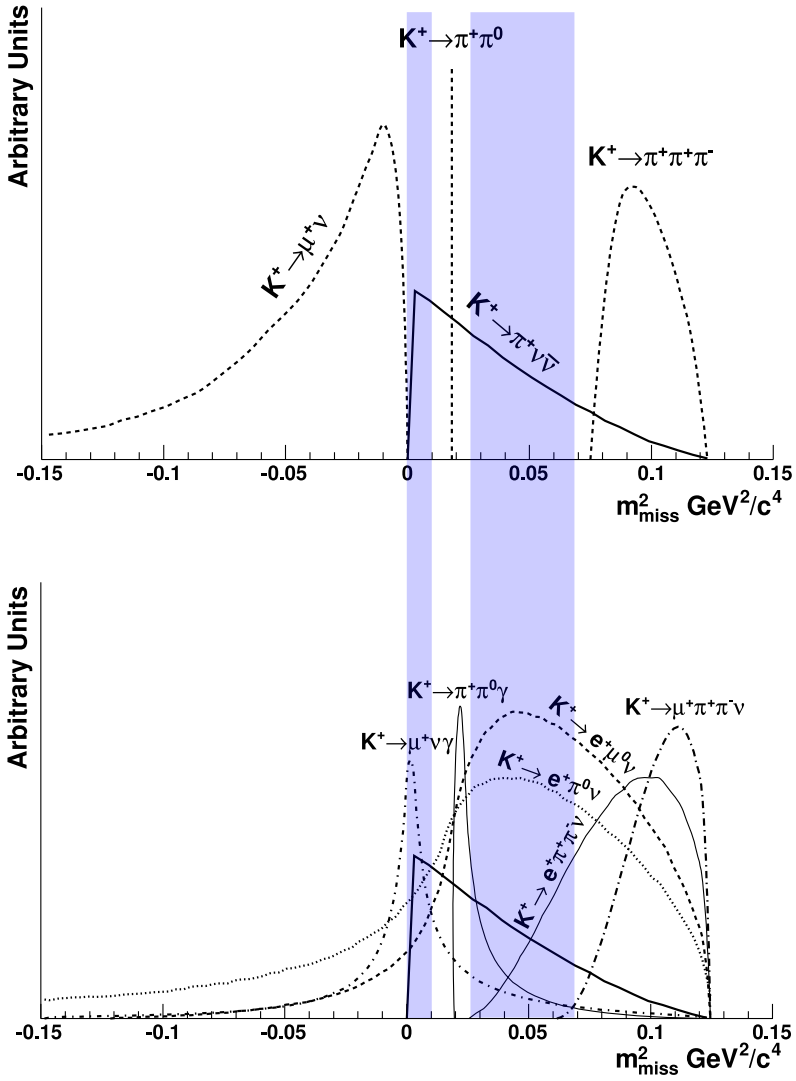


Figure 1.5: Distribution of the recostructed squared missing mass resulting from the decay of the generated kaon into the detected charged daughter, under the assumption that the latter is a pion. The solid line represents the $K^+ \to \pi^+\nu\bar{\nu}$ signal in both plots. The first sketch shows the shape of the kinematically constrained backgrounds, which are also the channels with the largest branching ratios. The second plot shows the other main background processes for which the reconstructed missing mass overlaps that of the signal. The distributions shown here were computed without taking into account the errors due to the finite resolution of the NA62 detector [23]. The darkened areas display the current choice for the fiducial signal regions.

2 Experimental setup

Contents

The NA62 experiment is located at the CERN-SPS North area, on the beam line originally used by the NA48 experiment. The NA62 collaboration devised a new experimental apparatus on the basis of the NA48 experience [48].

An unseparated 750 MHz beam composed by protons, pions and a fraction of 6% K^+ is produced by the collision of a 400 GeV/c proton beam on a Beryllium target. The fiducial decay region begins approximately 100 m downstream of the target, and ends 70 m farther, where the downstream

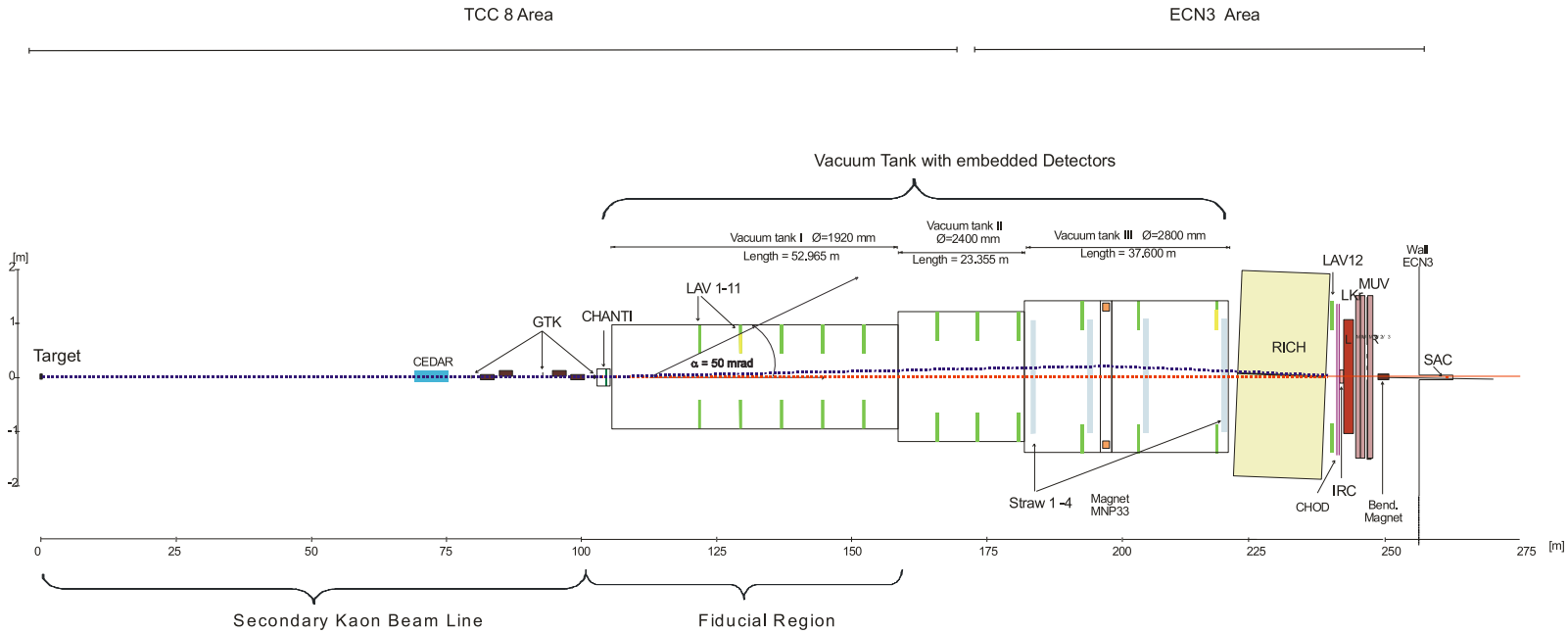


Figure 2.1: Schematic view of the NA62 detector.

detectors are located.

Three sub-detectors placed upstream of the decay region monitor the incoming beam conditions. Beam kaons are identified by the *CEDAR*, a Čerenkov Differential counter with Achromatic Beam focus. The *GTK* (Gigatracker) subdetector, a beam spectrometer composed by three stations of silicon micro-pixels, provides a precise measurement of the beam kaons momentum, direction and time. Finally, the *CHANTI* (Charged-Anti) scintillator rings that surround the last layer of the GTK veto large-angle charged particles before they enter the decay region.

Large-angle photon vetoes (*LAV*) surround the cylindrical walls of the vacuum chambers that host the decay region. Together with the downstream electromagnetic calorimeters, they ensure a photon rejection inefficiency smaller than 10^{-8} in a 50 mrad cone around the z axis (i.e. the initial direction of the kaon beam) [32].

Decay vertex, direction of flight and momentum of the charged decay products are measured by a straw chambers spectrometer (*STRAW*) and a *RICH*, namely Ring Imaging Čerenkov detector. The RICH is also used to provide π–μ discrimination in the $15 \leq P_z \leq 35$ GeV/*c* region, where P_z is the pion/muon momentum along the z axis.

A segmented scintillation hodoscope (*CHOD*) provides fast trigger signals for charged particles just after they emerge from the RICH vessel. Downstream, three calorimeters provide small-angle photon veto. The *LKr* (Liquid Krypton) detector is an electromagnetic calorimeter which can also provide a selective e^+/e^- trigger by measuring the energy deposit and the shape of the electromagnetic showers developed in its volume. An intermediate ring-shaped calorimeter (*IRC*) and the small-angle electromagnetic calorimeter (*SAC*) add photon suppression in the region not covered by the geometric acceptance of the LKr. In particular, a dipole magnet is installed in order to bend charged beam particles away from the z axis, so that only forward photons can hit the SAC.

Suppression of the $K^+ \to \mu^+\nu_\mu$ background requires fast and hermetic muon vetoing. A scintillator system (*MUV3*) detects muons emerging from an 80 cm thick iron block. Two additional muon-veto stations are provided by the *MUV1* and *MUV2* calorimeters, used to distinguish muons from hadrons by measuring the energy released and the shape of the hadronic showers initiated by the particle that has to be identified.

A longitudinal view of the experimental setup devised by the NA62 collaboration is shown in Figure 2.1. For clarity, I will divide the sub-detectors into two subsets: the beam detectors, upstream of the decay region, and the downstream detectors, used to detect and characterize the decay products.

2.1 Beam

Empirical results achieved in 1980 [12] show that it is convenient to use a high energy proton beam in order to maximize the production of positive kaons by beam interaction on a Beryllium target.

The highest kaon production is achieved at $P_K/P_p \simeq 0.35$, where P_p is the central proton beam momentum, and P_K is the momentum of the produced kaons. In addition, the number of kaons that decay in the fiducial region reaches its maximum for $P_K/P_p \simeq 0.25$. These quantities increase with P_K^2 and P_K, respectively [32]. Furthermore, the use of high energy kaons increases the detection efficiency of most sub-detectors. Due to these considerations, it was decided to fix the central beam momentum at 75 GeV/*c*.

The choice of positive kaons is determined by the following ratios of particles abundances in a beam produced with 400 GeV/*c* protons: [32]

$$K^+/K^- \simeq 2.1 \tag{2.1}$$

$$\frac{K^+/\pi^+}{K^-/\pi^-} \simeq 1.2 \tag{2.2}$$

A fiducial momentum range for the detected pion is defined between 15 and 35 GeV/*c*. In fact, in the RICH detector pions from K^+ decay are well separated from background muons only if their energy is higher than approximately 15 GeV (see Section 2.3.3). Moreover, an upper limit to the energy of accepted pions is set at 35 GeV, so that other particles originating from K^+ decays must carry an energy of 40 GeV, and thus be comfortably detectable if they are not neutrinos. This way, the detector hermeticity with respect to the background for the studied decay channel $K^+ \to \pi^+\nu\bar{\nu}$ is also increased.

Beam characteristics		60 GeV/c	75 GeV/c	120 GeV/c
Fluxes at production	p	89	171	550
	K^+	40	53	71
	π^+	353	532	825
	total	482	756	1446
Survival factor over 102 m	K^+	0.797	0.834	0.893
	π^+	0.970	0.976	0.985
Fluxes at 102 m from target	p	89	173	550
	K^+	32	45	63
	π^+	343	525	813
	total	464	743	1426
Decays in 60m fiducial length	K^+	3.9	4.5	4.1
	π^+	6.1	7.4	7.2
K^+ decays / π^+ decays in 60m		0.64	0.61	0.57
K^+ decays in 60m / total hadr. flux	$\cdot 10^{-3}$	8.4	6.1	2.9
$K \to \pi\nu\nu$ acceptance (R_1, no $P_{\pi max}$)		0.08	0.11	0.11
Acc. $K \to \pi\nu\nu$ / 10^{12} proton s^{-1}	$\cdot 10^6$ $B.R.$	0.31	0.50	0.45
Acc. $K \to \pi\nu\nu$ / π^+ decays in m	$\cdot B.R.$	0.052	0.067	0.062
Acc. $K \to \pi\nu\nu$ / total hadr. flux	$\cdot 10^{-3}$ $B.R.$	0.67	0.67	0.31

Table 2.1: Criteria for the choice of the central beam momentum [32]. Fluxes and decay rates are expressed in units of 10^6 s^{-1} and normalized to 10^{12} incident protons per second. Fluxes are measured in a solid angle $\Delta\Omega = 42$ μsr, and allow for a momentum spread $\Delta P/P = 1\%$.

Table 2.1 summarizes the criteria examined in order to choose the momentum of the kaon beam.

Let us briefly discuss how the final kaon beam is produced. A focalised 400 GeV/c proton beam extracted from the SPS is driven onto a 40 cm thick beryllium target with a diameter of 2 mm. The emerging beam passes through a 95 cm long copper collimator, which absorbs particles propagating at angles greater than 6 mrad before they can decay. Three subsequent quadrupole magnets define a spatial acceptance of 3 mrad along the horizontal direction (x) and 5.2 mrad along the vertical direction (y) around a central beam momentum of 75 GeV/c [32]. Two subsequent dipoles select the positive component of the beam. A 1.06 X_0 thick tungsten radiator is placed in the middle of this 4-dipole achromatic optic element, in order

Momentum	75 ± 0.9 GeV/c
Rate	750 MHz
Composition	70% π^+ 23% p^+ 6% K^+ 1% other

Table 2.2: Features of the final mixed beam entering the fiducial decay region.

to reduce the positron energy at such a level that the beam optics can reject positrons up to 99.6%. Finally, two remaining dipoles, and then three quadrupole magnets, refocus the beam onto its original direction along the $z-$axis. The whole system ensures a momentum acceptance band $|\Delta P/P| \simeq 1\%$.

After entering the vacuum chamber, the beam is focused again to an aperture of 4 mrad by a 1.8 m long collimator with a diameter of 28 mm.

The characteristics of the final beam are shown in Table 2.2.

2.1.1 Beam tracking

In Section 2.2.1 I will describe the CEDAR detector used to tag K^+ in the beam. The CEDAR is preceded by an adjustable collimator that ensures that the beam is sufficiently large and parallel to match the detector requirements, and by a vertical steering magnet. Two pairs of scintillator counters in coincidence monitor the divergence of the beam, and their feedback is used to tune the beam optics in order to suppress the beam divergence.

A system composed by the Gigatracker detector (described in Section 2.2.2), four achromatic C-shaped dipoles, and a horizontal steering magnet is used to track the beam kaons. The achromatic dipole system deflects the beam in the vertical direction, providing a 60 mm displacement that allows to measure the track momentum with a resolution of 0.2%.

Between the second and the third dipole there is a 5 m long toroidal collimator, made of magnetized iron, whose aim is to defocus the muon content of the beam. The "return fields" in the yokes of the following dipoles

supplement the defocussing action of the collimator. The overall system can intercept and deflect muons out of the spatial acceptance of the beam for momenta $P_\mu < 55$ GeV/c [32].

Just before the last station of the Gigatracker, a horizontal magnet deflects the beam by an angle $\theta_K = 1.2$ mrad towards the positive side of the x axis. This way, the subsequent deflection of -3.6 mrad towards $x < 0$ that allows the STRAW spectrometer to track the decay products (see Section 2.3.2) directs the beam back into the central hole of the LKr calorimeter (Section 2.3.1).

2.2 Beam detectors

2.2.1 The differential Čerenkov counter (CEDAR)

One disadvantage of high-energy beams is that kaons cannot be efficiently separated from the p/π content of the beam by means of the beam optics. As a consequence, upstream detectors are exposed to about 17 times the "useful" rate of particles. The identification of kaons before they decay is indeed a critical aspect in such a high-rate environment.

Positive kaon tagging is achieved by letting the beam traverse a differential Čerenkov counter (CEDAR). The detector is filled with hydrogen at a pressure of 3.6 bar, and it has a total thickness of 6.6 X_0.

A particle crossing a radiator with refractive index n at a speed β emits a cone of Čerenkov light at an angle $\theta_c(\beta, n)$ (see Chapter 2.3.3). Since the momentum of the beam is known, the Čerenkov angle, at a fixed gas pressure and therefore fixed n, is a function of the mass of the particle. The gas pressure is therefore adjusted so that only the wanted particle type can emit Čerenkov radiation at the chosen light detection angle.

The Čerenkov light is reflected by a spherical mirror onto a ring-shaped diaphragm that vehicles light into 8 clusters of 32 photomultipliers each [32]. The number of photomultipliers was increased, compared to the original CEDAR on the beam line, in order to decrease the photon rates on each readout device, reducing dead time and accidental noise as a consequence.

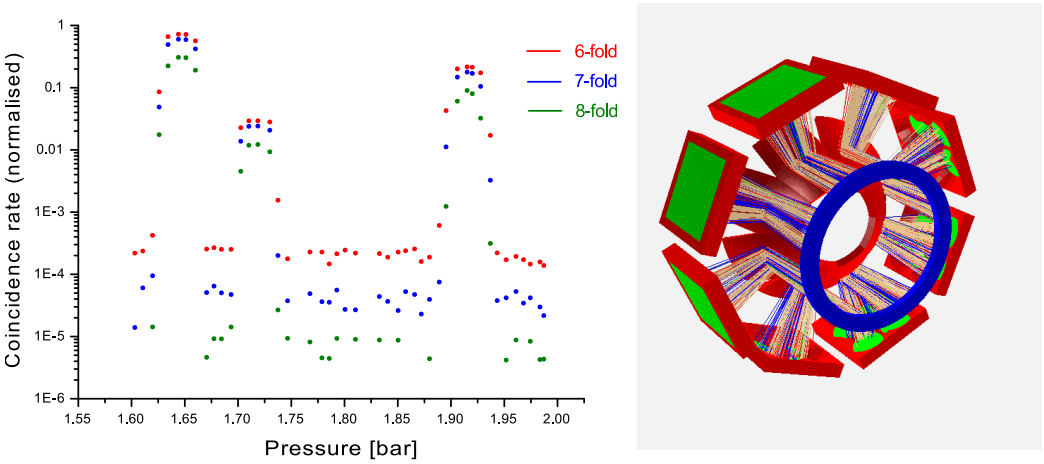


Figure 2.2: On the left, a pressure scan on a 75 GeV beam shows three peaks, corresponding to pions, kaons and protons respectively. The plot shows the counting rate of the CEDAR detector, normalized to the total beam rate, as a function of the pressure of the N_2 gas filling the chamber [23]. The panel on the right displays the photons for 100 simulated CEDAR events: ellipsoidal mirrors defocus the light onto the photodetector planes, in order to lower the counting rate of each photomultiplier [32].

2.2.2 The GigaTracker spectrometer

The name "Gigatracker" derives from the high rate of particles that this spectrometer must sustain. Due to the non-uniform 750 MHz beam rate, the particle flux presents a peak of 1.3 GHz/mm^2 around the centre of the detector.

The Gigatracker provides precise measurements of the angle, momentum and time of the crossing particle. The resolutions to be achieved are:

time	150 ps
momentum	$\Delta p \,/\, p \leq 0.2\%$
direction	16 μrad

Such accurate measurements are necessary in order to correctly associate the decaying kaon with the π^+ track detected downstream. Kinematic selection of $K^+ \to \pi^+ \nu\bar{\nu}$ events indeed relies on constraints based on the missing mass variable, defined as

$$\begin{aligned} m^2_{miss} &\equiv (P_K - P_\pi)^2 \\ &\simeq m^2_K \left(1 - \frac{P_\pi}{P_K}\right) + m^2_\pi \left(1 - \frac{P_K}{P_\pi}\right) - P_K P_\pi \theta^2_{K\pi} \end{aligned} \tag{2.3}$$

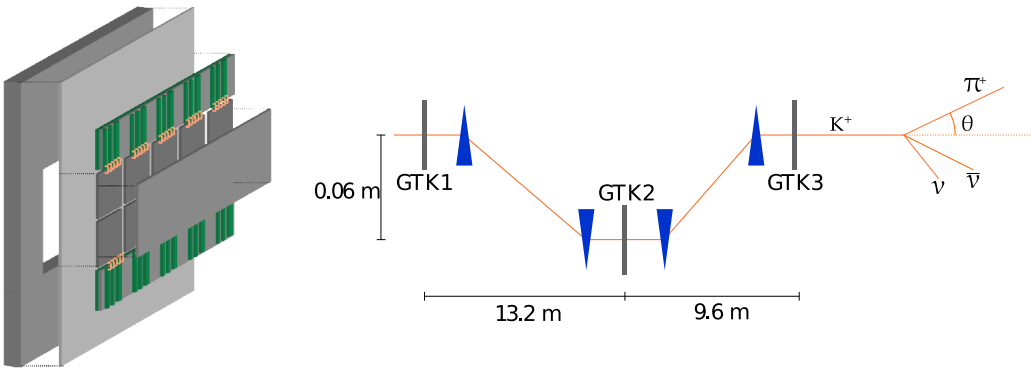


Figure 2.3: On the left, sketch of a Gigatracker station [23]. On the right, layout of the Gigatracker stations and magnets used to bend the beam [68].

In this equation, P_K entirely derives from the Gigatracker data, and the determination of $\theta_{K\pi}$ makes use of both the GTK and the downstream STRAW spectrometer.

In order to limit hadronic interactions and to preserve the beam divergence, the Gigatracker is composed of three stations for a total thickness lower than 0.5 X_0 [32]. Each station contains 18000 300×300 μm^2 silicon micro-pixels 200 μm thick bump-bonded to 10 readout ASIC chips 100 μm thick. A sketch of a Gigatracker station is shown in Figure 2.3(a).

The three stations of the GTK are mounted inside the vacuum tank preceding the decay region, and they are interlaced with 4 achromat magnets as shown in Figure 2.3(b).

An extensive amount of resources was committed to the design and development of this detector. Critical aspects are the high radiation hardness needed to sustain the rate of beam particles crossing the silicon sensors, and the very high time resolution required. In particular, the dissipation of the increasing leakage current due to radiation damage needs a complex cooling system. This will be based on a micro-channel cooling system.

2.2.3 The charged anti-counter

A vital requirement of the experiment is the reduction of accidental background to a level of 10^{-11}. The purpose of the CHANTI (*Charged Anti-counter*) is to tag particles propagating at angles larger than that allowed for the beam as they emerge from the last station of the Gigatracker. Such

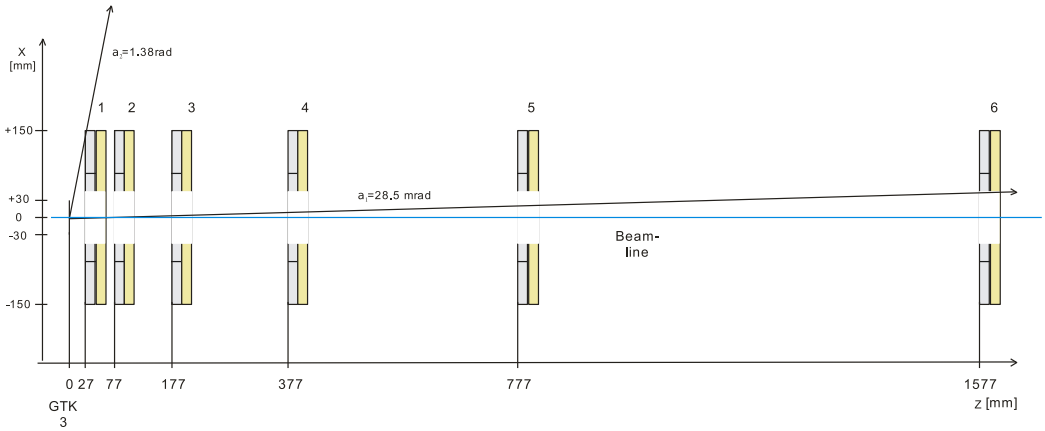


Figure 2.4: Sketch of the six CHANTI stations on the beam line [32].

particles can be produced by inelastic interaction of the beam with the collimator and upstream material.

A set of large angle guard counters is therefore installed immediately after the Gigatracker (see Figure 2.4). Further stations, sensitive in the closest region to the beam, can veto the beam halo muons.

The CHANTI is made of six double-layer stations [32]. Each station is a 30×30 cm^2 square with a 90×50 mm^2 rectangular hole to allow the passage of the beam. Each layer is composed of 24 (22) scintillator bars aligned to the x axis (y axis). Light is collected by wavelength-shifting fibers, and transported to one side of the station, where a silicon photomultiplier is placed.

2.3 Detectors downstream of the decay region

The downstream detector has been designed to detect K^+ decay products. Therefore, central beam holes have been arranged in all sub-detectors.

2.3.1 A hermetic setup for photon vetoing

Photons originating from one of the major backgrounds, $K^+ \to \pi^+\pi^0$ ($B.R. = 20.7\%$), propagate at angles greater than 50 mrad only in a 0.2% fraction of $\pi^+\pi^0$ events. A photon veto system was therefore developed, that ensures a rejection inefficiency lower than 10^{-8} in the fiducial energy range

of the signal. Photon veto detectors cover a 50 mrad angular region around the beam.

The photon veto system is partitioned in four sub-detectors that cover different angular regions and employ three different technologies:

- Twelve LAV stations cover the angular region between 8.5 and 50 mrad.
- SAC and IRC cover the inner angular region, from about 0 to 1 mrad.
- The LKr calorimeter covers angles between 1 and 8.5 mrad.

The **Large Angle Veto** (LAV) detector reuses the $10\times10\times37$ cm^3 lead-glass blocks from the OPAL electromagnetic calorimeter [3], arranged in 12 annular stations (Figure 2.5(a)).

Of the 12 LAV counters, 11 are placed inside the 3×10^{-7} mbar vacuum tank hosting the decay region, and one is placed between the RICH and the CHOD sub-detectors.

Each ring-shaped station of the LAV has an increasing diameter as the distance from the target grows, and consists of four to five rings of lead-glass blocks read out at the outer side by 76 mm diameter Hamamatsu photomultipliers.

Photons incident onto the LAV blocks start electromagnetic avalanches, detected through the collection of Čerenkov light emitted by e^+e^- pairs.

Thanks to its low threshold, the LAV system will be also able to detect muons and pions in the beam halo [49].

The **Liquid Krypton Calorimeter** (LKr), placed between the RICH and the MUV detectors, is the same used in the NA48 experiment. Its main purpose is to reject photons that fly at an angle between 1 and 8.5 mrad to a level of 10^{-5}. However, the LKr can also provide accurate measurements of the energy of electrons and positrons, useful for the rejection of the $K^+ \to \pi^0 e^+ \nu_e$ background.

The calorimeter volume is filled with 9 m^3 of liquid krypton, whose characteristics are summarised in Table 2.3.

In this quasi-homogeneous calorimeter, the active material is 127 cm (approximately 27 X_0) thick, and can fully contain 50 GeV showers. The

X_0	4.7 cm
Moliere radius	6.1 cm
Bath temperature	119.8 K

Table 2.3: Characteristics of the liquid krypton filling the calorimeter [67].

active volume is divided in 13248 $2\times2\times127$ cm^3 double ionization cells with Cu-Be ribbons as central anodes (Figure 2.5(b)). Despite being originally built for the NA48 experiment, the LKr calorimeter is still a state-of-the-art piece of hardware, featuring excellent resolution in energy, space and time:

$$\frac{\sigma_E}{E} = \frac{0.032}{\sqrt{E}} \oplus \frac{0.09}{E} \oplus 0.0042 \text{ GeV} \tag{2.4}$$

$$\sigma_x = \sigma_y = \frac{0.42}{\sqrt{E}} \oplus 0.06 \text{ cm} \tag{2.5}$$

$$\sigma_t = \frac{2.5}{\sqrt{E}} \text{ ns} \tag{2.6}$$

In the above formulas all energies are expressed in GeV.

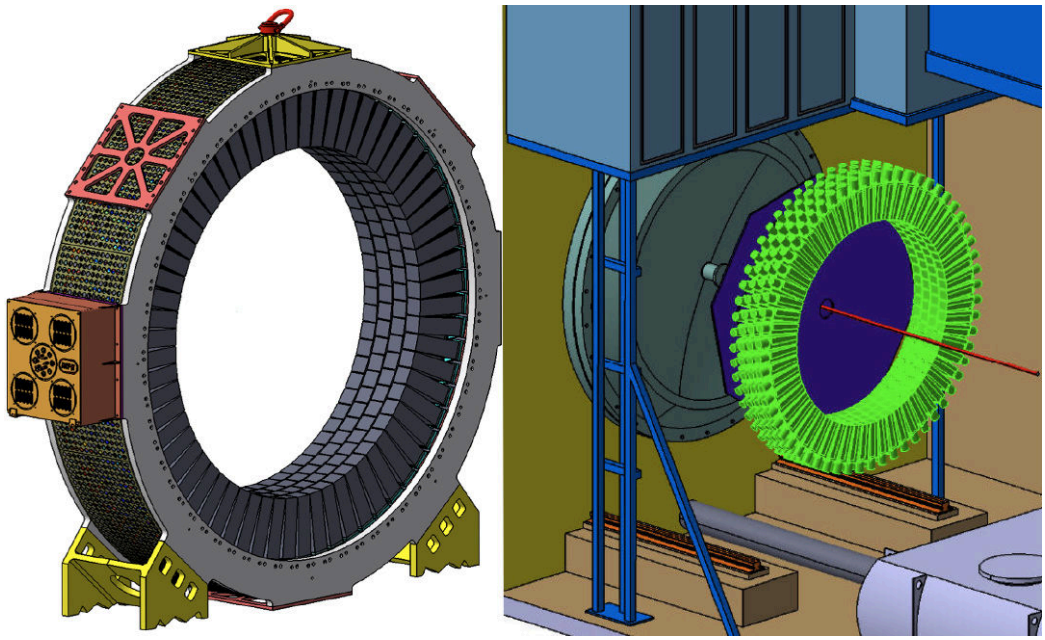
Both the small-angle veto calorimeters, **IRC** and **SAC**, are "*shashlyk*" type calorimeters, i.e. detectors made of lead absorber layers interspaced with plastic scintillator plates used as active material [13].

Due to the geometry of the experiment, photons hitting IRC or SAC will have an energy greater than 5 GeV. Interacting with the lead plates, they will then start electromagnetic showers. A wavelength-shifting dopage is added to the scintillator tiles in order to gather the light emerging from the showers into fibers read out by photomultipliers.

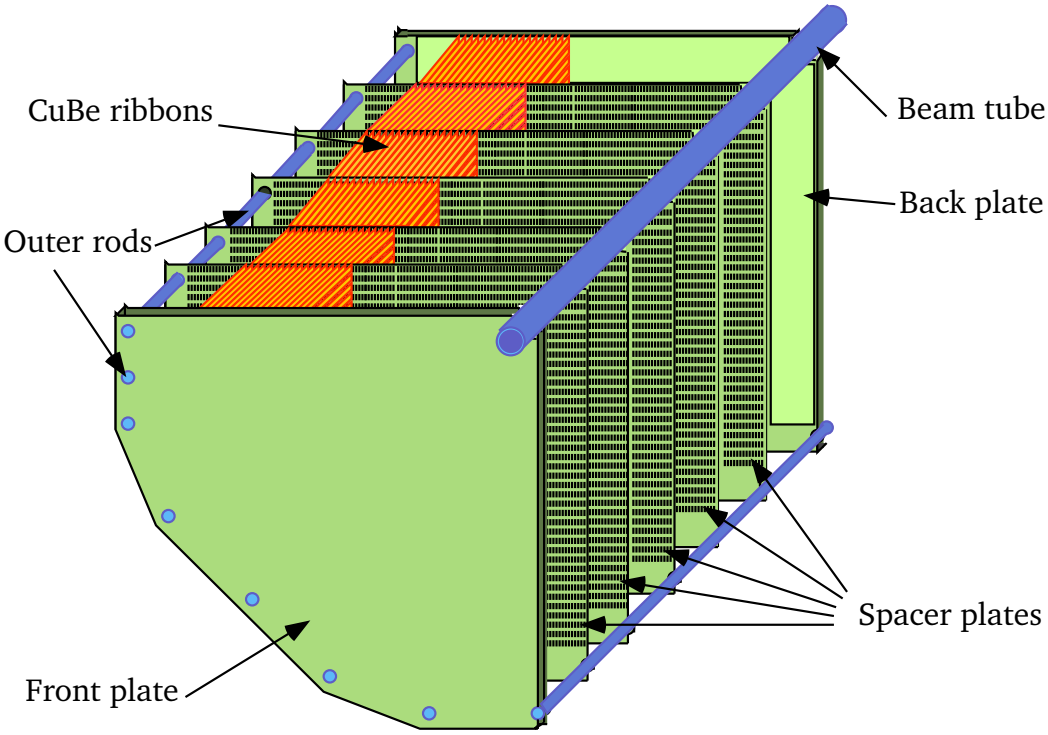
The **Ring-shaped Calorimeter** IRC is placed around the beam line in front of the LKr, and covers the angular region between such detector and the SAC.

A dipole magnet bends the beam so that charged particles cannot hit the SAC (**Small Angle Calorimeter**), the most forward detector in the NA62 setup.

The combined detection inefficiency of the small angle vetoes is requested to be lower than 10^{-8}.



(a) A view of the LAV-12 station.



(b) Liquid krypton calorimeter electrode structure [67].

Figure 2.5: Technical designs of a LAV station (top panel) and of the LKr calorimeter (bottom panel).

2.3.2 The STRAW magnetic spectrometer

The purpose of the *magnetic spectrometer* is to determine the directions and momenta of secondary particles originating from primary kaon decays. The kinematical constraints needed to reject most of the background require an accurate reconstruction of the track of the daughter charged particle. In particular, in order to achieve proper reconstruction, the needed resolutions are:

- decay angle: $\Delta\theta_{K\pi} \leq 60$ mrad
- momentum: $\Delta p/p \leq 1\%$
- spatial resolution: $\sigma_{x,y} \leq 130$ μm, in order to correctly trace back the decay vertex.

In addition to this, a number of experimental requirements need to be fulfilled. The tracker will be in fact integrated inside the vacuum chamber, and it needs to minimize the amount of material traversed by the particles as much as possible. The detector must also be able to operate in a high rate environment, with a particle flux close to 40 MHz cm^{-1} in the central area.

The spectrometer is based on the *straw* technology, its building blocks being ultra-light straw tubes 2.1 m long and 9.8 mm in diameter [32]. The material employed consists of 36 μm thin PET foils coated on the inner side with 50 nm of copper and 20 nm of aluminium, acting as cathode; the anode is a gold-plated tungsten wire, 30 μm in diameter, placed at the centre of the tube.

A simulation performed with Garfield[1] has allowed to compare the best working points for two different gas mixtures: a slower, but more "ageing-safe" isobutane mixture (CO_2 90% iso-C_4H_{10} 5% CF_4 5%), and a faster Ar 70% CO_2 30% mixture. Considerations about the possibility of pile-up of hits corresponding to different events have led to the choice of the faster gas mixture.

The full spectrometer consists of four chambers. A dipole magnet, placed between the second and the third chamber, generates a vertical field of 0.36 T, corresponding to a kick of 270 MeV/c directed towards the $x-$axis. Each chamber is composed of four complementary "views" (x, y, u and v), allowing for a redundance of 100%. Each view, finally, is made of 256 *straw* tubes. Figure 2.6 shows the layout of the views composing a STRAW

[1] http://garfield.web.cern.ch/garfield/

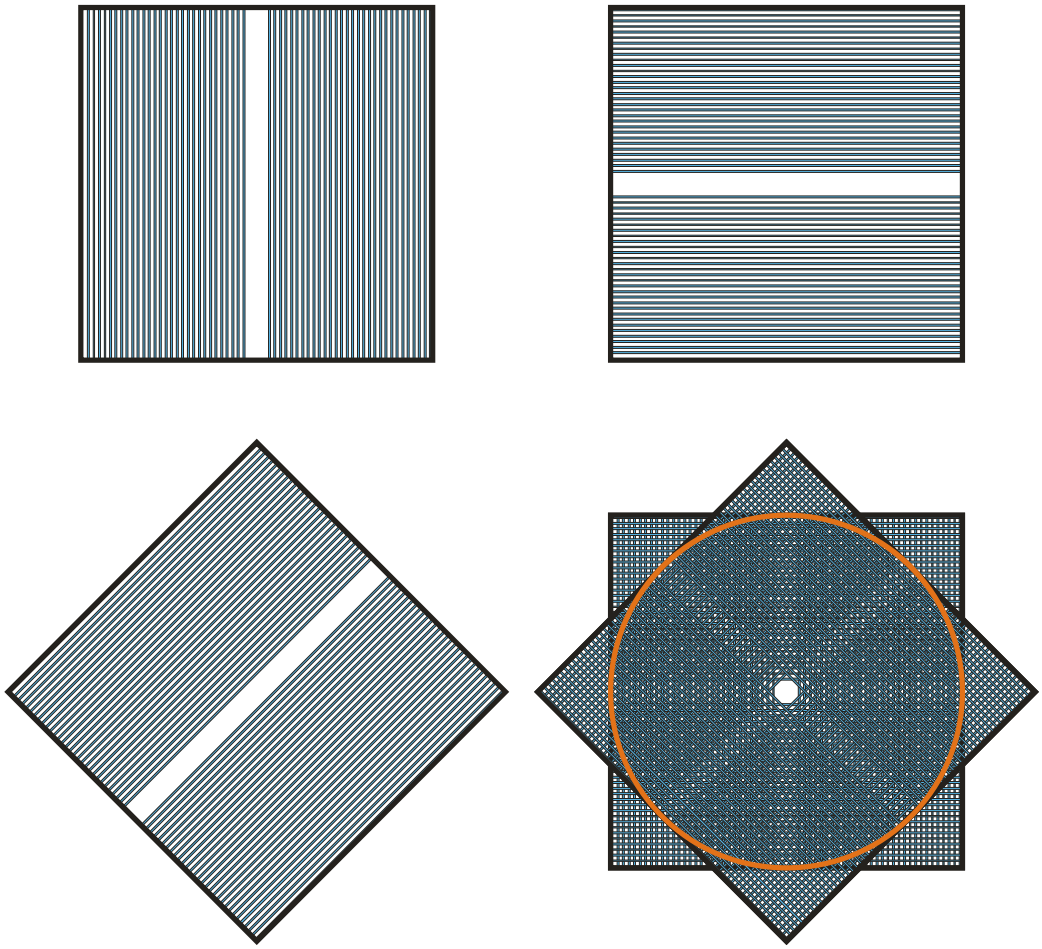


Figure 2.6: Sketch of the four "views" composing a straw chamber [32]. The last panel shows the superposition of x, y, u and v views, i.e. a complete station.

Figure 2.7: Details of: the straws for the downstream spectrometer, the CEDAR-KTAG photomultiplier housing frame, the ribbons defining the cells of the LKr calorimeter [51].

chamber, while a detailed view of the tubes composing a plane is visible in the first panel of Figure 2.7.

2.3.3 The RICH detector

The main background $K^+ \to \mu^+\nu_\mu$ should be suppressed by a factor 10^{-13} in order to achieve a signal-to-background ratio of the order of 10% for the $K^+ \to \pi^+\nu\bar{\nu}$ decay [8]. This task will be fulfilled by a combination of kinematic cuts and direct muon rejection at trigger level.

The **Ring Imaging Čerenkov** (RICH) sub-detector in the NA62 setup will be able to separate pions from muons in the momentum range comprised between 15 and 35 GeV/c up to a level of 5×10^{-3}; moreover, it will also provide a level 0 trigger primitive for charged tracks. Since this detector is central to the work described in this thesis, it is here discussed in more detail.

Let us describe the functioning of a RICH detector in general terms, before discussing the one devised for the NA62 setup.

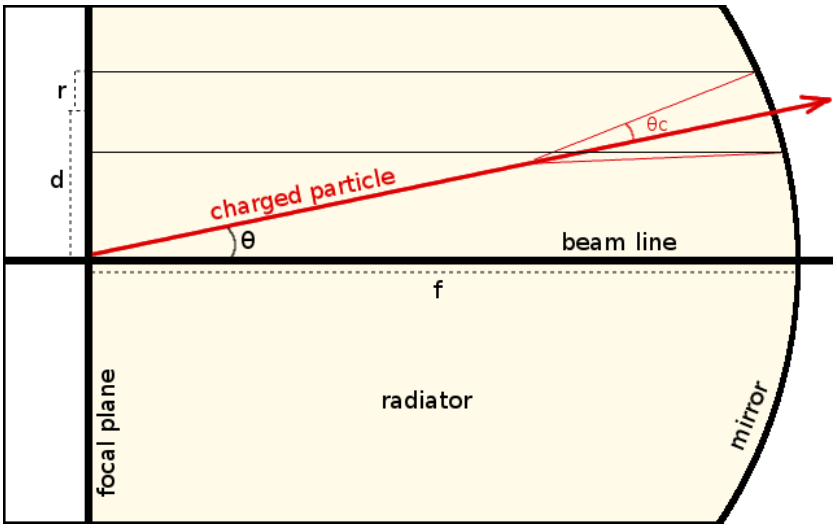


Figure 2.8: Draft of a simple RICH detector. The radius of the circle in the focal plane is determined by the velocity of the particle, while the position of its centre depends on the particle direction.

Principles of a RICH detector

Figure 2.8 describes how a RICH detector works. When a particle goes through a medium at a velocity $\beta = v/c > 1/n$, where n is the refractive index of the medium, it emits Čerenkov light at an angle θ_c relative to the particle trajectory, such that

$$\cos\theta_c = \frac{1}{n\,\beta} \tag{2.7}$$

thus forming what is called a *Čerenkov cone*. It follows that a threshold β_{th} exists below which no radiation is emitted:

$$\beta_{th} = \frac{1}{n} \qquad (\theta_c = 0) \tag{2.8}$$

while the maximum angle of emission is achieved for $\beta \to 1$:

$$\cos\theta_{max} \to \frac{1}{n} \qquad (\beta \to 1) \tag{2.9}$$

From Eqn. 2.8 we derive the threshold momentum P_{th} for a particle of mass m to emit Čerenkov radiation:

$$P_{th}(m) = \frac{m}{\sqrt{n^2 - 1}} \tag{2.10}$$

The light cone is projected on a focal plane, perpendicular to the beam direction, by means of a spherical mirror (or a system of mirrors, as it will be described in the following section) of focal length f. For particles travelling parallel to the beam line, the resulting image on the focal plane is a ring of radius

$$r_c = f \tan\theta_c \tag{2.11}$$

while, for particles travelling at an angle θ to the beam line, the same image appears shifted by a distance

$$d = f \tan\theta \tag{2.12}$$

from the focus. Figure 2.8 shows a sketch of a basic RICH detector.

Larger rings on the focal plane correspond to particles crossing the RICH radiator volume at a larger velocity (keeping the type of the particle fixed). On the other hand, if the momentum of the beam is known to an adequate precision, the radius of the reflected Čerenkov ring can be used to compute the mass of the crossing particle, and therefore to perform P.ID. (*Particle Identification*).

The following relation holds:

$$P(\theta_c) = \frac{m}{n} \frac{1}{\sqrt{\sin^2\theta_{max} - \sin^2\theta_c}} \tag{2.13}$$

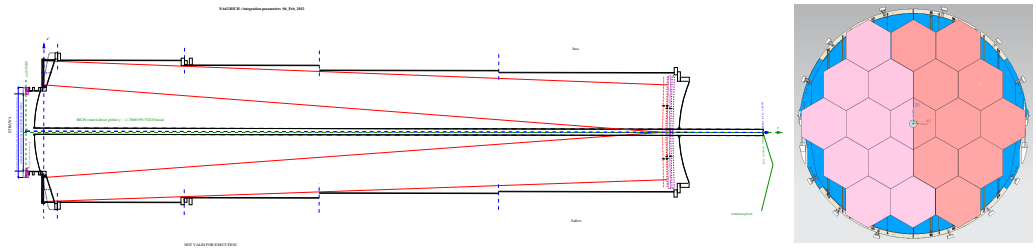
and therefore:

$$P(r_c) = m \frac{\sqrt{f^2 + r_c^2}}{\sqrt{r_{max}^2 - r_c^2}} \simeq \frac{m\, f}{\sqrt{r_{max}^2 - r_c^2}} \tag{2.14}$$

where $r_{max} = f\sqrt{n^2 - 1}$.

The NA62 RICH detector

The momentum range and the position of the Čerenkov threshold determined the choice of the gas and gas pressure: the vessel, a 18 m long and 2.8 m wide cylinder, will be filled with neon at atmospheric pressure. This way, the Čerenkov threshold momentum for a pion will be $P_{th} \simeq 12.5$ GeV/c, 20% smaller than the lower limit of the accepted momentum range. The refraction index n will be such that $(n-1) \simeq 60 \times 10^{-6}$ [32]: according to Eqn. 2.10, this value almost perfectly corresponds to the emission threshold



(a) (b)

Figure 2.9: Panel 2.9(a): technical drawing of the RICH vessel (the beam goes from right to left). Panel 2.9(b): front view of the system of 18 hexagonal and 2 semi-hexagonal mirrors.

for the mass of the charged pion. The Ne gas also guarantees small dispersion [8]. Figure 2.9(a) shows the technical design of the vessel that will host the RICH detector.

The vessel will be placed between the last STRAW chamber and the LKr calorimeter. It will be rotated by 2.4 mrad with respect to the $z-$axis: this way, its central hole around the beam pipe will gather the charged beam component (and the beam halo) bent by the spectrometer magnet, while most of the kaon decay products will cross the active volume.

A mosaic of 18 hexagonal and 2 semi-hexagonal mirrors made of aluminium-coated 25 mm thick glass covered with a thin dielectric film, with sides 35 cm long, reflects the Čerenkov cone onto the RICH focal plane. In order to avoid absorption of light by the beam pipe, the mirrors actually form two independent spherical surfaces (shown in Figure 2.9(b)), with the foci corresponding respectively to the two PMT flanges[2]. The collective 34 m curvature radius of the mirrors layout results in a nominal focal length $f = 17$ m.

Detector simulations showed that the best compromise between the requirements of photon acceptance and angular resolution, and the need to maintain the cost of the apparatus at an affordable level, can be achieved by arranging 1952 photomultipliers on the vertices of the cells of a compact hexagonal lattice. The layout will consist of two support flanges, 70 cm in diameter, each hosting 976 Hamamatsu photomultipliers [32].

[2]In the following, the left and right sides will be often referred to as Jura and Salève flanges respectively, after the two mountains overlooking the Geneva area.

Since the RICH will also participate to the level 0 trigger, producing a primitive each time a charged particle crosses its volume, a time resolution of about 100 ps is required, leading to the choice of fast single-anode photomultipliers.

The RICH performance has been tested and verified with particle beams using a full length prototype [8].

2.3.4 The charged hodoscope

A scintillator hodoscope (CHOD) will provide a fast signal to trigger data acquisition on the passage of a charged particle. In addition, CHOD information will be combined with data from the RICH in order to provide primitives useful for the selection of π^+ tracks at subsequent trigger stages. Due to its excellent time resolution ($\sigma_t \simeq 200$ ps), this detector will also be a useful tool during the offline analysis, allowing to match the detected track with that of the decaying kaon.

Initially, in 2014, the hodoscope will be a refurbished version of the detector used by the preceding NA48 experiment, to be later replaced with a new detector. The existing NA48 CHOD consists of two planes of 64+64 plastic scintillator tiles aligned respectively to the x and y directions. The scintillation light from the counters is collected by Photonis photomultipliers via short Plexyglas light guides. The "NEWCHOD" is currently in design phase. The proposed detector should be built with scintillator tiles of variable dimension according to the distance from the beam, so that the particle rate is below 500 kHz in each counter. Light would be read out via sets of wavelength-shifting fibres positioned in order to maximise the detection efficiency [44].

2.3.5 The muon veto detectors

In addition to what achieved with the RICH detector, further muon suppression is needed up to a level of 10^{-5}. A calorimetric **muon veto** system will fulfill this requirement in two steps [32]:

1. A fast muon veto detector (**MUV3**), with time resolution $\sigma_t \leq 1$ ns, will reject events featuring coincident signals in the GTK and CEDAR detectors. This will lead to a pions to muons ratio of about 20 already at the first trigger level.

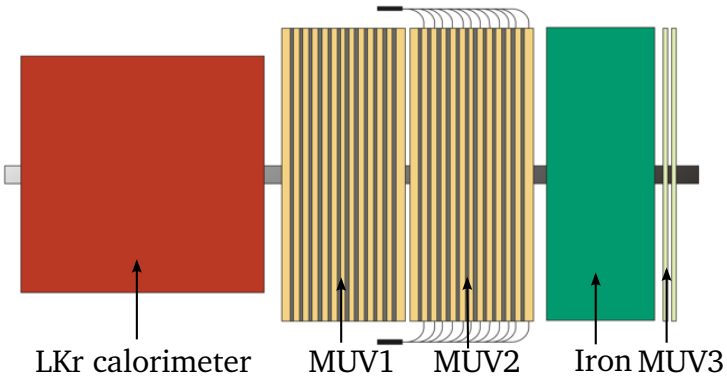


Figure 2.10: Layout of the three MUV detectors.

2. Two segmented calorimeters (**MUV1** and **MUV2**) will require crossing particles to deposit a significant amount of energy. In addition, measurements of the shower shape will allow to distinguish those muons undergoing catastrophic Bremsstrahlung or direct pair production from hadrons.

The first two modules, placed around the beam line next to the LKr calorimeter, are composed of alternate layers of scintillator (10 mm thick) and iron (25 mm thick). The total thickness of each module is 62.5 cm. The scintillator bars, 130 cm long and 4 to 6 cm wide, are alternatively oriented along the vertical and horizontal directions.

The third module is placed downstream of an 80 cm thick iron wall, and serves as fast level 0 trigger. It consists of a matrix of $20\times20\times5$ cm^3 scintillator blocks read out by photomultipliers.

Figure 2.10 sketches the layout of the three muon veto stations.

Part II

A RICH-based online trigger for $K^+ \to \pi^+\pi^0$ rejection: simulation and design

3 An online trigger using the RICH detector

Contents

3.1 Purpose

NA62 was designed to collect approximately 100 $K^+ \to \pi^+\nu\bar{\nu}$ events in two years of data taking. This exceptional statistics will make it possible to probe this ultra-rare K^+ decay channel with unprecedented precision.

Accounting for an acceptance of signal events between 10% and 20%, the experiment was planned in order to allow about 10^{13} K^+ decays in the fiducial region. With an unseparated ∼800 MHz hadron beam containing approximately 6% kaons, the output event rate of the detectors will be of the order of magnitude of 10 MHz.

However, a 10 MHz input rate is unsustainable for any reasonable-sized offline data acquisition system. A multi-level trigger system was therefore devised, with the purpose of scaling the data-saving rate down to few tens

of kHz. The following list describes the trigger chain to be used in the NA62 experiment:

L0: hardware synchronous level. Data rate reduction from 10 MHz to 1 MHz, with a maximum latency of 1 ms, using a few fast detectors only.
L1: simplified reconstruction of single detectors. Data rate reduction from 1 MHz to about 100 kHz.
L2: complete information, full reconstruction. Data rate reduction from about 100 kHz to about 15 kHz.

The L0 is a fast hardware system that collects information from a few detectors and uses it to perform a fast event rejection before readout from the temporary data buffers to the online PC farm. At this stage, simple assumptions may be done in order to discard a set of data clearly due to background processes, such as events with more than one charged track, or featuring signals on any chamber of the muon veto system. Chapter 3.3 describes how the L0 trigger signal is produced.

The maximum latency of the level 0 trigger, i.e. the available time for the first "decision making" process, was preliminarily set at 1 ms, a remarkably large value compared to other High Energy Physics experiments. In principle, this latency is large enough to allow for deeper analysis on separately read detectors.

The RICH detector was described in Chapter 2.3.3. Thanks to its fast response, it will be used in the L0 trigger by providing a multiplicity count. Much more information is however available from the RICH. The features of the rings generated by charged particles crossing the RICH volume would provide the direction and the velocity of the particle as independent measurements. Evaluating this information at the online trigger stage would allow an important reduction of the event flow. However, since the readout system of the RICH detector consists in a matrix of photomultipliers, information about the particle is available only after a first stage of ring identification. This identification would also need to be performed online, which requires very high-speed processing units.

Recent years witnessed the development of a new trend in Information Technology, i.e. "General Purpose computing on Graphics Processing Units" (**GPGPU**). Despite being originally designed for 3D computer graphics, modern commercial GPUs are often exploited to efficiently perform scientific

Sub-detector	Stations	Channels per station	Tot. channels	Hit rate (MHz)	Raw data rate (GB/s)
CEDAR	1	240	240	50	0.3
GTK	3	18000	54000	2700	2.25
LAV	12	320–512	4992	11	0.3
CHANTI	1	276	276	2	0.04
STRAW	4	1792	7168	240	2.4
RICH	1	1912	1912	11	0.09
CHOD	1	128	128	12	0.1
IRC	1	20	20	4.2	0.04
LKr	1	13248	13248	40	22
MUV	3	176–256	432	30	0.6
SAC	1	4	4	2.3	0.02

Table 3.1: Payload rates for the 12 sub-detectors of the NA62 experiment. These estimations date back to 2010 [32], and some detectors have been slightly revisited since then.

computations. Recently, major vendors such as Nvidia and ATI began to sell processing units and API (Application Programming Interface) libraries specially designed for this purpose.

In particular, today the computing power and parallel structure of GPUs seems capable to meet the timing requirements of a low level trigger for High Energy Physics experiments. The idea at the basis of this Master's thesis is to exploit the computing capability of GPUs in order to analyse the extra track information provided by the RICH detector at the L0 trigger stage.

3.2 Trigger and Data Acquisition in NA62

The high rate of events and the presence of 12 sub-detectors, for a total of about 90000 readout channels, results in such a high output data rate that it is impossible to save them on disk without some type of filtering. Table 3.1 summarizes the typical payload rates for the primary sub-detectors. A *trigger* system is therefore needed, which should identify the events to be saved and reject the rest.

The NA62 experiment will feature a unified *Trigger and Data Acquisition* (TDAQ) system. Trigger information will be assembled from readout-ready digitized data, simplifying the subsequent acquisition process [32]. Each

sub-detector readout system will be able to run individually, driven by a common 40 MHz clock generated by a single high-stability oscillator and distributed through optical fibres by the Timing Trigger and Control (TTC) system designed for the LHC[1]. The building block of the TDAQ system will be a common general-purpose integrated trigger and data acquisition board developed in Pisa, nicknamed TEL62 (*Trigger ELectronics for NA62*) [5].

The TEL62 board is an upgraded version of the TELL1 board used by the LHCb experiment at CERN [31]. The mechanical and electrical architecture has been maintained for compatibility, but the new board hosts more powerful FPGAs[2] and a large amount of DDR2 memory, whose size determines the maximum latency of the trigger process. In fact, four Altera Stratix-III pre-processing FPGAs (*PP*) are connected to dedicated TDC[3] boards and to 2 GB circular memory buffers, where data is stored during real-time evaluation. A central FPGA of the same type, named SyncLink (*SL*), is connected to the PPs and to an output mezzanine through high-speed buses. The SL links data and trigger primitives from all the PPs, stores data in Multi-Event Packets (*MEP*), and finally sends them to the output board [5].

Four high performance TDCs are mounted on custom TDCB boards specifically developed for this application. TDCBs (Time to Digital Converter Boards) are able to service 128 sub-detector channels with a time resolution of 100 ps. Four such boards can be plugged on a TEL62, for a total of 512 channels [25]. The output mezzanine, named Quad-GbE, is the same equipped by the original TELL1 board, and hosts four 1 Gbit Ethernet links that can be used to connect the TEL62 board to the central L0 Trigger Processor (*L0TP*) or to each other in a daisy-chain configuration. Several trigger primitives can be packed together into Multi-Trigger Packets (*MTP*), allowing a custom number of separate event primitives from the same detector to be transmitted to the L0TP at the same time, optimizing the bandwidth usage.

A hardware real time trigger, labelled *L0*, will command sub-detectors to transfer data through TEL62 boards to a PC farm through Gigabit Ethernet links, where data will be stored upon satisfaction of various criteria organized in levels of increasing complexity and sequentially controlled. Data are stored in UDP[4] packets for all Ethernet communications, because the

[1] http://ttc.web.cern.ch/TTC/intro.html
[2] Field Programmable Gate Arrays.
[3] Time to Digital Converter
[4] User Datagram Protocol

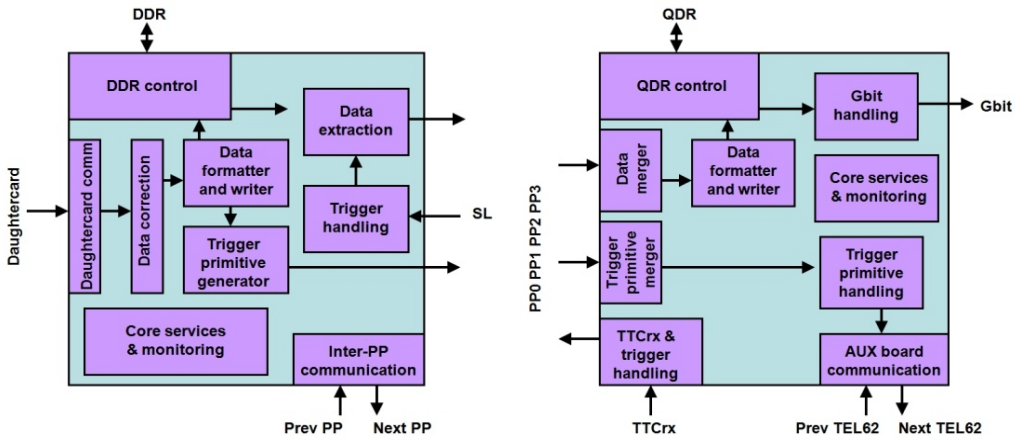


Figure 3.1: L0 trigger primitive production: PP and SL firmware block diagrams [5].

simple transmission model of this protocol allows to avoid data processing overhead at the network interface level. Moreover, the UDP header consists of only 8 B of information, and therefore allows for a large payload. A detailed description of the L0 trigger will be given in Section 3.3.

The software-based trigger hierarchy is organized in two levels, labelled *L1* and *L2* respectively, performed on raw data stored on a dedicated PC farm:

L1 Trigger decisions are taken independently, based on each complete sub-detector system; however, it is possible to perform logical operations on the outputs of different sub-detectors.

L2 Trigger decisions are taken on partially reconstructed events, based on pieces of information assembled from different sub-detectors.

3.3 The standard L0 trigger in NA62

The existing online trigger is hardware-implemented. Data are locally evaluated by each sub-detector involved. Each time a local trigger condition is satisfied, the sub-detector feeds a timestamped information into a central L0 processor. The latter matches such trigger primitives among all the detectors involved in the formation of the L0 trigger signal; it then broadcasts back a timestamped acquisition signal to the sub-detectors when the whole set of L0 requirements is satisfied. Triggered by this signal, the local front-end systems feed the corresponding data to the TDAQ PC farm.

The TEL62 firmware is sketched in Figure 3.1. At the online TDAQ level, the data flow is organized as follows:

1. Detector hits are digitized on the TDCB mezzanines.
2. The PPs receive data from TDCBs.
3. Part of the data is calibrated and analysed to generate L0 trigger primitives (this process is different for each sub-detector). Event data is stored into circular memory buffers.
4. The trigger primitives from all the PPs are collected by the SL, and merged if their time windows overlap.
5. A timestamped trigger primitive is assembled on the SL.
6. A number of trigger primitives is stored. When the predefined MTP size is reached, the Multi-Trigger packet is sent to the L0TP through a dedicated Ethernet link.
7. The L0TP matches the primitives from all the L0 trigger sources, and decides if a timestamped L0 trigger signal should be communicated to all the TEL62s.
8. Upon receiving a trigger signal, the SL commands the PPs to fetch data associated to the corresponding timestamp from the memory buffers.
9. Triggered event data are assembled on the SL until the predefined MEP size is reached.
10. A Multi-Event packet is assembled, and sent to the L1 PC farm through Ethernet links.

A L0 trigger based on a few simple, uncorrelated conditions is set up in order to quickly reduce data rates [62]. The sub-detectors responsible for the L0 trigger need to provide identification of charged tracks (**CHOD** / **RICH**), and vetoing of the most important backgrounds (**LKr** calorimeter for 2π and 3π events, and **MUV** for muons).

The following conditions were therefore devised to perform a first selection of $\pi\nu\bar{\nu}$ events over background [6]:

CHOD	At least one track candidate
RICH	Hit multiplicity: $5 \leq n \leq 32$
MUV3	No hits (MUV3 is the most downstream muon veto)
LKr	No more than one quadrant with energy deposit $E_{\mathrm{LKr}} \geq 5$ GeV

An optional additional contribution to the L0 trigger can be provided by the LAV:

LAV	No photons detected

Component	Initial	**CHOD**	**RICH**	**MUV3**	**LKr**	L0 output
$K^+ \to \pi^+\pi^0$	2140	1732	1179	1089	539	25.2 %
$K^+ \to \mu^+\nu_\mu$	6585	4204	3817	29	29	0.4 %
$K^+ \to \pi^+\pi^+\pi^-$	579	458	225	180	171	29.5 %
$K^+ \to \pi^+\pi^0\pi^0$	182	156	98	89	18	9.9 %
$K^+ \to \pi^0 e^+\nu_e$	525	403	253	248	84	16 %
$K^+ \to \pi^0\mu^+\nu_\mu$	347	272	194	17	14	4.0 %
Total rate		7226	5765	1651	854	11.8 %

Table 3.2: Rate of events inside NA62 acceptance after each step of L0 trigger. All rates are expressed in kHz. Study performed with a set of 10^5 simulated events for each K^+ decay mode. The last line highlights the total event rate rescaled to the firing rate of the CHOD detector, to which the beam upstream and downstream muon halos contribute as well. Data from [6].

In the NA48 hodoscope, a primitive featuring at least one coincidence between corresponding quadrants on the vertical and horizontal planes defines a track candidate. Charged particles may start showers in the Liquid Kripton Calorimeter: therefore, no multiple (spatially separated) energy deposit clusters are allowed on the LKr, since this would mean that more than one pion have been detected, or protons or $e^\pm$. This condition is intended to set an upper limit to the number of hadrons detected in the same event. However, the possibility of two showers clustering in the same sector of the calorimeter, within its spatial resolution, limits the 2π and 3π rejection level achievable in this way.

Table 3.2 reports the most recent simulation study of the rejection capability of each of the conditions listed above [6].

While the existence of a track detected by the CHOD sets the main acceptance window, the condition on the multiplicity of hits on the RICH detector effectively weakens the 3π and beam halo components (the latter is not shown in the table). Events featuring one or more π^0 are also partially rejected in this way, due to frequent photon conversion. A slight increase of π^0 rejection would be enabled by the availability of LAV sub-detectors. The condition on the LKr cluster multiplicity helps rejecting events with photons, and the contribution of the most downstream muon veto sub-detector allows for a muon suppression to the level of percent.

3.4 Use of GPUs in triggers

Researchers from all over the world are beginning to develop GPU algorithms for a variety of applications, including medicine, finance, chemistry, experimental and theoretical physics, network science and many more. This new branch of research is often referred to as **GPGPU**, i.e. General-Purpose computing on Graphics Processing Units, a term coined in 2002 which refers to an early trend of using GPUs for non-graphics applications [34].

High Energy Physics experiments rely on the acquisition and analysis of large amounts of information. Modern particle detectors feature very fast time responses, as required by the always increasing pace of accelerator technology. Most detectors feature a binned bi- or tri-dimensional geometry, providing accurate spatial information.

Such structured systems would natively benefit from parallel-oriented analysis techniques. Moreover, particle physics computing is an intrinsically parallel problem: experiments produce lots of of events, which are computationally *local*. Analysis is indeed carried out independently for each event on different memory locations. Hence the use of *multi-thread* techniques is expected to bring huge improvements in data processing.

Experiments such as ATLAS and ALICE at the LHC are starting to explore the advantages of using GPU (Graphical Processing Unit) programming in their *High-Level Triggers* (HLT), in order to perform faster analysis on Terabytes of recorded data. In particular, a track selection algorithm was developed for ALICE, that combines a fast Cellular Automaton[5] method for pattern recognition and a Kalman Filter[6] for track fitting [30]. Kalman Filters are also used in ATLAS to speed up the process of vertex finding. ATLAS is also designing a parallelized version of a toolkit for Multi-Variate Analysis of events [45, 70].

The Bejing Spectrometer (BES-III) experiment at BEPC, designed to perform a thorough study of the spectra of light hadrons, is successfully using a framework for partial wave analysis implemented on graphics processors [15]. Other ion and nucleon spectroscopy facilities such as FAIR (Facility for

[5] Cellular automata are discrete deterministic mathematical models for scientific computation. An automaton consists of a grid of cells whose evolution is a function of the current state of the cell and its two immediate neighbors only.

[6] A Kalman Filter is a statistical algorithm that produces an estimate of the system state by evaluating several streams of "noisy" data.

Antiproton and Ion Research at the GSI in Darmstadt) already make use of GPUs in their simulation and data analysis frameworks [52].

A project is being developed within the NA62 collaboration, which aims to integrate GPUs into the lowest-level trigger for the first time in High Energy Physics research. The use of GPUs in such a hard real-time system has not been attempted so far, but it looks a realistic and challenging possibility. The online use of GPUs would allow the computation of complex trigger primitives while providing a highly scalable trigger architecture. Emergent increased speed requirements would be handled just by adding more GPUs. Cards with hundreds of parallel floating-point units are extremely powerful hardware available at a relatively small cost, thanks to the continuous development driven by the huge market of computer games and by the broad excitement about GPGPU applications.

3.5 The $K^+ \to \pi^+\pi^0$ background

The aim of this work is to further reduce the rate of $K^+ \to \pi^+\pi^0$ events fed to the L1 farm. This is the most prominent background after the standard L0 trigger selection, as shown in Table 3.2.

Among the kinematically constrained backgrounds (see Chapter 1.4), $K^+ \to \pi^+\pi^0$ is the only process falling in the same missing mass region as the signal. This $2-$body process features a closed kinematics and could therefore be detected as a Dirac delta in the spectrum of the missing mass in the decay of the beam kaon into π^+ (neglecting resolution effects). When the distribution of experimental errors is convoluted with the expected spectrum for this process, the squared missing mass m^2_{miss} distributes as a Gaussian around the squared mass of the neutral pion.

From an experimental point of view, the signature of the $K^+ \to \pi^+\pi^0$ process consists of one or more charged tracks detected (to account for the charged pion and the eventual conversion of photons from π^0 decay).

This process may yield multiple Čerenkov rings in case of:

- inelastic scattering on the material of the beam pipe or of the upstream detectors
- Dalitz decay of the neutral pion: $\pi^0 \to e^+e^-\gamma$
- photon conversion $\gamma \to e^+e^-$ in a sufficiently upstream region.

The identification of a Čerenkov ring on the RICH detector provides information on both the direction and the velocity of the particle that has crossed the detector volume. In principle, the relationship between these quantities can tell whether a detected pion comes from a 2-body decay or not. Let us go through the kinematics involved in the $K^+ \to \pi^+\pi^0$ decay.

Let P_K, P_π and P_0 be the four-momenta of the kaon, of the charged pion and of the neutral pion respectively. In the centre of mass frame, the equality $P_K = P_\pi + P_0$ reads

$$\begin{cases} m_K = E_\pi^* + E_0^* \\ \vec{0} = \vec{P_\pi^*} + \vec{P_0^*} \end{cases} \tag{3.1}$$

where E_π^* and E_0^* are the centre of mass energies of the charged and neutral pion respectively. The second relation can be written as

$$(E_\pi^*)^2 - m_\pi^2 = (E_0^*)^2 - m_0^2 \tag{3.2}$$

and therefore, using the first of Eqns. 3.1, we find the centre of mass energy of the π^+, that is single-valued:

$$E_\pi^* = \frac{m_K^2 + m_0^2 - m_\pi^2}{2m_K} \tag{3.3}$$

Now consider the invariant product $P_K^\mu P_{\pi\mu}$. In the rest frame of the decaying kaon it reads

$$\left(P_K^\mu P_{\pi\mu}\right)^* = m_K E_\pi^* - \vec{0} \cdot \vec{P_\pi^*} = m_K E_\pi^* \tag{3.4}$$

On the other hand, in the laboratory frame we have to consider the momenta of both particles:

$$P_K^\mu P_{\pi\mu} = E_K E_\pi - \vec{P_K} \cdot \vec{P_\pi} = E_K E_\pi - |\vec{P_K}||\vec{P_\pi}| \cos\theta_{K\pi} \tag{3.5}$$

where $\theta_{K\pi}$ is the angle comprised between the K^+ and the π^+ tracks.

The position of the Čerenkov ring of the pion provides a measurement of the direction of the pion, which can be related to $\theta_{K\pi}$ by means of the simple backwards propagation steps described in Chapter 4, and of the velocity β_π of the pion. Therefore, if we assume that the detected particle is really a charged pion, we can compute its energy and momentum:

$$E_\pi = \gamma_\pi m_\pi = \frac{m_\pi}{\sqrt{1-\beta_\pi^2}} \tag{3.6}$$

$$|\vec{P_\pi}| = \beta_\pi E_\pi = \frac{\beta_\pi m_\pi}{\sqrt{1-\beta_\pi^2}} \tag{3.7}$$

If we substitute Eqns. 3.6 and 3.7 in Eqn. 3.5, and match the two expressions 3.4 and 3.5 for the $P_K^\mu P_{\pi\mu}$ invariant, we find that the velocity of the pion is constrained by the laboratory-frame decay angle $\theta_{K\pi}$ through the relation

$$\theta_{K\pi}(\beta_\pi)\Big|_{K^+\to\pi^+\pi^0} = \arccos\left[\frac{1}{\beta_\pi|\vec{P_K}|}\left(E_K - \frac{m_K}{m_\pi}E_\pi^*\sqrt{1-\beta_\pi^2}\right)\right] \quad (3.8)$$

that arises from the closed kinematics of 2-body decays.

In principle, E_K and $|\vec{P_K}|$ are beam parameters that are known to a precision of 1%; therefore, a RICH measurement alone would allow to check if Eqn. 3.8 holds. In case of a positive result, a $K^+ \to \pi^+\pi^0$ event could be identified and safely discarded.

3.6 Feasibility study

The GPU rejection algorithm for $K^+ \to \pi^+\pi^0$ events would add to the hardware-implemented L0 trigger described in Section 3.3.

As a first step, I had to verify:

- the resolution of the process of reconstructing the particle kinematic variables by fitting a Čerenkov ring to the hits on the photomultiplier flanges;
- the background rejection and signal acceptance: these quantities have a strong dependence on the reconstruction resolution;
- whether the data flow and algorithm execution chain speed can be optimized in order to match the latency of the L0 trigger (1 ms).

Since positive results from such feasibility studies were a prerequisite for the conception of the actual GPU algorithm, a thorough analysis was performed by means of the NA62 software framework. The NA62 detector is still being commissioned, therefore I have used Montecarlo simulations in order to define the design of a RICH-based trigger. The results of this study are reported in Chapter 5.

4 RICH reconstruction

Contents

The project described in this thesis was partly carried out by means of the NA62 software. "NA62FW", i.e. the NA62 software suite, is still being actively developed. It includes a Montecarlo event generator, featuring Geant4[1] modelling and event reconstruction routines for most NA62 detectors. This framework makes use of many Physics utility libraries: detector geometry and particle propagation are implemented through Geant4 *physics lists*, while some ROOT classes handle input/output and data processing, and define hit and event structures. While this thesis was being developed, hardly any Physics analysis algorithm was implemented in NA62FW. The author designed the reconstruction and track propagation methods described in this Chapter.

In order to consider all the material in which particles could interact, I simulated $K^+ \to \pi^+\nu\bar{\nu}$ and $K^+ \to \pi^+\pi^0$ decays enabling all the subdetectors upstream of the RICH, that was also included. I then used the reconstruction routines supplied with the NA62 software to examine the events detected by the RICH. Figure 4.1 shows a typical output of the RICH reconstruction routine: a Čerenkov cone emitted by a charged particle results in a ring of photomultipliers firing.

[1] Geant4 is a toolkit to simulate the passage of particles through matter. http://geant4.cern.ch/

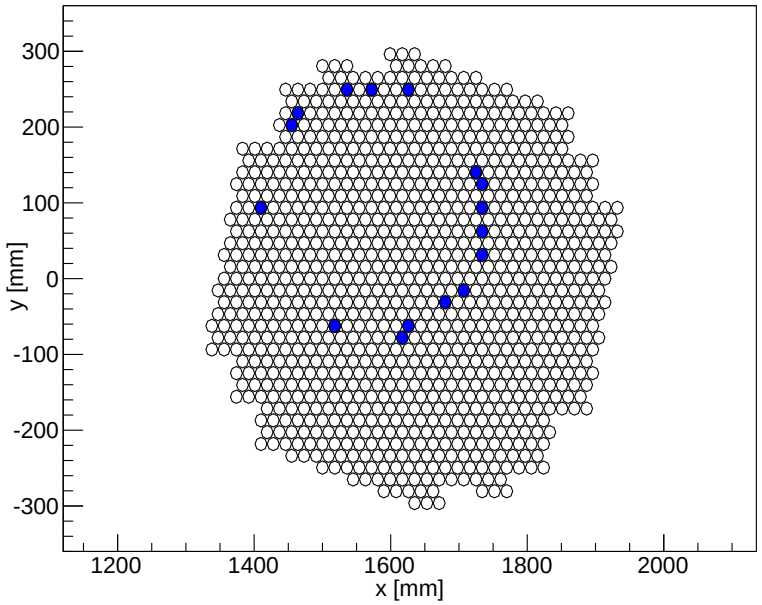


Figure 4.1: Čerenkov ring produced by a simulated π^+ with momentum $15 \leq P_z \leq 35$ GeV/c on one of the two PM spots of the RICH. Each circle represents a photomultiplier.

The RICH reconstruction algorithm that was provided by the NA62 software at the time of my work may be summarized as follows:

Algorithm 1: RICH event reconstruction

input : coordinates of a set of firing photomultipliers (RICH *event*)
output: Čerenkov ring parameters $(\vec{C}, R)$ and timestamp T

$\bar{x}$, $\bar{y} \leftarrow$ compute centre of gravity of the hits;

initial guess: $\vec{C} \leftarrow \bar{x}$, $\bar{y}$
initial guess: R $\leftarrow$ 18.8 cm
fit circle$(\vec{C}, R)$ to $\{\text{hit}\}_{i=0\ldots n_\gamma}$

$T \leftarrow$ compute average time of the hits

Here, $\bar{x}$ and $\bar{y}$ are the coordinates of the centre of gravity of the hits, $\vec{C}$ is the center of the fitted ring and R is its radius. This procedure shows some clear limitations. In fact, it only tries to adapt a single ring to the whole set of photon hits in input. RICH events often feature more than one charged

particle producing Čerenkov rings. Moreover, the geometry of the detector is not completely taken into account. Finally, the whole framework lacks simulation and reconstruction of piled-up events.

Therefore, I modified the code provided, and took the geometry features that were missing into consideration. I will briefly explain how I achieved this in Section 4.1.

Events pile-up and the possibility of detecting more than one ring were not included at this time. The strategy the experiment will adopt in order to manage the time superposition of events is still evolving, and anyway it does not represent a critical aspect of this project. Regarding the possible multiplicity of Čerenkov rings, a big effort has been put into the design of a GPU-based 'multiple rings' fitting algorithm. However, during the previous Montecarlo characterization step that allowed me to study the rejection efficiency of such trigger algorithm, the presence of more than one ring was represented simply as a high χ^2 in a single-ring fit.

My trigger algorithm requires some level of physics reconstruction on top of the detection of a Čerenkov ring. In fact, the kinematic quantities involved in the decay process are modified by the MNP-33 magnet of the *STRAW* spectrometer. In addition, the original direction of the decaying kaon has to be taken into account when computing the laboratory-frame decay angle $\theta_{K\pi}$. In Section 4.2 I will describe the procedure I used in order to reconstruct physical information about the process responsible for a detected RICH event.

4.1 Geometric corrections

The existing reconstruction algorithm included in the NA62 software did not take into account that:

- the vessel of the RICH detector is rotated with respect to the beam line, in order to minimize the possibility that beam particles spill into the detection volume (see Chapter 2.3.3). The rotation angle is tuned on the magnetic kicks due to the GTK and STRAW dipoles.
- the two mirrors reflecting photons onto the focal plane are tilted at different angles relative to the vessel axis. This asymmetry is meant to spread light onto the central regions of the photomultipliers flanges.

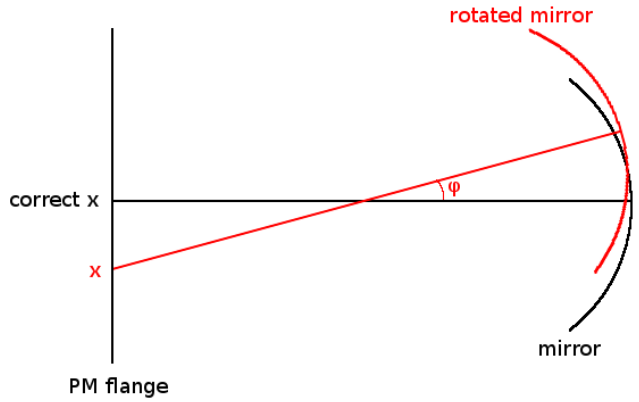


Figure 4.2: Sketch representing the effects of the tilt of the RICH mirrors.

Figure 4.2 illustrates the rotation of the mirrors. The tilts applied to the two mirrors are different, hence I corrected this effect on a hit-by-hit basis. On the contrary, I fixed the global vessel rotation through a single operation on the position of the centre of the Čerenkov ring, that is the only parameter affected.

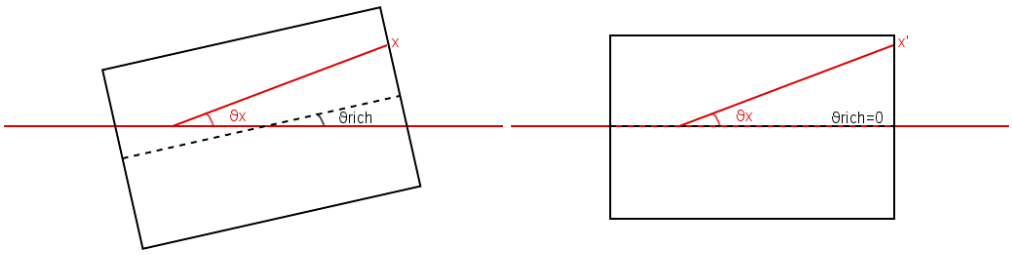
The 1952 PMTs are placed on the RICH flanges so that the first 976 are on the left-side frame, and the others on the right-side frame. Let φ and φ' be the Salève and Jura mirrors angles to the vessel axis, respectively. Hit by hit, I shift the x coordinate of the hit by a quantity φR or $\varphi' R$ depending on which spot is illuminated:

$$x \equiv \begin{cases} x - 2f\varphi & 0 \leq \text{Channel ID} < 976 \\ x - 2f\varphi' & 976 \leq \text{Channel ID} < 1052 \end{cases} \tag{4.1}$$

where $f = R/2$ is the focal lenght of the device, R being the curvature radius of the mirrors.

Let us now see how to account for the global rotation of the vessel, represented in Figure 4.3. I treated this by shifting the centre of the reconstructed Čerenkov ring along the $x-$axis by an adequate quantity. Let $\vec{C} = (x, y)$ be the centre $\vec{C}$ of the ring, θ_x be the direction of the particle projected onto the $x-$axis of the laboratory frame, and $\theta'_x \equiv \theta_x - \theta_{\text{rich}}$ the x component of the direction of the particle in the RICH frame. The following relation holds:

$$\tan\theta'_x = \frac{x}{f} \tag{4.2}$$



(a) **(b)**

Figure 4.3: Sketch representing the effects of the rotation of the RICH vessel.

Therefore we can expand θ_x as:

$$\theta_x = \theta'_x + \theta_{rich} = \arctan\frac{x}{f} + \theta_{\text{rich}} \tag{4.3}$$

$$\tan\theta_x \equiv \frac{x'}{f} \Longrightarrow x' = f\tan\left(\arctan\frac{x}{f} + \theta_{\text{rich}}\right) \tag{4.4}$$

The resulting x' is the coordinate where the centre of the ring should be moved to account for the vessel rotation. The y coordinate remains unchanged.

These two simple geometrical corrections allow the achievement of an angular resolution of the order of $\sigma(\theta_x) \leq 450$ μrad. Figure 4.4 shows the reconstruction accuracy achieved for simulated $K^+ \to \pi^+\pi^0$ events, computed using 4 different subsets of data:

- events with hits on one spot only (Jura or Salève),
- events with rings spread across the two spots,
- the complete set of events.

4.2 Track propagation: upstream magnets

Through the assessment of the parameters of the Čerenkov ring, we can now compute the components of the direction of the particle crossing the RICH:

$$\theta_x = \arctan(x/f) \tag{4.5}$$

$$\theta_y = \arctan(y/f) \tag{4.6}$$

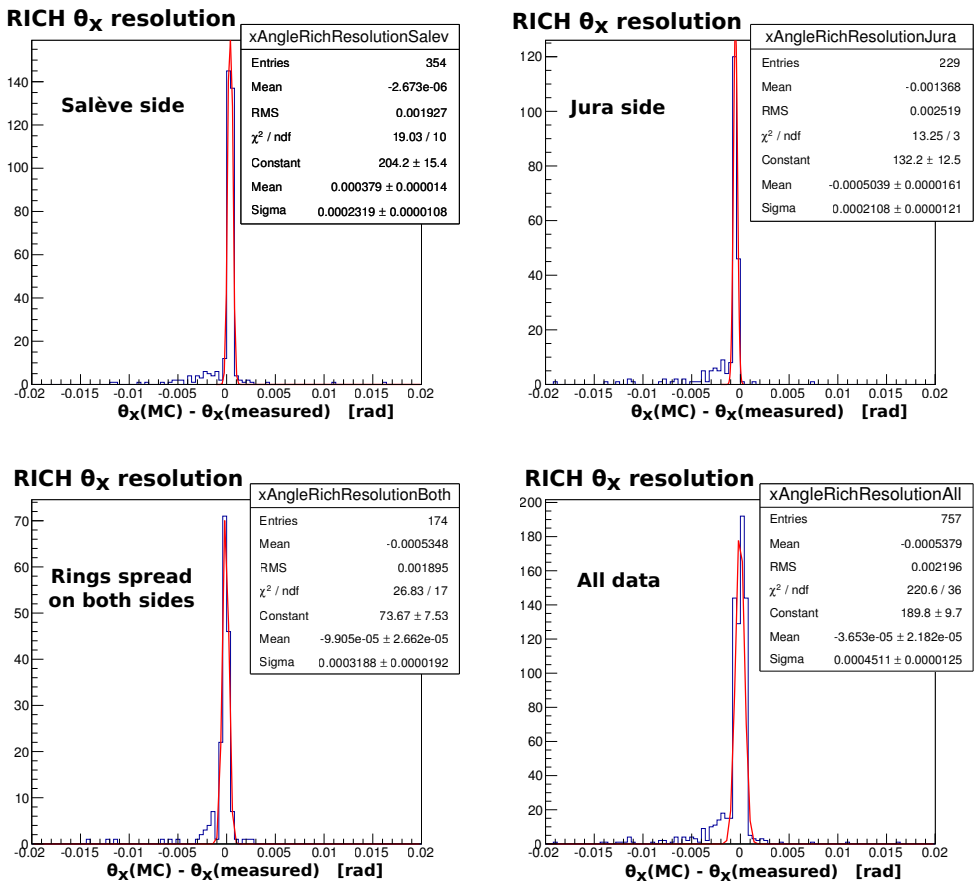


Figure 4.4: Resolution achieved on the x component of the direction of pions from $K^+ \to \pi^+\pi^0$ crossing the RICH detector.

where x and y are the coordinates of the ring centre, already corrected for the RICH geometry as explained in Section 4.1.

In order to trace the decay angle $\theta_{K\pi}$ back, we must propagate the detected track backwards to the decay vertex. This implies reverting the magnetic kick due to the dipole of the downstream spectrometer. If we assume that the particle is a pion, we can compute the kick given to it through the approximate formula

$$\theta_{\text{kick}} = \frac{P_{\text{kick}}}{P_\pi(\beta)} \tag{4.7}$$

valid for small deflections ($|P_{\text{kick}}/P_\pi| \approx 10^{-2}$ in the fiducial momentum region), where $P_\pi(\beta) = \beta m_\pi / \sqrt{1-\beta^2}$. Notice that we are using the value of β computed from the radius of the detected Čerenkov ring (see Chapter 2.3.3), introducing therefore an experimental uncertainty.

A sketch representing the original direction of the charged particle and its direction after the spectrometer magnet is shown in Figure 4.5. Once θ_{kick} is computed, we may trace the original direction $\theta_{x,\text{orig}}$ back by subtracting θ_{kick} from the direction θ_x at the RICH:

$$\theta_x = \theta_{x,\text{orig}} + \theta_{\text{kick}} \tag{4.8}$$

Like those described in the previous section, the corrections shown here only modify the x component of the direction of the particle.

Let us now assert the physical meaning of the angle we have just found. Downstream of the Gigatracker, the beam spectrometer described in Chapter 2, the emerging K^+ beam is bent at an angle θ_K on the x–z plane. The angle $\theta_{x,\text{orig}}$ we have just computed does not represent the decay angle, i.e. the angle between the decaying beam kaon and its charged daughter. All the directions defined so far are relative to the $z-$axis of the laboratory frame, that coincides with the direction of the beam upstream of the Gigatracker. The relationship between the computed angle and the x component of the true decay angle $\theta_{K\pi}$ is shown in Figure 4.6, where I used the following labels:

- $\theta_{x\pi}$ is the x component of the decay angle, i.e. of the angle measured in the laboratory frame between the kaon and pion tracks;
- θ_x is the angle measured between the direction of the detected particle and the $z-$axis of the laboratory frame ($\theta_{x,\text{orig}}$ in the above formulas);

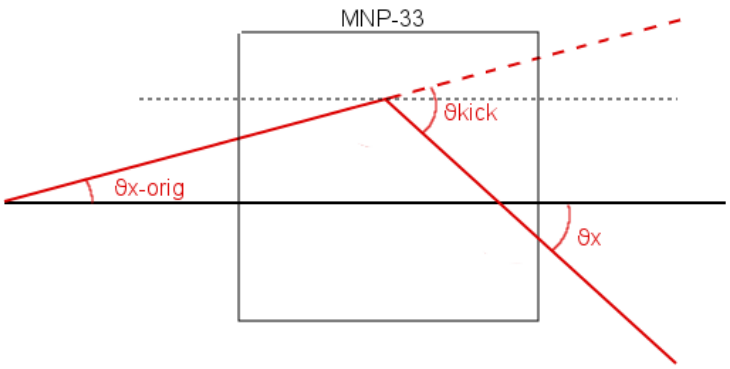


Figure 4.5: Sketch showing the effect of the spectrometer magnet on the direction of charged particles.

- θ_K is the direction of the beam, after being bent by the GTK magnet, relative to the laboratory frame.

Reverting to the notation I have used so far, we observe that the original decay angle $\theta_{K\pi}$ can be obtained with the following formula:

$$\theta_{x(K\pi)} = \theta_{x,\text{orig}} - \theta_{\text{kick}} - \theta_K \tag{4.9}$$

where $\theta_{x,\text{orig}}$ was defined in Eqn. 4.5, θ_{kick} is computed as in Eqn. 4.7, and θ_K was defined above.

4.3 Reconstruction accuracy

In the previous sections I examined the simple analysis chain I have used in order to extract physics information from the RICH data. The relevant parameters used are: [32]

- $\theta_{\text{rich}} = +0.00175571$ rad
- $P_{\text{kick}} = -270$ MeV/c
- $\theta_K = +0.0012$ rad

Let us summarize the outcome of the analysis.

- The velocity of the particle is computed from the radius of the detected ring as of Eqn. 2.14, and does not undergo any further modification.

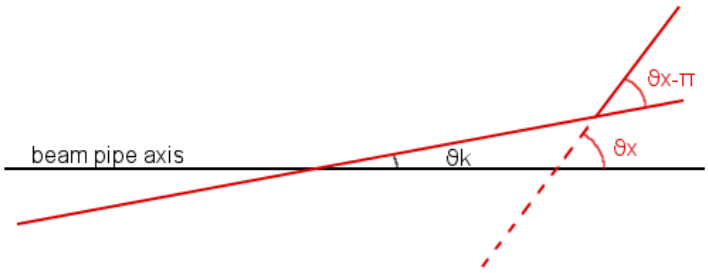


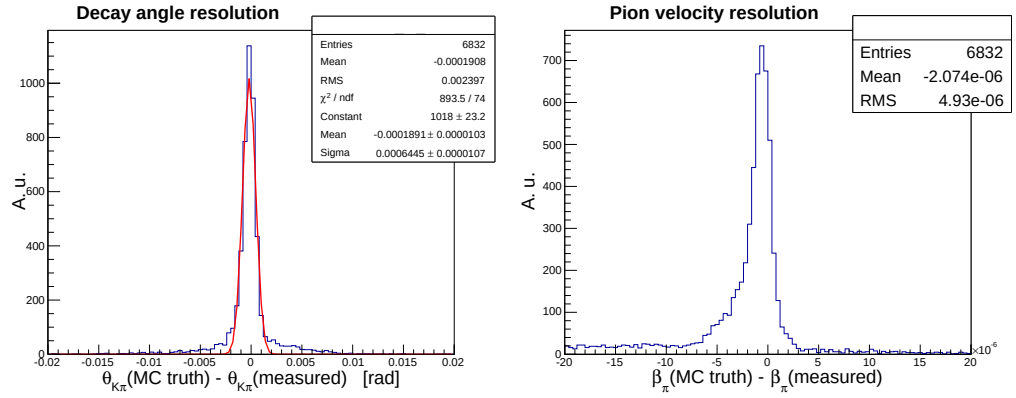
Figure 4.6: Sketch representing the decay angle $\theta_{K\pi}$ in the laboratory frame and in the rest frame of the decaying kaon.

- The first adjustment performed consists in a hit-by-hit shift of the PMT coordinates, needed in order to take the mirrors tilt into account.
- Once the hits are repositioned, we can fit a ring to the data. We then change the x coordinate of the found ring centre according to Eqn. 4.4.
- We now use the x and y coordinates to compute the θ_x, θ_y components of the particle direction (Eqns. 4.5, 4.6). Of these, θ_x is propagated back according to Eqn. 4.9.

We finally obtain the decay angle:

$$\theta^2_{K\pi} = \theta^2_{x(K\pi)} + \theta^2_y \tag{4.10}$$

Figure 4.7 shows the accuracy achieved in the reconstruction of β_π and $\theta_{K\pi}$. The decay angle is reconstructed with a resolution $\sigma(\theta_{K\pi}) \approx 650\ \mu$rad (Figure 4.7(a)). On the other hand, the distribution of β (Figure 4.7(b)) shows a slight asymmetry towards values higher than the true speed of the particle. This is probably due to an uneliminable effect of the discrete binning of photomultipliers. The asymmetry arises as the Čerenkov light emission angle is generally overestimated, as it can be inferred from Figure 4.8. In fact, the distributions shown here were obtained by use of the Taubin algorithm for circle fitting, described in Appendix A.3.1, which is one of the most robust single-ring fit procedures available with respect to the radius of the circle. Quoting the RMS of the distribution as a measurement of the reconstruction accuracy, we may state that the resolution in β is approximately $\sigma(\beta) \approx 5 \times 10^{-6}$.



(a) Decay angle resolution **(b)** Particle speed resolution

Figure 4.7: RICH reconstruction resolution.

Let us finally discuss a few critical aspects concerning this analysis. At the beginning of this section, the magnetic kick due to the spectrometer dipole was computed assuming the detected particle is a π^+, therefore using its mass in Eqn. 4.7. In fact, most events without pions will be rejected by the hardware L0 trigger. However, on some occasions we might still wrongly assume a particle is a pion and propagate it with a wrong formula. This would not in any way affect the efficiency for the signal of the trigger presented in this thesis, since events without charged pions would be discarded during the stage of offline analysis. Besides, the Montecarlo study reported in this work does not examine background channels different from $K^+ \to \pi^+\pi^0$, as the analysis process involved would be substantially different.

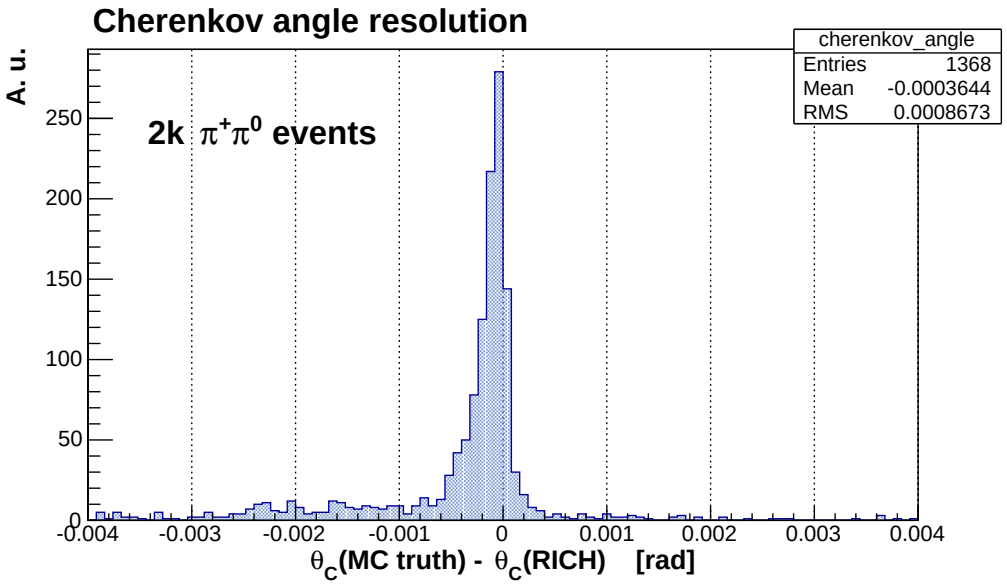


Figure 4.8: Reconstruction resolution on the Čerenkov light emission angle. An asymmetry is clearly visible between RICH reconstructed data and the "Montecarlo truth".

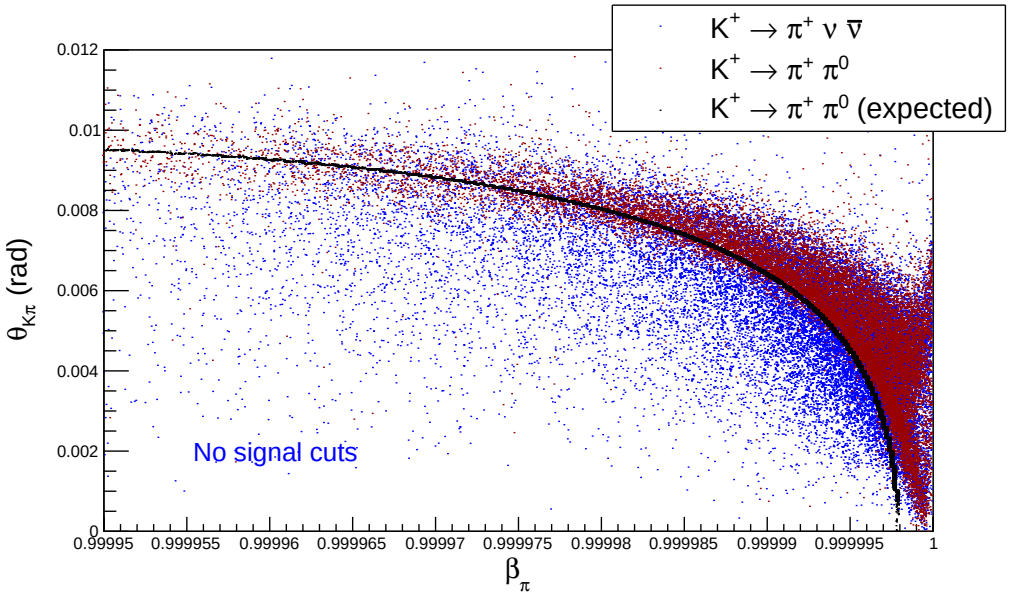
5 Trigger characterization

Contents

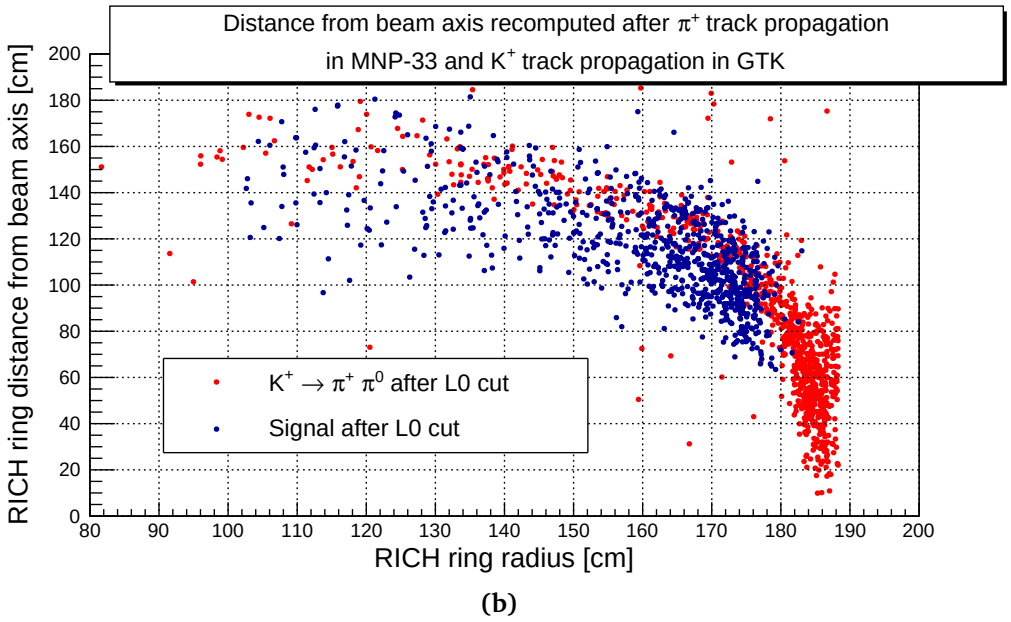
In order to achieve background rejection while preserving as much signal as possible, we must look for a set of variables whose distribution allows to separate $K^+ \to \pi^+\nu\bar{\nu}$ events from the target background $K^+ \to \pi^+\pi^0$. We may then set appropriate cuts on these distributions, and transmit to higher trigger levels only the portion of data fulfilling these requirements.

5.1 $\beta_\pi - \theta_{K\pi}$ correlation

In Chapter 3.5 I outlined the kinematics of the $K^+ \to \pi^+\pi^0$ decay and introduced Eqn. 3.8, that could be used to identify this background. The $K^+ \to \pi^+\nu\bar{\nu}$ decay is a 3-body process, and therefore its kinematics is not constrained. Background rejection could be obtained by determining $(\theta_{K\pi})_{2-body}$ as a function of the RICH-measured β_π for each event, and discarding those events whose reconstructed $\theta_{K\pi}$, computed from the direction θ_π detected at the RICH, is close to $(\theta_{K\pi})_{2-body}$.



(a)



(b)

Figure 5.1: Top plot **(a)**: the distributions of the signal and of the $\pi^+\pi^0$ background in the $(\theta_{K\pi})_{2-body} - \beta_\pi$ plane. Bottom plot **(b)**: the same plane is remapped to the raw Čerenkov ring observables, and L0 trigger and standard signal cuts (see Table 5.1) are applied to the data sets.

Unfortunately, the Montecarlo simulations I will report in this chapter have revealed that the resolution of the RICH is not sufficient to successfully exploit this approach. In Figure 5.1(a), the black dots corresponds to the exact mathematical relation between the decay angle and the velocity of the pion for the $K^+ \to \pi^+\pi^0$ decay, described in Eqn. 3.8. The red dots correspond to RICH-reconstructed $(\beta_\pi, \theta_{K\pi})$ pairs, computed as described in Chapter 4, for simulated $\pi^+\pi^0$ events, while blue dots represent simulated $\pi^+\nu\bar{\nu}$ events.

It can be inferred from Figure 5.1(a) that:

- $\pi^+\pi^0$ events (red dots) cover a wide area of the $(\theta_{K\pi})_{2-body} - \beta_\pi$ plane, and there is a slight asymmetry compared to the expected occupancy (black dots); this means that the resolution of a single detector might not be sufficient to perform such bi-dimensional selection.
- The regions occupied by the two data sets – signal (blue) and $\pi^+\pi^0$ background (red) – completely overlap.

It follows that a significant bi-dimensional cut set up on relation 3.8 would reject a large amount of signal events, and therefore it is not feasible.

Figure 5.1(b) is a re-parametrisation of the plot shown in 5.1(a). Here, the radius and position of the Čerenkov ring are put in the relation emerging from Eqn. 3.8:

$$d(r_c) = d\left(\theta\left(\beta\left(r_c\right)\right)\right) \tag{5.1}$$

where $d(\theta)$ is the ring centre distance to the focus, defined in Eqn. 2.12, $\theta(\beta)$ was defined in Eqn. 3.8, $\beta(r_c)$ in Eqns. 2.11 and 2.7 and r_c is the Čerenkov ring radius. The aim of this test was to understand if the spread of the distributions of Figure 5.1(a) was due to the intrinsic resolution of the RICH detector or if it was a consequence of computational issues arising during the evaluation of β_π and $\theta_{K\pi}$. From Figure 5.1(b) we see that such spread is due to the finite resolution of the RICH, and unfortunately this issue cannot be overcome in the contest of a real-time trigger. Moreover, the signal selection and standard L0 trigger cuts applied in this plot have the effect of thinning the area covered by $\pi\nu\bar{\nu}$ data, thus making a bi-dimensional selection of $\pi^+\pi^0$ unattainable.

5.2 Missing mass

In principle, what best distinguishes a 3-body from a 2-body decay is the energy spectrum of the decay products. In a 2-body decay, the energy spec-

trum of the π^+ in the rest frame of the decaying kaon is single-valued (see Chapter 3.5).

Instead, for a 3-body decay (such as the signal) the rest frame energy of the π^+ belongs to a continuum that spreads roughly from $m_K - m_\pi$ to as near to zero as our detectors can measure (the mass of the neutrinos is negligible with respect to the precision of these instruments). Figure 1.5 shows the shapes of the squared missing mass to the detected pair $(K^+,\ \pi^+)$ for the target process and the most important K^+ decay modes.

For the two-body decay this quantity is theoretically expected to distribute as a Dirac delta centred in $m^2_{\pi^0} \simeq 0.0182$ GeV^2/c^4. Taking the finite resolution of our detectors into account, we expect a Gaussian distribution around this value.

I explored the possibility of using the missing mass, computed using exclusively the information originating from the RICH detector, as a cut variable. However, as shown in Figure 5.2, due to the limited momentum resolution of the RICH it is not possible to separate the two processes this way either. In fact, the two sets of simulated events (respectively signal and $\pi^+\pi^0$ background) end up in wide superposition on this variable.

At the final level of data analysis, NA62 expects to put strong contraints on the events to be tagged as **signal**. These should be set as follows:

- $15 \leq P_\pi \leq 35$ GeV/c, i.e. the range of momenta for which the μ–π separation achieved is maximum ($P_\pi \geq 15$ GeV/c), with an upper limit that ensures that the other particles from K^+ decay carry a momentum of at least 40 GeV/c and are therefore detectable. This way, for example, at least one photon from $\pi^0 \to \gamma\gamma$ decay is bound to fall in the acceptance of the NA62 detector.
- $105 \leq z_{\text{vtx}} \leq 165$ m, where z_{vtx} is the z coordinate of the decay vertex, i.e. the K^+ decay happened in the fiducial region.
- $m^2_{\text{miss}} \in R_1 = (0, 0.01)$ or $\in R_2 = (0.026, 0.068)$ GeV^2/c^4, in order to explore only "background-free" regions (see Figure 1.5).

Events not fulfilling these requirements will be left out of the fiducial sample for data analysis. Such events would be discarded in any case at a later stage, but we may exploit the signal selection to reject them already at the online L0 stage. This will obviously reduce the amount of data fed to higher level triggers, thus allowing more efficient evaluations at those stages.

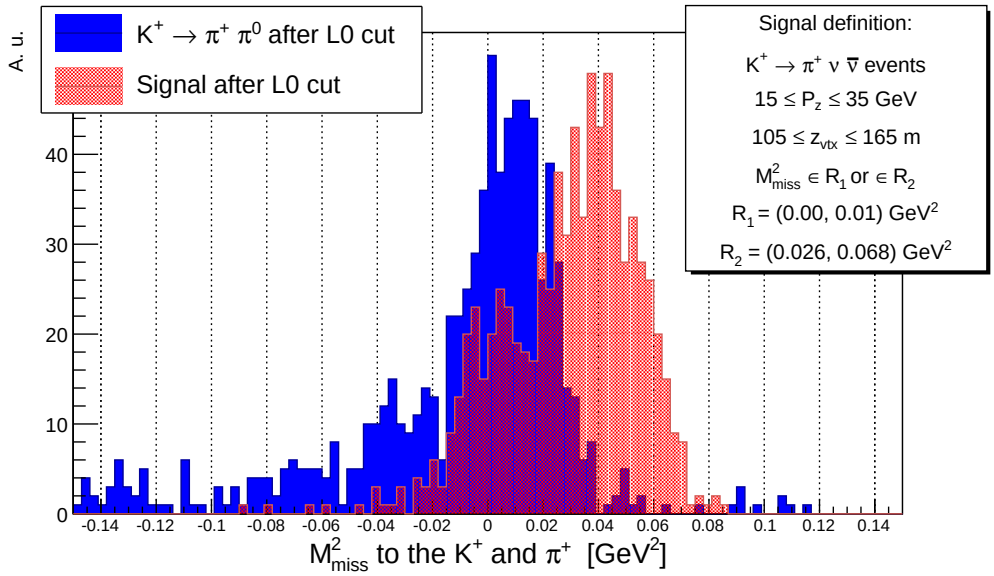


Figure 5.2: Spectrum of the missing mass to the K^+ and π^+ for the signal and the $K^+ \to \pi^+\pi^0$ background. The missing mass was computed using the information from the rings detected on the RICH, assuming that the particle crossing the detector was a π^+. These spectra were obtained from the superposition of two sets of simulated events, one containing only $K^+ \to \pi^+\nu\bar{\nu}$ and one only $K^+ \to \pi^+\pi^0$ decays.

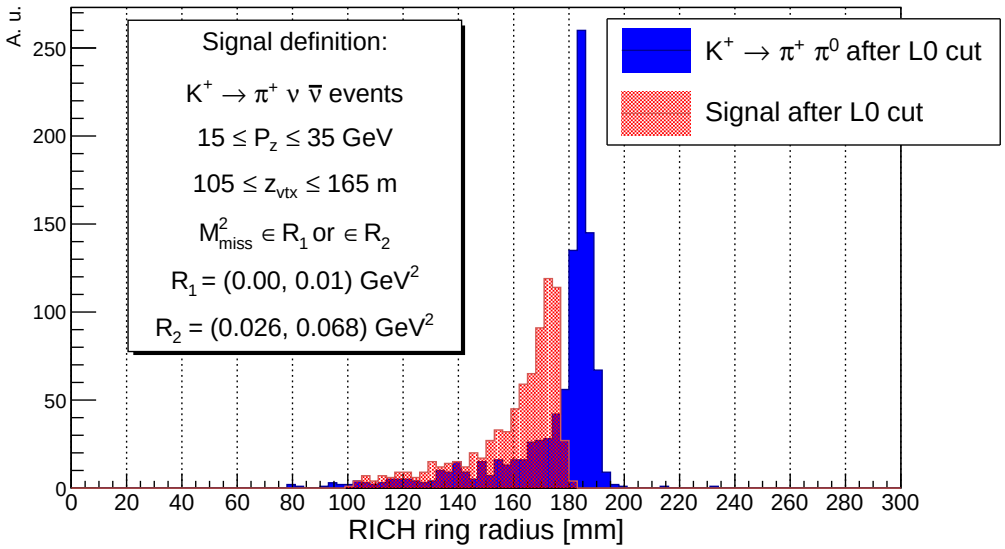


Figure 5.3: Reconstructed Čerenkov ring radius for the signal and the $K^+ \to \pi^+\pi^0$ background after L0 cuts have been applied (Montecarlo-simulated data).

5.3 Čerenkov ring radius

First of all, we notice that the signal is distributed along a limited range of momenta, 15 to 35 GeV/c. This means that the velocity of a charged particle should be in a range defined by its mass, in order for the particle not to be discarded. In the RICH detector, this results in limits to the radius of the Čerenkov rings. A Montecarlo simulation performed with the NA62 software allowed me to find the distributions of the radius of the Čerenkov rings for signal and background events. The results are shown in Figure 5.3, where we can see that this quantity provides a good option for events separation.

In particular, we can see that the peak value of the Čerenkov radius is on average larger for $\pi^+\pi^0$ events than it is for $K^+ \to \pi^+\nu\bar{\nu}$ events.

If the signal had not been momentum-selected, we would have expected the peak of $\pi^+\pi^0$ events to be lower than that of the signal, due to the different kinematics of the two processes. In the two-body decay, the charged and neutral pion masses are similar, and therefore they carry similar fractions of the kaon energy, about 50% each in the rest frame of the kaon. On the other hand, the sought 3-body decay with two neutrinos contains only one pion, and it is the only particle with a mass of the same order of magnitude as that

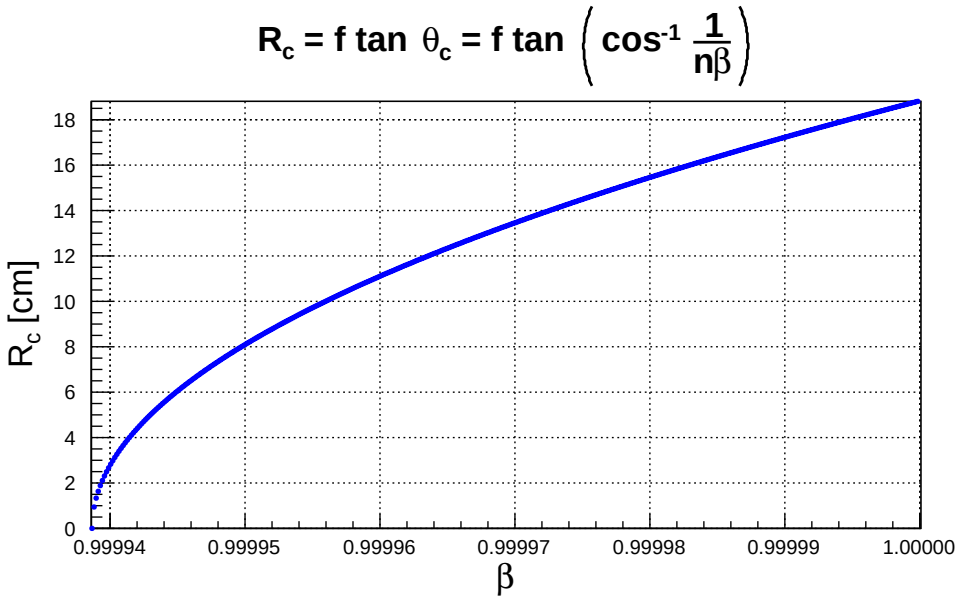


Figure 5.4: Expected radius of the Čerenkov ring for a range of velocities from $\beta = 1/n$ to $\beta = 1$. In this study I used $f = 17$ m and $n = 1.0000613422636$, the design values of the RICH detector for NA62.

of the kaon. In this case, the larger fraction of the kaon energy is statistically carried by the pion, and smaller fractions contribute energy to the neutrinos. As a consequence, in the $K^+ \to \pi^+\nu\bar{\nu}$ decay the π^+ is on average faster than when originating from $K^+ \to \pi^+\pi^0$ events. A higher velocity β produces larger circles on the focal plane of the RICH detector, as shown in Figure 5.4.

However, the fiducial momentum region defined for the signal allows us to discard a priori those events where the detected pion has a momentum greater than 35 GeV/c. This way, as Figure 5.3 suggests, we can suppress a major component of the $\pi^+\pi^0$ background.

5.4 Other possible optimizations

As discussed in Section 2.3.3, the RICH detector not only allows to perform particle identification by distinguishing between muons and pions: it also provides information about the flight direction of the charged particle crossing its volume. Figure 5.2 shows that it is inefficient to use the missing mass as a discriminating variable, with the resolution available online. However, there may be some other variables which could exploit the whole information we get from both the detection and our understanding of the decay

process responsible for $\pi^+\pi^0$ events. Herein I will show how to look for such variables and to exploit them. The results achieved so far have only provided marginal improvements to the background rejection level already obtained. This approach necessarily involves mathematical difficulties; nevertheless, even a numerical approximation, if found, could provide interesting results in the future.

In the two-body decay scenario, the correlation between the ring radius and the position of its centre, that reflects the one existing between the momentum and angle of the π^+ (see Chapter 3.5), would be a valuable way to separate signal from background. However, it also causes difficult computational issues. In fact, the variance is greater for a function of two correlated variables than it is for a function of uncorrelated variables. Such variance can be computed according to the general error propagation formula.

Given a set $\vec{x}$ of n random variables with expectation values $\vec{\mu}$, and a function $f(\vec{x})$, assuming we can perform a Taylor expansion around $\vec{\mu}$:

$$f(\vec{x}) = f(\vec{\mu}) + \sum_{i=1}^{n}(x_i - \mu_i)\frac{\partial f}{\partial x_i}\bigg|_{\vec{x}=\vec{\mu}} + \dots \tag{5.2}$$

Truncating at first order:

$$\begin{aligned}
\mathrm{E}(f) =& f(\vec{\mu}) + \dots && (5.3)\\
\mathrm{Var}(f) =& \mathrm{E}\left((f - \mathrm{E}(f))^2\right) && (5.4)\\
=& \mathrm{E}\left((f(\vec{x}) - f(\vec{\mu}))^2\right) && (5.5)\\
\simeq& \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial f}{\partial x_i}\frac{\partial f}{\partial x_j}\bigg|_{\vec{x}=\vec{\mu}}\right)\mathrm{Var}_{ij}(\vec{x}) && (5.6)
\end{aligned}$$

For a function f of two variables x and y:

$$\mathrm{Var}(f) = \sigma_x^2\left(\frac{\partial f}{\partial x}\bigg|_{\vec{x}=\vec{\mu}}\right)^2 + \sigma_y^2\left(\frac{\partial f}{\partial y}\bigg|_{\vec{x}=\vec{\mu}}\right)^2 + 2\,\mathrm{Cov}(x,y)\left(\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}\bigg|_{\vec{x}=\vec{\mu}}\right) \tag{5.7}$$

In the case of the missing mass, the small RMSs in the reconstructed kinematic variables β_π and $\theta_{K\pi}$ add up in a nonlinear way, thus making it an unconvenient statistical variable from a computational point of view.

The RICH reconstruction algorithm as it is now implemented exhibits a slight asymmetry in the reconstruction of the particle velocity β, which is usually overestimated (as shown in Figure 4.7 in Chapter 4.3). This results in an inaccurate estimate of the particle flight direction, through the assessment of its momentum $P(\beta, m_\pi)$ and the backwards propagation of the track across the two upstream magnets (see Chapter 4). In particular, the magnitude of the magnetic deflection due to the spectrometer magnet is inversely proportional to the momentum of the particle. The correlated deviations of the reconstructed position and radius of the Čerenkov rings therefore add up when used to compute functions $f(\beta_\pi, \theta_{K\pi})$ of both variables, e.g. the function evaluating the missing mass $m^2_{\rm miss}(K^+, \pi^+)$.

The problem of finding the function $g(\beta_\pi, \theta_{K\pi})$ that makes use of as much information as possible while keeping its variance low is very interesting from a mathematical point of view. Minimizing a functional ($\mathrm{Var}(f)$ in this case) is a variational problem without a general solution. Unfortunately, our knowledge is limited to the covariance matrix of the two variables, and this does not allow for an algebraic solution to this problem. Moreover, in the 2-body decay β_π and $\theta_{K\pi}$ are not only strongly correlated, but also functionally dependent. Therefore, the only reasonable solution would be a variable obtained with as few computation steps as possible, but containing full information about the physical process: for example

$$\delta \equiv \theta_{K\pi} - \theta_{K\pi}(\beta_\pi) \tag{5.8}$$

would exploit the functional relation between $\theta_{K\pi}$ and β_π in the $K^+ \to \pi^+\pi^0$ decay, while preventing further computational steps that might increase the variance of the test variable.

It is expected that the distribution of δ for $\pi^+\pi^0$ events is peaked at 0. On the other hand, nothing can be said a priori about the distribution of δ for events $\pi^+\nu\bar{\nu}$, which do not feature such functional dependence. The width of the distribution for background events might be small enough to allow a cut in a short range around 0.

Unfortunately, this is not a completely acceptable solution either, as shown by the simulation results reported in Figure 5.5. The distribution in δ for the signal events, in fact, is very narrow and centred so close to 0 that it almost completely overlaps the peak arising from $\pi^+\pi^0$ events. Furthermore, the signal lies in the momentum range for which the RICH provides maximum accuracy in the reconstruction of β; since this is not true

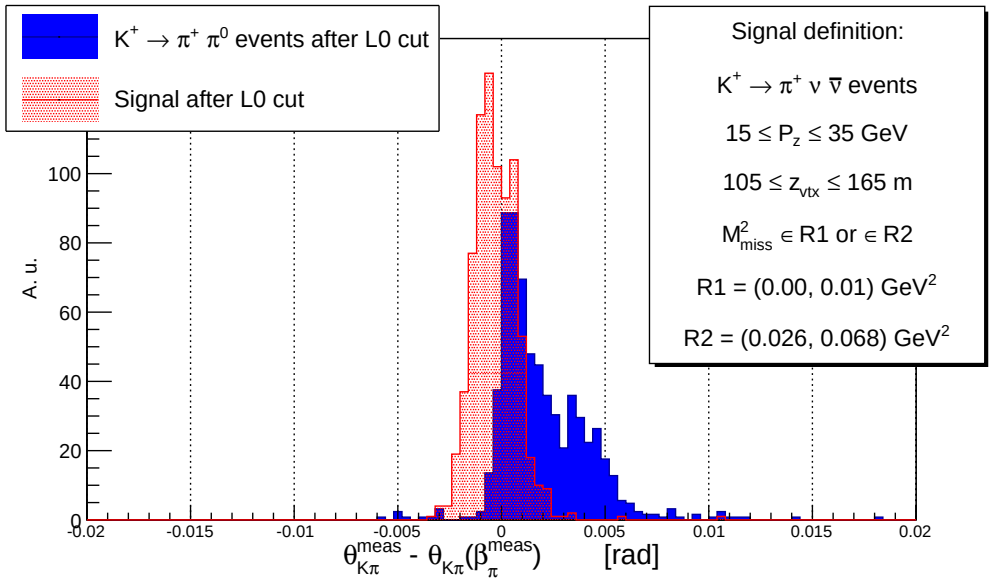


Figure 5.5: Distribution of δ defined in Eqn. 5.8 for signal and background events sets.

for the background, an asymmetry arises in the reconstruction of β, that spreads the corresponding distribution towards the positive side.

However, we may once more exploit the fact that the background is not filtered according to the pion momentum, and remove those events for which the reconstruction asymmetry exceeds a given threshold from the data pool, for example by requiring $\delta \leq 0.002$ at trigger level.

5.5 Performance together with the standard L0 trigger

According to the distributions presented in the previous sections, the rejection power of the RICH-based trigger algorithm I developed for the NA62 experiment mostly emerges from a selection of the radius of the Čerenkov rings. Therefore, I performed a study on the rejection achievable with the cut

$$R_c \leq R_{\rm th} \tag{5.9}$$

Table 5.2 shows both the signal acceptance and the background rejection for various threshold values. The first column lists the threshold values $R_{\rm th}$

Signal definition:

Event type	$K^+ \to \pi^+\nu\bar{\nu}$
π^+ momentum along z	15 to 35 GeV/c
Decay vertex z position	105 to 165 m
Square missing mass to the K^+ and π^+	0 to 0.01 GeV$^2/c^4$
	0.026 to 0.068 GeV$^2/c^4$

Table 5.1: Offline analisys requirements for the selection of $K^+ \to \pi^+\nu\bar{\nu}$ events in the NA62 experiment.

used. The second column shows the percentage of signal events that pass the selection, computed as $n_{\text{pass}}/n_{\text{tot}}$. The signal is selected according to the requirements listed on Table 5.1. The third column shows the $\pi^+\pi^0$ rejection capability of the RICH-based L0 algorithm alone, which in this study consists in the $R_c \leq R_{\text{th}}$ cut only. The rejection power is computed as $1 - n_{\text{pass}}/n_{\text{tot}}$. The "RICH | L0" column is the most significant one. It highlights the rejection power of my algorithm rescaled to the background events that have passed the selection performed by the standard hardware L0 trigger. Finally, the last column shows the combined rejection level achieved by the two trigger algorithms – RICH-based and standard L0 – running in parallel. The last row shows the $\pi^+\pi^0$ background rejection achieved by the standard hardware L0 trigger described in Chapter 3.3.

The selection performed by the $R_c \leq R_{\text{th}}$ requirement makes use of different observables than the existing hardware-based L0 trigger, and it makes it possible to cut the data rate fed to the following trigger stage by an additional 60% to 70%. Such reduction may in turn allow the L1 stage to perform deeper analisys, without stretching its latency or increasing the amount of resources required.

The boldface row in Table 5.2 is an attempt to select a starting working point for further studies, as a reasonable compromise between signal efficiency and backround rejection. The efficiency for the signal for the quoted $R_{\text{th}} = 17.9$ cm is (98.6 $\pm$ 0.4)%, which means that the signal will remain essentially unchanged. The background rejection achieved over the L0 trigger is (64.8 $\pm$ 1.2)% and the overall $\pi^+\pi^0$ rejection adds up to (91.8 $\pm$ 0.2)%.

R_{th} (cm)	Signal efficiency	RICH rejection	RICH | L0	L0+RICH
17.60	92.6 ± 0.8	63.4 ± 0.5	70 ± 1	93.0 ± 0.2
17.70	96.0 ± 0.6	61.3 ± 0.5	68 ± 1	92.6 ± 0.2
17.75	97.1 ± 0.6	59.8 ± 0.6	68 ± 1	92.5 ± 0.2
17.80	97.6 ± 0.5	58.5 ± 0.6	67 ± 1	92.3 ± 0.2
17.85	98.3 ± 0.4	56.8 ± 0.7	66 ± 1	92.1 ± 0.2
17.90	$\mathbf{98.6 \pm 0.4}$	$\mathbf{55.2 \pm 0.8}$	$\mathbf{65 \pm 1}$	$\mathbf{91.8 \pm 0.2}$
17.95	99.0 ± 0.4	53.3 ± 0.8	64 ± 1	91.5 ± 0.2
18.00	99.5 ± 0.3	51.3 ± 0.8	62 ± 1	91.2 ± 0.2
18.10	99.6 ± 0.3	47.0 ± 1.0	59 ± 1	90.4 ± 0.3
Standard L0 rejection			76.7 ± 0.04	

Table 5.2: Rejection power and signal acceptance achievable with a cut on the radius of Čerenkov rings. All values are expressed as percentages, and the errors shown are pure statistical uncertainties due to the size of the data samples. The last row shows the $\pi^+\pi^0$ background rejection achieved by the standard hardware L0 trigger. For this study I used two separate data sets consisting of 10000 K^+ decays each. One data set contained only $\pi^+\nu\bar{\nu}$ events, while the other one consisted of $K^+ \to \pi^+\pi^0$ decays.

In a subsequent study I tried to estimate the effect of a cut on the δ variable introduced in Section 5.4. This selection does not add much to that on the Čerenkov radius, as it operates on correlated variables: the faster a pion crosses the RICH detector, the worse its characteristics can be measured. However, I found out that a cut

$$R_c \leq 17.9 \text{ cm} \tag{5.10}$$
$$\delta \leq 0.0025 \tag{5.11}$$

yields a maximum $\pi^+\pi^0$ rejection of $(71.5 \pm 0.9)\%$ on particles that passed the standard L0 selection, while a cut

$$R_c \leq 17.9 \text{ cm} \tag{5.12}$$
$$\delta \leq 0.002 \tag{5.13}$$

yields a slightly higher rejection of $(72 \pm 1)\%$, maintaining a signal acceptance of $(98.0 \pm 0.4)\%$.

Finally, I introduced a "ring fit quality" selection. A greater background rejection can be expected if there is a possibility of detecting the presence of more than one Čerenkov ring, for the reasons explained in Chapter 3.5.

The simplest way to check if the hits detected belong to one or more rings is to perform a cut on a χ^2 variable computed on the result of the ring fit. Montecarlo simulations show that a threshold $\chi^2_{th} = 1$ allows to discriminate between clean 1-ring events and multi-ring or noisy events. Higher values of χ^2_{th} did not sensibly change the result. The following cuts:

$$R_c \leq 17.9 \text{ cm} \tag{5.14}$$

$$\delta \leq 0.002 \tag{5.15}$$

$$\chi^2_{ring} \leq 1 \tag{5.16}$$

therefore allow to attain (77 $\pm$ 1)% additional $\pi^+\pi^0$ rejection after the standard L0 cuts, with (96.2 $\pm$ 0.7)% signal acceptance.

These numbers should be interpreted as the best achievable with a RICH-based online trigger, since both the characteristics of the target processes and the acceptance and resolution of this detector are taken into account.

Part III

Algorithm development and test

6 "Ptolemy", a two-step algorithm

Contents

6.1 The necessity for a multi-ring algorithm

In order to be used as lowest-level trigger, a ring-fitting algorithm needs to be:

- **seedless**: it will be fed with raw RICH data, with no previous information on the ring position from other detectors;
- **fast**: it will run concurrently with the hardware L0 trigger, with a maximum latency (*decision making* time) of 1 ms and an event rate of about 10 MHz;
- **accurate**: while the aim of this project is to reduce the background events rate to as low as possible, we still need to maintain the signal acceptance close to 100%. Our constraints therefore will be defined by the finite combined resolution of the RICH detector and of the ring finding algorithm;

- **"multi-ring friendly"**, to account for inelastic hadron scattering and π^0 Dalitz decays, for the $\pi^+\pi^-\pi^+$ background, and to allow positive identification of rare di-lepton decay modes.

When I started working on this thesis, there were no multi-ring parallel fitting algorithms in literature ready to be adapted to RICH applications. For this reason, I tried to figure out how to discard events with more than one ring, i.e. with more than one charged track, that would be rejected during the final $K^+ \to \pi^+\nu\bar{\nu}$ analysis.

If there are only single-ring algorithms available, the easiest way to check whether the data points really belong to a single circle is to compute a χ^2 variable on the ring candidate. Except in the rare case when two rings are concentric and similar in radius, a *least squares* analysis will return a higher χ^2 when the points do not belong to the same ring.

However, as reported in Chapter 5, a background rejection algorithm is more effective if it is designed to provide at least basic kinematical information. The background process $K^+ \to \pi^+\pi^0$ analysed in this work may create multiple rings in the cases reported in Chapter 3.5. The presence of more rings (or parts of rings), and possibly of noise hits on the photomultipliers, would make it difficult to evaluate the radius and position of a "real" π^+ Čerenkov ring.

In addition, NA62 is a promising experiment for the study of other K decay channels, besides $\pi^+\nu\bar{\nu}$: the large statistics it will collect will make it possible to probe other ultra-rare or forbidden decay channels with unprecedented precision. Most current BSM theories predict some degree of Lepton Flavour Violation (LFV). A non-exhaustive list of such theories includes Supersymmetry (SUSY), Technicolor, Little Higgs models, extra dimensions and even the introduction of heavy neutrinos.

SM-forbidden decays such as $K^+ \to \pi^+\mu^\pm e^\mp$ and $K^+ \to \pi^-\ell^+\ell'^+$ (with $\ell, \ell' = \mu, e$) would feature a very clean experimental signature. The high statistics of NA62 – of the order of 10^{13} kaon decays – would allow the current limits to be improved by a factor of about 10: the existing limits for these processes are listed in Table 6.1.

Current limits from Table 6.1 date back to 2005 (1996 data) for BNL experiments, and to 2011 for NA48/2 [4, 14, 60]. The sensitivity in NA62 is expected to be of the order of 10^{-12}, and it should be possible to improve the

K^+ decay mode	SM violation	Branching ratio at 90% C.L.	Experiment
$\pi^+\mu^+e^-$	LF	$< 1.3 \times 10^{-11}$	BNL E777 – E865
$\pi^+\mu^-e^+$	LF	$< 5.2 \times 10^{-10}$	BNL E865 – CERN NA48/2
$\pi^-\mu^+e^+$	L	$< 5.0 \times 10^{-10}$	BNL E865 – CERN NA48/2
$\pi^-e^+e^+$	L	$< 6.4 \times 10^{-10}$	BNL E865 – CERN NA48/2
$\pi^-\mu^+\mu^+$	L	$< 1.1 \times 10^{-9}$	CERN NA48/2

Table 6.1: Lepton Number (*L*) and Lepton Flavour Number (*LF*) violating K^+ decay modes

current branching ratio limits by at least one order of magnitude. Moreover, NA62 will be equipped with better spectrometers compared to the previous generation kaon experiments, providing a better mass resolution and allowing cleaner separation between background events such as $K^+ \to \pi^+\pi^+\pi^-$ and the signal region. This will allow direct searches in regions with low expected Physics background.

LF violating decays may be pursued by NA62 to sensitivities corresponding to possible tree-level contribution of new particles, where the decay mode suppression arises from the mass of the exchanged field. A precise measurement would rule out a fraction of theories providing LFV.

The detection of a lepton pair would result in a trigger opportunity for such forbidden decay modes: allowed decays with more than one lepton in the final state are highly suppressed, and always imply the presence of a neutrino. As a consequence, they can be identified and discarded by the analysis of the missing energy. The probability of misidentifying a pion as a lepton is low, due to the presence of several P.ID.[1] detectors. The high-resolution NA62 spectrometer makes it possible to apply stringent cuts on the decay vertex, and to remove the background due to the pile-up of different decay events.

For all these reasons, a real-time algorithm capable of fitting multiple rings would be valuable.

Since no seedless multiple-ring-fitting algorithm exists, we chose to split the problem in a preliminary "**pattern recognition**" step, needed to divide the data set into single-ring candidates, and a subsequent **single-ring fitting**

[1] Particle Identification.

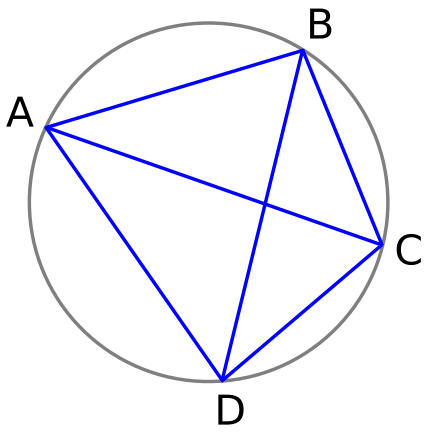


Figure 6.1: Ptolemy’s theorem is a relation among the lengths of the sides and those of the diagonals in a cyclic quadrilateral.

stage.

6.2 Ptolemy’s theorem

For the pattern recognition step we chose to exploit Ptolemy’s trigonometric theorem [27] about quadrilaterals inscribed in a circle, as first proposed by G. Lamanna [43]:

Ptolemy’s Theorem. *For a cyclic quadrilateral, the sum of the products of the two pairs of opposite sides equals the product of the diagonals.*

For our application this theorem translates to the subsequent formula (refer to Figure 6.1 for vertices and segments naming):

$$\overline{AB} \cdot \overline{CD} + \overline{AD} \cdot \overline{BC} - \overline{AC} \cdot \overline{BD} = 0 \tag{6.1}$$

Given the coordinates of three points on a circle, we use this theorem to check whether a fourth point belongs to the same circle or not. If it belongs to the same circle, it is added into an array that contains the set of points belonging to the given ring candidate.

Ideally, this process would be repeated until there are no points left, and then the ring candidates would be fed to a ring-fitting algorithm that would find the best centre and radius.

Unfortunately, our algorithm needs to be seedless, meaning that we do not know a priori how many rings the current event contains, and we are unable to identify three starting points belonging to the same ring. We must therefore find an approximate way to define adequate starting "triplets" of hits that will be fed to the Ptolemy algorithm.

In this chapter I will examine the building blocks of the trigger algorithm developed for this project. In Section 6.4 I will examine the procedure devised for the definition of four initial suitable triplets in detail. However, we first need to go through a short description of the data fed to the procedure (Section 6.3). Chapter 7 will describe the GPU implementation of the whole trigger program more thoroughly.

6.3 Reparametrization of the photomultipliers lattice

The test setup and the data format used to send data to the GPU will be described in Chapter 7.2.1.

In order to store x and y coordinates in 8-bit words, as demanded by the data format described in Figure 7.4, we reparametrized the frame enclosing the RICH focal plane in a way that is computationally efficient. Each of the two round flanges hosts 976 photomultipliers distributed on a compact hexagonal lattice (see Figure 6.2). We have assigned a progressive integer to each x-axis point occupied by a vertex of the lattice, and another progressive integer to each y position. As a result, we are effectively defining a lattice where half of the vertices are occupied by the centre of a photomultiplier; this way, a 38×66 lattice is enough to contain all the readout devices of a flange, whose coordinates can therefore be represented by two 8-bit integers. The two flanges are equal and will be superimposed in the process of ring fitting, therefore only one 38×66 lattice is needed. Figure 6.3 shows a reparametrized RICH flange.

Coordinate reparametrization is operated through the following formulas:

$$x'(x) = \text{round}\left(n_x \frac{x - x_{min}}{x_{max} - x_{min}}\right) \tag{6.2}$$

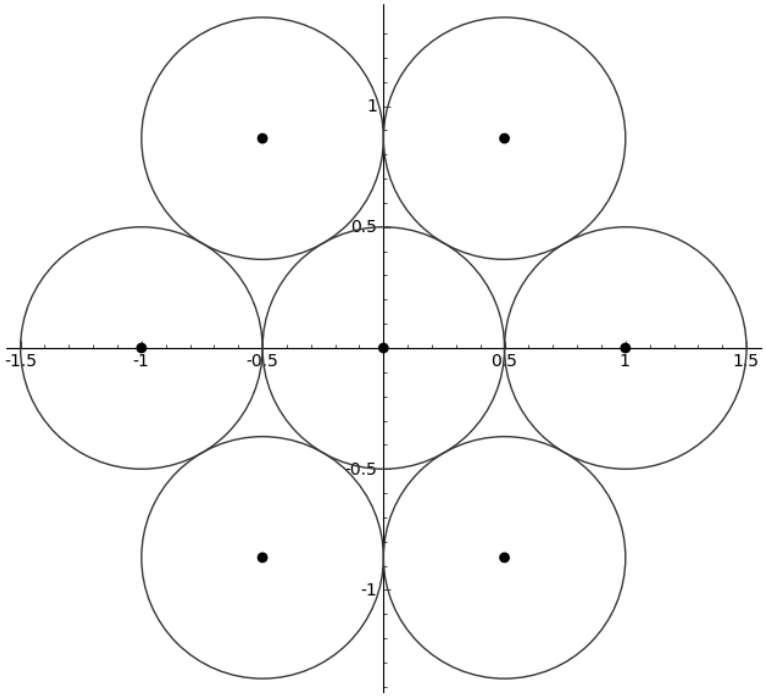


Figure 6.2: Hexagonal packing of circles. Circle centres are spaced by $\Delta x = 1/2$ in x, and by $\Delta y = \sqrt{3}/2$ in y. If Δx and Δy represent the gaps in a lattice, half of the lattice points can host the centre of a circle [61].

$$y'(y) = \sqrt{3} \cdot \text{round}\left(n_y \, \frac{y - y_{min}}{y_{max} - y_{min}} \right) \tag{6.3}$$

with

$$\begin{aligned} x_{min} &= 0 \\ x_{max} &= 66 \\ y_{min} &= 0 \\ y_{max} &= 38 \end{aligned}$$

so that the centre of each PM, residing in (x, y) in the laboratory frame, is represented by a (x', y') pair in the reparametrized frame. The roundings are used to ensure that the output is the closest integer, rather than using truncations. The vertical coordinates are inflated by a factor $\sqrt{3}$ in order to account for the asymmetrical filling factors of a compact hexagonal lattice.

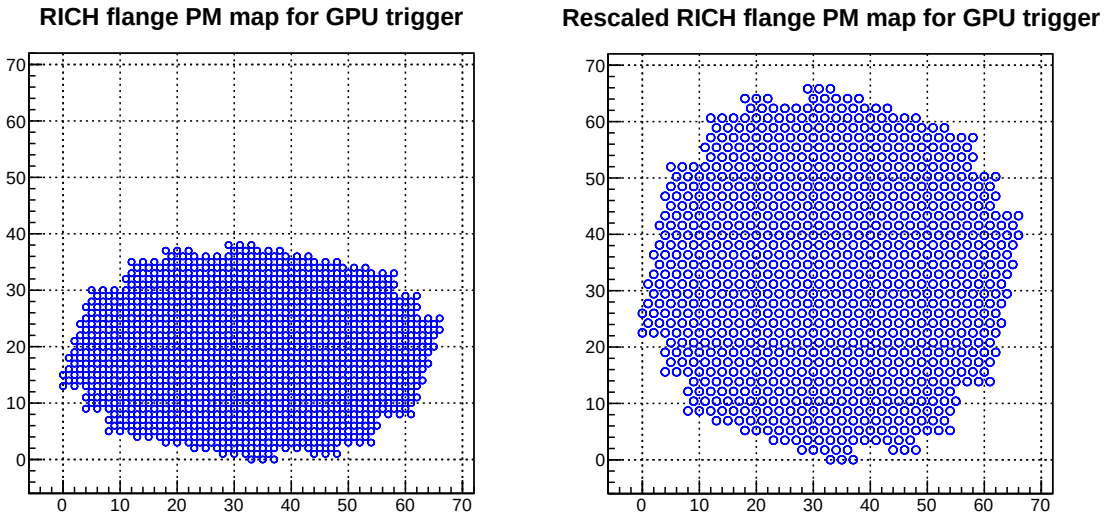


Figure 6.3: Representation of the reparametrized RICH photomultiplier-holding flange. The first frame shows the new PMTs coordinates, while the second one shows the lattice that is used for GPU operations: vertical distances are increased by a factor $\sqrt{3}$ because of the filling characteristics of the compact hexagonal lattice.

The transformation can be inverted, obtaining:

$$x(x') = x_{min} + \frac{x_{max} - x_{min}}{n_x}\, x' \tag{6.4}$$

$$y(y') = y_{min} + \frac{y_{max} - y_{min}}{\sqrt{3}\, n_y}\, y' \tag{6.5}$$

The above conversion formulas prove useful to cast the results back in the laboratory frame. If the program will return the coordinates (a', b') of the centre and the radius r' for each ring, the true values for these parameters can be obtained simply by computing

$$a = x(a') \tag{6.6}$$

$$b = y(b') \tag{6.7}$$

$$r = \frac{x_{max} - x_{min}}{n_x}\, r' \tag{6.8}$$

The last formula is obtained making use of the equivalence between the two factors

$$\frac{x_{max} - x_{min}}{n_x} \simeq \frac{y_{max} - y_{min}}{\sqrt{3}\, n_y} \simeq 9 \tag{6.9}$$

(up to the small rounding needed to cast the laboratory coordinates as the closest integers), that describes the scaling between rescaled and laboratory coordinates, and computing the distance between the centre and one point of the circle:

$$\begin{cases} x' = a' + r' \cos\theta \\ y' = b' + r' \sin\theta \end{cases} \xrightarrow{\theta \equiv 0} \begin{cases} x'_0 \equiv a' + r' \\ y'_0 \equiv b' \end{cases} \tag{6.10}$$

Here θ is an arbitrary phase, that we set to 0 for simplicity. This point corresponds, in the laboratory frame, to the point $(x\,(a' + r')\,, y\,(b'))$. The distance to the centre will then be:

$$r^2 = [x(a') - x(a' + r')]^2 + [y(b') - y(b')]^2 \tag{6.11}$$

$$= [x(a') - x(a' + r')]^2 \tag{6.12}$$

$$r = \left|x(a') - x(a' + r')\right| = \frac{x_{max} - x_{min}}{n_x}\, r' \tag{6.13}$$

6.4 Pattern recognition

A convenient way must be found to initialize the process of checking the data points through the Ptolemy's theorem by providing sets of hit triplets.

The points assigned to each triplet should:

- be well separated, in order to optimize the subsequent selection of hits satisfying the Ptolemy's theorem;
- maximise in some way the probability of belonging to the same circle, which would result in a better capability of identifying the maximum number of Čerenkov rings.

All the possible triplets of points would have to be tried in order to maximise the efficiency of the pattern recognition step. However, this approach is extremely time expensive and therefore not feasible at an online level within a maximum latency of 1 ms. It is also unclear whether this approach would work even in an offline approach, as the amount of memory needed to store all those subsets of data could be higher than what is available on certain GPU devices. For example, for a 20-hit event, there are 1140 possible triplets. This means that, for each event, 6.84 kB of memory are needed only to store the 16 bit indexes of the hits forming the triplets. Supposing half of the triplets match a 20-hit ring, 1.49 GB of data would need to be stored and processed per batch of processed events (see Chapter 7). The NVIDIA Tesla K20 GPU used for this work, that is quite a state-of-the-art

device, features 5 GB of global memory in total (see Chapter 7.1), but "only" 369 MB per multiprocessor, i.e. available at the same time.

On the other hand, the fastest approach would consist in choosing three points at random. Again, this procedure would not be optimal, as the ring-detection efficiency would be low.

A procedure is needed, according to which the triplets of points should be selected. Looking at some events from a $K^+ \to \pi^+\pi^0$ sample, we may notice that, on a statistical basis, except for events with hits caused by noise or by particles from the beam halo, the Čerenkov rings are rarely concentric or close to each other. A feasible approach may therefore consist in selecting the "border" hits, as there is a high chance that they belong to the same circle. Figure 6.4 shows two typical $\pi^+\pi^0$ events.

We chose to select:

- the leftmost three points (*XMIN* triplet)
- the rightmost three points (*XMAX* triplet)
- the three points at the bottom of the frame (*YMIN* triplet)
- the three points on top of the frame (*YMAX* triplet)

with the following condition on the distance d between the points $\vec{t}_i$, $\vec{t}_j$ belonging to the same triplet:

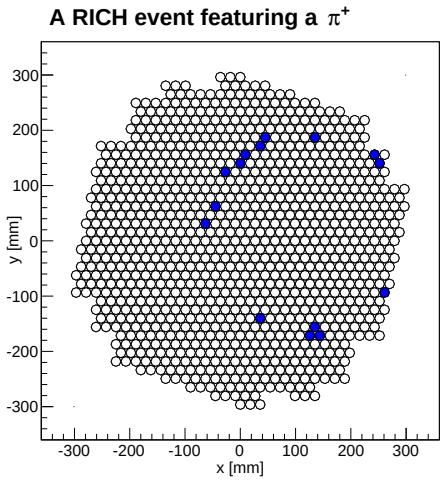
$$d^2\left(\vec{t}_i, \vec{t}_j\right) > d_{th}^2 \tag{6.14}$$

where the minimum square distance was set to $d_{th}^2 = 8$ (in the rescaled units) after empirical tests: greater threshold distances do not increment the identification efficiency of ring candidates, and they increase the probability of selecting hits belonging to different Čerenkov rings.

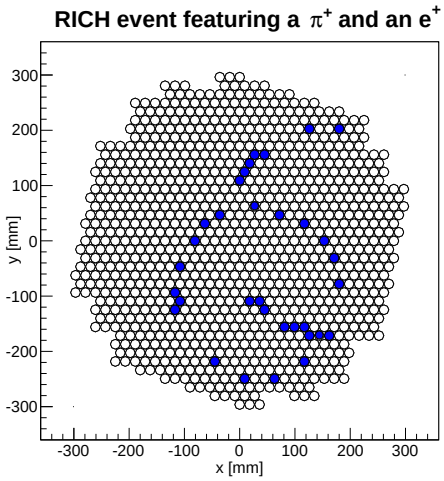
The choice of triplets for the two-rings event shown in Figure 6.4(b) is illustrated in Figure 6.4(c).

The maximum number of rings that can be detected in this way is four. The NA62 experiment does not focus on the study of events with more than one charged particle, anyway. In fact, the result we need to achieve at this stage is an efficient detection of at least one circle, possibly in presence of other hits or parts of additional rings.

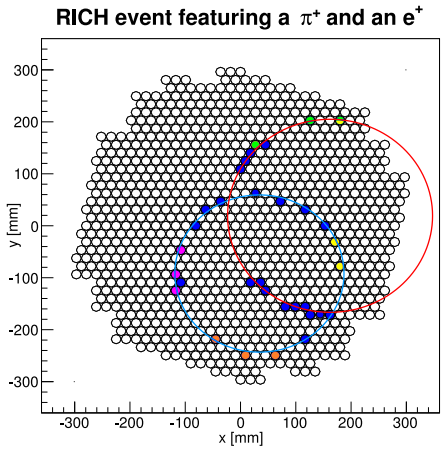
Events featuring less than 5 or more than 32 hits were discarded for this study, whose aim is in fact to strengthen the $\pi^+\pi^0$ rejection achieved



(a) One-ring event



(b) Two-ring event



(c) Two-ring event: pattern recognition

Figure 6.4: Two typical $\pi^+\pi^0$ events as detected by the RICH subdetector. In the event represented in the top left panel, only a π^+ was detected. On the top right panel we see two rings, one of which arose from a π^0 Dalitz decay: a positron from $\pi^0 \to e^+e^-\gamma$ was detected as well. Finally, the bottom panel demonstrates the initialization of a pattern recognition procedure. Starting from each side of the frame, the outermost three hits are selected, which undergo a condition of reciprocal distance greater than a minimum threshold. Each point can be chosen to belong to more than one “triplet”.

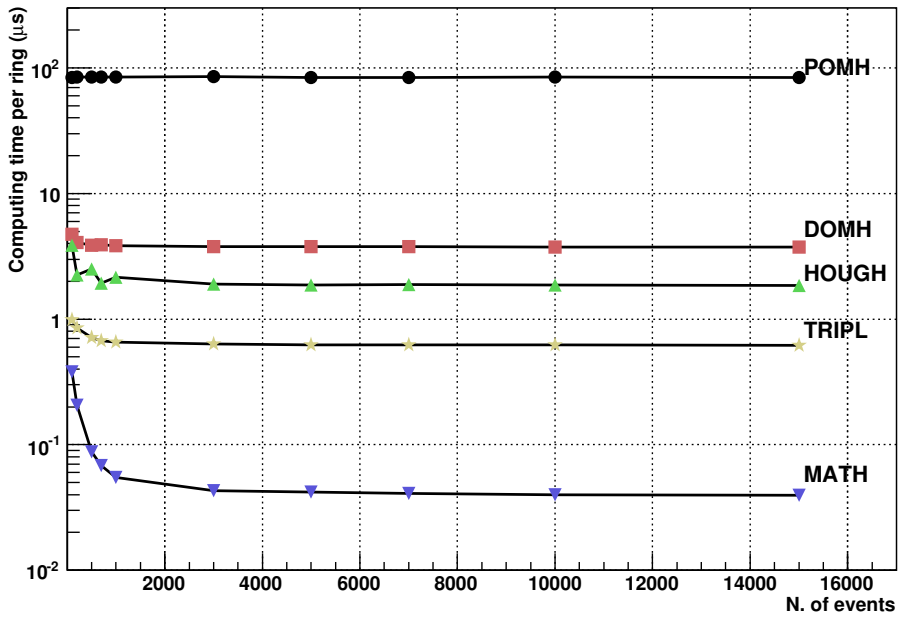


Figure 6.5: Kernel execution time per single-ring event (20 hits) as a function of the number of events processed in one batch on a NVIDIA Tesla C1060 GPU [26].

at the standard L0 trigger level. The upper limit condition may be dropped in order to develop a positive trigger for multi-particle events, such as the lepton number or lepton family number violating channels $K^+ \to \pi^-\mu^+\mu^+$ (B.R. $< 3 \cdot 10^{-9}$) or $K^+ \to \pi^+\mu^-e^+$ (B.R. $< 5 \cdot 10^{-10}$); in this case, however, more GPU threads should be allocated for each triplet, allowing for a higher number of hits, and therefore fewer events could be processed simultaneously.

6.5 Single-ring fit

For the ring fitting stage, several single-ring fitting algorithms were previously examined in [26]. Five non-iterative procedures were implemented and tested using Montecarlo generated data with rings of variable position, radius and number of hits.

Two of the tested algorithms, nicknamed “POMH” (Problem-Optimized Multi-Histograms) and “DOMH” (Device-Optimized Multi-Histograms), em-

ployed parallelization at the algorithm level only, with approximately 1000 GPU cores being used concurrently to process a single event.

Another tested algorithm ("HOUGH") was based on a series of Hough Transforms[2] and reduced the problem to that of finding intersections between circles centred on each hit photomultiplier.

A geometrical approach was examined with the "TRIPL" algorithm, for which the centre is determined by the intersection of the mid-points axes of pairs of segments defined by randomly chosen triplets of hits. In order to achieve a good resolution and to gain robustness against the noise, several triplets had to be used for each event, averaging the results.

Finally, a simple approach was tested, which used a least square method to identify the best circle parameters ("MATH"). In this case, parallelization was exploited only by processing several events at the same time.

The best results in terms of fit accuracy and execution time per event, shown in Figure 6.5, were obtained with this last algorithm, suggesting that simple geometrical or algebraic approaches are most advisable in a context where reliability of results is a key issue.

For the above reason, I examined a number of algebraic ring fitting methods, mostly described in [24], in order to select the best one in terms of robustness and execution time. A procedure devised by Gabriel Taubin in 1991 [66] turns out to be the most computationally safe algorithm, being also quite robust with respect to noise that could affect the estimation of the radius, which "MATH" is not. A mathematical description of the "MATH" and Taubin algorithms is available in Appendix A.

I implemented both procedures in a simple test framework and measured their execution times, in order to check if the Taubin algorithm could be used in our kernel in place of MATH. The two kernels were executed on a Nvidia Tesla K20 GPU with an increasing number of single-ring events per execution batch. The result, shown in Figure 6.6, highlights the negligible time difference between the two. As a consequence, we could safely decide to adopt the Taubin algorithm to perform the single-ring fitting part of our program.

[2]The Hough Transform is a pattern extraction technique aimed at finding instances of objects of a certain shape in a noisy environment, by means of a voting procedure.

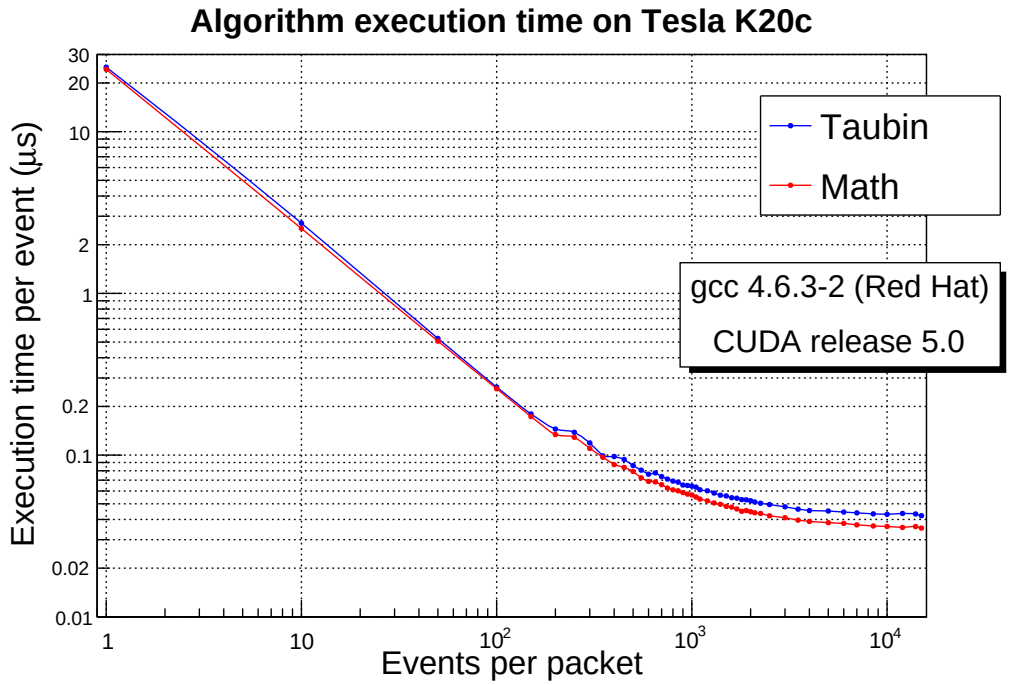


Figure 6.6: Execution times of the *MATH* and *Taubin* kernels as a function of the number of events processed in one batch on a NVIDIA Tesla K20c GPU.

7 Implementation on GPUs

Contents

7.1 GPU architecture and CUDA framework

Originally designed for the video-game market and the handling of screen graphics, GPUs are massively parallel multiprocessors equipped with large fast access on-board memory. Unlike CPUs, much more silicon area is devoted to computing units than to control structures (Figure 7.1). The computing power of GPUs arises from the large number of processing cores installed on the device, rather than from the chip clock speed (as for CPUs). Graphic card devices are indeed used to execute highly parallelized tasks.

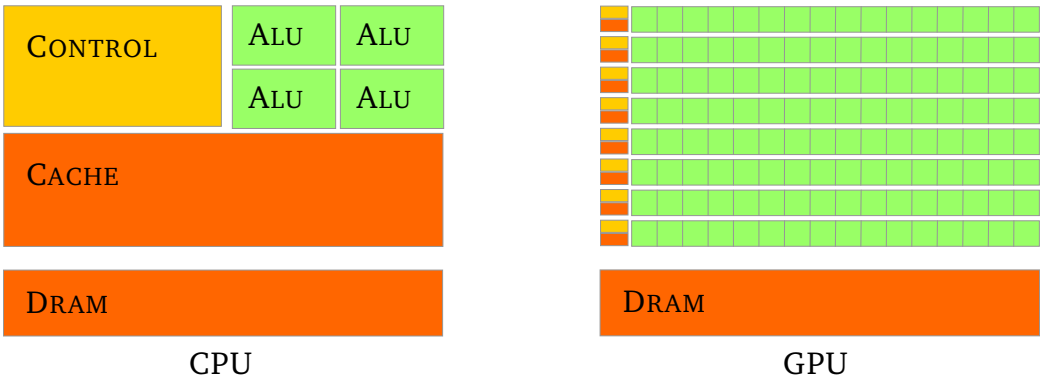


Figure 7.1: Compared to the CPU, the GPU devotes more transistors to data processing [53].

The trigger algorithm and the test framework I will describe in Section 7.2.1 use the CUDA (Compute Unified Device Architecture) toolkit[1]. CUDA is a platform for parallel programming and computing developed by NVIDIA[2], compatible with GeForce, Quadro and Tesla GPUs. This platform exposes GPUs for computing just like any usual processor, through CUDA-accelerated libraries and extensions to the most popular programming languages.

A set of C/C++ libraries enables heterogeneous programming and provides straightforward APIs[3] for device and memory management. GPUs can be embedded in the PC motherboard, or on the CPU die, or reside in dedicated *graphics cards* connected to the host PC via PCI Express links, as in this work. Hereinafter, I will call *host* the CPU and its memory, and *device* the GPU. Serial functions, decorated with the `__host__` prefix, are coded in standard C and execute on the processor; the host can call a `__device__` function (*kernel*) at any moment, that will run on the GPU.

A few definitions should be given before discussing the GPU architecture:

Thread: the smallest sequence of instruction that can be run independently.

Warp: the minimum *work group size*, i.e. the maximum number of threads that can execute the same instruction simultaneously, in SIMD mode (*Single Instruction – Multiple Data*), within a single CUDA multiprocessor. Currently, all NVIDIA GPUs feature warps of 32 threads.

[1] http://www.nvidia.com/object/cuda_home_new.html
[2] http://www.nvidia.com/
[3] Application Programming Interface

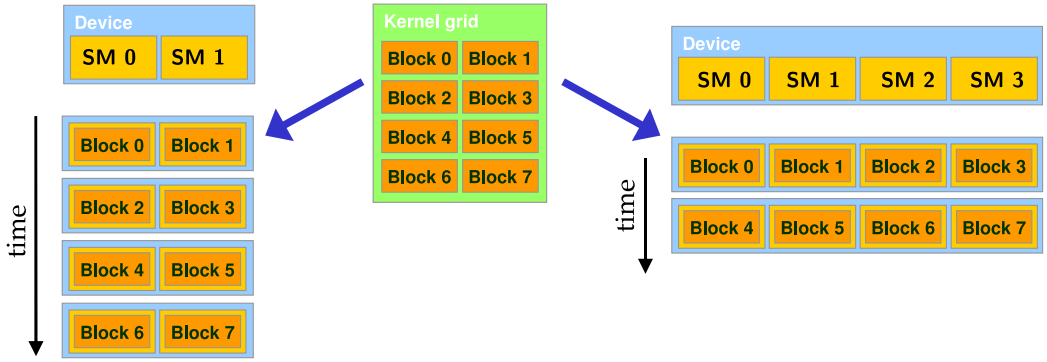


Figure 7.2: Concurrent execution of thread blocks on devices with a different number of multiprocessors: 2 SMs (left) and 4 SMs (right) [64].

Block: the basic element of a GPU program. The threads of a block execute concurrently on one multiprocessor, and multiple blocks can run at the same time. New blocks are launched on the free multiprocessors, as the previous ones terminate.

The building block of the GPU architecture is the Streaming Multiprocessor (SM), which hosts a number of single-precision CUDA cores (see Appendix C) executing identical sets of instructions, and a block of high-speed on-chip memory. Up to 24 warps can be active at the same time on a single SM, depending on their memory usage and on the number of registers available on the SM. Each block of threads can be scheduled in any order on any of the available SMs, allowing for program scalability: devices with more SMs automatically outperform older GPUs, as demonstrated in Figure 7.2 [53]. One of the most important characteristics of the CUDA architecture, indeed, is that kernels are scalable across any number of parallel SMs to adapt to different hardware.

The CUDA runtime manages the number of blocks processed simultaneously by the GPU SMs, that is closely linked to the availability of hardware resources.

A kernel is launched with a call

```
mykernel<<<numberOfBlocks,threadsPerBlock,sharedMemorySize,
    streamID>>>(input* somestuff, output* someresults);
```

where:

- the triple angle brackets denote a call to a *device* function by a *host* function;

- the kernel is only executed by the stream `streamID` (see Section 7.1.2);
- the input parameters point to the device memory, that must be allocated before the kernel call;
- `mykernel` is executed at the same time by a grid of `numberOfBlocks` independent blocks;
- each block is split into `threadsPerBlock` threads that can synchronize and communicate to each other through access to the shared memory;
- a portion of shared memory (see Section 7.1.1) of size `sharedMemorySize` is allocated for each block.

The same kernel is executed by all the threads. Data-parallel programming maps data elements to parallel processing threads: inside the kernel function, jobs can be distributed to the blocks and split between the threads of a block by means of the built-in three-dimensional indices `blockIdx` and `threadIdx`. This provides a natural way to invoke computation across the elements of a multidimensional domain, such as matrices and vectors [53]. Individual threads executing in a warp are free to execute independently via data-dependent conditionals branches. In this case, all the branched paths are scheduled serially. Threads can be synchronized, i.e. can execute a common instruction at the same time, through dedicated CUDA functions, and automatically synchronize upon convergence to the same instruction branch.

7.1.1 CUDA memory hierarchy

The CUDA architecture features a hierarchy of memory spaces that are accessible in different scopes. All threads have read/write access to a large memory space called *global* (some GB), persistent across subsequent kernel launches within the same application. A different space, residing on each Streaming Multiprocessor, is devoted to independent portions of memory – referred to as *shared memory* – visible to all the threads of the same block. The lifetime of the shared memory coincides with the execution time of the block. Finally, each thread has access to a small private local memory space [53].

7.1.2 Streams and concurrency

Since the development of the Fermi[4] architecture, NVIDIA GPUs can execute up to three *streams* at the same time, allowing to concurrently manage

[4] http://www.nvidia.com/object/fermi-architecture.html

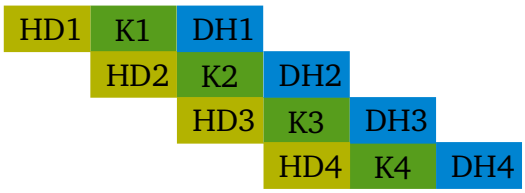


Figure 7.3: *3-way concurrency* demonstrated for a four-step program [57]. At the beginning, data is copied from the host memory into the device memory (*HD*). As the copy is executed, the stream is free to undertake another operation. At the second step, one stream executes the kernel (*K*), and another one concurrently copies more data into the device memory. At the following step, as soon as the previous two operations are concluded, three streams can run at the same time: the first one pulls the results of the previous execution of the kernel from the device, and stores them back into the host memory (*DH*); the second executes the kernel on the last data copied into the device memory; the third pushes some more data into the device memory.

processing and data transfers between host and GPU of different data sets.

A **CUDA stream** is a sequence of operations that execute in issue-order on the GPU [57]. CUDA operations from different streams may run concurrently and be interleaved. As an example, the concurrent execution of a small kernel in up to four streams may use as much as possible of the GPU computing resources at once. The maximum number of kernels that a device can run at the same time is four [69].

A typical example of concurrent execution of streams emerges from the possible overlapping of *host to device* and *device to host* memory copies and of the execution of a device kernel. We have adopted this solution, often referred to as “3-way concurrency”, for this project. Figure 7.3 explains the concurrent execution of such three streams.

7.2 Multi-ring algorithm implementation

7.2.1 Test framework, data format and input

A framework was built by F. Pantaleo [55], that consists of a multi-threaded software platform designed in order to optimize the execution latency of a given set of instructions. The platform executes on the CPU as host for GPU-based applications.

The main components are four:

- Network communication manager
- Process scheduler
- Device kernel (on the GPU)
- Trigger monitor

Network communication is handled by means of a modified API for the "Direct NIC Access" (DNA) driver[5]. This driver represents an alternative to the usual ethernet drivers, and it exposes the memory buffers stored on the Network Interface Controller (NIC) board to the user space. This way, data can be accessed directly from the NIC buffers, without the need of copying them to the host RAM first: this results in a lower and much more stable latency.

This part of the framework allows the user to use a large number of network interfaces. A thread is spawned for each interface, to handle the network communication.

The framework includes a scheduler in charge of handling a smart queue, managing multiple "producer" and asynchronous "consumer" processes. Data produced by threads running the network communication is accumulated in host memory buffers. Each buffer is copied to the device memory only when a predefined size is reached or when a "timeout" flag is raised: in order to sustain the throughput, the host waits for a good number of MTP packets within a given maximum time window. After that, the GPU kernel (where the trigger algorithm resides) is executed, and the results are copied back to the host memory. The full pipeline is managed by three independent software threads.

Finally, information about the status of the network communication and, possibly, real-time trigger results are displayed by a monitor refreshing every 5 s. The data shown include the number of UDP[6] packets read from each interface, the saturation of the network links and the number of MTPs processed by the GPU.

The firmware of the readout system for NA62 is still under development. For the tests reported in Chapter 8 we assumed that the TEL62 boards reading out the signals from the photomultipliers of the RICH will also take care of the conversion between the PMT channel IDs and their 8-bit coordinates. A simple look-up table will be implemented for this purpose on the board

[5] http://www.ntop.org/products/pf_ring/dna/
[6] User Datagram Protocol

firmware.

I customized the Montecarlo simulation and event reconstruction framework provided by NA62 to prepare six different sets of data:

- $K^+ \to \pi^+\nu\bar{\nu}$ events selected with the standard *signal* and *L0* cuts;
- $K^+ \to \pi^+\nu\bar{\nu}$ events selected with the standard *signal* cuts and relaxed hit multiplicity $5 \leq n \leq 64$ on the RICH;
- $K^+ \to \pi^+\pi^0$ events selected with the standard *L0* cuts;
- $K^+ \to \pi^+\pi^0$ events with hit multiplicity $5 \leq n \leq 64$ on the RICH;
- $K^+ \to \pi^+\pi^+\pi^-$ events selected with the standard *L0* cuts;
- $K^+ \to \pi^+\pi^+\pi^-$ events with hit multiplicity $5 \leq n \leq 64$ on the RICH.

Each data set was divided in files containing 256 events each.

In addition, we supposed that the geometric correction related to the tilts of the RICH mirrors (Eqn. 4.1) would also be computed by the TEL62 boards, before data are sent to the GPU. As a consequence, the lattice introduced in Section 6.3 had to be modified in order to avoid the repositioning of hits on "phantom" lattice vertices with negative x coordinates. Therefore, the whole frame was shifted by 85 units towards the positive side of the x axis, i.e. the 8-bit coordinates were computed as

$$x'(x) = \text{round}\left(n_x \frac{x - x_{min}}{x_{max} - x_{min}}\right) + 85 \tag{7.1}$$

instead of using Eqns. 6.2 and 6.3.

The UDP packets sent to the PC by the TEL62 boards contain all the relevant data we need for the analysis. Each packet consists of a header and a data container with the structure shown in Figure 7.4. These data are prepared by the TEL62 boards as UDP packets and sent to the NIC.

7.2.2 Data stream flow and triplet forming

It is convenient to define some constants before going through the implementation of the algorithm:

```
#define MAXHITS (int) 32
#define MAXEVENTS (int) 256
```

Here, `MAXHITS` represents the maximum number of hits allowed per event. This value was initially set to 32 in order to analyse events that have

Source ID	Format	Reserved
Source sub-ID	Num. of events	Total MGP length
Event data		
Event data		
...		

Event timestamp			
Reserved	Event fine time	Number of hits	
Hit ID y	Hit ID x	Hit ID y	Hit ID x
Hit ID y	Hit ID x	Hit ID y	Hit ID x
...		...	

Bits 31 16 15 0

Figure 7.4: Structure of the data files produced by the TEL62 boards. The top panel shows a GPU multi-trigger primitive packet. The bottom panel shows how data are packed for each event.

already passed the standard online selection[7]. Input from the NIC board is continuously read; the number of events packed in a UDP packet is read from the UDP header. Event data is written into `Element` structures, defined below, that are copied to the local memory of the GPU when a predefined size (`MAXEVENTS`) or data accumulation time has been reached. I will discuss the choice of `MAXEVENTS`=256 in Chapter 8.3.

```
typedef struct {
        uint8_t x[MAXHITS*MAXEVENTS];
        uint8_t y[MAXHITS*MAXEVENTS];
        int triplet[4*3*MAXEVENTS];
        uint16_t length[MAXEVENTS];
        uint16_t actualsize;
} Element;
```

In particular, the `x[]` and `y[]` arrays are filled with the coordinates of *all* the events of the batch, i.e. of the set of `MAXEVENTS` events processed concurrently:

$$x[\,] = \underbrace{x_0\, x_1\, x_2 \cdots x_{31}}_{\text{evt 0}}\ \underbrace{x_{32} \cdots x_{63}}_{\text{evt 1}}\ x_{64} \cdots \tag{7.2}$$

$$y[\,] = \underbrace{y_0\, y_1\, y_2 \cdots y_{31}}_{\text{evt 0}}\ \underbrace{y_{32} \cdots y_{63}}_{\text{evt 1}}\ y_{64} \cdots \tag{7.3}$$

The data format is fixed and independent from the actual hit multiplicity: if an event counts less than `MAXHITS` hits, some locations are left empty. Let us

[7] In Chapter 8 I will also test the possibility of increasing (actually doubling) this limit: if the GPU-based trigger proves efficient enough in the high-multiplicity region, the multiplicity cut could be eliminated from the hardware L0 trigger.

also define an array of indices that will be used later:

$$I[\,] = \underbrace{0\,1\,2\,\cdots\,31}_{\text{evt 0}}\ \underbrace{32\,\cdots\,63}_{\text{evt 1}}\ 64\,\cdots \tag{7.4}$$

This way, the x and y coordinates of the n-th hit in the scope of the current event can be accessed through the same index: $x_n = x\big[I[n]\big]$, and similarly $y_n = y\big[I[n]\big]$.

While the 8-bit coordinates of hit PMTs are read and stored in `x` and `y` arrays, the host program also checks if hits can be assigned to a triplet. For each event, in fact, a `triplet[]` array is defined, that hosts the indices of the hits selected for the initialisation of the pattern recognition step:

$$\texttt{triplet}[\,] = \underbrace{t_0\,t_1\,t_2}_{\text{XMAX}}\ \underbrace{t_3\,t_4\,t_5}_{\text{XMIN}}\ \underbrace{t_6\,t_7\,t_8}_{\text{YMIN}}\ \underbrace{t_9\,t_{10}\,t_{11}}_{\text{YMAX}} \qquad (\forall \text{ event}) \tag{7.5}$$

The first element of the `triplet[]` array is initialised with the index of the first hit. The indexes of those subsequent hits whose coordinates satisfy the conditions discussed in Section 6.4 are then written into the other elements. As an example, the following code shows how the `XMIN` portion of the triplet is populated.

```
#define THD (int) 8
#define sq_dist(j,k) ( pow(x[I[j]] - x[triplet[k]],2)
                     + pow(y[I[j]] - y[triplet[k]],2) )

//i = index of the i-th hit of the current event
//I[i] = index of the i-th hit of the batch
if (i > 2) {
        if (x[I[i]] < x[triplet[2]] && sq_dist(i,1) > THD
            && sq_dist(i,0) > THD) {
                if (x[i] > x[triplet[1]]) triplet[2]= I[i];
                else {
                        if (x[i] > x[triplet[0]]) {
                                triplet[2] = triplet[1];
                                triplet[1] = I[i];
                        }
                        else {
                                triplet[2] = triplet[1];
                                triplet[1] = triplet[0];
                                triplet[0] = I[i];
                        }
                }
        }
}
else if (i == 0) {
        triplet[i] = I[i];
}
else if (i == 1 && sq_dist(i,0) > THD) {
        if (x[I[i]] < triplet[0]) {
                triplet[i] = triplet[0];
                triplet[0] = I[i];
```

```
        }
        else triplet[i] = I[i];
}
else if (i == 2 && sq_dist(i,1) > THD
         && sq_dist(i,0) > THD) {
        if (x[I[i]] < triplet[1]) {
                if (x[I[i]] < triplet[0]) {
                        triplet[i] = triplet[1];
                        triplet[1] = triplet[0];
                        triplet[0] = I[i];
                }
                else {
                        triplet[i] = triplet[1];
                        triplet[1] = I[i];
                }
        }
        else triplet[i] = I[i];
}
```

Similarly, `XMAX`, `YMIN` and `YMAX` portions are populated. We obtain a 12-elements array $\{t_n\}$ such that:

$$x[t_0] \leq x[t_1] \leq x[t_2] \leq x[t_3] \leq x[t_4] \leq x[t_5] \tag{7.6}$$

$$y[t_6] \leq y[t_7] \leq y[t_8] \leq y[t_9] \leq y[t_{10}] \leq y[t_{11}] \tag{7.7}$$

These operations are carried out once per event. The indices used in the above code only fit the first event of the stream, while for the subsequent events the triplet index is transformed as $t_n \longrightarrow t_n + 12k$, where k is the event index. The array containing triplets will then have the form

$$\texttt{triplet}[\,] = \underbrace{t_0 \cdots t_{11}}_{\text{evt } 0}\ \underbrace{t_{12} \cdots t_{23}}_{\text{evt } 1}\ t_{24} \cdots \tag{7.8}$$

The operations described so far are executed on an event-by-event basis by the host program, running on the CPU, while event data are read from the NIC memory. This way, the population of the `triplet[]` array is interlaced with the readout and accumulation of UDP packets, so that smaller sets of instructions need to be processed by the GPU cores.

When the event batch is ready, data is copied to the global memory of the GPU, where the kernel is executed, and the results are copied back to the host memory by means of dedicated CUDA functions:

```
cudaMemcpyAsync((void*)gpu_mem, (void*)host_mem, sizeof(Element),
    cudaMemcpyHostToDevice, stream_id);

multiring<<<blocksPerGrid, threadsPerBlock, smemSize, stream_id>>>(
    gpu_mem, gpu_results, utils);

cudaMemcpyAsync((void*)host_results, (void*)gpu_results,sizeof(Result),
    cudaMemcpyDeviceToHost, stream_id);
```

where `stream_id` refers to the running stream, `gpu_mem` is a pointer to an `Element` of data stored in the global memory, `gpu_results` points to the global memory allocated for the `Result` structure, and `utils` provides temporary storage in a `UTILITY` structure for variables that need to be accessed while executing the circle fitting function:

```
typedef struct {
  int16_t x_candidate[4*MAXHITS*MAXEVENTS];
  int16_t y_candidate[4*MAXHITS*MAXEVENTS];
  float xHit[4*MAXHITS*MAXEVENTS];
  float yHit[4*MAXHITS*MAXEVENTS];
  float xm[4*MAXEVENTS];
  float ym[4*MAXEVENTS];
  float u[4*MAXHITS*MAXEVENTS];
  float v[4*MAXHITS*MAXEVENTS];
  float u2[4*MAXHITS*MAXEVENTS];
  float v2[4*MAXHITS*MAXEVENTS];
  float z[4*MAXHITS*MAXEVENTS];
  float z2[4*MAXHITS*MAXEVENTS];
  float uz[4*MAXHITS*MAXEVENTS];
  float vz[4*MAXHITS*MAXEVENTS];
  float uv[4*MAXHITS*MAXEVENTS];
  float zav[4*MAXEVENTS];
  float z2av[4*MAXEVENTS];
  float u2av[4*MAXEVENTS];
  float v2av[4*MAXEVENTS];
  float uvav[4*MAXEVENTS];
  float uzav[4*MAXEVENTS];
  float vzav[4*MAXEVENTS];
} UTILITY;

typedef struct {
  uint16_t actualsize;
  float xCenter[MAXEVENTS*4];
  float yCenter[MAXEVENTS*4];
  float radius[MAXEVENTS*4];
  int nHitsPerCandidate[MAXEVENTS*4];
  int nHitsPerEvent[MAXEVENTS];
} Result;
```

In the above code, `u` and `v` are the hit coordinates relative to the hits center of gravity frame; the other `UTILITY` variables are polynomial combinations of these two, that are used to compute the best ring parameters (see Appendix B for the actual CUDA implementation of the ring fit).

The kernel call parameters are defined as

```
int threads = MAXHITS;
const unsigned int eventsPerBlock = 1;
const unsigned int tripletsPerEvent = 4;
dim3 threadsPerBlock(threads,1,1);
int blocksPerGrid = ceil(MAXEVENTS*tripletsPerEvent
                         / (float)eventsPerBlock);
int smemSize = 7*sizeof(int)+MAXHITS*sizeof(float);
```

so that 4 blocks execute for each event, each one handling a pattern recognition stage initialised with a different triplet, and the corresponding single

ring fitting stage. Each block contains a number of threads equal to the maximum number of hits per event. Finally, `MAXEVENTS` events are evaluated concurrently. The amount of shared memory that has to be allocated will be discussed later.

7.2.3 Implementation of the kernel

The `multiring` kernel consists of 3 main subsets of instructions.

1. An array containing the integer x and y coordinates of the hits forming the current triplet (of dimension `6*sizeof(int)`) is created in the shared memory, visible to all the threads of the block.
2. Each thread evaluates the Ptolemy's theorem on a different hit of the event, and if it stands it copies the hit coordinates to the `UTILITY` structure. A counter, stored in the shared memory (`1*sizeof(int)`), is incremented every time a hit is accepted by this pattern recognition procedure.
3. A semi-parallelized version of the Taubin ring fit algorithm, discussed in Appendix A, is executed, and the `Result` structure is populated.

```
__device__ inline int Ptolemy(const uint8_t& x, const uint8_t& y,
                              const int* triplet);

__global__ void multiring(Element* Event, Result* Result, UTILITY* utils)
{
  //Allocate block shared memory
  __shared__ int triplet_s[6];
  __shared__ unsigned int length;
  __shared__ float reduction[32];

  //Distribute jobs to blocks and threads
  unsigned int eventIdx = blockIdx.x/4;
  unsigned int tripletIdx = blockIdx.x%4;
  unsigned int hitIdx = threadIdx.x + eventIdx * MAXHITS;
  unsigned int utilsIdx = MAXHITS*blockIdx.x + threadIdx.x;
  unsigned int elemIdx = 12*eventIdx+threadIdx.x
                         +3*tripletIdx;

  //Copy the current triplet in shared memory
  if(threadIdx.x < 3) {
      triplet_s[2*threadIdx.x] = Event->x[Event->triplet[elemIdx]];
      triplet_s[2*threadIdx.x+1] = Event->y[Event->triplet[elemIdx]];
    }
  __syncthreads();

  //Pattern recognition: copy 'good' hits
  if (Ptolemy(Event->x[hitIdx],Event->y[hitIdx], triplet_s)
     && (Event->x[hitIdx] != 0) && (Event->y[hitIdx] != 0)) {
      utils->x_candidate[utilsIdx] = Event->x[hitIdx];
      utils->y_candidate[utilsIdx] = Event->y[hitIdx];
```

```
    } else {
      utils->x_candidate[utilsIdx] = 0;
      utils->y_candidate[utilsIdx] = 0;
    }
  __syncthreads();

  //Compute number of hit candidates per ring
  length = 0;
  if(utils->x_candidate[utilsIdx] != 0) atomicAdd(&length, 1);
  __syncthreads();

  //Fill some results
  if (threadIdx.x == 0) {
      Result->actualsize = Event->actualsize;
      Result->hitsPerEvent[eventIdx] = Event->length[eventIdx];
      Result->nHitsPerCandidate[blockIdx.x] = length;
    }
  __syncthreads();

  //Execute ring fit
  Taubin(utils, reduction, Result);
}
```

The code implementing the `Taubin` function is available in Appendix B. This function finds the parameters of the best fitting circle in the reparametrized frame introduced in Section 6.3, also reverting the arbitrary 85-units shift applied to Montecarlo-generated data (see Eqn. 7.1). The results are then converted back to the laboratory frame.

Several arrays need to be reduced[8]: to achieve this, a modified version of the `reduce_kernel` function provided by the CUDA Data Parallel Primitives Library (CUDPP)[9] is used, that allows to compute the sum of an array in a small number of steps, as sketched and explained in Figure 7.5. All the threads of the block participate to the reduction, and therefore a portion of shared memory of size `MAXHITS*sizeof(float)` is allocated.

Both the kernel and the triplet finding procedures have been designed together with F. Pantaleo.

7.2.4 Implementation of the trigger

After the execution of the kernel, results are copied back to the host memory and made accessible to the user. A selective trigger program has been de-

[8] Those array operations that return a result with a smaller rank are referred to as "reductions". In this case, the sum of the array elements is computed (rank 0), in order to find their average.

[9] https://code.google.com/p/cudpp/

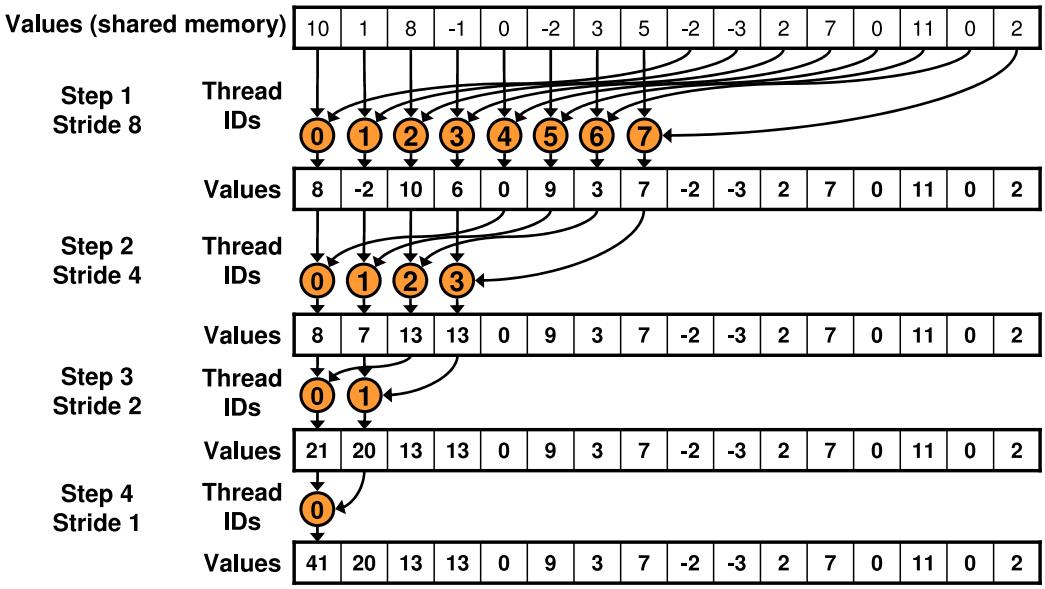


Figure 7.5: Parallel reduction of arrays within a single thread block [33]. The algorithm recursively reduces the output size, by computing the local sum of 2 array elements. Results are stored in the shared memory, visible to all the threads. A sequential addressing approach is used for memory access, in order to prevent memory bank conflicts. This way, the complexity of the algorithm is $\mathcal{O}(\log_2 N)$ where N is the size of the array.

veloped, that makes use of the ring fit results to perform basic kinematical analysis on the events.

Four conditions are set to check the quality of the identified rings:

- the radius belongs to a reasonable interval of values: $R_{min} < R < R_{max}$
- the number of hits on which a single-ring fit was executed exceeds a minimum threshold: $n > n_{th}$
- the fraction of hits assigned to the ring candidate, compared to the total number of hits in the event, exceeds a minimum threshold: $n/n_{hit} > f_{th}$
- the rings differ from each other: $|R_i - R_j| > \Delta R_{th}$ and the distance d_{ij} between the centres is $d_{ij} > D_{th}$.

where:

$$R_{min} = 50 \text{ cm} \tag{7.9}$$
$$R_{max} = 350 \text{ cm} \tag{7.10}$$
$$n_{th} = 5 \tag{7.11}$$
$$f_{th} = 1/3 \tag{7.12}$$

$$\Delta R_{th} = 1 \text{ cm} \tag{7.13}$$

$$D_{th} = 2 \text{ cm} \tag{7.14}$$

These numbers have been optimised in order to minimise the number of times signal events are erroneously rejected, while keeping the efficiency of the trigger as high as possible.

Rings that do not satisfy the above condition are discarded and not used to produce trigger decisions. The default behaviour is to produce a positive decision, i.e. the current event is forwarded to the next trigger levels unless some conditions, described below, are satisfied. For each event, if at least a "good" ring is identified, a negative trigger decision is produced if one of the following conditions holds:

- at least two different rings are detected, and the following conditions are satisfied:
  - the number of hits in the event is compatible with the presence of more than one charged particle: $n_{hit} > 16 + n_{th} \cdot (N_R - 1)$, where n_{hit} is the number of hits of the event and N_R is the number of different rings found;
  - the rings were found using different hits: $\sum_{i=1}^{N_R} n_i < 1.3 \cdot n_{hit}$ (a 30% tolerance is introduced to account for partial rings overlapping).
- at least one ring is detected whose radius exceeds $R_{th} = 179$ mm, according to the simulation results reported in Chapter 5;
- at least one ring is detected, corresponding to a reconstructed kinematics such that $\delta > 0.002$

where R_{th} and δ are the cut variables introduced in Chapter 5.

If none of the above conditions is satisfied, a positive trigger decision is produced.

8 Tests and conclusions

Contents

8.1 Trigger efficiency

A total of 5120 Montecarlo-generated background events and 1536 signal events from different samples were fed to the test framework. The number of times the algorithm successfully identified and rejected a background event is shown in Table 8.1. The efficiency for the signal was also computed and it is shown in the same table. The errors shown represent pure statistical uncertainties.

The trigger was run on two series of Montecarlo-generated data sets. One series consisted of events that had passed the standard L0 selection (see Chapter 3.2). This means that the multiplicity of hit PMTs was constrained between 5 and 32. The other series was not filtered through the L0 trigger requirements, but it was requested that each event had a number of hits comprised between 5 and 64. The latter events were analysed in order to explore the possibility of running a GPU-based trigger in parallel to a less biased hardware trigger, where the RICH hit multiplicity cut is relaxed or completely dropped. This way it would be possible to design special triggers

Data type		Sample	Discarded	Trigger efficiency (%)
$\pi^+\pi^0$	L0 selected	1280	800	62.5 ± 1.4
$\pi^+\pi^+\pi^-$	L0 selected	1536	920	59.9 ± 1.3
$\pi^+\pi^0$	$5 \leq n_{hit} \leq 64$	1024	618	60.4 ± 1.5
$\pi^+\pi^+\pi^-$	$5 \leq n_{hit} \leq 64$	1280	936	73.1 ± 1.2
$\pi^+\nu\bar{\nu}$	L0 + Signal cuts	768	38	95.1 ± 0.8
$\pi^+\nu\bar{\nu}$	Signal cuts	768	38	95.1 ± 0.8

Table 8.1: GPU trigger performance. The first four rows show the achieved rejection level for four different background samples. The last two rows show the $K^+ \to \pi^+\nu\bar{\nu}$ efficiency achieved before and after a standard L0 selection is performed (the rejected events differ in the two cases, even if the result is the same).

to explore additional Physics channels.

The data sets were further divided into subsets according to the number of hits in each event, to analyse the behaviour of the algorithm with respect to this variable. The performance of the trigger on the various subsets is shown in Figures 8.1(a) to 8.1(e).

We can infer from Figure 8.1(a) that the rejection for L0 filtered $\pi^+\pi^0$ events increases with the number of hits. In particular, the low rejection $\varepsilon_{\leqslant 10} = (40.5 \pm 3.4)\%$ shown for events with $5 \leq n_{hit} \leq 10$ arises from the difficulty of fitting a ring with so few points. The three following bins yield increasing rejection levels of $\varepsilon_{\leqslant 16} = (58.5 \pm 2.6)\%$, $\varepsilon_{\leqslant 23} = (69.4 \pm 2.0)\%$ and $\varepsilon_{\leqslant 32} = (76.0 \pm 3.1)\%$, respectively: as the number of hits per event increases, chances are that the fraction of spurious hits due to the presence of other partial rings decreases. The identification of rings is therefore easier, with the effect of increasing the rejection capability of the trigger (as discussed in Section 7.2.4, events are rejected only if the quality of the identified rings exceeds a threshold defined by a few parameters).

On the other hand, the initial low-multiplicity rejection capability achieved for 3π events with few hits is higher (plot 8.1(c)): in this case, $\varepsilon_{\leqslant 10} = (49.3 \pm 3.1)\%$. In fact, in this case it is more probable that at least one Čerenkov ring is present in the RICH detector. However, Figure 8.1(c) highlights that for 3π events there is no steep increase of rejection power with the number of hits, as there is for the 2-body background. This is probably due to the fact that the trigger is not optimized for the $\pi^+\pi^+\pi^-$

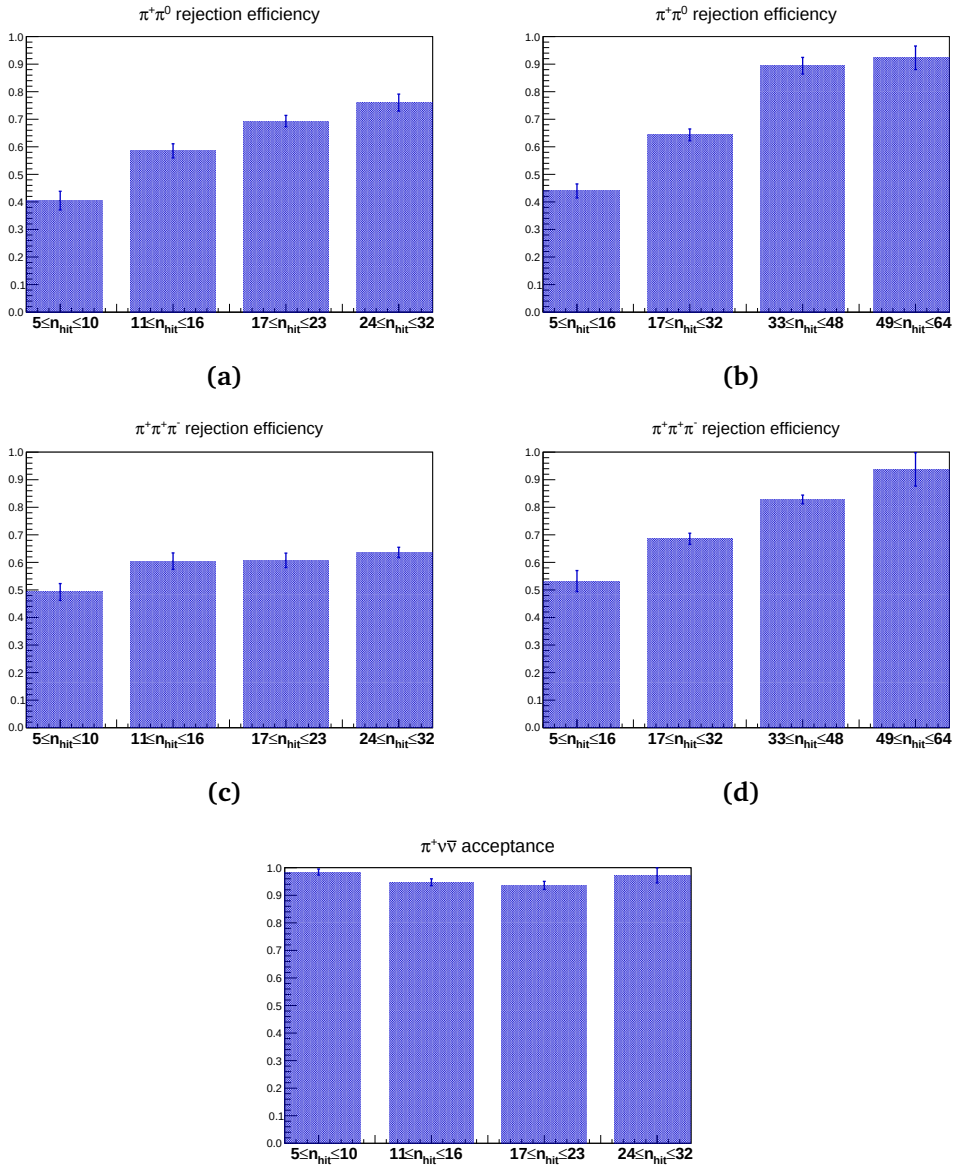


Figure 8.1: Trigger rejection for $\pi^+\pi^0$ **(a, b)** and $\pi^+\pi^+\pi^-$ **(c, d)** backgrounds, and signal efficiency **(e)**. The distributions on the right differ from the ones on the left for the data set used: on the left **(a, c)**, the trigger was run on Montecarlo-simulated events that passed the standard L0 selection. On the right **(b, d)**, only a relaxed $5 \leq n_{hit} \leq 64$ cut was applied to the input data set. Standard L0 selection and signal cuts (see Table 5.1) were applied for the $\pi\nu\bar{\nu}$ data set shown in the bottom panel **(e)**.

background. As the number of hits increases, it is easier to detect a single ring, but it becomes more difficult to separate an increasing number of rings. In principle, this difficulty might be overcome by using more triplets in the pattern recognition procedure. The trigger rejection computed in the other three bins reads: $\varepsilon_{\leqslant 16} = (60.4 \pm 3.0)\%$, $\varepsilon_{\leqslant 23} = (60.7 \pm 2.7)\%$ and $\varepsilon_{\leqslant 32} = (63.6 \pm 1.9)\%$.

Plots 8.1(b) and 8.1(d) display the level of background rejection achieved on the second series of input data, that was selected with a relaxed multiplicity cut only ($5 \leq n_{hit} \leq 64$).

In particular, from Figure 8.1(b) we learn that most $\pi^+\pi^0$ events can be correctly identified, and therefore rejected, when the number of hits found is greater than 32:

$$\varepsilon_{\leqslant 16} = (44.0 \pm 2.5)\% \tag{8.1}$$
$$\varepsilon_{\leqslant 32} = (64.3 \pm 2.1)\% \tag{8.2}$$
$$\varepsilon_{\leqslant 48} = (89.4 \pm 3.0)\% \tag{8.3}$$
$$\varepsilon_{\leqslant 64} = (92.3 \pm 4.3)\% \tag{8.4}$$

The level of event rate suppression achieved for the set of $\pi^+\pi^+\pi^-$ events can be read from plot 8.1(d):

$$\varepsilon_{\leqslant 16} = (53.2 \pm 3.8)\% \tag{8.5}$$
$$\varepsilon_{\leqslant 32} = (68.6 \pm 2.0)\% \tag{8.6}$$
$$\varepsilon_{\leqslant 48} = (82.8 \pm 1.6)\% \tag{8.7}$$
$$\varepsilon_{\leqslant 64} = (93.8 \pm 6.1)\% \tag{8.8}$$

The trigger proves most powerful for high-multiplicity events: it could be therefore feasible, in principle, to drop the RICH multiplicity cut from the standard trigger, and to rely only on the GPU-based alternative.

The results plotted in Figure 8.1(e), finally, ensure that the design of this algorithm is optimized to minimize unwanted suppression of the signal. The plot shows the efficiency for $K^+ \to \pi^+\nu\bar{\nu}$ decays that passed the standard L0 selections, for different intervals of hit multiplicity:

$$1 - \varepsilon_{\leqslant 10} = (98.4 \pm 1.1)\% \tag{8.9}$$
$$1 - \varepsilon_{\leqslant 16} = (94.7 \pm 1.2)\% \tag{8.10}$$
$$1 - \varepsilon_{\leqslant 23} = (93.6 \pm 1.5)\% \tag{8.11}$$
$$1 - \varepsilon_{\leqslant 32} = (97.2 \pm 2.7)\% \tag{8.12}$$

A study of the signal efficiency carried out on a set of $\pi^+\nu\bar{\nu}$ data not preliminarily filtered by the standard L0 selections yielded the same global results (Table 8.1). However, no $\pi^+\nu\bar{\nu}$ events with hit multiplicity higher than 32 could be found after performing the signal selection described in Table 5.1.

It must be pointed out that none of the selections implemented in the trigger (see Section 7.2.4) were designed specifically to cope with the 3π background. The algorithm should be further optimized in order to suppress the rate of these events. The tests reported in plots 8.1(b) and 8.1(d) were only intended to verify, on events that may feature more than one Čerenkov ring, the efficiency of the 'multi-ring' algorithm I developed.

Plots 8.2(a) to 8.2(e) analyse the data sets used for the tests described in this section. The multiplicity of hits in the events is shown as a function of the number of different rings detected, for the data sets described above. We can observe that:

- The number of times no rings could be identified approaches zero, for $\pi^+\pi^0$ events, as the number of hits in the event grows, and in particular for $n_{hit} > 32$. This explains why the trigger is more powerful for high-multiplicity primitives.
- While $\pi^+\pi^0$ events mostly exhibit a single Čerenkov ring, in a significant fraction of the 3π events analysed two rings were detected. This fraction increases with the hit multiplicity up to approximately $n_{hit} \simeq 35$, and then it decreases. This is a hint that the 4-triplets approach used for this work cannot efficiently handle events with a very large hit multiplicity.
- The number of times three different rings were detected is very small. However, it is not possible to state that this is due to a problem in the algorithm discussed here. The current NA62 analysis software does not provide any straightforward way to know the actual number of Čerenkov rings formed, or the actual number of charged particles that crossed the RICH volume. Besides, in order to verify the real ring detection efficiency, the format of data fed to the GPU should be modified to include "Montecarlo-truth" information. A "toy" simulation with a custom ring generating tool might be designed to perform these tests.

The algorithm I designed represents only a starting point for a real-time application of GPUs in High Energy Physics. The results achieved with this work can still be improved by devoting further effort to the development of a parallelized trigger algorithm, and to the study of its performance.

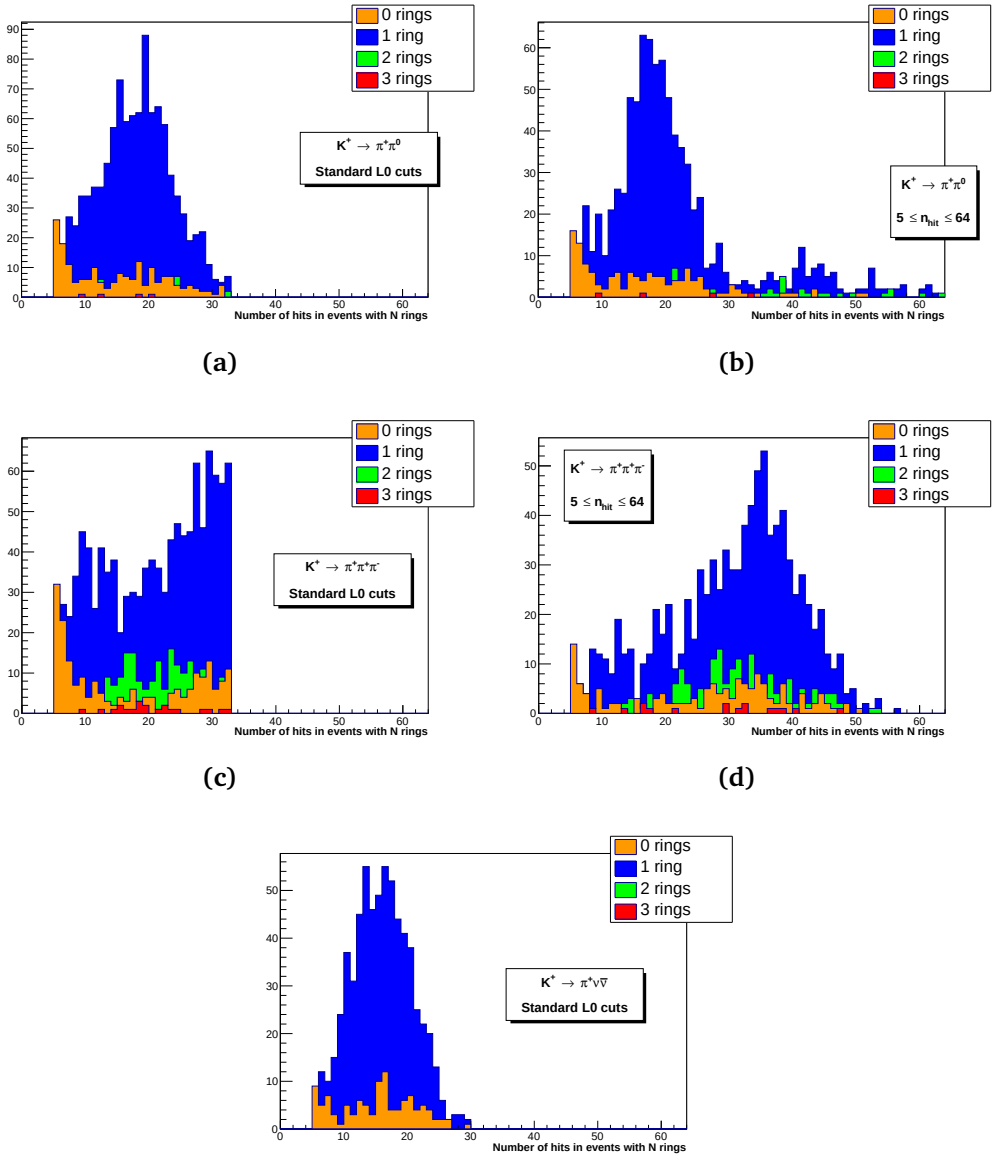


Figure 8.2: Hit multiplicity in the triggered data sets.

8.2 Timing tests

The possibility of actually implementing a useful GPU trigger algorithm ultimately depends on the possibility of running it at a high input rate, and within a time lower than the predefined trigger latency.

Before analysing the real time performance of the trigger, it is useful to remind that all GPU-related latencies depend on the specifications of the device on which the program is compiled and executed. Appendix C lists all the relevant features of the NVIDIA Tesla K20 GPU that was used for the tests reported here. In particular, this device has a warp size of 32 threads. This means that, if we set a number of threads per block equal to the maximum hit multiplicity, those data sets with hit multiplicity $n_{hit} \leq 32$ will be processed with a single warp execution, while the $5 \leq n_{hit} \leq 64$ data sets will require a queue of two warps for each triplet (triplets are processed concurrently, thanks to the high number of available CUDA cores).

The histograms of Figure 8.3 report separately the three CUDA latencies adding up to the total latency of the GPU trigger:

1. the time $\Delta t_{H\to D}$ needed to copy the data corresponding to a batch of events into the global memory of the GPU;
2. the kernel execution time Δt_{kernel};
3. the time $\Delta t_{D\to H}$ needed to copy the results back to the host user space (an eventually to the TDAQ system).

The measurements were repeated 100 times, using the same L0-filtered $\pi^+\pi^0$ data set (256 events). The measured latencies are:

$$\Delta t_{H\to D} = 27 \pm 2\ \mu s \tag{8.13}$$
$$\Delta t_{kernel} = 196 \pm 2\ \mu s \tag{8.14}$$
$$\Delta t_{D\to H} = 24 \pm 2\ \mu s \tag{8.15}$$

summing up to a total latency Δt_{tot} (Figure 8.4(a))

$$\Delta t_{tot} = 247 \pm 6\ \mu s \tag{8.16}$$

and therefore to a total latency *per event*:

$$\Delta t_{evt} = 0.97 \pm 0.02\ \mu s \tag{8.17}$$

since 256 events are processed per batch.

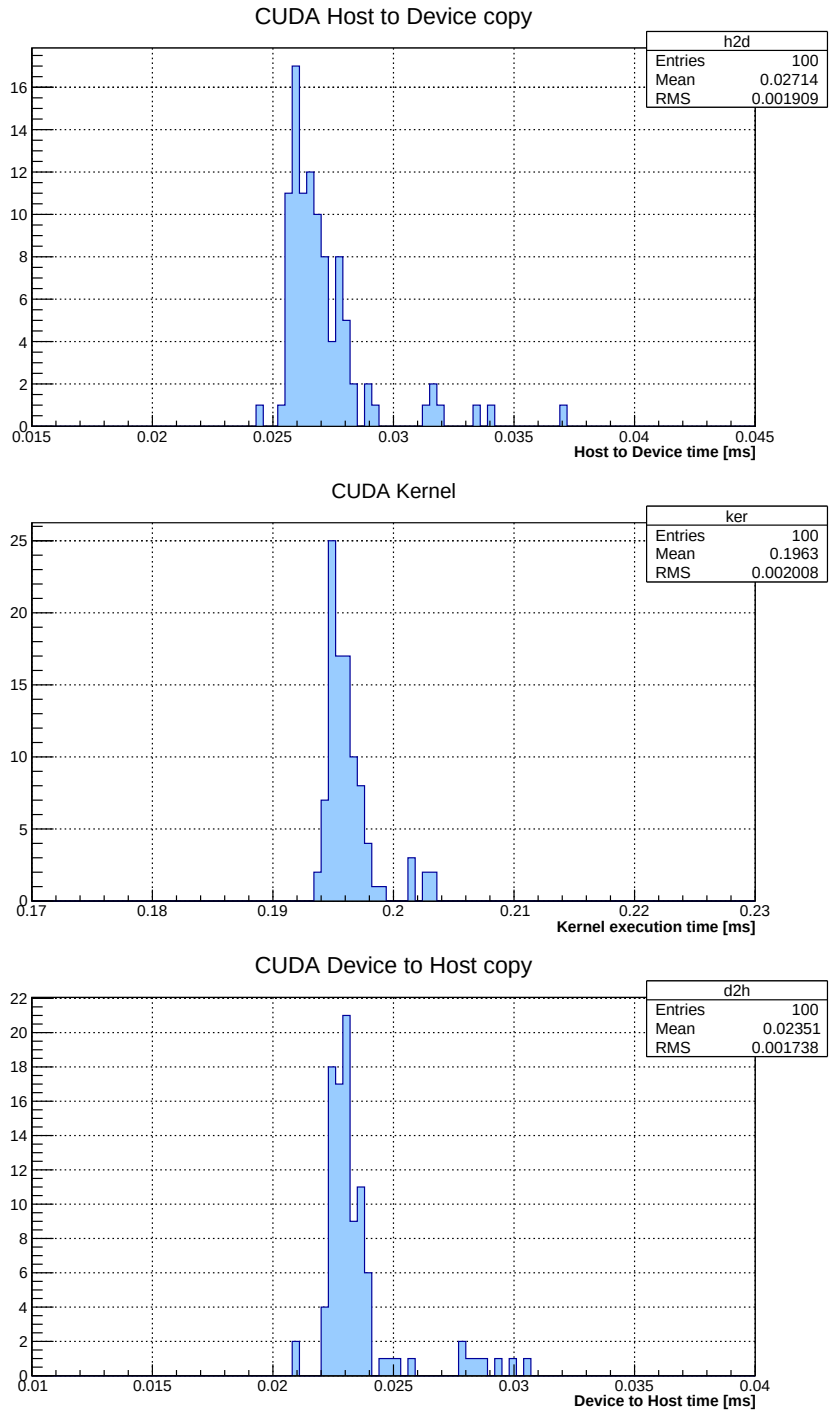


Figure 8.3: GPU execution latencies for a batch of 256 $\pi^+\pi^0$ events preliminarily filtered by the standard L0 selection. The measurement was repeated 100 times. The first panel shows the time needed to copy the whole data set from the host RAM to the global memory of the GPU. The second panel shows the kernel execution time. The last one shows the time needed to copy the results back to the host.

These simulated events were preloaded in an internal emulation buffer of the TEL62 TDAQ boards and sent from there to the NIC at 600±25 kHz rate. Due to the limited buffer memory of the TEL62 board, it was not possible to prepare large data files with more than 256 events; nor was it possible to transmit different samples in a queue. Therefore, in order to measure the time response stability of the trigger, we had to capture[1] a set of UDP packets (256 events) and transmit them in a loop from an Ethernet interface to another one on the same PC, using the `pfsend` application provided with the DNA driver APIs[2]. In the future, it will be useful to test the framework with a continuous stream of real data from TEL62 boards.

Figure 8.4(b) reports the results of some time stability tests. The total execution time was analysed using all the data sets discussed in Section 8.1. Here, the total GPU latency is reported as a function of the number of the processed batch: 100 batches = 25600 events were transmitted to the NIC in a row, with event packets sent at a rate of 0.1 GB/s, that corresponds to about 0.7 to 5.6 MHz event rate depending on the number of hits. These results show that the GPU performs best when it is under full load, i.e. the maximum execution speed is not reached for the first and last few executions. However, this should not be a problem if the trigger operates with a stable particle beam.

From Figures 8.4(a) and 8.4(b) we can also infer that the *plateau* latency of the trigger is not doubled when the number of threads per block is raised from 32 to 64 ($5 \leq n_{hit} \leq 64$ event sets). In fact, the total kernel execution time results from the sum of two contributions: a kernel launch time, that is fixed and not measurable, and the time needed to perform the operations issued in the kernel function. For example, the total time needed to process 256 3π events, with hit multiplicity $5 \leq n_{hit} \leq 64$, amounts to

$$\Delta t_{tot} = 271 \pm 5\ \mu s \tag{8.18}$$

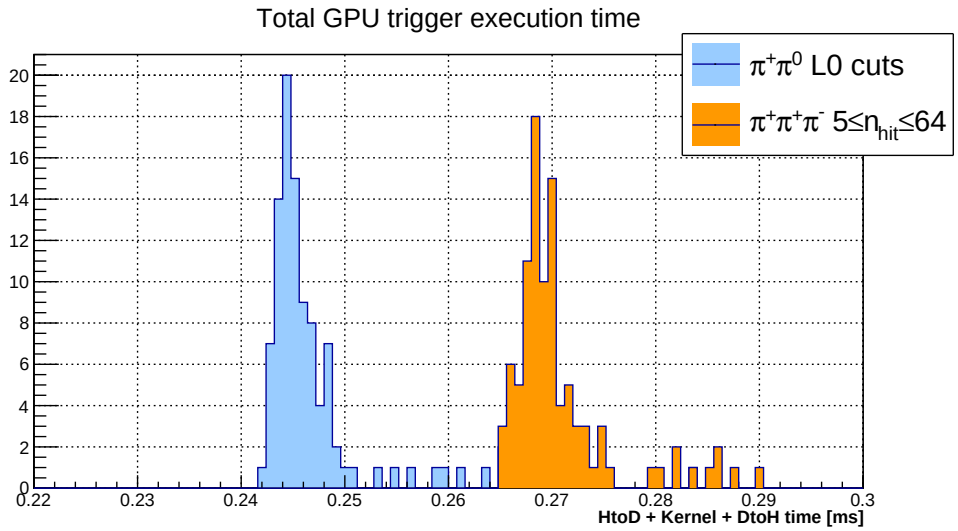
corresponding to

$$\Delta t_{evt} = 1.06 \pm 0.02\ \mu s \tag{8.19}$$

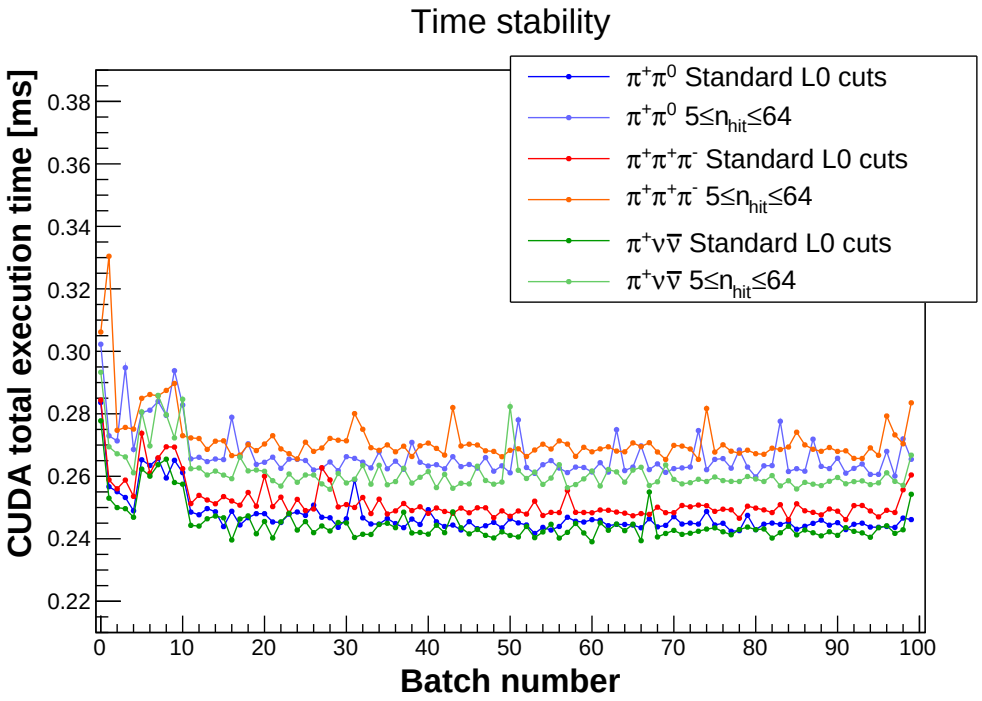
per event, to be compared with the result reported above in Eqn. 8.17, that was achieved by running the kernel function on 32 threads per block.

[1] The Wireshark network protocol analyser (http://www.wireshark.org/) was used for this purpose.

[2] http://www.ntop.org/solutions/wire-speed-traffic-generation/



(a) Total GPU trigger latency for a batch of 256 events



(b) Total GPU latency as a function of the number of iterations.

Figure 8.4: Total trigger execution time for 256 L0-filtered $\pi^+\pi^0$ and 256 unfiltered $\pi^+\pi^+\pi^-$ events (top panel). The measurement was repeated 100 times. The bottom plot shows the same quantity as a function of the number of iteration, and for all the data sets used for the tests discussed in Section 8.1.

The above reported timing tests are encouraging. In particular, from the results in Eqns. 8.17 and 8.19 we learn that 10 Tesla K20-equivalent GPUs would be enough to handle a data input rate of 10 MHz. A "triggerless" approach using normal processors would require a much more larger and expensive PC farm.

8.3 Possible improvements and outlook

The algorithm developed for this thesis focusses on the suppression of residual $K^+ \to \pi^+\pi^0$ background after the standard hardware L0 selection. However, the results shown in the preceding sections suggest that such an approach could allow to drop the hit multiplicity requirement from the standard hardware L0 trigger.

A GPU trigger might therefore be optimized to this end, and it should be investigated whether it could be used to suppress both the $\pi^+\pi^0$ and the 3π backgrounds to the levels currently achieved by the standard trigger (Table 3.2).

In particular, a more thorough Montecarlo study should support the design of a $\pi^+\pi^+\pi^-$ rejection trigger. If the average ring multiplicity proves to be sensibly higher than 2 for an important fraction of events, the idea of creating more triplets should be considered. For example, after the definition of the initial `XMAX`, `XMIN`, `YMAX` and `YMIN` triplets, one could rotate the PMTs frame by 45°, and define 4 new triplets as before.

A dedicated Montecarlo simulation should be prepared in order to test the ring identification efficiency in 3-ring and 4-ring events.

The time performance of the trigger might be further improved in many ways. As a starting point, the framework should be tested with a continuous stream of data (if the TEL62 boards could not be used for this purpose, with controllable simulated data, a very large batch of data could be looped between two Ethernet interfaces of the same PC). In this way one could optimise the maximum number of events processed per batch (`MAXEVENTS`), that was set to 256 in order not to increase the actual largest latency of the process, i.e. the time elapsed while the host accumulates the data to process on the GPU, that scales almost linearly with the number of events [42, 55].

Moreover, alternative designs might be taken into consideration. In the

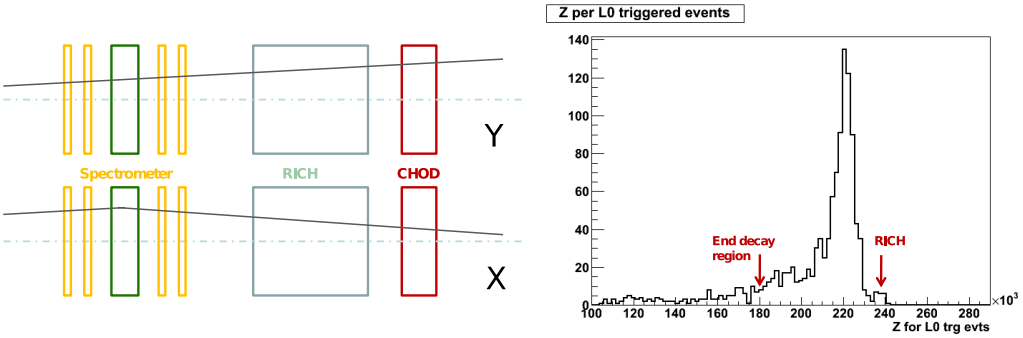


Figure 8.5: Using RICH and CHOD at the L1 trigger level, it would be possible to reduce the rate of $\pi^+\pi^0$ down to approximately 20%, even with a limited resolution on the reconstruction of the decay vertex. Plot from [41].

present implementation of the algorithm, during the single-ring fit routine only one thread executes the operations issued, except for the reduction of the arrays. In particular, the `Taubin` function, listed in Appendix B, exploits a Newton method to find the roots of the characteristic polynomial from which the best ring parameters are later extracted. This part of the fitting routine cannot be parallelized, and it might be moved to the host.

Another solution was proposed by G. Lamanna: we could exploit workload parallelization only at the event level, dropping the organization in blocks and allocating a thread for each event. This way the code would be serialized, and therefore easier to maintain; however, the number of events processed per batch would be much larger, and tests should be done to verify if the whole process – accumulation of data plus execution of the trigger process – exhibits a latency low enough to be used as real-time trigger.

From the point of view of the selections that could be performed at the first trigger levels, additional background rejection could be obtained at L1 by relating the information from the RICH to that of another detector able to measure the position of the crossing track, like the CHOD. Figure 8.5 shows that most of the residual $\pi^+\pi^0$ content after the L0 selection is characterised by a production vertex very close to the RICH detector, out of the fiducial z_{vtx} region allowed for the signal analysis [41]. The angle information from the RICH and the track impact point on the CHOD could be combined to propagate the track backwards until it crosses the beam axis. This way, even with a relatively poor longitudinal vertex resolution of $\sim$10 m, up to 80% of the residual $\pi^+\pi^0$ events could be rejected.

8.4 Conclusions

This research project represents the first approach to a GPU-based real-time trigger solution in High Energy Physics experiments. It started as a bare idea, supported by early tests by the NA62 collaboration [26, 43, 55], and that idea has now been implemented into a working online trigger framework prototype.

A fast, parallelized and seedless algorithm has been developed, capable of finding multiple rings at real-time trigger level, a challenging idea that proved to be feasible. The possibility to fit Čerenkov rings online allows selective trigger conditions to be set, based on kinematical constraints: the radius and centre of the rings in fact provide a rough measurement of the particle momentum and of its direction, with an assumption on the particle type.

The analysis reported in Chapter 5 demonstrates that the main background processes can be suppressed, by at least 60%, in addition to what can be achieved by a non-selective hardware L0 trigger. Basic kinematical analysis can be performed on data coming from a single detector in order to provide early trigger decisions. If this idea is pursued, it could lead to a "triggerless" design of future experiments: data could be evaluated in real time in order to assess the signal and background content, with no need for a previous event selection performed in hardware.

Within the scope of the NA62 experiment, a GPU-based trigger could be built, that reduces the data rate arising from $\pi^+\pi^0$ events up to 92% for high PMT hit multiplicity events. The results reported in Section 8.1 demonstrate that also the $\pi^+\pi^+\pi^-$ background can be suppressed down to a similar level, even with a non optimized algorithm. Additional trigger requirements may be set up in order to further reduce this rate and to increase the trigger rejection for low multiplicity events. Moreover, this approach makes it possible to set up specific triggers for lepton number and lepton flavour violating K^+ decay modes, as discussed in Section 6.1.

The latency of the GPU-based algorithm developed for this thesis proved to be stable, and low enough to be compatible with the requirements of a real-time trigger for high rate experiments.

The results reported in this thesis prove that alternative trigger designs are feasible for the NA62 experiment, and dedicated triggers to study other

K^+ decay modes are accessible and should be further investigated.

The graphics cards industry has seen lively development since the first Personal Computers were released. Today, this sector is one of the most supported by the IT industry. General-Purpose computing on GPUs is a new research branch in fast expansion, that has already introduced advantages for scientific computing.

This work represents a starting point to introduce the use of flexible GPU-based real-time triggers in High Energy Physics.

A

Ring fitting algorithms

Contents

A.1 Problem definition

Two approaches exist to the problem of fitting a circle to experimental data. Our final purpose is to minimize the distance between our set of data points (x_i, y_i) and a generic circle, finding the parameters identifying the best fitting ring. This can be achieved in one of two ways:

- iterative methods that converge to the minimum of

$$\mathcal{F} = \sum_i d_i^2 \tag{A.1}$$

$$d_i = \sqrt{(x_i - a)^2 + (y_i - b)^2} - R \tag{A.2}$$

where the d_i are the **orthogonal** (geometric) distances from the experimental points (x_i, y_i) to the hypothetical circle of centre (a, b) and radius R;

- approximate algorithms that replace the distances with some other quantities f_i defined by simple algebraic rules, and minimize the resulting sum

$$\mathcal{F}' = \sum_i f_i^2 \tag{A.3}$$

These two types of fitting algorithms are referred to as **geometric** and **algebraic** algorithms, respectively.

In the next sections I will briefly discuss the algorithms used for this project. The **math** algorithm is the simplest mathematical approach to the problem, whereas the **Taubin** method makes use of some approximation to improve robustness and computational ease. They are both algebraic fitting algorithms. Nevertheless, after the fitting, the circle must be expressed in its geometrical parametrisation in order to extract the data we need, i.e. the radius and the position of the centre. It will be convenient to start the discussion from the two possible parametrisations, geometric and algebraic, and the relationship between them.

A.1.1 Geometrical parametrisation

The canonical equation of a circle of centre (a, b) and radius R is

$$(x-a)^2 + (y-b)^2 - R^2 = 0 \tag{A.4}$$

with the constraint $R > 0$. If we define $r_i(a,b) \equiv \sqrt{(x_i-a)^2 + (y_i-b)^2}$, then the minimization function becomes

$$\mathcal{F}(a,b,R) = \sum_i (r_i - R)^2 \tag{A.5}$$

and its minimum with respect to R is attained at $\hat{R} = \frac{1}{n}\sum_i r_i = \bar{r}$, where n is the number of data points. $\hat{R}$ is the best estimator for the radius R in this picture. We can then express $\mathcal{F}$ as a function of the two variables a and b only:

$$\begin{aligned} \mathcal{F}(a,b,\bar{r}) &= \sum_i (r_i - \bar{r})^2 \qquad (A.6) \\ &= \sum_i \left[\sqrt{(x_i-a)^2 + (y_i-b)^2} - \frac{1}{n}\sum_j \sqrt{(x_j-a)^2 + (y_j-b)^2} \right]^2 \end{aligned}$$

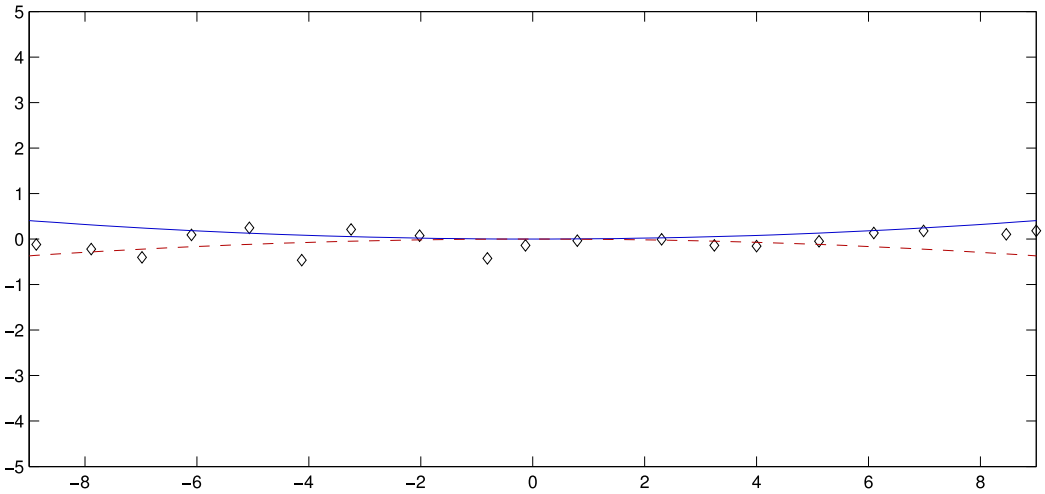


Figure A.1: Data points sampled along a short arc. The blue line represents the correct fit. A small inaccuracy may produce a wrong fit (dashed line) with opposite curvature [24].

A drawback of this parametrisation is that its robustness with respect to the parameter R is poor when treating experimental points that lie on a very short arc. A small perturbation may result in arbitrarily large values of R, even with opposite signs (Figure A.1).

A.1.2 Algebraic parametrisation

An more elegant approach to the problem was proposed by Pratt in 1987 [56]. A circle is algebraically described by the equation

$$A\left(x^2+y^2\right)+Bx+Cy+D=0 \tag{A.7}$$

or, equivalently,

$$\left(x+\frac{B}{2A}\right)^2+\left(y+\frac{C}{2A}\right)^2-\frac{B^2+C^2-4AD}{4A^2}=0 \tag{A.8}$$

Compared to Eqn. A.7, Eqn. A.8 has the advantage of making two constraints explicit:

$$A\neq 0 \tag{A.9}$$
$$B^2+C^2-4AD>0 \tag{A.10}$$

An additional constraint may be set, since the circle parameters only need to be determined up to a multiplicative factor. Indeed, setting $A=1$ leads

to the usual canonical equation A.4.

Pratt's choice is more convenient: the choice of

$$B^2 + C^2 - 4AD = 1 \tag{A.11}$$
$$A > 0 \tag{A.12}$$

automatically ensures the constraint $B^2 + C^2 - 4AD > 0$. The condition on the sign of A ensures a unique correspondence between a set of parameters (A, B, C, D) and a circle. With this parametrisation, the distance of a point (x_i, y_i) to the circle is given (after some algebraic manipulations) by

$$l_i = \frac{2F_i}{1 + \sqrt{1 + 4AF_i}} \tag{A.13}$$

where

$$F_i = A\left(x_i^2 + y_i^2\right) + Bx_i + Cy_i + D \tag{A.14}$$
$$1 + 4AF_i = \frac{(x_i - a)^2 + (y_i - b)^2}{R^2} \tag{A.15}$$

The term $1+4AF_i$ inside the square root is always positive, hence l_i is always computable. This formula can be proved to be more numerically robust as compared to the geometric parametrisation [24].

Conversion formulas between algebraic and geometric parameters

$$a = -\frac{B}{2A} \qquad b = -\frac{C}{2A} \qquad R^2 = \frac{B^2 + C^2 - 4AD}{4A^2} \tag{A.16}$$
$$A = \pm\frac{1}{2R} \qquad B = -2Aa \qquad C = -2Ab \qquad D = \frac{B^2 + C^2 - 1}{4A} \tag{A.17}$$

Let us now see the two algorithms *math* and *Taubin* I mentioned at the beginning of this chapter.

A.2 The "math" algorithm

Let us define

$$f_i \equiv (x_i - a)^2 + (y_i - b)^2 - R^2 \tag{A.18}$$

The simplest algebraic model to fit a circle $(x_i - a)^2 + (y_i - b)^2 + R^2 = 0$ to a set of points would minimize the algebraic expression [19, 24]

$$\mathcal{F}_1 = \sum_i f_i^2 \tag{A.19}$$

Note that f_i does *not* represent the square orthogonal distance to the circle of the point (x_i, y_i) defined in Eqn. A.2. However, f_i is small if and only if the point lies near to the circle: for this reason, some authors [56] call f_i the *algebraic distance* from (x_i, y_i) to the circle. This approach is also the base for the Kåsa, Pratt, and Taubin algorithms.

The change of parameters described in Eqns. A.16 and A.17 with the choice $A \equiv 1$ linearises the derivatives of the objective function $\mathcal{F}_1$:

$$\mathcal{F}_1 = \sum_i \left(x_i^2 + y_i^2 + Bx_i + Cy_i + D\right)^2 \tag{A.20}$$

Now differentiating $\mathcal{F}_1$ with respect to the parameters B, C, D yields a system of linear equations, by solving which we can compute B, C, D and, finally, convert them back to the geometrical parameters a, b, R [24].

A.2.1 Implementation of the "math" algorithm

We will solve the problem in a suitable (u, v) coordinate system, and then transform the solutions back to the laboratory frame (x, y) [19]. Writing

$$\bar{x} = \frac{1}{n}\sum_i x_i \qquad \bar{y} = \frac{1}{n}\sum_i y_i \tag{A.21}$$

$$u_i \equiv x_i - \bar{x} \qquad v_i \equiv y_i - \bar{y} \tag{A.22}$$

we obtain:

$$\mathcal{F}_1{}' = \sum_i g_i^2 \tag{A.23}$$

$$g_i = (u_i - u_c)^2 + (v_i - v_c)^2 - R^2 \tag{A.24}$$

where (u_c, v_c) are the coordinates of the centre of the circle, as computed in the (u, v) frame:

$$u_c = a - \bar{x} \tag{A.25}$$

$$v_c = b - \bar{y} \tag{A.26}$$

In order to minimize $\mathcal{F}_1{}'$ we compute its derivatives with respect to the parameters u_c, v_c and R^2:

$$\frac{\partial \mathcal{F}_1{}'}{\partial R^2} = 2\sum_i g_i \frac{\partial g_i}{\partial R^2} = -2\sum_i g_i \tag{A.27}$$

$$\frac{\partial \mathcal{F}_1{}'}{\partial u_c} = 2\sum_i g_i \frac{\partial g_i}{\partial u_c} = -4\sum_i u_i\, g_i + 4u_c \sum_i g_i \tag{A.28}$$

$$\frac{\partial \mathcal{F}_1{}'}{\partial v_c} = 2\sum_i g_i \frac{\partial g_i}{\partial v_c} = -4\sum_i v_i\, g_i + 4u_c \sum_i g_i \tag{A.29}$$

and then set them to zero. This system gives the unique solution

$$\sum_i g_i = 0 \qquad \sum_i u_i\, g_i = 0 \qquad \sum_i v_i\, g_i = 0 \tag{A.30}$$

Now let

$$S_u \equiv \sum_i u_i \qquad S_{uu} \equiv \sum_i u_i^2 \qquad S_{uuu} \equiv \sum_i u_i^3 \quad \text{etc.} \tag{A.31}$$

and similarly for S_v, S_{uv} and so on, where $S_u = S_v = 0$ by definition. Adopting this notation, if we expand Eqns. A.30 we obtain the system

$$u_c S_{uu} + v_c S_{uv} = \frac{1}{2}\left(S_{uuu} + S_{uvv}\right) \tag{A.32}$$

$$u_c S_{uv} + v_c S_{vv} = \frac{1}{2}\left(S_{vvv} + S_{uuv}\right) \tag{A.33}$$

$$n\left(u_c^2 + v_c^2 - R^2\right) + S_{uu} + S_{vv} = 0 \tag{A.34}$$

which in turn yelds the solutions

$$u_c = \frac{\frac{S_{uv}}{2}\left(S_{vvv} + S_{uuv}\right) - \frac{S_{vv}}{2}\left(S_{uuu} + S_{uvv}\right)}{S_{uv}^2 - S_{vv}^2} \tag{A.35}$$

$$v_c = \frac{\frac{1}{2}\left(S_{uuu} + S_{uvv}\right) - u_c S_{uu}}{S_{uv}} \tag{A.36}$$

$$R^2 = u_c^2 + v_c^2 + \frac{S_{uu} + S_{vv}}{n} \tag{A.37}$$

The centre of the circle in the original coordinate system will be $(a, b) = (u_c, v_c) + (\bar{x}, \bar{y})$.

A.3 The Taubin algorithm

The very simple ring fitting algorithm developed by Kåsa and the later, more stable and "elegant" Pratt modification are throughly described in [24, 56, 66] and we will not discuss them here in detail.

Another method for algebraic circle fitting was proposed in 1991 by Gabriel Taubin [66]. His method is very similar to the Pratt algorithm both in design and performance, but:

- it can be generalized to ellipses and other algebraic curves;
- it features smaller bias and higher statistical accuracy;
- it requires simpler computations, and therefore it is less "expensive".

The Taubin algorithm is based on the minimization of the function

$$\mathcal{F}_2 = \frac{\sum_i \left[(x_i - a)^2 + (y_i - b)^2 - R^2\right]^2}{\sum_i \left[(x_i - a)^2 + (y_i - b)^2\right]} \tag{A.38}$$

as a result of the development of few simple algebraic ideas.

Take the so-called algebraic distances f_i as defined in Eqn. A.18. Now consider the expansion

$$r_i \equiv \sqrt{(x_i - a)^2 + (y_i - b)^2} \tag{A.39}$$

$$f_i = (r_i - R)(r_i + R) \tag{A.40}$$

The Kåsa and "math" methods effectively minimize the sum $\mathcal{F}_1 = \sum_i d_i^2 D_i^2$, where we can identify the two factors as the distances from the $i-$th point to the *nearest* point of the circle ($d_i = r_i - R$, as defined in Eqn. A.2) and to the *farthest* point on the circle ($D_i = r_i + R$). Unfortunately, this is the reason why the Kåsa method, and therefore also the "math" mathod, are heavily biased towards smaller circles, underestimating the radius when the data points lie on a short arc. By minimizing the averaged product of d_i^2 and D_i^2 one may only find the best trade-off between the two distances: a smaller circle would yeld smaller D_i^2, minimizing $\mathcal{F}_1$ regardless of the larger d_i^2 [24]. The solution found by Pratt [56] solves the statistical bias introduced by Kåsa, but involves matrix algebra and therefore it is less numerically stable.

Now observe that, if the points are close to the circle,

$$\left.\begin{array}{l} D_i = d_i + 2R \\ |d_i| \ll R \end{array}\right\} \Longrightarrow \mathcal{F}_1 \approx 4R^2 \sum_i d_i^2 \tag{A.41}$$

A natural assumption $|d_i| \ll R$ yields:

$$R^2 \approx (R + d_i)^2 = (x_i - a)^2 + (y_i - b)^2 = r_i^2 \tag{A.42}$$

Positive and negative fluctuations will tend to cancel out if we average over the whole sample:

$$R^2 \approx \frac{1}{n} \sum_i \left[(x_i - a)^2 + (y_i - b)^2 \right] = \frac{1}{n} \sum_i r_i^2 \tag{A.43}$$

Eqn. A.41 now factorizes into

$$\mathcal{F}_1 \approx \left(\frac{4}{n} \sum_i r_i^2 \right) \times \left(\sum_i d_i^2 \right) \equiv \mathcal{F}_r \times \mathcal{F}_d \tag{A.44}$$

We ultimately wish to minimize the second factor ($\mathcal{F}_d$). Comparing Eqns. A.18–A.19 and the expression A.41 for $\mathcal{F}_1$, we can define a new minimization function $\mathcal{F}_2 \equiv \mathcal{F}_1 / \mathcal{F}_r$

$$\begin{aligned} \mathcal{F}_2 &= \frac{\sum_i \left[(x_i - a)^2 + (y_i - b)^2 - R^2 \right]^2}{4/n \sum_i \left[(x_i - a)^2 + (y_i - b)^2 \right]} \\ &= \frac{\sum_i \left(z_i - 2ax_i - 2by_i + a^2 + b^2 - R^2 \right)^2}{4/n \sum_i \left(z_i - 2ax_i - 2by_i + a^2 + b^2 \right)} \end{aligned} \tag{A.45}$$

by minimizing which we can approximately achieve our goal. Above, we denoted $z_i \equiv x_i^2 + y_i^2$ for brevity.

Switching to the algebraic parameters (A, B, C, D) defined in Eqn. A.17, we get

$$\mathcal{F}_2 = n \frac{\sum_i \left(A z_i + B x_i + C y_i + D \right)^2}{\sum_i \left(4A^2 z_i + 4AB x_i + 4AC y_i + B^2 + C^2 \right)} \tag{A.46}$$

The minimization of A.46 is equivalent to minimizing

$$\mathcal{F}_3 \equiv \sum_i \left(A z_i + B x_i + C y_i + D \right)^2 \tag{A.47}$$

with respect to the parameters (A, B, C, D), with the constraint

$$4A^2 z_i + 4AB x_i + 4AC y_i + B^2 + C^2 = 1 \tag{A.48}$$

as discussed in Section A.1.2.

Finally, in **matrix form**, Taubin's algorithm requires the minimization of

$$\mathcal{F}_3(\mathbf{A}) = \mathbf{A}^T \left(\mathbf{X}^T\mathbf{X}\right) \mathbf{A} \tag{A.49}$$

with the constraint $\mathbf{A}^T\mathbf{T}\mathbf{A} = 1$, where:

$$\mathbf{A} = \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} 4\overline{z} & 2\overline{x} & 2\overline{y} & 0 \\ 2\overline{x} & 1 & 0 & 0 \\ 2\overline{y} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.50}$$

$$\mathbf{X} = \begin{bmatrix} z_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ z_n & x_n & y_n & 1 \end{bmatrix} \tag{A.51}$$

where $\bar{z} = \frac{1}{n}\sum_i z_i$, and $\mathbf{X}$ is called the *extended data matrix*.

A.3.1 Implementation of the Taubin algorithm

Eqns. A.46, A.47 and A.48 can be much simplified, reducing the problem to that of finding the roots of a third order polynomial [24].

$\mathcal{F}_2$ is a second order polynomial in D, and therefore admits a unique minimum in $D = -A\overline{z} - B\overline{x} - C\overline{y}$. Substituting in Eqn. A.48, we get the equations of the two constraints to the minimization of $\mathcal{F}_2$ (Eqn. A.47):

$$D = -A\overline{z} - B\overline{x} - C\overline{y} \tag{A.52}$$
$$B^2 + C^2 - 4AD = 1 \tag{A.53}$$

Now perform the change of coordinates described in Eqn. A.22, and let $w_i \equiv u_i^2 + v_i^2$. The above contraints become

$$D = -A\overline{w} \tag{A.54}$$
$$A_0^2 + B^2 + C^2 = 1 \tag{A.55}$$

having defined $A_0^2 \equiv 4\overline{w}A^2$, and the function to minimize can be written as

$$\mathcal{F}_5 = \sum_i \left(\frac{w_i - \overline{w}}{2\sqrt{\overline{w}}}A_0 + Bu_i + Cv_i\right)^2 \tag{A.56}$$

In matrix form, this corresponds to

$$\begin{cases} \mathcal{F}_5(\mathbf{A_0}) = \mathbf{A_0}^T \left(\mathbf{U}^T\mathbf{U}\right) \mathbf{A_0} \\ \mathbf{A_0}^T\mathbf{A_0} = 1 \end{cases} \tag{A.57}$$

where

$$\mathbf{A_0} = \begin{bmatrix} A_0 \\ B \\ C \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} (w_1 - \overline{w})/2\sqrt{\overline{w}} & u_1 & v_1 \\ \vdots & \vdots & \vdots \\ (w_n - \overline{w})/2\sqrt{\overline{w}} & u_n & v_n \end{bmatrix} \tag{A.58}$$

are the modified vector of parameters and data matrix.

The minimization can be performed using a Lagrange multiplier η:

$$\mathcal{G}(\mathbf{A_0}, \eta) \equiv \mathbf{A_0}^T \left(\mathbf{U}^T\mathbf{U}\right) \mathbf{A_0} - \eta \left(\mathbf{A_0}^T\mathbf{A_0} - 1\right) \tag{A.59}$$

$$\frac{\partial \mathcal{G}}{\partial \mathbf{A_0}} = \left(\mathbf{U}^T\mathbf{U}\right) \mathbf{A_0} - \eta\, \mathbf{A_0} \equiv 0 \tag{A.60}$$

$$\left(\mathbf{U}^T\mathbf{U}\right) \mathbf{A_0} = \eta\, \mathbf{A_0} \tag{A.61}$$

Our algorithm reduces to the **eigenvalue problem** of Eqn. A.61. Observe that

$$\mathcal{F}_5(\mathbf{A_0}) = \mathbf{A_0}^T \left(\mathbf{U}^T\mathbf{U}\right) \mathbf{A_0} = \eta\, \mathbf{A_0}^T\mathbf{A_0} = \eta \tag{A.62}$$

The minimum of Eqn. A.56 is therefore attained by the eigenvector $\mathbf{A_0}'$ of $\mathbf{U}^T\mathbf{U}$ corresponding to its smallest eigenvalue η' [24]. $\mathbf{U}^T\mathbf{U}$ is symmetric and positive-definite, therefore its eigenvalues are all real and positive. The characteristic equation of this problem

$$\det\left(\eta\mathbf{I} - \mathbf{U}^T\mathbf{U}\right) = 0 \tag{A.63}$$

can be written as a third order polynomial in η:

$$p(\eta) = c_0 + c_1\eta + c_2\eta^2 + c_3\eta^3 = 0 \tag{A.64}$$

with

$$\begin{aligned} c_0 &= \overline{uw}\,(\overline{uw}\,\overline{v^2} - \overline{vw}\,\overline{uv}) \\ &\quad + \overline{vw}\,(\overline{vw}\,\overline{x^2} - \overline{uw}\,\overline{uv}) - \mathrm{Var}\, z\ \mathrm{Cov}(x, y) && \text{(A.65)} \\ c_1 &= [\mathrm{Var}\, z + 4\mathrm{Cov}(x, y)]\,\overline{w} - \overline{uw}^2 - \overline{vw}^2 && \text{(A.66)} \\ c_2 &= -3\overline{w}^2 - \overline{w^2} && \text{(A.67)} \\ c_3 &= 4\overline{w} && \text{(A.68)} \\ \mathrm{Var}\, z &= \overline{w^2} - \overline{w}^2 && \text{(A.69)} \\ \mathrm{Cov}(x, y) &= \overline{u^2}\,\overline{v^2} - \overline{uv}^2 && \text{(A.70)} \end{aligned}$$

Observing the coefficients and the form of $p(\eta)$, we find that:

- $p(\eta = 0) = \det\left(-\mathbf{U}^T\mathbf{U}\right) < 0$ (the matrix results singular only in the case that all the experimental points lie exactly on a circle).
- $p''(\eta = 0) < 0$, and the only root of $p''(\eta) = 0$ is $\eta_2 = -2c_2/6c_3 > 0$. Hence, $p''(\eta) < 0$ in the interval between 0 and its lowest root.

The latter point implies that the function $p(\eta)$ is convex upwards. Therefore a **Newton** procedure supplied with the initial guess $\eta = 0$ is bound to converge to the smallest positive root of $p(\eta)$ [24].

Once the eigenvalue η' is computed, the circle parameters can be found simply solving the equation for the corresponding eigenvector $\mathbf{A}'_\mathbf{0}$:

$$\left(\mathbf{U}^T\mathbf{U} - \eta'\mathbf{I}\right)\mathbf{A}'_\mathbf{0} = 0 \tag{A.71}$$

CUDA code for the Taubin algorithm

```
//Modified CUDA utility for parallel array reduction
__device__ inline void reduce_kernel(float* g_idata, float* g_odata,
                                     float* sdata,
                                     unsigned int blockSize);

//Semi-parallel implementation of the Taubin ring fit algorithm
__device__ void Taubin(UTILITY* utility,    //Global memory: utilities
                       float* s_reduction, //Shared memory: array sums
                       Result* Result)      //Global memory: results
{
  float xav,yav;
  int length=Result->nHitsPerCandidate[blockIdx.x];

  //Distribute jobs to blocks and threads
  unsigned int eventIdx = blockIdx.x/4;
  unsigned int tripletIdx = blockIdx.x%4;
  unsigned int beginIdx = 4*MAXHITS*eventIdx + MAXHITS*tripletIdx;
  unsigned int hitIdx = 4*MAXHITS*eventIdx + MAXHITS*tripletIdx +
      threadIdx.x;

  //Execute function only on busy blocks
  if(eventIdx < Result->actualsize)
    {
      //Erase hit arrays
      utility->xHit[hitIdx] = 0;
      utility->yHit[hitIdx] = 0;

      //If current hit belongs to triplet
      //copy its coordinates and revert shift
      if (utility->x_hits_on_ring[hitIdx] != 0 && utility->y_hits_on_ring
          [hitIdx] != 0)
      {
          utility->xHit[hitIdx] = utility->x_hits_on_ring[hitIdx]
            - XSHIFT;
          utility->yHit[hitIdx] = SQRT3*utility->y_hits_on_ring[hitIdx];
      }
      __syncthreads();

      //Compute center of gravity
      reduce_kernel(&utility->xHit[beginIdx], &utility->xm[blockIdx.x],
```

```
                s_reduction, MAXHITS);
    reduce_kernel(&utility->yHit[beginIdx], &utility->ym[blockIdx.x],
                s_reduction, MAXHITS);
    xav = utility->xm[blockIdx.x]/length;
    yav = utility->ym[blockIdx.x]/length;

    //Erase utility arrays
    utility->uzav[4*eventIdx + tripletIdx] = 0.f;
    utility->z2av[4*eventIdx + tripletIdx] = 0.f;
    utility->u2av[4*eventIdx + tripletIdx] = 0.f;
    utility->v2av[4*eventIdx + tripletIdx] = 0.f;
    utility->uvav[4*eventIdx + tripletIdx] = 0.f;
    utility->uzav[4*eventIdx + tripletIdx] = 0.f;
    utility->vzav[4*eventIdx + tripletIdx] = 0.f;

    //If current hit belongs to triplet
    //fill utility arrays
    if(utility->xHit[hitIdx]!=0
       && utility->yHit[hitIdx]!=0)
      {
        utility->u[hitIdx] = utility->xHit[hitIdx] - xav;
        utility->v[hitIdx] = utility->yHit[hitIdx] - yav;
        utility->u2[hitIdx] = utility->u[hitIdx] * utility->u[hitIdx];
        utility->v2[hitIdx] = utility->v[hitIdx] * utility->v[hitIdx];
        utility->uv[hitIdx] = utility->u[hitIdx] * utility->v[hitIdx];
        utility->z[hitIdx] = utility->u2[hitIdx] + utility->v2[hitIdx];
        utility->z2[hitIdx] = utility->z[hitIdx] * utility->z[hitIdx];
        utility->uz[hitIdx] = utility->u[hitIdx] * utility->z[hitIdx];
        utility->vz[hitIdx] = utility->v[hitIdx] * utility->z[hitIdx];
      }

    //Compute sum of arrays
    reduce_kernel(&utility->u2[beginIdx], &utility->u2av[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->v2[beginIdx], &utility->v2av[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->uv[beginIdx], &utility->uvav[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->z[beginIdx], &utility->zav[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->z2[beginIdx], &utility->z2av[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->uz[beginIdx], &utility->uzav[blockIdx.x],
                s_reduction, MAXHITS);
    reduce_kernel(&utility->vz[beginIdx], &utility->vzav[blockIdx.x],
                s_reduction, MAXHITS);

    //The following code is executed once per block
    if(threadIdx.x == 0)
      {
        //Compute average
        float zav = utility->zav[blockIdx.x]/length;
        float z2av = utility->z2av[blockIdx.x]/length;
        float u2av = utility->u2av[blockIdx.x]/length;
        float v2av = utility->v2av[blockIdx.x]/length;
        float uvav = utility->uvav[blockIdx.x]/length;
        float uzav = utility->uzav[blockIdx.x]/length;
        float vzav = utility->vzav[blockIdx.x]/length;

        //Characteristic polynomial coefficients
        float CovXY = u2av*v2av - uvav* uvav;
        float VarZ = z2av-zav * zav;
```

```
            float c0 = uzav*(uzav*v2av - vzav*uvav)
              + vzav*(vzav*u2av - uzav*uvav) - VarZ*CovXY;
            float c1 = VarZ*zav  + 4*CovXY*zav - uzav*uzav - vzav*vzav;
            float c2 = -3*zav*zav - z2av;
            float c3 = 4*zav;
            float c22 = c2*2;
            float c33 = c3*3;

            //Find the roots of the characteristic polynomial
            //P(eta) = c0 + c1*eta + c2*eta^2 + c3*eta^3 = 0
            float eta, eta_new, poly, poly_new, derivative;
            eta = 0.f;
            poly = c0;
            //Newton's method
            for(int i=0; i<MAX_ITERATIONS; i++)
              {
                derivative = c1 + eta*(c22 + eta*c33);
                eta_new = eta - poly/derivative;
                if( eta_new==eta || isnan(eta_new) )
                   break;
                poly_new = c0 + eta_new*(c1 + eta_new*(c2 + eta_new*c3));
                if( abs(poly_new) >= abs(poly) )
                   break;
                eta = eta_new;
                poly = poly_new;
              }

            //Compute ring parameters
            float det = eta*eta - eta*zav + CovXY;
            float uc = (uzav*(v2av - eta) - vzav*uvav)/(2*det);
            float vc = (vzav*(u2av - eta) - uzav*uvav)/(2*det);
            float alpha = uc*uc + vc*vc + zav;

            //Converted output result
            Result->radius[blockIdx.x] = sqrtf(alpha)*((XMAX - XMIN)/NX);
            Result->xCenter[blockIdx.x] = XMIN
              + (uc + xav)*((XMAX - XMIN)/NX);
            Result->yCenter[blockIdx.x] = YMIN
              + ((vc + yav)/SQRT3)*((YMAX - YMIN)/NY);
          }
      }
}
```

C

Specifications of the NVIDIA Tesla K20 GPU

Device 0:	"Tesla K20c"
CUDA Driver Version / Runtime Version	5.0 / 5.0
CUDA Capability Major/Minor version number:	3.5
Total amount of global memory:	4800 MBytes (5032706048 bytes)
(13) Multiprocessors x (192) CUDA Cores/MP:	2496 CUDA Cores
GPU Clock rate:	706 MHz (0.71 GHz)
Memory Clock rate:	2600 Mhz
Memory Bus Width:	320-bit
L2 Cache Size:	1310720 bytes
Total amount of constant memory:	65536 bytes
Total amount of shared memory per block:	49152 bytes
Total number of registers available per block:	65536
Warp size:	32
Maximum number of threads per multiprocessor:	2048
Maximum number of threads per block:	1024
Maximum sizes of each dimension of a block:	1024 x 1024 x 64
Maximum sizes of each dimension of a grid:	2147483647 x 65535 x 65535
Maximum memory pitch:	2147483647 bytes
Texture alignment:	512 bytes
Concurrent copy and kernel execution:	Yes with 2 copy engine(s)
Run time limit on kernels:	No
Integrated GPU sharing Host Memory:	No
Support host page-locked memory mapping:	Yes
Alignment requirement for Surfaces:	Yes
Device has ECC support:	Enabled
Device supports Unified Addressing (UVA):	Yes
Device PCI Bus ID / PCI location ID:	132 / 0

Table C.1: Technical characteristics of the GPU on which this project was implemented and tested. Data obtained by means of the `deviceQuery.cu` script provided with the CUDA libraries (`http://docs.nvidia.com/cuda/cuda-samples/index.html`).

Bibliography

[1] R. AAIJ et al. Measurement of the $B_s^0 - \bar{B}_s^0$ oscillation frequency Δm_s in $B_s^0 \to D_s^-(3)\pi$ decays. *Phys. Lett. B* 709 (2012), pp. 177–184. DOI: 10.1016/j.physletb.2012.02.031.

[2] A. ABULENCIA et al. Observation of B0(s) - anti-B0(s) Oscillations. *Phys. Rev. Lett.* 97 (2006). DOI: 10.1103/PhysRevLett.97.242003.

[3] K. AHMET et al. The OPAL detector at LEP. *Nucl. Instrum. Meth. A* 305 (1991), pp. 275–319. DOI: 10.1016/0168-9002(91)90547-4.

[4] D. AMBROSE et al. New limit on muon and electron lepton number violation from $K_L^0 \to \mu^\pm e^\mp$ decay. *Phys. Rev. Lett.* 81 (1998), pp. 5734–5737. DOI: 10.1103/PhysRevLett.81.5734.

[5] B. ANGELUCCI et al. TEL62: an integrated trigger and data acquisition board. *Journal of Instrumentation* 7.02 (2012). DOI: 10.1088/1748-0221/7/02/C02046.

[6] B. ANGELUCCI. "Trigger simulation update". NA62 Physics WG meeting. Aug. 2013.

[7] V. V. ANISIMOVSKY et al. Improved measurement of the $K \to \pi\nu\bar{\nu}$ branching ratio. *Phys. Rev. Lett.* 93 (2004), p. 031801. DOI: 10.1103/PhysRevLett.93.031801.

[8] G. ANZIVINO et al. Construction and test of a RICH prototype for the NA62 experiment. *Nucl. Instrum. Meth. A* 593 (Aug. 2008), pp. 314–318. DOI: 10.1016/j.nima.2008.05.029.

[9] A. V. ARTAMONOV et al. New measurement of the $K^+ \to \pi^+\nu\bar{\nu}$ branching ratio. *Phys. Rev. Lett.* 101 (2008), p. 191802. DOI: 10.1103/PhysRevLett.101.191802.

[10] A. V. ARTAMONOV et al. Study of the decay $K^+ \to \pi^+\nu\bar{\nu}$ in the momentum region $140 < P_\pi < 199$ MeV/c. *Phys. Rev. D* 79 (9 May 2009). DOI: 10.1103/PhysRevD.79.092004.

[11] Y. ASANO et al. Search for a rare decay mode $K \to \pi\nu\bar{\nu}$ and axion. *Physics Letters B* 107.1-2 (Dec. 1981), pp. 159–162. DOI: 10.1016/0370-2693(81)91172-2.

[12] H. W. ATHERTON et al. *Precise measurements of particle production by 400 GeV/c protons on beryllium targets*. Geneva: CERN, 1980.

[13] G. ATOIAN et al. Lead scintillator electromagnetic calorimeter with wavelength shifting fiber readout. *Nucl. Instrum. Meth. A* 320 (1992), pp. 144–154. DOI: 10.1016/0168-9002(92)90773-W.

[14] J. BATLEY et al. New measurement of the $K^\pm \to \pi^\pm\mu^+\mu^-$ decay. *Phys. Lett. B* 697 (2011), pp. 107–115. DOI: 10.1016/j.physletb.2011.01.042.

[15] N. BERGER. Partial wave analysis at BES III harnessing the power of GPUs. American Institute of Physics Conference Series 1374 (Oct. 2011). Ed. by D. ARMSTRONG et al., pp. 553–556. DOI: 10.1063/1.3647201.

[16] J. BROD, M. GORBAHN and E. STAMOU. Two-loop electroweak corrections for the $K \to \pi\nu\bar{\nu}$ decays. *Phys. Rev. D* 83 (3 Feb. 2011). DOI: 10.1103/PhysRevD.83.034030.

[17] G. BUCHALLA and A. J. BURAS. QCD corrections to the anti-s d Z vertex for arbitrary top quark mass. *Nucl. Phys. B* 398 (1993), pp. 285–300. DOI: 10.1016/0550-3213(93)90110-B.

[18] G. BUCHALLA and A. J. BURAS. The rare decays $K \to \pi\nu\bar{\nu}$ and $K_L \to \mu^+\mu^-$ beyond leading logarithms. *Nucl. Phys. B* 412 (1994), pp. 106–142. DOI: 10.1016/0550-3213(94)90496-0.

[19] R. BULLOCK. *Least-Squares Circle Fit*. Developmental Testbed Center. 2006.

[20] N. CABIBBO. Unitary Symmetry and Leptonic Decays. *Phys. Rev. Lett.* 10 (12 1963), pp. 531–533. DOI: 10.1103/PhysRevLett.10.531.

[21] G. D. CABLE et al. Search for Rare K^+ Decays. II. $K^+ \to \pi^+\nu\bar{\nu}$. *Phys. Rev. D* 8 (11 Dec. 1973), pp. 3807–3812. DOI: 10.1103/PhysRevD.8.3807.

[22] U. CAMERINI et al. Experimental Search for Semileptonic Neutrino Neutral Currents. *Phys. Rev. Lett.* 23 (6 Aug. 1969), pp. 326–329. DOI: 10.1103/PhysRevLett.23.326.

[23] A. CECCUCCI. *NA62/P-326 Status Report*. Tech. rep. SPS Experiments Committee, CERN, Nov. 2007.

[24] N. CHERNOV. *Circular and linear regression: Fitting circles and lines by least squares*. Chapman & Hall/CRC, 2010. ISBN: 1439835906.

[25] G. COLLAZUOL et al. "Fast FPGA-based Trigger and Data Acquisition System for the CERN Experiment NA62: Architecture and Algorithms". In: *11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*. 2008, pp. 405–412. DOI: 10.1109/DSD.2008.99.

[26] G. COLLAZUOL et al. Fast online triggering in high-energy physics experiments using GPUs. *Nucl. Instrum. Meth. A* 662 (2012), pp. 49–54. DOI: 10.1016/j.nima.2011.09.057.

[27] H. S. M. COXETER and S. L. GREITZER. *Geometry Revisited*. The Mathematical Association of America, 1967. ISBN: 0883856190.

[28] FLAVIANET MARIE CURIE TRAINING NETWORK. *Working Group on Precise SM Tests in K Decays*. June 2010. URL: http://www.lnf.infn.it/wg/vus/.

[29] J. M. FLYNN, M. PAULINI and S. WILLOCQ. *WG2 Conveners' Report: Vtd and Vts, B-Bbar mixing, radiative penguin and rare (semi)leptonic decays*. Tech. rep. Nov. 2003. arXiv:hep-ph/0311018.

[30] S. GORBUNOV et al. ALICE HLT High Speed Tracking on GPU. *IEEE Transactions on Nuclear Science* 58.4 (2011), pp. 1845–1851. DOI: 10.1109/TNS.2011.2157702.

[31] G. HAEFELI et al. The LHCb DAQ interface board TELL1. *Nucl. Instrum. Meth. A* 560 (2006), pp. 494–502. DOI: 10.1016/j.nima.2005.12.212.

[32] F HAHN et al. *NA62: Technical Design Document*. Tech. rep. NA62-10-07. Geneva: CERN, Dec. 2010.

[33] M. J. HARRIS. "Optimizing Parallel Reduction in CUDA". NVIDIA Developer Technology. URL: http://developer.download.nvidia.com/assets/cuda/files/reduction.pdf.

[34] M. J. HARRIS. *Real-Time Cloud Simulation and Rendering*. PhD thesis. Department of Computer Science, University of North Carolina at Chapel Hill, 2003. URL: http://www.markmark.net/dissertation/.

[35] T. HURTH et al. Constraints on New Physics in MFV models: A Model-independent analysis of $\Delta F = 1$ processes. *Nucl. Phys. B* 808 (2009), pp. 326–346. DOI: 10.1016/j.nuclphysb.2008.09.040.

[36] G. ISIDORI, F. MESCIA and C. SMITH. Light-quark loops in $K \to \pi\nu\bar{\nu}$. *Nucl. Phys. B* 718 (2005), pp. 319–338. DOI: 10.1016/j.nuclphysb.2005.04.008.

[37] E. IWAI. Status and prospects of J-PARC KOTO experiment. *Nucl. Phys. Proc. Suppl.* 233 (2012), pp. 279–283. DOI: 10.1016/j.nuclphysbps.2012.12.090.

[38] D. E. JAFFE and S. YOUSSEF. Bayesian estimate of the effect of $B^0\bar{B}^0$ mixing measurements on the CKM matrix elements. *Comput. Phys. Commun.* 101 (1997), p. 206. DOI: 10.1016/S0010-4655(96)00171-3.

[39] M. KOBAYASHI and T. MASKAWA. CP Violation in the Renormalizable Theory of Weak Interaction. *Prog. Theor. Phys.* 49 (1973), pp. 652–657. DOI: 10.1143/PTP.49.652.

[40] J. LAIHO, E. LUNGHI and R. S. VAN DE WATER. Lattice QCD inputs to the CKM unitarity triangle analysis. *Phys. Rev. D* 81 (2010). DOI: 10.1103/PhysRevD.81.034503. URL: http://www.latticeaverages.org/.

[41] G. LAMANNA. "2 rings trackless fitting for L1/L0 RICH trigger using GPU". NA62 Trigger and Data Acquisition WG meeting. May 2011.

[42] G. LAMANNA. "GPU for realtime processing in HEP experiments". In: *EPS-HEP 2013, Stockholm*. URL: https://indico.cern.ch/contributionDisplay.py?contribId=187&confId=218030.

[43] G. LAMANNA. "GPUs for fast triggering in NA62 experiment". In: *Technology and Instrumentation in Particle Physics*. June 2011. URL: http://indico.cern.ch/contributionDisplay.py?contribId=108&confId=102998.

[44] I. MANNELLI. "NEWCHOD: Preliminary construction ideas". NA62 joint MUV/CHOD WG meeting. June 2013.

[45] H. MCKENDRICK. *Analysis Acceleration in TMVA for the ATLAS Experiment at CERN using GPU Computing*. PhD thesis. School of Informatics, University of Edinburgh, 2011.

[46] F. MESCIA and C. SMITH. Improved estimates of rare K decay matrix-elements from Kl3 decays. *Phys. Rev. D* 76 (2007). DOI: 10.1103/PhysRevD.76.034017.

[47] M. MISIAK and J. URBAN. QCD corrections to FCNC decays mediated by Z penguins and W boxes. *Phys. Lett. B* 451 (1999), pp. 161–169. DOI: 10.1016/S0370-2693(99)00150-1.

[48] NA48 COLLABORATION et al. The beam and detector for the NA48 neutral kaon CP violation experiment at CERN. *Nucl. Instrum. Meth. A* 574 (May 2007), pp. 433–471. DOI: 10.1016/j.nima.2007.01.178.

[49] NA62 COLLABORATION. *2013 NA62 Status Report to the CERN SPSC*. Tech. rep. SPS and PS Experiments Committee, CERN, Mar. 2013.

[50] K. NAKAMURA et al. Review of Particle Physics. *Journal of Physics G: Nuclear and Particle Physics* 37.7A (2010). URL: http://pdg.lbl.gov.

[51] F. O. NEWSON. "Kaon experiments at CERN: recent results and prospects". In: *PSI 2013*. 2013. URL: https://indico.psi.ch/contributionDisplay.py?contribId=14&confId=2036.

[52] M. NICULESCU and S.-I. ZGURA. Computing trends using graphic processor in high energy physics. *ArXiv e-prints* (June 2011). arXiv:1106.6217 [cs.DC].

[53] NVIDIA CORPORATION. *NVIDIA CUDA C Programming Guide*. July 2013. URL: http://docs.nvidia.com/cuda/index.html.

[54] F. PANTALEO et al. "Real-Time Use of GPUs in NA62 Experiment". In: *13th International Workshop on Cellular Nanoscale Networks and their Applications*. Geneva, Aug. 2012.

[55] F. PANTALEO et al. "Real-time triggering in HEP using GPUs". In: *GPU Technology Conference*. Mar. 2013. URL: https://registration.gputechconf.com/form/session-grid.

[56] V. PRATT. Direct least-squares fitting of algebraic surfaces. *SIGGRAPH Comput. Graph.* 21.4 (Aug. 1987), pp. 145–152. DOI: 10.1145/37402.37420.

[57] S. RENNICH. "CUDA C/C++ Streams and Concurrency". GPU Computing webinars, NVIDIA Corp. URL: https://developer.nvidia.com/gpu-computing-webinars.

[58] A. ROMANO. *Leptonic decays and Kaon identification at the NA62 experiment at CERN*. PhD thesis. School of Physics and Astronomy, University of Birmingham, Dec. 2012.

[59] A. SERGI. *Recent results from Kaon Physics*. Tech. rep. Mar. 2013. arXiv:1303.0629.

[60] A. SHER et al. An Improved upper limit on the decay $K^+ \to \pi^+\mu^+e^-$. *Phys. Rev. D* 72 (2005). DOI: 10.1103/PhysRevD.72.012005.

[61] K. SHUM. *Distance distribution in the hexagonal packing*. Dec. 2012. URL: http://home.ie.cuhk.edu.hk/~wkshum/wordpress/?p=1003.

[62] M. SOZZI. "A concept for the NA62 trigger and data acquisition". NA62 Internal Note. Nov. 2007.

[63] M. SOZZI et al. "NA62 online software and TDAQ interface". NA62 Internal Note. Feb. 2011.

[64] M. STEIN. "CUDA Basics". ManyCores Reading Group, New York University. URL: http://www.cs.nyu.edu/manycores/.

[65] S. TARIQ. "An Introduction to GPU Computing and CUDA Architecture". GPU Computing webinars, NVIDIA Corp. 2011. URL: https://developer.nvidia.com/gpu-computing-webinars.

[66] G. TAUBIN. Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 13.11 (1991), pp. 1115–1138. URL: http://dblp.uni-trier.de/db/journals/pami/pami13.html#Taubin91.

[67] G. UNAL. "Performances of the NA48 Liquid Krypton Calorimeter". In: *CALOR2000, Annecy*. 2000. URL: http://calor.pg.infn.it/calor2000/Contributions/Ionization/Guillaume_Unal.pdf.

[68] B. VELGHE. "GigaTracker, a Thin and Fast Silicon Pixels Tracker". In: *RD13 - 11th International Conference on Large Scale Applications and Radiation Hardness of Semiconductor Detectors*. July 2013. URL: https://agenda.infn.it/contributionDisplay.py?contribId=34&confId=6120.

[69] J. VERSCHELDE. "Concurrent Kernels & Multiple GPUs". Graduate seminar lecture notes. University of Illinois, Chicago. Apr. 2012. URL: http://homepages.math.uic.edu/~jan/mcs572/main.html.

[70] A. WASHBROOK. "Algorithm Acceleration from GPGPUs for the ATLAS Upgrade". In: *Conference on Computing in High Energy and Nuclear Physics*. Oct. 2010. URL: http://cds.cern.ch/record/1300750.

[71] L. WOLFENSTEIN. Parametrization of the Kobayashi-Maskawa Matrix. *Phys. Rev. Lett.* 51 (21 Nov. 1983), pp. 1945–1947. DOI: 10.1103/PhysRevLett.51.1945.

Acknowledgements

I would like to express my gratitude to my supervisor, who has always been available and attentive, and has patiently guided and encouraged me through the steps of my work with many enlightening advices and a maddening attention to spelling and punctuation.

I thank Gianluca Lamanna, whose «Would it be easy to try this?...» suggestions I learned to expect and to treasure (and they will probably never stop until the very day of my graduation!).

Thanks to Felice, for his valuable help and company during the long afternoons we spent writing and erasing code in front of the lab PCs.

Thanks to Roberto, Bruno and Jacopo for having always been willing to help me during these long months. My deepest gratitude goes to all of them.

I cannot keep myself from thanking all of my friends. I know that every one of them will bring to themselves and to the people around them the same enjoyable happiness that they have often brought to me.

Thanks to Martina, Alberto, Luigi, Daniele, Leslie, Massimo, Elena, Paola, Marco, Sara, Andrea, Giovanni, Paola, Francesco, and to my dear flatmates Cinzia, Eliana and Irene. I hope my fellow students did not mind that I haunted our department as a thesis-writing ghost all the time. A special mention to Francesco and Luca, my fellow coffee-seeking ghosts!

A heartfelt thought goes to Arianna: our paths split as we left Siena for new adventures, but I know my friend will always be there when I need her.

I take this opportunity to express my profound gratitude to my family, and especially to my parents, that have been a constant source of support during my university years. It is to them that this thesis is dedicated.

Last but not least, my deepest gratitude goes to Salvo. Without his loving support, it would have been much more difficult for me to go through these challenging and sometimes stressful times.