POLISH ACADEMY OF SCIENCES

INSTITUTE OF FUNDAMENTAL TECHNOLOGICAL RESEARCH

Doctoral Dissertation

Evolutionary Algorithm for Particle Trajectories Reconstruction

Oskar Wyszyński

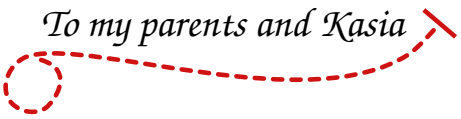
Dissertation supervisor:

prof. dr hab. Roman Płaneta

Auxiliary supervisor:

dr Wiesław Chmielnicki

Warsaw 2016


To my parents and Kasia

Acknowledgments

I would like to express my deepest gratitude to the supervisors prof. dr hab. Roman Płaneta, Wiesław Chmielnicki and András László for support, excellent guidance and allowing me to have the freedom to choose.

A debt of my great appreciation goes to Marek Gaździcki for leading me through complex aspects of working in an international collaboration.

Last but not least, I would like to thank Balázs Kégl for fruitful discussion, which had big impact on the final shape of this thesis.

This work was supported by the National Science Center of Poland (grant UMO-2015/18/M/ST2/00125).

Abstract

Significant progress has been made in the derivative-free optimization algorithms over past two decades, causing rise in popularity of those methods.

The most prominent area of their application is parameters estimation of complex models during machine learning process. Complex model may take forms of non-linear, non-smooth, non-convex functions but also discontinuities might be present. Therefore, group of traditional derivative based methods are failing in this respect.

One of the most promising stochastic methods is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES). This stochastic and derivative-free method, is suitable for optimization problems, where very little assumptions on the nature of the underlying objective function can be made. Fewer assumptions will allow to solve larger set of problems, particularly desirable when dealing with difficult objective functions. Another benefit is better separation between model and optimization algorithm, providing means for flexible, multi-model algorithms.

Within this work, feasibility of application and performance of the CMA-ES method for particle trajectory reconstruction were studied. The method was used in order to train a model for each particle trajectory by continuous optimization of its parameters. The result of the studies is a novel method of particle trajectory reconstruction, targeting small and medium experiments as well as detector prototype testing, where high reconstruction efficiency and short development time is of

greater importance than cutting edge execution time.

Additionally, the same optimization technique has been used for training a multiclass classifier, for the purpose of nuclei identification produced in heavy ion reactions at intermediate energies. Resulting algorithm is the first fully automated software tool, which can be used for identification in other nuclear physics experiments as well.

In summary, the objective of this work was development of two algorithms based on stochastic optimization methods. The first one has been used to find and reconstruct particle trajectories within gaseous detectors. Whereas the second algorithm has been applied to identify nuclear reaction products, registered by telescope detectors.

Streszczenie

W ostatnich dwóch dekadach dokonał się ogromny postęp w dziedzinie technik optymalizacji nieopartych na gradientach, co spowodowało wzrost zainteresowania tymi technikami.

Najbardziej obiecującym obszarem zastosowań jest optymalizacja parametrów złożonych modeli w dziedzinie uczenia maszynowego. Jedną z najbardziej obiecujących metod jest stochastyczna strategia ewolucji wykorzystująca adaptację macierzy kowariancji (CMA-ES). Metoda ta nie wymaga gradientów i sprawdza się w sytuacjach, gdzie nie można poczynić daleko idących założeń dotyczących natury badanej funkcji. Mniejsza liczba założeń pozwala na zastosowanie metody do rozwiązania szerszego zestawu problemów, szczególnie tych zdefiniowanych przy użyciu nieróżniczkowalnych, niewypukłych oraz nieciągłych funkcji. Kolejną zaletą jest lepsza separacja między modelem a algorytmem optymalizacji, umożliwiająca tworzenie elastycznych algorytmów wykorzystujących wiele modeli.

W ramach tej pracy zbadano możliwość zastosowania metody CMA-ES do rekonstrukcji trajektorii cząstek elementarnych, poruszających się w polu magnetycznym. Metodę zastosowano w celu wyszukania parametrów modelu trajektorii każdej cząstki, poprzez ciągłą optymalizację jego parametrów. Wynikiem badań jest nowatorska metoda rekonstrukcji trajektorii cząstek, ukierunkowana na potrzeby małych oraz średnich eksperymentów, ale także do wykorzystania podczas testów prototypowych detektorów, gdzie wysoka efektywność rekonstrukcji i krótki czas

rozwoju oprogramowania ma większe znaczenie niż czas wykonywania algorytmu.

Dodatkowo metodę CMA-ES zastosowano do wytrenowania modelu klasyfikatora wieloklasowego, służącego do identyfikacji jąder powstałych w reakcjach ciężkich jonów przy pośrednich energiach i zarejestrowanych w wielodetektorowym systemie. Przedstawiony algorytm jest pierwszym w pełni zautomatyzowanym narzędziem, który można wykorzystać do identyfikacji powstałych w wyniku reakcji fragmentów w wielu eksperymentach fizyki jądrowej.

Podsumowując, celem tej pracy było opracowanie dwóch algorytmów opartych o metodę stochastycznej optymalizacji – CMA-ES. Pierwszy z nich został użyty do znalezienia oraz rekonstrukcji trajektorii cząstek elementarnych zarejestrowanych w detektorze gazowym. Podczas gdy drugi algorytm zastosowano do identyfikacji produktów reakcji jądrowych zarejestrowanych w detektorach teleskopowych.

Contents

List of Figures

Chapter 1

Introduction

Significant progress has been made in the derivative-free optimization algorithms over past two decades, causing rise in popularity of those methods among machine learning community [1, 2, 3]. In particular, stochastic methods of arts performance [4, 5, 6].

The most prominent area of their application is parameters estimation of complex models during machine learning process. Complex model may take forms of non-linear, non-smooth, non-convex functions but also, discontinuities might be present. Therefore, group of traditional derivative based methods are failing in this respect.

One of the most promising stochastic methods is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [7, 8, 9, 10, 11]. It is a stochastic, derivative-free method for finding global optimum with very little assumptions on the nature of the underlying objective function. Fewer assumptions will allow to solve larger set of problems, particularly desirable when dealing with difficult objective functions. Another benefit is better separation between model and optimization algorithm, providing means for multi-model type algorithms, making it desirable for difficult optimization tasks in machine learning application. Such separation will not only

simplifies software development process by offering greater flexibility, but also this approach tends to be less error prone.

On the other hand, according to the No Free Lunch (NFL) theorem [12], there is no free lunch and therefore mentioned above benefits come at a price. That means lower performance on solving problems where *a priori* knowledge can be used. However, growing computing power and development of parallel processing technologies such as GPU[13] or FPGA[14] indicate that even though stochastic optimization methods are still too computationally expensive for many problems, this may change drastically in the coming years. Nevertheless, those methods are already pretty effective in solving problems, where execution time is not of the first importance [1, 2, 3].

Unfortunately, stochastic optimization techniques are neglected in the number of scientific fields such as **particle** and **nuclear physics**. Therefore, within this work, research on feasibility of the CMA-ES application in those two fields was conducted. The biggest emphasis was put on **particle physics** where, during my participation in Doctoral Student Program at CERN, the author of this dissertation contributed in the whole experimental infrastructure of the NA61/SHINE experiment (descirbed in chapter 2). Whereas my contribution in the field of **nuclear physics** is limited to development of automated algorithm for nuclear fragments identification, using mentioned optimization technique. The algorithm is a consequence of participation in the International Workshop on Nuclear Dynamics and Thermodynamics which took place at Texas A&M University in College Station.

In the following paragraphs, brief introduction to particle physics and nuclear physics regarding this topic is given.

The field of **Particle Physics**, also called High Energy Physics (HEP), studies properties of the matter and aims to answer the most fundamental question such

as: where the matter comes from, what the mass is, what matter consists of and many more questions. In order to answer them, various types of experiments have to be conducted. Basic instruments of studies are particle accelerators such as linear accelerators, cyclotrons or synchrotrons.

Those accelerators produce particle stream called beam, which typically is collided with a fixed target or an another beam, in order to initiate a reaction. Then, the products of the reaction are registered by a set of detectors and delivered in form of electronic signals for further processing and analyzing. In the following manner, many discoveries have been made, including recent discovery of Higgs boson, which has been awarded with the Nobel prize. The most probably, more secrets held by the Universe will be uncovered using mentioned accelerators.

However, to study reactions of collisions one has to collect, calibrate and reconstruct the data in the first place. Each step is complex and essential for the success of an experiment. In order to collect the data from previously designed and built detectors, a readout system has to be built and a proper software has to be developed.

In general, every readout system should present a good compromise between overall build cost and performance. On one hand, plenty of data to collect implies higher cost of the system, on the other hand, too little data will not provide sufficient statistics required for an analysis. In order to maximize production of expected data, a trigger system should be used. This system aims at selecting the most desirable data (so-called events [1]) thus the amount of data needed to achieve a particular statistics is reduced significantly. Furthermore, a real-time compression algorithm typically is implemented in the readout system, providing an additional data reduction, which in turn improves overall performance. Both, readout and

[1] An event refers to the results of a single fundamental interaction, which took place between subatomic particles.

trigger system in the NA61/SHINE experiment are subjects of this thesis.

The next step is to calibrate and reconstruct recorded data using custom software. In order to facilitate this task, typically a common software framework is developed. Such framework provides common means of data and detector calibration manipulation tools, which helps to accelerate data analysis process and significantly reduces probability of human error.

The calibration process is not a part of this work, however a strong emphasizing will be given on the most crucial part of reconstruction process – particle trajectory reconstruction.

Particle trajectory reconstruction is a complex task, which requires great amount of development effort where time span is counted in years. Typical reconstruction algorithm consists of three parts:

- trajectory model – describes trajectory and free parameters of a particle. For each trajectory, there should be a unique set of parameters that fits the trajectory.

- assignment rules – govern which measurement data is used for trajectory model training.

- optimization method – used for training the model while adding new measurements.

Although in theory this division is often possible, it is abandoned in the chase for greater execution speed. Convolution of those parts is made in order to exploit *a priori* knowledge about the model during optimization process, such as gradients, parameter ranges or by using anticipated values as initial values for parameters. Such algorithms come with the high development and maintenance cost, especially when few interchangeable models are preferred e.g. due to change of experimental conditions such as magnetic field, or simply due to will of study a new model.

Although, convoluting together those elements is justified when the exceptional performance is required, it is a sub-optimal solution for mediocre performance tasks.

Reconstruction software used by small and medium HEP experiments, mostly adapt software prepared by big experiments in hope to reduce development workload. Unfortunately, because of mentioned convolutions, in the end, required effort approaches the one required for development of a new tailored solution. Therefore, in cases where top performance is not required, development of stochastic, derivative-free based methods are much simpler.

Surprisingly, software from big experiments were also adapted for prototype detector tests e.g. in 2016 the author of this dissertation participated in several tests and small experiments[15, 16], where my major task was adaptation of the ATLAS[17] software. The software consists of around 5 millions lines of code, and it is used for parallel data processing, running on over 4000 PC class machines and digesting data from a 46 m long, 25 m high and 25 m wide detector system. However, using it to process data from a detector prototypes of a suitcase size is just an unjustifiable action.

In the field of **Nuclear Physics**, a great effort is devoted to the investigation of heavy ion collisions at intermediate energies. The progress made in detector physics results in large acceptance multi detector arrays, which allows for more precise measurements of emitted nuclei fragments from a reaction and consequently for more advanced analysis techniques. However, similarly to experiments in HEP, the data must be pre-processed before those advanced techniques can be used. Noteworthy, the pre-processing process consequently become more sophisticated as well.

Thankfully, increasing computing power makes implementing complex simulation, calibration and nuclei identification methods not only possible but also affordable. In particular, improving identification methods is specially interesting as all of present methods require, to a greater or lesser extent, human intervention in the process. In other words, identification is partially done by operator of a tool. This makes identification process time consuming, tiresome and prone to human error.

Therefore, in this dissertation, feasibility of application of CMA-ES method for automated nuclear fragments identification process, was studied as well.

Within this work, feasibility of application and performance of CMA-ES method for particle trajectory reconstruction were studied. The method was used in order to train a model for each particle trajectory by continuous optimization of its parameters. Those parameters, resulting from model training, are of great importance for the physics analysis as they represents basic particle properties such as particle momentum and charge. Those properties, among others, are required in the process of particle identification.

The result of the studies is a novel method of particle trajectory reconstruction, tailored for small and medium experiments including detector prototype testing, where high reconstruction efficiency and short development time is of greater importance than cutting edge execution time.

Additionally, the same optimization technique has been used for training a multiclass classifier, for the purpose of nuclei identification produced in heavy ion reactions at intermediate energies. Resulting algorithm is the first fully automated software tool, which can be used for identification in other nuclear physics experiments as well.

1.1 State of the Art

In the following sub-sections, the state of the art of particle trajectory reconstruction techniques in **Particle Physics** as well as the state of the art of atomic nucleus identification in **Nuclear Physics** is outlined.

1.1.1 Trajectory Reconstruction

The choice of the algorithm depends on the detector type. Some types of detector give only two-dimensional information, another ones give space points. Additionally, complex shape of the detector and presence of the magnetic field makes the mathematical model of a track even more complicated. Such model may significantly narrows down the number of suitable methods. Therefore, designing a detector should go along with development of reconstruction algorithm. Failing to do so, may lead to difficulties in designing of algorithm with satisfactory performance. In extreme cases, it might not be possible to design such algorithm at all due to imperfections of the detector such as noise, distortions or uneven sensitivity.

The first commonly used techniques were based on Hough transformation for track finding [18, 19]. The Hough transformation is a feature extraction technique invented by Hough in 1959 for analysis of bubble chamber photographs. The method proved to be effective and fast algorithm and therefore it superseded human labor in trajectory reconstruction task. Although it was revolutionary step in this field, Hough transformation method showed poor efficiency when dealing with high density of trajectories offered by newer generation of tracking detectors.

In this method, the data points from different trajectories will vote for the same accumulation cell in the parameter space if the number of cells are too low. If using higher number of accumulating cells, the votes will spread among many cells making the tracks indistinguishable from other tracks and from noise. Furthermore,

when having large number of parameters (more than three) or trajectories have relatively low number of data points compared to the overall data, the cells will not accumulate enough votes. The flaws of employed methods became more visible during the quest for scientific discoveries which lead to higher energies, implying higher multiplicity of particles.

In 1960 a new powerful tool that addresses the problem, called the Kalman filter, has been presented [20, 21]. The filter, also known as linear quadratic estimation, uses series of measurements containing noise over the time and produces estimates of a future state. Named after Rudolf E. Kálmán, the filter became the standard solution for particle trajectory reconstruction. The great success of the Kalman filter is due to its small computational requirement (so important in the 1980s), being an optimal minimum variance estimator and its recursive nature [22]. One of the most famous application of the Kalman Filter was in the Apollo navigation computer that took safely Neil Armstrong and Buzz Aldrin to the Moon and back [23]. However, the Kalman Filter (KF) is suitable only for linear and close to linear problems. Therefore, the effort to extend its capabilities to non-linear problems has been taken.

The first extensions and generalizations of the method called Extended Kalman Filter (EKF) have been developed few years later [24]. The EKF approximates the state-transition and observation matrices with first-order linearization by expanding them in Taylor series at current state and use that approximation to predict a next state. Therefore the model may not be linear but it must be locally linear and it must be differentiable. When a function is highly non-linear, this approach often introduces large errors in estimates and consequently leads to sub-optimal results or even may lead to divergence. Furthermore, mostly due to approximating, the EKF tends to be difficult to implement, difficult to tune and only reliable for systems that are almost linear [25].

In 1997, Julier and Uhlman [26] proposed an improvement over the EKF called Unscented Kalman Filter (UKF). Instead of approximating an arbitrary nonlinear function, it is easier to propagate a probability distribution using sample through a non-linear function than a covariance matrix[25]. This modification makes UKF perform better for slightly nonlinear models when compared to an EKF in terms of accuracy and robustness, whereas the computational cost is of the same order as of EKF[27]. Nevertheless, most flaws of the original method are present, namely the model is convoluted with the optimization step and on top of that, the uncertainties must be estimated quite accurately.

In late 90's, the KF was introduced to event reconstruction in HEP by Billoir [28, 29, 30] and Fruhwirth [31], becoming the most popular algorithm for particle trajectory reconstruction and being used up to now. Regardless of its draw backs, one of KF flavors lies under most of track reconstruction algorithms used in modern HEP experiments, including those located at the Large Hadron Collider (LHC) [32] at European Organization for Nuclear Research at CERN[33] in Geneva, Switzerland. All located there LHC experiments (ATLAS, CMS, LHCb, ALICE) have based their algorithms on the Kalman filter algorithm [34, 35, 36, 37, 38, 39, 40, 41], although implementation is known to be not a trivial task, using required low execution time to justify the choice.

Despite of the success of Kalman filter, applying it to a nonlinear system can be difficult. It is especially difficult to create a state transition, control and observation matrices, when dealing with multidimensional data and complicated differential equations. Therefore, there is an effort to develop a new tracking algorithm which is much simpler to implement, and at the same time allowing models to evolve.

All main Large Hadron Collider (LHC) experiments are searching for a more flexible algorithm[42, 43]. For instance, many techniques were tested such as the Cellular Automaton for the LHCb Outer Tracker [44], a recursive neural network

known as Hopfield Network in the LHCb Muon System [45] or the Denby-Peterson network for the Inner Tracking System in the ALICE experiment [46, 47, 48]. However, in the end, those new methods are still mostly accompanied by the Kalman Filter[44, 46, 47, 48] e.g. for extending tracks or filtering outliers.

Therefore, this work is dedicated for development of flexible algorithm for particle trajectory reconstruction by means of derivative-free stochastic optimization method called CMA-ES combined with probabilistic machine learning techniques. Those techniques were chosen in order to build the models required for finding trajectories and for measurements classification. Because those models cannot be defined as closed-form expressions, they are empirical distributions built using simulated data.

1.1.2 Nuclei Identification

The products of a nuclear reaction are often detected in so-called telescopes – stacks of different thickness detectors arranged in various configurations depending on the type of the research. Such detection systems have been developed in many research institutes [49, 50, 51, 52, 53, 54, 55, 56]. The number of telescopes in such arrays vary from few hundreds [54] to over a thousand [53]. The measurements of fragment energy loss in each layer of the telescope are the key point in the identification process. The correlation of energy losses between two chosen layers, reveals specific pattern of curved lines, which obey a function proposed by L.Tassan-Got [57].

The general performance of the detectors depends obviously on their quality and the associated electronics but also on the homogeneity of their response and their stability over long periods of time (e.g. temperature). Thus, each telescope can give a different ΔE-E matrix, which can change also during data-taking. Therefore, the identification procedure has to be executed several times for each telescope

before physics analysis can start.

Nowadays there are several methods in use, however all existing methods require user-intervention. The method presented in [58, 57, 59], beside being very sensitive to the size of data set, requires setting up quite precisely initial values and boundaries of model parameters for each detector separately. For methods [60, 61, 56, 62] one has to know with good precision each detector properties. To obtain those properties, additional measurements are required. The most automated methods available today still require from a user to setup the initial values for parameters and histogram binning must be chosen carefully [54, 63]. However, it is noteworthy that the method presented in [63] can be used not only for ΔE-E correlations.

The common draw back of all mentioned methods is that they require a dedicated graphical interface for human interaction. Therefore, processing data from few hundred telescopes is tiresome work.

The method presented in this dissertation, does not require any user interaction, precise initial values for parameters, defining boundaries or, consequently graphical interface.

1.2 Aims of the Dissertation

Within this work two aims can be distinguished. The first aim is to find and reconstruct particle trajectories registered by gaseous detectors with high accuracy. The second aim is to accurately identify products of nuclear reactions, using information delivered by telescope detectors.

The first aim of this thesis can be formulated as follows:

Stochastic black-box optimization techniques allow local[2] particle trajectory reconstruction in the time projection chambers with high accuracy reaching over 90%[3].

The objective is research and development of a particle trajectory reconstruction algorithm, which should fulfill the following requirements:

- to separate trajectory model and optimization algorithm – such design provides great flexibility
- to allow for local tracking in high trajectory density events – in such events, distance between tracks exceeds maximal resolution of the detector.
- to reach over 90% of reconstruction efficiency – because of high particle beam production cost, this is the lowest acceptable value.

The execution time is of secondary importance. In order to obtain an implementation design where all requirements are satisfied, a black-box optimization method will be combined with machine learning techniques.

The second aim of this thesis can be formulated as follows:

Stochastic black box optimization techniques allow full automation of nuclear fragments identification, produced in heavy ion collisions at intermediate energies and registered by telescope detectors.

[2]Local trajectory reconstruction means reconstruction within a signle detector such as Vertex TPC 1 (see fig. 2.3)

[3]This threshold is calculated based on maximal allowed measurement error.

The objective is research and development of nuclei identification algorithm, which should fulfill the following requirements:

- to separate energy-loss model and optimization algorithm – such design provides great flexibility
- to accept models defined as algebraic functions.
- to reach over 95%[4] of identification efficiency – lower efficiency would jeopardize quality of physics results

Similarly, the execution time is of secondary importance.

1.3 Research Methodology

In the following two sub-sections, methodology is presented for two research studies: feasibility of application of CMA-ES for particle trajectory reconstruction in the field of **particle physics**, and feasibility of application of CMA-ES for nuclei identification in the field of **nuclear physics**. The reason for choosing CMA-ES as optimizing method is described in the first two subsections of this chapter.

1.3.1 Trajectory Reconstruction

Particle trajectory reconstruction is a complex problem to solve, requiring multistage acquisition and preparation of data. The work presented in this dissertation tackles all aspects required to be done, before trajectories can be reconstructed. Therefore, the methodology of the research study presents as follows:

1. Development of a data acquisition system – the system was required in order to digitize, serialize and store all data coming from detector apparatus.

[4]This threshold is calculated based on maximal allowed measurement error.

2. Development of data selection system – also called trigger system, was required to overcome limitation of processing power, therefore a selection of events must be performed.

3. Development of data processing software framework – a framework facilities development of new methods by providing the common data manipulation tools and algorithms.

4. Simulation of training data using Monte Carlo methods – also called simulated training events, were required to train trajectory seed model.

5. Simulation of test data using Monte Carlo methods – also called simulated test events, were used to assess performance of new trajectory reconstruction algorithm

6. Obtaining labeled real detector noise data – in order to train detector noise model

7. Superimposing a real detector noise sample on simulated test events – in order to make the test more realistic, random real detector noise event were added to each simulated test event.

8. Training a trajectory seed model using simulated training events – this model was required to foreseen evolution of each trajectory seed.

9. Training detector noise model using labeled samples of real events representing detector noise – this step was crucial for building a noise model, which will compete for measurement points with every trajectory model.

10. Application of the CMA-ES based particle trajectory reconstruction algorithm – the algorithm will process test data in order to assess its efficiency.

11. Efficiency assessment of trajectory reconstruction – output of processed simulated test events were compared with particle list used as input for the data simulation.

Comparison mentioned in item 11, was performed using well established matching method within NA61/SHINE collaboration. The method is standard procedure called matching of reconstructed to simulated tracks, based on position of clusters and with a trajectory overlap threshold being set to 80 %. Within this work, presented steps were executed and the results of test using simulated data are presented.

1.3.2 Nuclei Identification

Problem of nuclei identification is less complicated than the one of particle trajectory reconstruction. Input data used for nuclei identification is only 2-dimensional compared to 3-dimensional spacial positions used to describe trajectory. However, this is still challenging problem, which is still not solved in fully automated manner. The research study steps of feasibility of CMA-ES application are as follows:

1. Building the model using energy loss correlation function.
2. Generating labeled test data using the model.
3. Superimposing a Gaussian noise.
4. Fitting model to the test data using CMA-ES.
5. Efficiency assessment using labels.

The details along with the results are presented in chapter 10.

1.4 Outline of the Thesis And Contribution

The chapters chapters 2 to 9 are related to the application of CMA-ES in the field of the **particle physics** whereas chapter 10 presents the application of the same method in the field of the **nuclear physics**.

At the beginning, in chapter 2, a general experiment description is provided, where only the elements necessary to understand aspects treated in this thesis are described. In particular, the experimental facility and working principal of the gas detectors are outlined. Afterward, in chapter 3, one can find an overview of modular electronics commonly used in HEP community. Most of presented technologies have been used to build the readout system of the NA61/SHINE experiment.

Further chapter is not essential to understand the main goals of this thesis, however it gives more in-deph knowledge, how such systems such as Trigger and DAQ are constructed. In chapter 4 the DAQ system is presented in details. This component of the NA61 facility is responsible for pre-processing and storage of the data recorded by experimental detectors. However, due to enormous amount of data, a data filtration system had to be developed. The system, called Trigger system, is described in chapter 5. The author of this dissertation contributed to both of these systems [64, 65].

Because of the complexity of the NA61/SHINE experimental facility, a data processing framework had to be developed. Such framework include all tools needed for data calibration and reconstruction such as data input/output interface, calibration data and detector geometry handling of various coordinate systems. Furthermore, a framework provides a common ground for exchange of algorithmic ideas. My contribution was to adapt the core of the framework to the NA61/SHINE experiment requirements as well as reconstruction and simulation modules. Part of those modules encapsulated modified versions of previously used legacy FORTRAN and C based algorithms, which were inherited from predecessor experiment, called

the NA49 experiment[66]. Another modules, provided an interface between legacy algorithms and the modern C++ based framework [67, 68]. Those algorithms were ported to the new framework in order to maintain backward compatibility with the data obtained during The NA49 experiment. Description of the Shine Offline framework is laid out in chapter 6.

The main part of reconstruction process is to find and extract physics properties of particle trajectories which traversed detector apparatus. The new trajectory reconstruction algorithm is the main subject of this thesis and it utilizes stochastic, derivative-free optimization method called CMA-ES. The optimization method is briefly outlined in chapter 7.

The particle reconstruction algorithm itself, is described in chapter 8. Whereas the results of its performance are presented in chapter 9.

Furthermore, another CMA-ES based algorithm has been presented in chapter 10. This algorithm is a fully automated solution to a problem of nuclei identification in the field of nuclear physics. The final summary and thesis conclusions about two algorithms are discussed in chapter 11

Chapter 2

NA61/SHINE Experiment

The NA61/SHINE (SPS Heavy Ion and Neutrino Experiment) is a large acceptance hadron spectrometer[69] designed for comprehensive studies of hadron production in hadron-proton, hadron-nucleus and nucleus-nucleus collisions at the European Organization for Nuclear Research (CERN). The SPS Heavy Ion and Neutrino Experiment (SHINE) [70] is one of the Super Proton Synchrotron (SPS) [71, 72] experiments, located in the North Area of CERN. The SPS is used as the LHC injector as well as accelerator for protons and ions for fixed target experiments like NA61/SHINE. The SPS accelerator is 6.9 km long (circumference) and accelerates protons from 20 GeV/c to 450 GeV/c. The transverse beam size decreases with the square root of the beam energy during

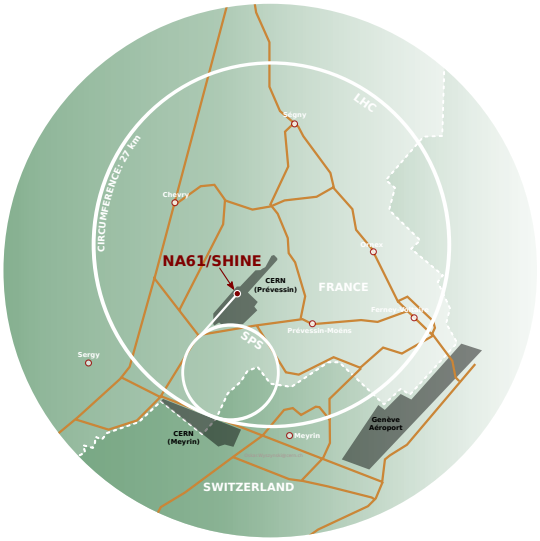


Figure 2.1: Location of the NA61/SHINE experiment.

acceleration and is typically of the order of few millimeters for the highest beam momentum (450 GeV/c).

Numerous components of the NA61/SHINE setup were inherited from its predecessors, in particular from the last one, the NA49 experiment[66]. The experiment reuses the main tracking system (fig. 2.2) including important upgrades as well as simulation and reconstruction software with corresponding modifications.

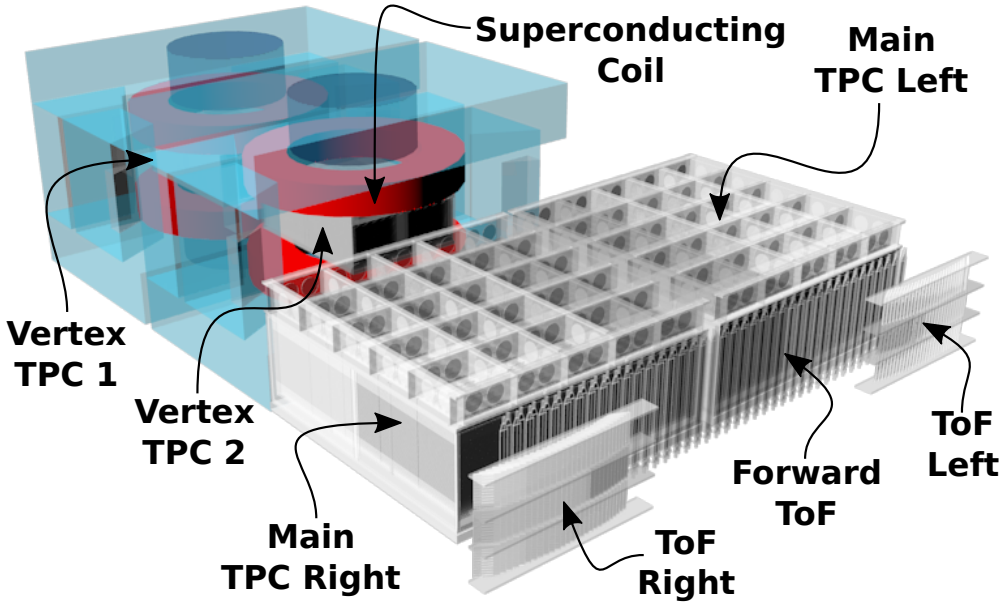


Figure 2.2: Rendering of the NA61/SHINE hadron spectrometer.

2.1 Detector Overview

The NA61/SHINE facility [69], fig. 2.2 and fig. 2.3, consists of five Time Projection Chamber (TPC) [66], namely Main TPCs, Vertex TPCs immersed in homogeneous magnetic field and Gap TPC. Each Main TPC and Vertex TPC detector is constructed using 25 or 6 TPC sectors respectively. In order to extend identification abilities to low momentum particles, Time Projection Chambers

are supplemented by Time of Flight (ToF) detector walls and Low Momentum Particle Detector (LMPD) [73]. Recently installed calorimeter, Projectile Spectator Detector (PSD), is used to select events representing central collisions also at the trigger level.

The beam position detectors located upstream of the target, together with scintillation and Cherenkov counters, provide precise position measurement and timing references for particles in the beam. The Trigger system uses a set of counters along with PSD calorimeter in order to determine time of appearance of the events of interest. Afterwards it sends the Main Trigger signal to data acquisition (more precisely, to Busy Logic), where decision is made whether electronics is ready to record a new event or not. Details about these processes will be described in the following chapters.

2.2 The Time Projection Chambers

A TPC, shown in fig. 2.4, is a particle detector placed inside a volume of noble gas located in a quasi-homogeneous electric field. A particle passing within the TPC sensitive volume, ionizes the gas.

The resulting primary ionization electrons drift upward the amplification-and-readout plane due to the electric field, as shown in fig. 2.4. The amplification-and-readout plane contains an array of readout pads, wired to an amplifier and charge digitizer. All together provide the digital data about the ionization trace/track such as coordinates X, Y, Z and electric charge of freed and amplified electrons.

The electric charge generated by ionization is specific to the particle velocity within the detection gas. Furthermore, trajectory curvature, caused by the bending force of magnetic field (Lorentz force), indicates momentum of a particle. Electric charge together with the momentum information can be used for particle identifi-

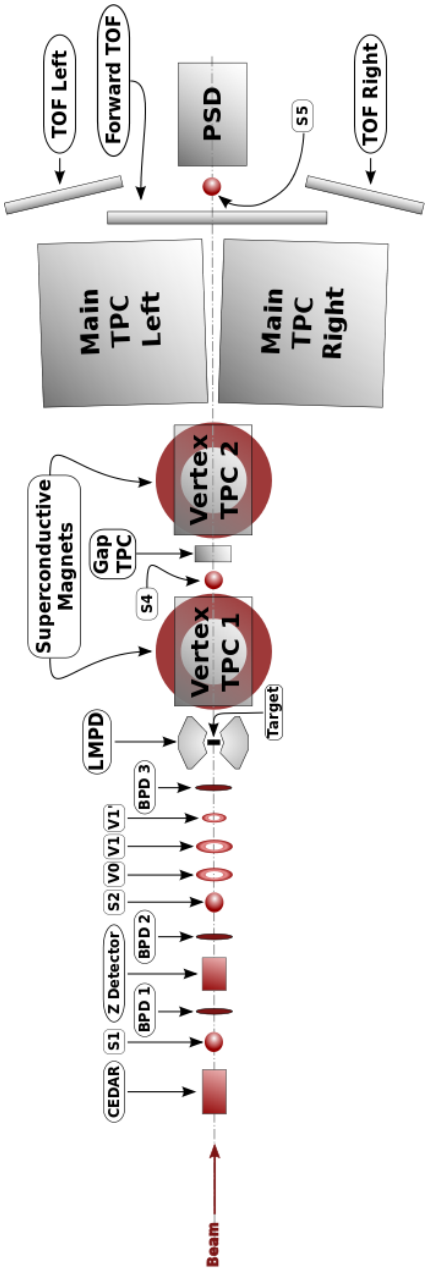


Figure 2.3: The NA61/SHINE detector overview. All detectors before taget (left of target) are called beam counters. They provide timing (SX, VX), identification(CEDAR, Z detector) and position (BPDX) information necessary to identify a moment when DAQ system should start recording. Behind target, is a region of high magnetic field, bending particle trajectories, which are registered by set of TPC, ToF and PSD detectors.

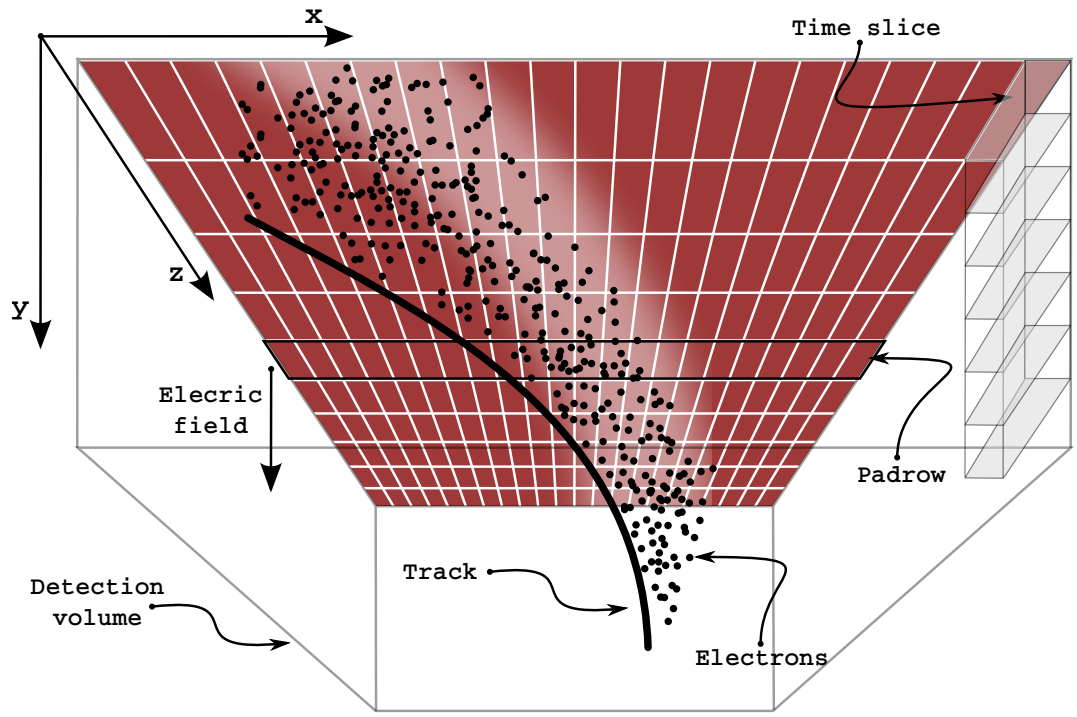


Figure 2.4: TPC principle of work: a particle traveling through detector ionizes gas. Freed electrons drift upwards due to electric field. The charge transported by electrons is multiplied near pads due to electron avalanche process (not shown). The final charge is collected by pads, which are wired to Front-End Electronics (FEE).

cation. Example [1] of digitized signals of electric charge is shown on fig. 4.3 and fully reconstructed event is presented on fig. 2.5 .

A number of detector effects can slightly influence the shape of measured charge deposited by a cluster. For instance, particles traversing the bottom part of the detector and thus having the largest drift path, produce clusters with larger radius

[1] Depending on the reaction, different number of particles can be produced. Therefore, in one event we can observe from several (low multiplicity) to hundreds (high multiplicity) of trajectories in the chambers.

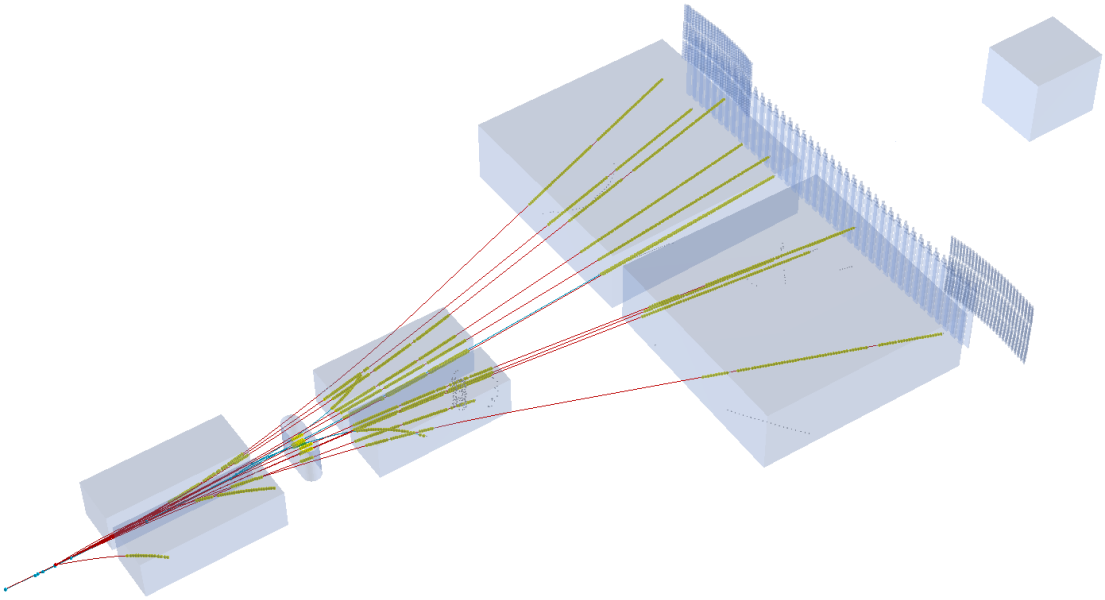


Figure 2.5: An example of a reconstructed event. Visible chambers are: VTPC1 (leftmost), GTPC, VTPC2. Two big MTPCs, ToF walls and PSD (right most). Detailed picture of the NA61/SHINE apparatus is shown on fig. 2.3.

due to charge cloud diffusion in the working gas. This implies a deterministic distortion effect on the signal.

On the other hand, fluctuations of the ionization process along with the electronic pickup noise on the readout pads, superimpose a statistical uncertainty on the measured charge depositions. The ensemble of all such detector effects complicates the task of track pattern recognition in the system.

2.3 Data processing

In the high energy physics, gaseous detectors since the beginning were used for identification of particles produced in nuclear interactions. Initially, particle trajectories were visible with the naked eye and events were recorded using photographic

film. Further, the photographs were analyzed by a team of specialists in order to determine particle trajectories. Various features of those particles were calculated, allowing to identify each of them.

Nowadays, designing and building of an experiment in the high energy physics is a multi stage and time consuming process. With an event rate of the order of 100Hz, photographic films are not sufficient anymore. This amount of data is impossible to handle without appropriate, often custom made electronics. Therefore, it is required to utilize advanced data acquisition systems, trigger systems and complex software infrastructure including advance algorithms for an event reconstruction.

This work addresses designing and building a data acquisition system and trigger system as well as reconstruction software used to process data collected by mentioned systems. Considering the complexity of detector facility, a common software framework called Shine had been created before works on reconstruction software could began.

The final stage of this work was design, implementation and performance study of the particle trajectories reconstruction algorithm. The algorithm had to be elastic in terms of adaptation to new conditions, it shall work in quasi-homogeneous magnetic field and the configuration shall not require user's in-depth knowledge about algorithm implementation. In addition, the particle trajectories should be reconstructed with high accuracy for high multiplicity events such as lead-lead collisions.

So far, the data acquisition and trigger systems have collected considerable amount of precious data. Analysis of the data with Shine software framework and reconstruction algorithms led to many new physics discoveries, which have been published in prestigious journals.

Chapter 3

Modular Electronics

This chapter is intended to provide more in-depth knowledge about modular electronics commonly used in particle as well as nuclear physics. Comprehension of this dissertation does not depend on this content.

Modular electronics is a term that refers to a concept, where only electronics in form of specialized modules are used, supported by a common infrastructure. This concept is commonly used for trigger electronics and data acquisition in physics, but also in other areas including National Aeronautics and Space Administration (NASA)[74], military[75, 76] or aerospace[77, 78] or in commercial applications such as telecommunication networks[79].

Modular electronics is commonly used in detector readout systems because it provides a great cost reduction due to rapid development and reusability. The flexibility it offers, allows rapid design, development, testing and deployment (in a matter of days or weeks) as an experiment is being put together. Afterwards, the modules can be removed and used again for the purpose of a new experiment.

A common infrastructure, in form of a crate, provides standardization of module power supply and means of communications. A crate is a box (chassis) that mounts

Figure 3.1: Left: A standard NIM crate produced by Wiener. Right: Front and back panels of a dual gate generator.

in an electronic modules with an opening in the front facing the user. For and example see fig. 3.1.

In the following sections, the most popular in HEP community modular electronics are briefly presented.

3.1 NIM

The simplest and the first standard of modular electronics, presented on fig. 3.1, is NIM[80]. The standard, first defined in late 60s by the U.S. Atomic Energy Commission's report, defines mechanical and electrical aspects. The backplane does not offer any means of communication between modules, it provides only power. Any communication is made using module front panels, what in case of more complicated setups, results in difficult to maintain net of cables. Such a design along with support of hot swapping, provides great flexibility and interchangeability

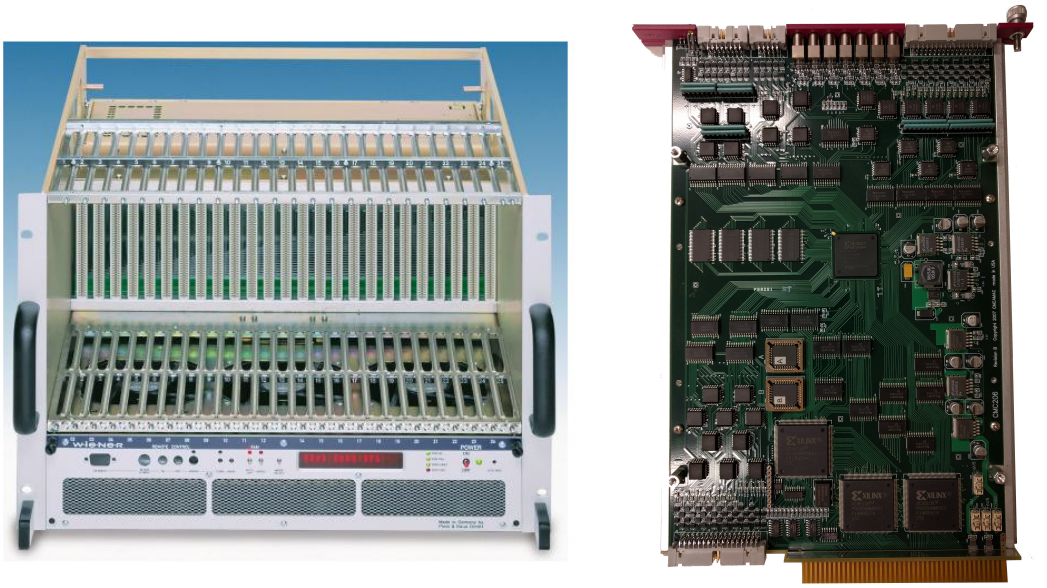


Figure 3.2: Left: A standard CAMAC crate produced by Wiener. Right: A FPGA based CAMAC module. It is used as the core of the NA61/SHINE trigger system.

between various modules. The user may quickly build a system for a particular application, and later easily restructure the instrumentation as required for different experiments or measurements. There are many standard modules available on the market such as: discriminators, fan-in fan-out units, pulse generators or converters between popular signal standard (Transistor–Transistor Logic (TTL), NIM, Emitter-Coupled Logic (ECL), Low-Voltage Differential Signaling (LVDS)).

This standard is heavily used in almost every system of the NA61/SHINE facility.

3.2 CAMAC

The Computer-Aided Measurement And Control (CAMAC), presented on fig. 3.2, is an international standard of modularized electronics, created by joint work of U.S. NIM and the European ESONE Committees. The standard (EUR

4100) [81] has been released in 1972 and it has been used at almost every nuclear physics research laboratory all over the world.

The CAMAC standard covers electrical and physical specifications for the modules, instrument housings or crates, and a crate backplane. The primary application of CAMAC was data acquisition but over the time it have been used for control applications as well.

The typical crate contains upto 25 module slots, which communicate using a standardized backplane called dataway. The rightmost module slot is reserved for a crate controller, which plays a role of gate between dataway and communication medium such as Ethernet. In the 80's the CAMAC has been superseded by much faster VMEbus (presented later in this chapter).

Noteworthy, the NA61/SHINE trigger system (see chapter 5) is based on this technology.

3.3 FASTBUS

FASTBUS (IEEE 960) [82] is a sophisticated data acquisition system standard developed by the U.S. NIM Committee in collaboration with the European ESONE Committee (ANSI/IEEE STD 960-1986). Essentially, the FASTBUS backplane was intended to replace CAMAC as a high-speed (upto 40MB/s), expandable data acquisition framework. Providing a more densely packed system than CAMAC system, it reduces dramatically the per-input cost.

However, the power requirements of the system build with FASTBUS were enormous. Each module may dissipate up to 70W, whereas a full-size crate holds upto 26 modules, results in total power upto 1820W. Typically, to provide enough current to every module, a high current (200A-300A) switched-mode power supplies had to be used.

Figure 3.3: Left: A standard VMEbus 64x crate produced by Wiener. Center: Front panel of an intel i7 based SBC VMEbus module from Concurrent Technologies. Right: Interior of SBC module using CompactFlash memory to hold an operating system.

Therefore, cooling and air handling were always a significant issue. Furthermore, production of modules having large physical size (355 mm by 381 mm) was more expensive than 3U and 6U VMEbus modules. Nonetheless, if needed, VMEbus offers a similar to FASTBUS size – 9U (360 mm by 340 mm) Consequently, the battle between FASTBUS and VMEbus had turned in favor of the latter one.

The FASTBUS is still in used in the NA61/SHINE experiment, for reading out Time of Flight detectors.

3.4 VMEBus

The VMEbus is a high performance bus system with powerful interrupt management and multiprocessor capability. It has been the most popular modular electronics and it is still used in many scientific facilities around the World. It

was first developed in 1981 and standardized by The International Electrotechnical Commission in 1987. The VMEbus specification has since then been refined few times adding new capabilities, the last VME 2eSST extension (VME 320 / VITA 1.5-2003) defines a synchronous data strobe and achieves 320 MB/s bandwidth. The list of VMEbus standards includes:

- 1982 - VME32 revision A – 32bit parallel bus. Transfers up to 40MB/s.
- 1987 - VME revision C – based ANSI/IEEE standard.
- 1987 - VXI – VME extension for instrumentation
- 1990 - VME430 – CERN nuclear VME 30 pin Paux connector additional -5.2V, -2V, +/-15V.
- 1994 - VME64 – Multiplexed 64 bit. Transfers up to 80MB/s
- 1996 - VME64x – New 160 pin connectors metric P0 connector.
- 1998 - VME64xP – VIPA (Physics), redefined P0. Adding 9U x 400mm module size.
- 2003 - VME 2eSST – Transfers increased up to 320MB/s.
- 2003 - VXS – Serial high speed fabric in P0 connector.

The mechanical design is based on the Eurocard form factor. Standard VME modules are 6U high and 160mm deep. For physics instrumentation a 9U x 400m form factor was added. The 3U exists as well, however it is rarely used. All VME modules (except 3U) are equipped with two 3-row DIN-96 or 5-row (VME64x) pin type connectors P1/P2 which match the backplane connectors J1/J2 (see fig. 3.4).

The VME64x standard provides an optional 95 pin 2 mm hard metric P0/J0 connector for more user defined I/O. Notably, the CERN VME 430 (“Nuclear

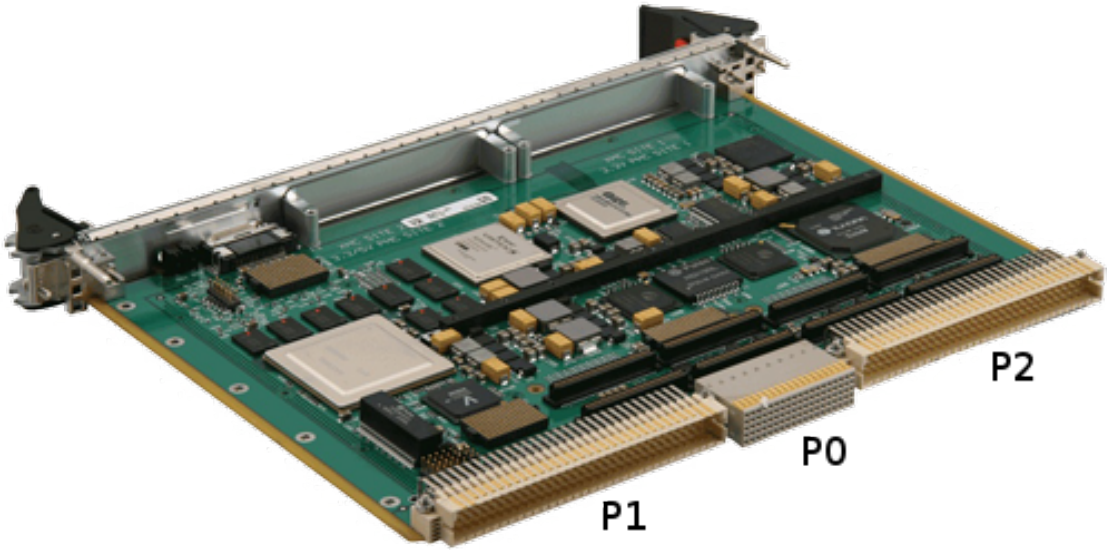


Figure 3.4: Connector names of a VMEbus module – P1,P0,P2. Corresponding crate side names are J1,J0,J2 respectively.

VME") Standard extends the backplane by adding a third dataway and connector row Jaux. Jaux was added in the place of P0/J0 connector between J1 and J2 to provide additional pins for DC power, geographic addressing and 3 differential bus signals (clocks and timing). It uses 3-row J1/Jaux/J2 connectors, which are backwards compatible – any standard VME / VME64 module will work in a CERN VME 430 compliant crate.

Typical VME crates hold up to 21 modules where the left-most module (slot 1) is called a VMEbus Bus arbiter. The main role of the arbiter is granting (using BG0OUT-BG3OUT signals) the bus to any module which first sent the bus request signal (BR0*-BR3*). Arbitration can be prioritized with 7 bus request levels or round robin. VME also defines a flexible prioritized interrupt subsystem. The VMEbus is a multi-master bus, however it can have only one arbiter.

The last extension, VMEBus Switched Serial (VXS), incorporates to the VME interface additional channels of communication such as PCI Express, RapidIO,

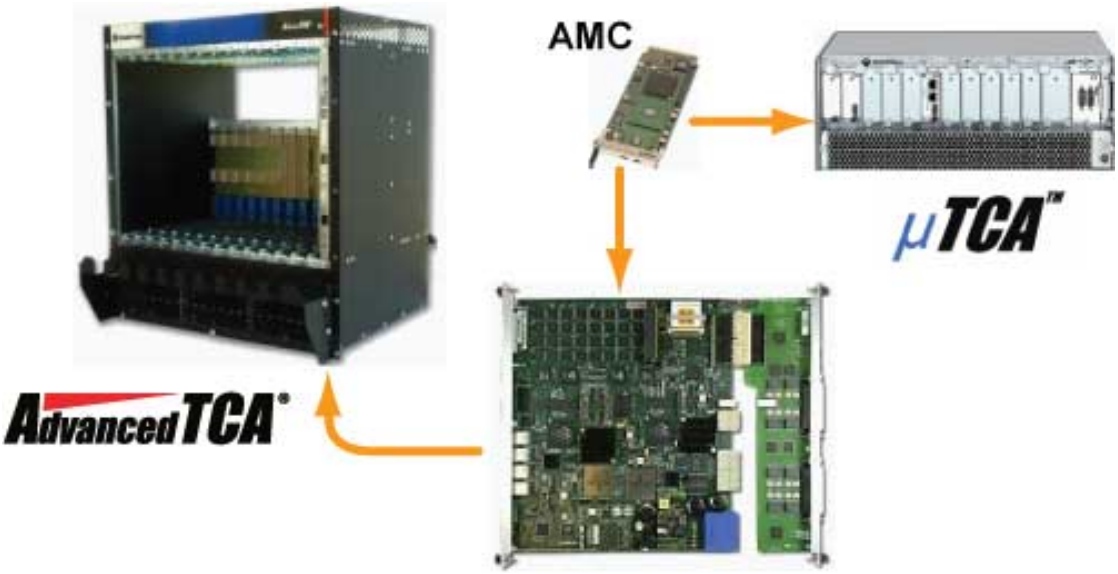


Figure 3.5: Left: ATCA crate. Center bottom: A blade. Center top: Mezzanine card AMC. Right: μTCA crate.

StarFabric and InfiniBand. The VXS standard did not gain much popularity, probably because the PICMG 3.x specification of ATCA was ratified by the PICMG organization in December 2002, year before VXS standard. Nevertheless, the VME64/VME64x are still very popular and seems to coexists with ATCA as an inexpensive solution for less demanding tasks, offering variety of modules delivered by various vendors.

The VMEBus is used in the data acquisition system (described in chapter 4) of the NA61/SHINE experiment.

3.5 ATCA

The newest technology and the last described in this chapter is Advanced Telecom Computing Architecture (ATCA). The ATCA was defined by PCI Industrial

Computers Manufacturing Group (PICMG) to address the need for commercial off-the-shelf components capable to provide high operational availability of "five nines" class (99.999%) by using redundancy technique [83].

Thanks to its open, multi-vendor architecture, more than 100 organizations currently support ATCA including Intel, Motorola, Alcatel-Lucent, Fujitsu, Sun Microsystems, Lockheed-Martin, the U.S. Department of Defense, and the U.S. Department of Energy. The ATCA is clearly superseding an over 30 years old VMEbus technology, however due to high complexity, VMEbus crates and modules (including new designs) are still produced.

An ATCA board (called blade) is 280 mm deep and 322 mm high, so they are slightly smaller than FASTBUS and 9U size VMEbus. Blades may carry on few small Advanced Mezzanine Cards (AMCs), which also plugs directly into a small version of ATCA crate – μTCA. This relation is depicted on section 3.5.

The AdvancedTCA backplane provides point-to-point connections between the boards and, in contrary to VMEbus, does not use a data bus. The backplane connectors are divided into so-called zones: Zone-1, Zone-2, and Zone-3.

The Zone-1 connector provides redundant power and system management signals (Intelligent Platform Management Interface (IPMI), Inter-Integrated Circuit (I2C) and Field Replaceable Units (FRU) data). The Zone-2 provides 1 to 5 connectors used for data transfers. All Fabric connections use point-to-point 100Ω differential signals. Zone-2 supports up to 200 connections of any communication standard that can use 100Ω differential signals. All connections are divided into 4 groups:

- 64 pairs for Base Interface – usually Ethernet
- 120 pairs for Fabric Interface in star configuration or full mesh – Ethernet, PCIe, Infiniband, serial RapidIO or StarFabric.
- 6 pairs for Clock Synchronization

- 10 pairs for Update Channel

The connectors in Zone-3 are user defined and are usually used to connect to a Rear Transition Module.

Clearly the ATCA is the most powerful and flexible modular electronics standard than anything before. Unfortunately, the great complexity and variety of communication protocols requires a complex, standard compliant software. Furthermore, many features of the standard are optional, what may lead to incompatibilities between modules from different vendors. Additionally, there are not many modules on the market which are suited for physics applications. Consequently, the VMEbus standard is still widely used, offering simplicity, low cost and often acceptable data transfer rates.

Chapter 4

Data Acquisition System

This chapter is intended to provide more in-depth knowledge about data acquisition system of the NA61/SHINE experiment. Comprehension of this dissertation does not depend on this content, however, it explains how data are obtained. The author contributed to the software development as well as maintenance of this system. The work presented in this chapter has been published [65].

The readout system, called also Data Acquisition (DAQ), is an electronic system which processes analog signals given by detectors and stores in digitized form for further analysis. Typically a structure of a DAQ system can be described as four main parts: Front-End Electronics (FEE), Back-End Electronics (BEE), an event builder and a data storage.

The FEE is electronics integrated with a detector or attached directly to a detector. In general, it performs amplification of analog signals, such as TPC pad charges, and sometimes digitization. Digitization, if possible, is applied in order to avoid signal deformation (due to transmission line length) which requires signal shaping.

The BEE is an equipment which is located far from detector e.g. control room.

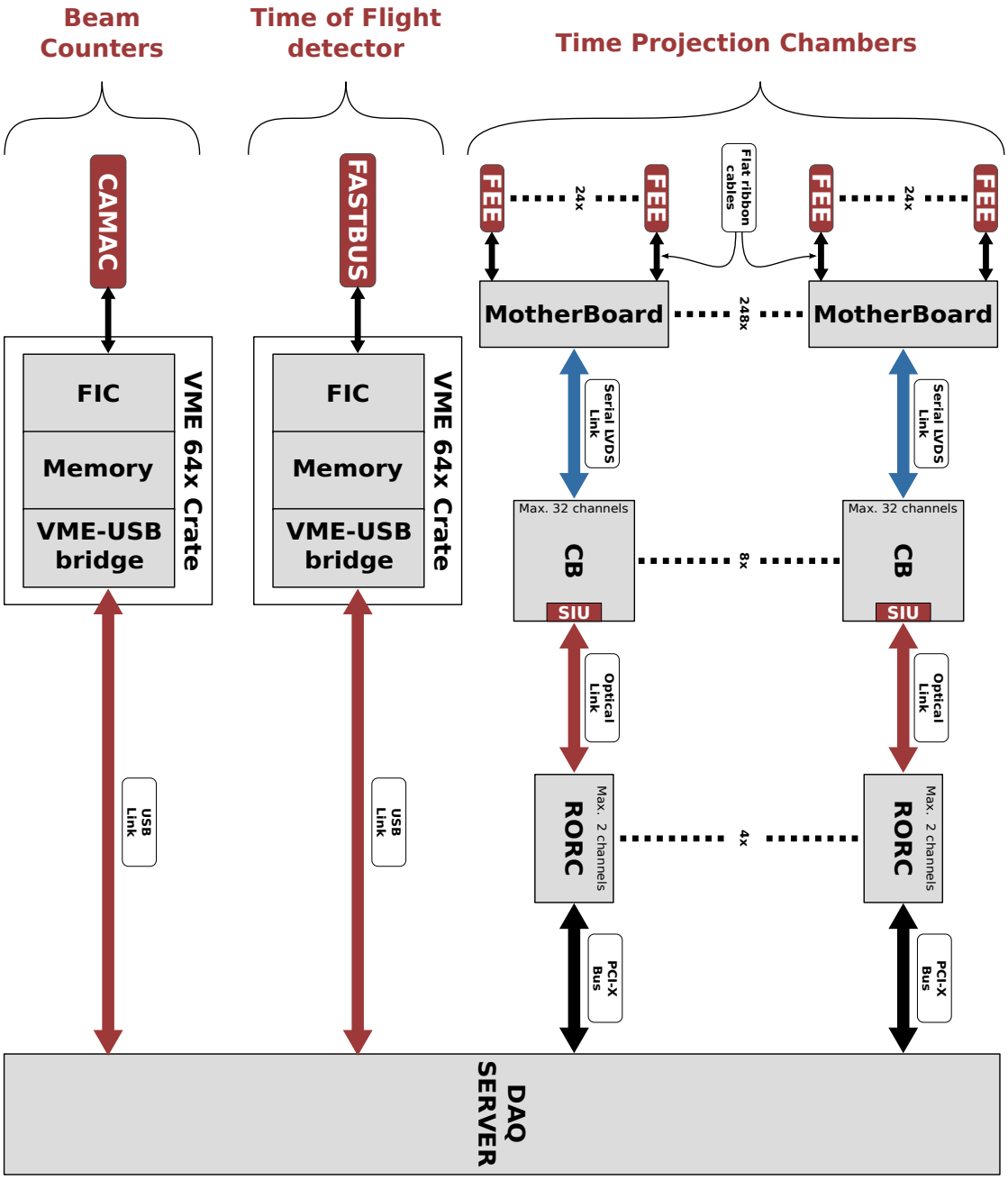


Figure 4.1: Overview of the NA61/SHINE data acquisition system.

It works mostly with already digitized signals provided by FEE. In contrast to FEE, mostly modular electronics such as NIM [80], VMEbus [84], CAMAC [85], ATCA [86] are used to implement this part of readout system. The main goal of BEE is to collect signals (mostly digitized) from detector, which will be gathered into an event by an event builder.

An event builder usually is a PC class computer running GNU/Linux system. Its main role is to collect data fragments provided by BEE and merge them into a single event. It includes a timing synchronization, which assures that data fragments belonging to the same event will be merged together and not mixed.

The data storage is a system capable of storing collected data. In the NA61/SHINE experiment the CERN CASTOR storage is used, which provides roughly 100 PB of storage. However, it is shared with other CERN experiments.

The NA61 readout system, presented on fig. 4.1, consists of three parts: beam counters readout, ToF readout and TPC readout. It is divided into three parts because each part uses a different technology.

Beam counters use a CAMAC and VMEbus standards whereas ToF uses FASTBUS (IEEE 960) and VMEbus standard. On the other hand, the FEE of TPCs are readout by motherboards, which are custom FPGA based electronics.

The NA61/SHINE [69] is a successor of the NA49 experiment [66]. Therefore it inherited all hardware and software [87]. However, already few upgrades have been performed, such as: TPC Readout, Trigger system and software. The effort to upgrade further parts of the experiment and unify this technological mosaic is ongoing.

In the following sections, the main electronic components of NA61/SHINE DAQ are briefly described. More detailed description on the data acquisition system is provided by [65].

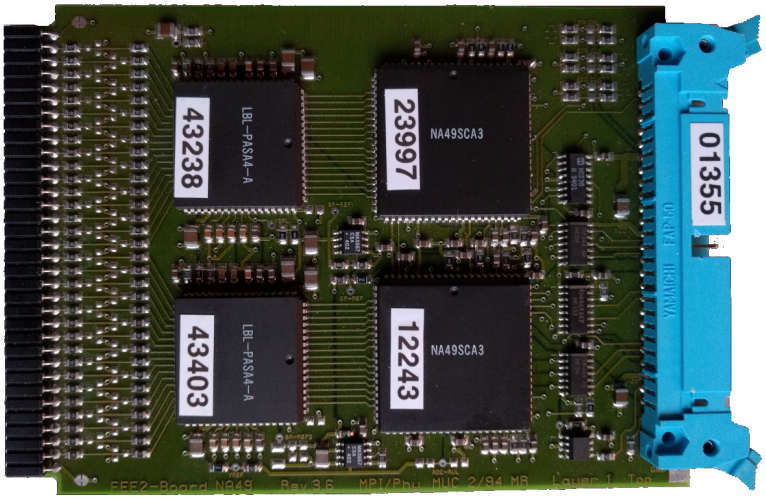


Figure 4.2: Front-End card. The card is connected to pads with the black (left) connector and it is connected to a motherboard with the blue (right) connector. The LBL-PASA4-A chip [88] is a fully integrated CMOS pulse shaping amplifier. The NA49SCA3 chip [89, 90] is a 16 channel, 12bit (dynamic range) Analog Digital Converter (ADC) which uses Switched Capacitor Array (SCA). It is a custom IC developed for the NA49 and STAR experiments.

4.1 Front-End Cards

In the NA61/SHINE experiment, FEE card (fig. 4.2) upon trigger arrival, samples the pre-amplified and shaped pad charges using the LBL-PASA4-A chip [88]. Then the samples (256 or 512 so-called time-slices) are digitized using the NA49SCA3 chip [89, 90]. It is a Wilkinson type 12bit (dynamic range) Analog to Digital Converter (ADC) using Switched Capacitor Array (SCA) and an external 25Mhz clock. The clock is centrally generated in order to avoid clock phase mismatch.

The final digital signal of a FEE card presents 9 bit charge value, which is sent to a motherboard, where due to pedestal subtraction and noise suppression, the

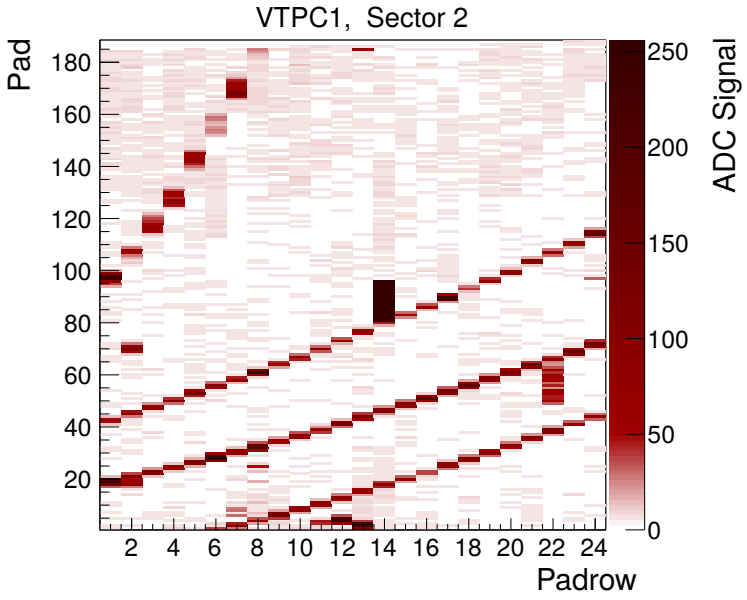


Figure 4.3: Example of digitized electric charge signals, in readout plane projection (pad-array) of a single TPC sector. The maximum value of all time-slices is chosen.

ADC values are truncated into 8 bits. One FE card can handle 32 TPC detection pads, so 182000 pads require 5687 FEE cards. The maximum values per pad registered by a TPC sector pads can be seen on fig. 4.3.

4.2 Motherboards

Motherboards are the FPGA based boards used to readout 24 FEE cards. The experimental facility has 5687 cards, which requires 236 motherboards. A motherboard reads values registered by FEE cards (9 bit long) and performs pedestal subtraction in order to remove a signal offset.

Values of pedestals to be subtracted are calculated from signals produced by electronics between data taking periods. Those values tends to vary mostly due to change of environmental conditions (e.g. temperature). Next, the noise suppression

is performed by substituting with zeros all channels below certain threshold, typical value of ADC signal to be nullified is 6 (maximum is $2^8 - 1 = 255$).

Afterwards, the data is sorted according to their pad number. The sorted data, before it is sent to a Concentrator Box, is compressed using zero compression algorithm. The zero compression algorithm is very simple and goes as follows:

- the consecutive zero bytes are replaced by two bytes: a zero and the number of zeros. So if we have 10 zeros, we send 0x00 and 0x0A (representing it in hexadecimal numerals).
- If the number of zeros are more than 255 then only 255 zeros are replaced, followed again by a zero and the number of remaining zeros. For example, 260 zeros will be compressed to 0x00,0xFF,0x00 and 0x05.
- A single zero is replaced by two bytes containing the values 0x00 and 0x01.

The event size without pedestal subtraction, noise suppression and zero compression is 100 MB for 512 time-slices mode and 50 MB for 256 time-slices mode. However when the above mentioned techniques are applied, the event size drops to a value between 1.5 and 5 MB, depending on number of registered trajectories. The serialized data of the FEEs are then sent by the Motherboard to Concentrator Box using LVDS connection with an effective bandwidth of $50 Mbit/s$ on the maximum distance of $15m$. The LVDS transmission medium is a STP (Shielded Twisted Pair) cable.

4.3 Concentrator Boxes

Concentrator Boxes are standalone serialization boxes with 32 bidirectional LVDS inputs/outputs. Up to 32 Motherboards can be connected to a single Concentrator Box. During the readout, the Concentrator Box serializes the incoming

data streams onto a single 32 bit wide data stream. Then the 32 bit data is transmitted to the DAQ server PC through DDL [91, 92]connection.

In the other direction, Concentrator Boxes transceive commands, such as pedestal table and control commands, from the DAQ PC to Motherboards.

4.4 DAQ Computer

The DAQ (Data AcQuisition) computer is based on x86-64 architecture, containing Detector Data Link (DDL) cards. The DDL [91, 92] is a high-speed serial optical link with parallel interfaces developed for the ALICE experiment. A link consists of a SIU (Source Interface Unit) card, RORC (Readout Receiver Card) and a maximum 200m length optical cable.

The DDL transfers the acquired detector data, data blocks or status info to the Central DAQ computer, and using a backward channel can load data blocks or commands from the Central DAQ system to the lower level systems. The bandwidth is guaranteed to be a minimum of 200 MB/s in both directions.

The DAQ computer also performs event building and quality assessment. Afterwards, it sends data to a tape based common storage (CASTOR [93]). The description of output raw data format can be found in [94].

4.5 Performance

The Data Acquisition system serves as a very reliable system for the extended data taking periods of 2–5 months per year. The data acquisition system was used to record a large amount of physics data on 30 types of reactions, with the number of events ranging from 1 to 40 millions (4 millions on average), summing up to 200 millions physics events recorded during past 7 years. It is expected that a further

20 types of reactions will be recorded until 2020.

The system dead-time[1] of 12 ms, limited by front-end cards processing speed, restrains a data rate to 83 Hz. The data rate proved to be sufficient for recording all data accordingly to the NA61/SHINE physics program and satisfies the present needs given the upper safety limit of usable beam intensity.

The performance of the data acquisition have met all the NA61/SHINE experiment requirements, recording data used in various physic analyses. Scientific results achieved with collected data have been published in prestigious journals [95, 96, 97, 98, 99, 100, 101, 102, 103].

4.6 Future Upgrades

The key motivation for developing the upgrade of Trigger and DAQ systems is measurement of charm hadron production, which requires a tenfold increase of the data taking rate from 80Hz to 1KHz[104, 105].

Furthermore, evolution of physics program requires adding new detectors to our DAQ system in an easy manner. Current DAQ system is already on the edge of scalability limit and bandwidth limit. Consequently, it has been decided to design a new system which meets requirements of the NA61/SHINE experiment.

The new DAQ system is already under development and it is foreseen to collect first data in 2021. The author contributes to this project as main designer and coordinator.

[1] the dead time is the time after each event during which the system is not able to record next event, because it still processes the previous one.

Chapter 5

Trigger System

This chapter is intended to provide more in-depth knowledge about trigger system of the NA61/SHINE experiment. Comprehension of this dissertation, does not depend on this content, however, it explained how data are selected. The author of this thesis contributed to the software development as well as maintenance of this system. In particular, the trigger monitor was entirely designed and developed by myself [64].

The aim of the trigger system is to select class of events that are of particular interest. Simultaneously, it reduces the data throughput that the data acquisition must handle, consequently reducing overall building cost of an experiment.

Furthermore, modern trigger systems allow to select few class of events as well as the proportions between them. Such filtration, significantly simplifies further data processing as the events are usually labeled by trigger with event class name. In the NA61/SHINE trigger system there is possible to define 4 event classes (T1, T2, T3, T4) as well as so-called pre-scaling levels for each of the classes, which define ratio between number of events in each class.

The Trigger system [64] uses a set of scintillator counters along with the PSD

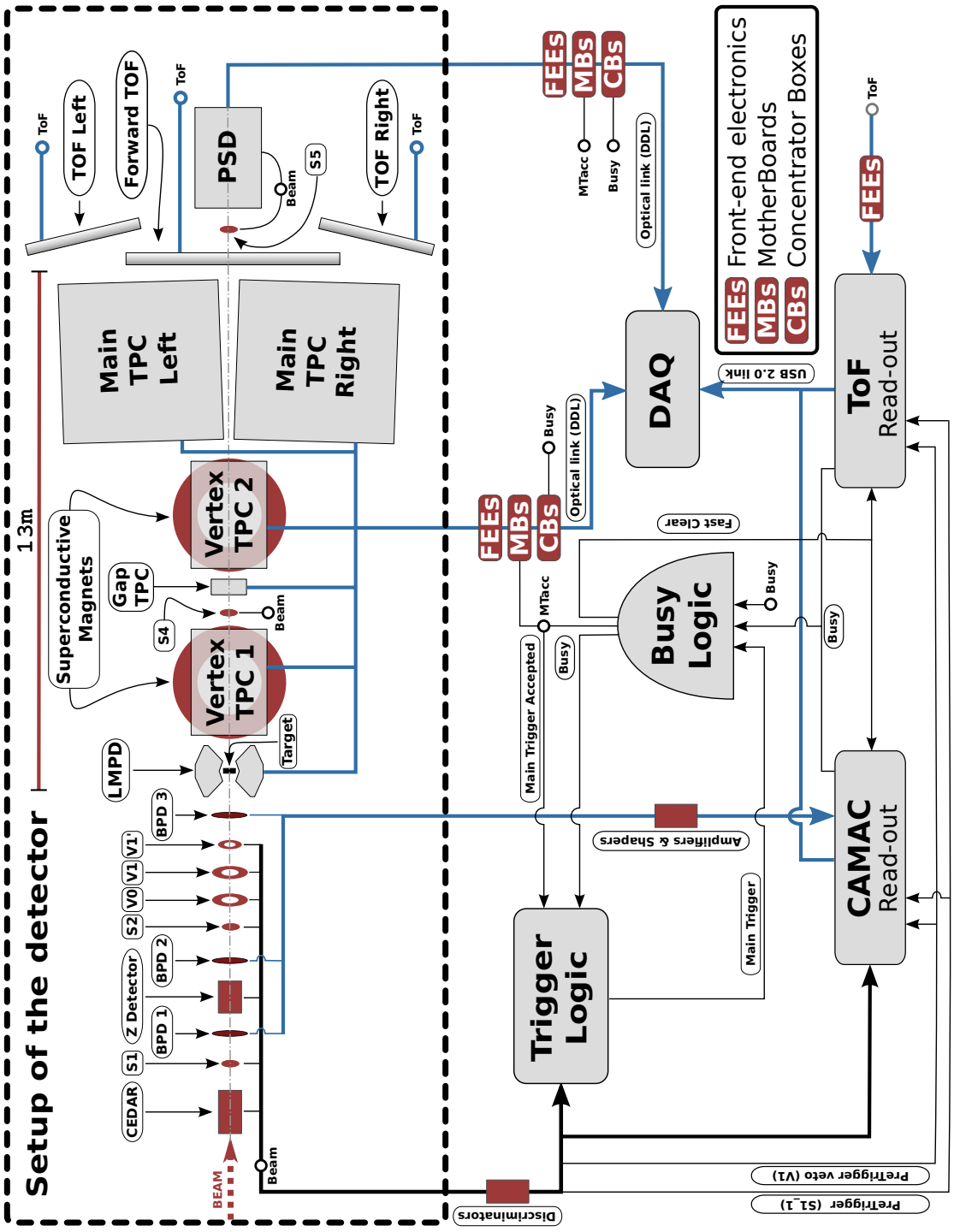


Figure 5.1: General overview of the NA61 readout system which is steered by the Trigger and Busy logic.

calorimeter in order to determine an appearance of the events of interest. In case the trigger detects a desired signature, it sends a signal (called Main Trigger) to the Busy logic. Busy logic is a system built on modular electronics called NIM, which makes decision whether accept an event or not. It is crucial part of a whole readout system, which mediates between Trigger and DAQ systems, as it is shown on fig. 5.1. If DAQ system is busy reading out a previous event, the Busy logic refuses further requests from Trigger system. The time during which DAQ system is unable to collect new events is called **dead time**.

The Trigger system is used to reduce the event rate and store the most interesting ones. In fact, it leads to tremendous reduction of data. Typical beam spill intensity counts about 100,000 particles. If we multiply it by typical size of an event 1.6MB (after compression, pedestal subtraction and noise suppression), we get 160GB of data. Considering the fact that a spill duration is typically about 10s, we end up with data rate about 16GB/s (peak value). Such an amount would increase significantly overall cost of readout system.

5.1 Beam Counters and Signal Processing

Signals from scintillation and Cerenkov counters as well as summed signal from PSD calorimeter are used by the Trigger system. Those signals provide precise position and timing measurement of particle beams. In total up to 16 signals have to be processed for a trigger decision:

- S1,S2,S3,S4,S5 are signals produced by scintillator detectors, located on the beam, when a particle pass through them. They are treated as positive signals in coincidence.
- V0, V1, V1' are also signals coming from scintillator, however they are located around the beam. Those signals are vetoing coincidence, because they indicate

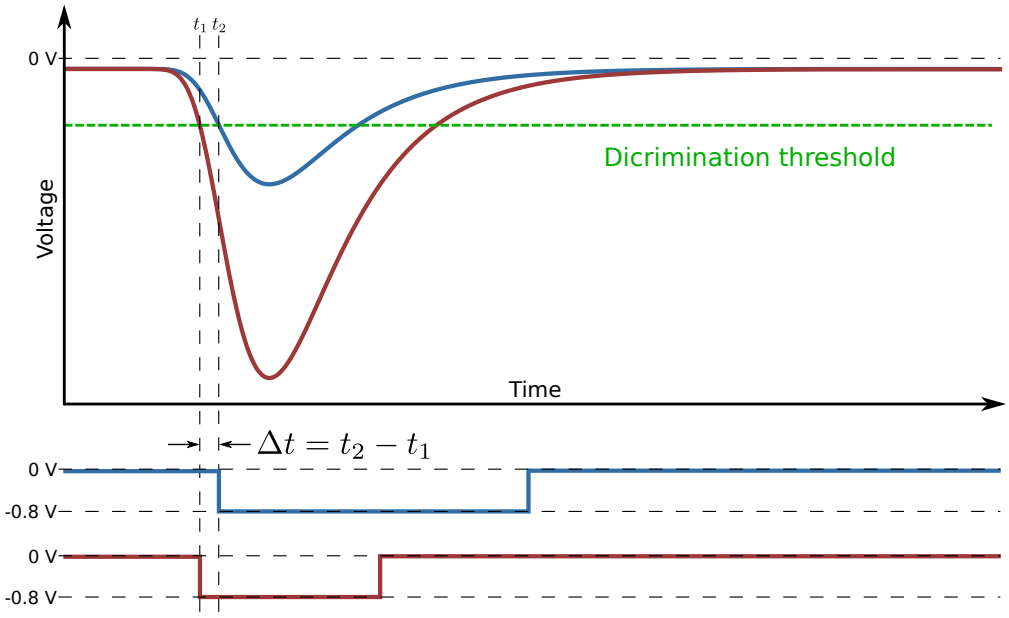


Figure 5.2: Illustration of LED output signal walking, caused by difference in input signal amplitude.

particles which diverged from the beam line.

- Cerenkov detector (Z detector) is used to roughly identify beam particles using Cerenkov radiation effect. It produces signals if a particle travels faster then the phase velocity of light in that medium.

- CEDAR counters are **CE**renkov **D**ifferential counters with **A**chromatic **R**ing focus. In contrast to ordinary Cerenkov detectors, it can detect particular type of particle.

- PSD is a calorimeter used to measure particle energy deposition and to determine centrality of an interaction. Simply speaking, a central interaction did not happen if PSD detects a projectile (particle).

Before the signals can be used, they have to be digitized. Digitization is performed by so-called discriminators, typically implemented as NIM based modules

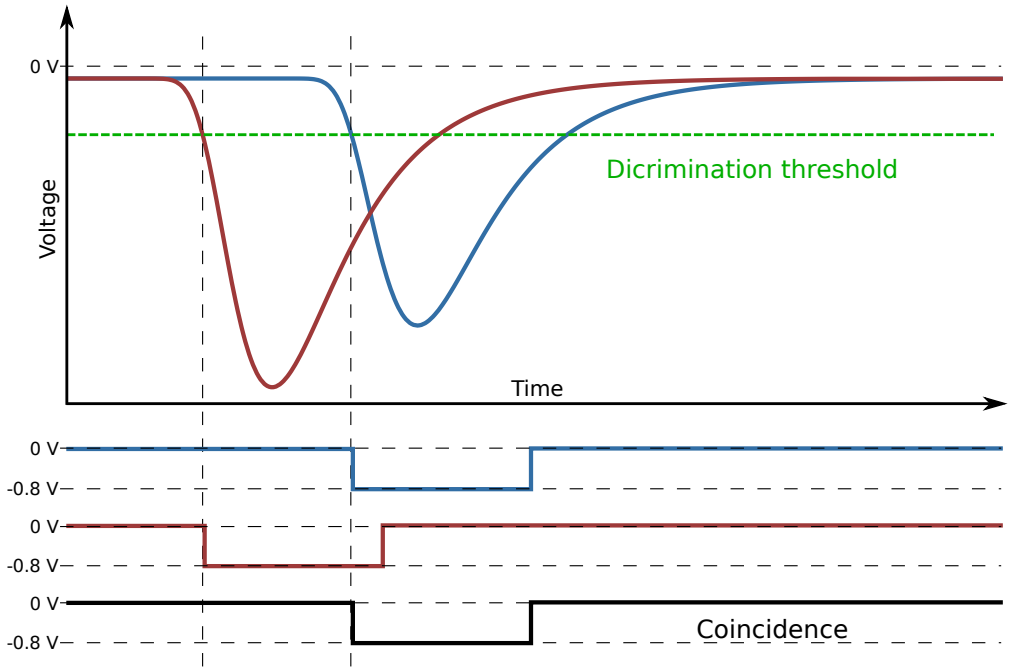


Figure 5.3: Input analog signals (top plot), followed by their digital forms of discriminated input signals (red and blue rectangular signals). The result of coincidence of two discriminated signals is represented by bottom black plot.

[80]. Two types of discriminators are used: simpler LED and more complex CFD.

The LED is commonly used in applications where high precision (order of ps) is not required. The working principle is fairy simple, it produces an digital signal every time when analog input signal passes a voltage threshold. This feature is implemented using operational amplifiers (e.g. CA747). However, LEDs have serious drawback which decreases time resolution, it is so-called time walking effect.

The effect presented on fig. 5.2 is the major effect which causes digital signals to jitter. Therefore, precise measurement of the time of flight and consequently, particle identification (using ToF) would be impossible. In applications where this drawback is not acceptable, CFD are used.

The Constant Fraction Discriminator solves this problem by using a constant

fraction of the input pulse to precisely determine the timing of the output pulse relative to the input signal. It is done by splitting the input signal and attenuating first one so that it is a certain fraction of the original amplitude, delaying and inverting the other signal. The attenuated pulse and the secons one (delayed and inverted) are then added together. Computed zero crossing value is our discrimination threshold.

After discrimination of signals, logic operations such as AND/OR can be performed using coincidence NIM modules. The idea of discrimination and coincidence are presented on fig. 5.3 with an example signals.

5.2 Trigger Logic

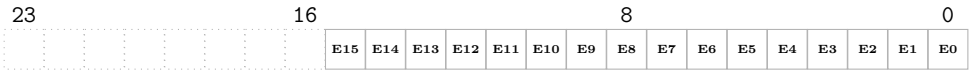
The core of the system is based on a single FPGA, which holds a Trigger logic able to handle 16 input signals. The FPGA (Xilinx Spartan-3 XC3S1500-4) is housed in a commercial CMC206 Universal Logic Module according to the CAMAC [85] standard. The FPGA is RAM based, thus the synthesized code of trigger logic must be reloaded from a flash memory each time a power is applied. It runs with a 120 MHz clock offering a resolution of 8.3 ns.

The CMC206 module implementation provides a possibility of parallel event selection which satisfies up to four user defined trigger definitions. In other words, we can define up to four different event classes/types using combinations of input signals, which may have a physical meaning e.g. centrality of interaction.

Events corresponding to these trigger definitions are recorded with relative frequencies which can be selected using 12 bit pre-scalers. The working parameters of the trigger, such as trigger definitions, pre-scaller values, coincidences and delays, are set up remotely via a Java application, designed to be handled by non-expert users. Those parameters are written into 24-bit configuration registers (located in

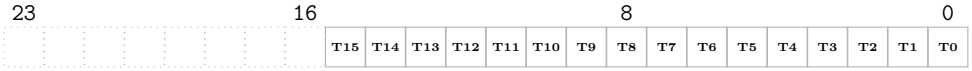
CMC206 module), namely:

- ENABLE (0..4, one per trigger definition):



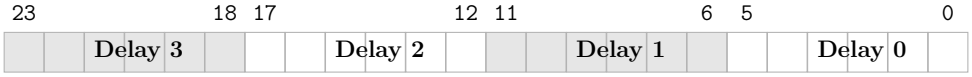
  It disables (0) or enables (1) an individual input signal (16 input signals in total).

- TRIGGER/VETO (0..4, one per trigger definition):

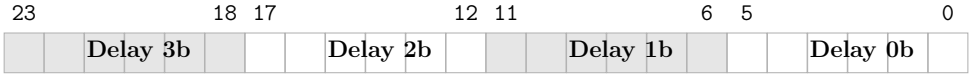


  Set an individual signal as a positive (trigger) or negative (veto) signal in coincidence

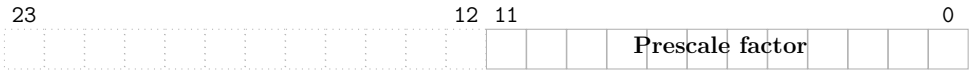
- DELAY (common for all trigger definitions, presented on fig. 5.4 and fig. 5.5):



- DELAYb (common for all trigger definitions, presented on fig. 5.6):



- PRESCALE (0..4, one per trigger definition):



Specifies the prescale factor for each trigger.

- `0x000` - all triggers pass through (every first trigger)
- `0x001` - every second trigger pass through
- `0x002` - every third trigger pass through
- `0xFFE` - every 4095th trigger pass through
- `0xFFF` - all triggers are blocked

- INHIBIT (common for all trigger definition):

It specifies an internal busy time for all triggers, during which new trigger generation is inhibited. The inhibit value is translated in the following manner: $(2 + 2 \cdot Inhibit) \cdot 8.33ns$, except $Inhibit = 0x0000$ (no internal busy is generated).

- `0x0000` = 0 ns
- `0x0001` = 33.3 ns
- `0xFFFF` = 1.09182976 ms

The configuration is uploaded to the trigger module via Ethernet-CAMAC bridge module, into registers read by FPGA. Whole communication with CAMAC crate is performed using a C library which provides functions to send CAMAC commands and addresses. The same bridge is used to readout data from Scaler modules. Scaler is a module, which counts number of electric digital pulses.

Scaler data are mostly for monitoring purposes, however sometimes they are useful for physics analysis. Therefore, the data are sent do data acquisition to

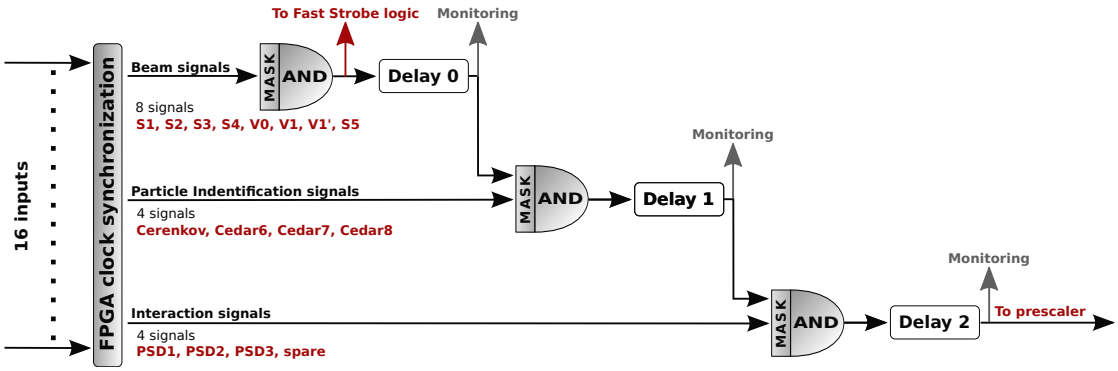


Figure 5.4: A single trigger logic. Beam signals inform that a particle has been detected and it hit the target. Particle identification signals provide information about particle type (proton, kaon etc.). Interaction signals notify about interaction centrality. If PSD produces a signal, it means there was no central collision.

be recorded. This task is done by trigger server, which using above mentioned C library, reads data from all monitoring modules and distribute it to connected clients. The communication is done using standard network sockets, which conform to Portable Operating System Interface for Unix (POSIX) standard.

An internal schematic of a single trigger logic (four in total) is presented on fig. 5.4. A trigger accepts up to 16 digital input signals within three categories:

- beam scintillator counters (up to 8 signals)
- signals from detectors used for particle identification (up to 4 signals)
- signals indicating interactions (up to 4 signals)

Three categories are used due to a big time differences between signal arrivals e.g. time difference between S1 (abbreviation for Scintillator number 1) signal and PSD is around 300 ns. Compensation of this difference using only one stage, would require delaying all beam counter signals by 300 ns outside of trigger. Delaying 8 signals would be very problematic regardless of delay type (analog or digital). In

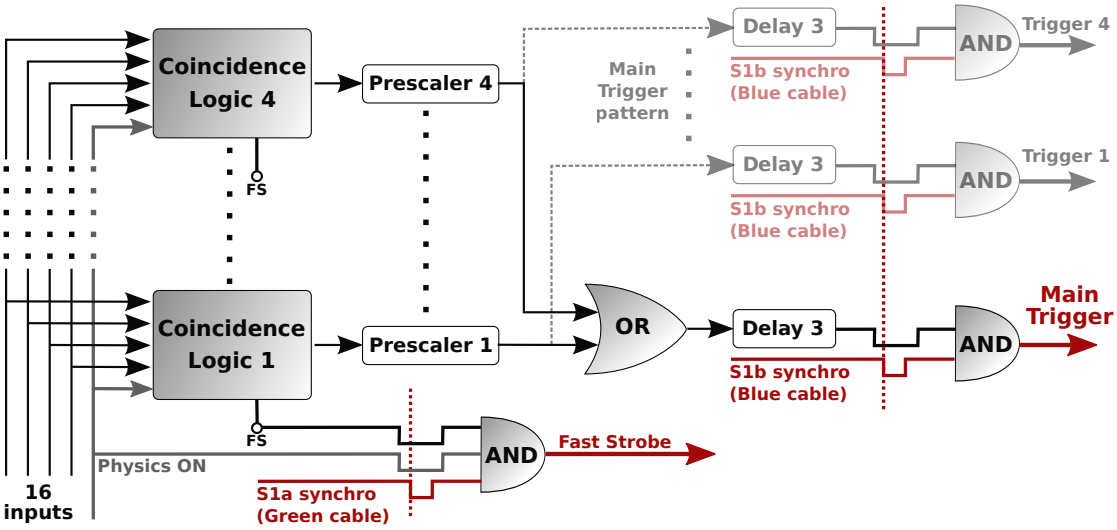


Figure 5.5: Relation between four triggers. Detailed scheme of Coincidence logic 1..4 is presented on fig. 5.4. Description can be found in section 5.2.

presented architecture, it is simpler to compensate any difference up to 531.2 ns ($2^6 \cdot 8.3ns = 531.2ns$, where 8.3ns is a FPGA clock period) by changing values of Delay0, Delay1 and Delay2. The delays are presented on fig. 5.4.

The trigger system contains four triggers joined as it is presented on fig. 5.5. The output signal of each trigger logic is sent to its own pre-scaller, which transmits every n^{th} signal (blocking other signals) according to the configuration.

Then, all signals are split to an OR gate (producing Main trigger) and to a dedicated monitoring outputs (Trigger 1..4). Delay3 and AND gate are used for internal synchronization ensuring constant processing time of the trigger logic. Fast Strobe signal is produced using only beam signals and it is meant as a pre-trigger for very slow detectors, thus it must be emitted as soon as possible.

The fig. 5.6 presents Busy logic which can inhibit all output signals from being produced. The busy signal is produced by DAQ while still processing previous event.

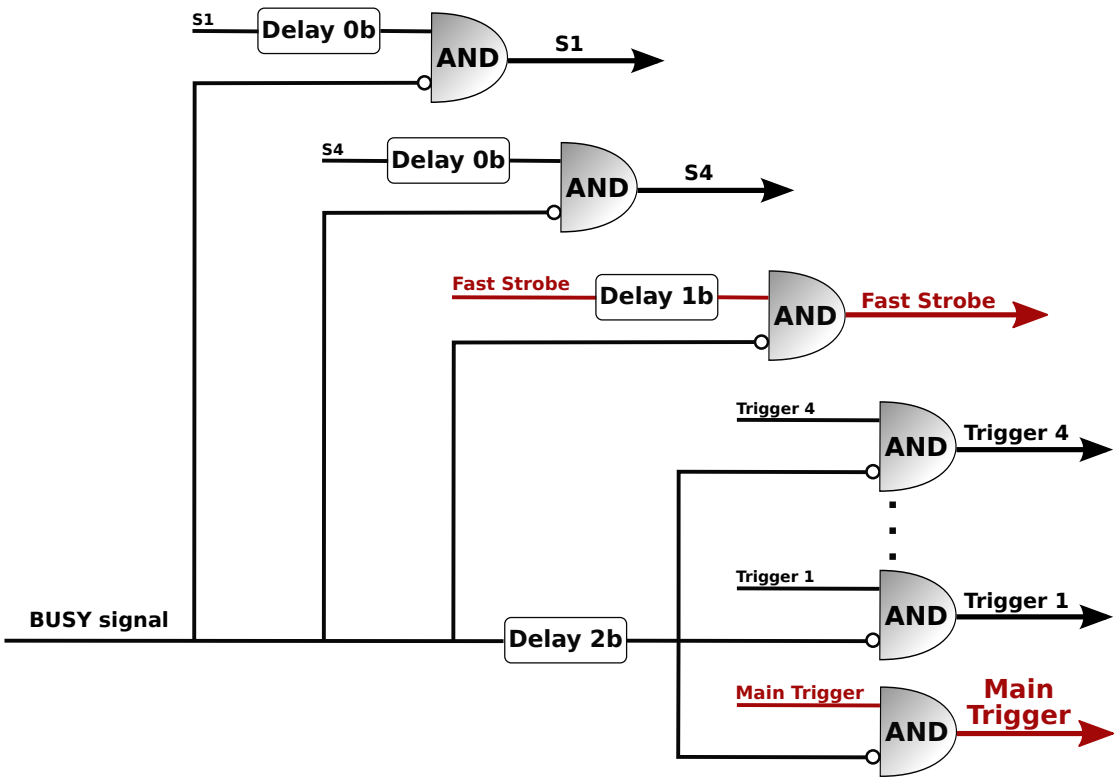


Figure 5.6: Inhibiting of all trigger signals with BUSY signal.

5.3 Monitoring

As it was mentioned in section 5.2, a trigger computer is connected with the trigger crate via Ethernet-CAMAC bridge module. The trigger computer, using a dedicated library, communicates with modules located in the crate. In order to upload a configuration or to download data from Scaler modules (counting impulses). The configuration is uploaded using, mentioned earlier, Java application.

However, data are downloaded using C++ application which plays a role of trigger server. The trigger server distributes monitoring data to all clients connected to the server. The communication is performed using POSIX sockets. For the time being, there are three important clients:

- DAQ computer – it stores monitoring data as it is sometimes useful for physics

analysis;

- Trigger monitor – it analyses and displays all crucial data in order to detect if any failure occurred. It is used by shift crew to monitor data taking process.

- Distributed Control System (DCS) – it collects detector related data in order to discovery failures of detector. Furthermore, data is stored for the future reference.

Noteworthy, the trigger monitor is a multi-threaded application, written in python, which implements observer design pattern. It contains a main readout thread and many monitoring threads registered in the main readout thread in order to receive data. The main readout thread informs other threads when data is ready for processing.

Until now, few monitoring interfaces have been written such as:

- QT based monitor – is used by shift crew to monitor data-taking. The spill structure tab of this interface is presented on fig. 5.8 and it is used to monitor quality of delivered beam. On fig. 5.7 the statistics tab is presented. It shows count values registered by scalers as well as user defined values such as trigger ratios or average values.

- Django [106] framework based monitor – used to publish data-taking status on a web page for people being outside of control room.

- Analyzing monitor – meant to perform various analysis of parameters sent by the trigger server e.g. Fast Fourier Transform (FFT) analysis of reference clock.

Those monitors all together help to control data-taking in very fine details and greatly decrease the risk of loosing expensive beam time, due to malfunction.

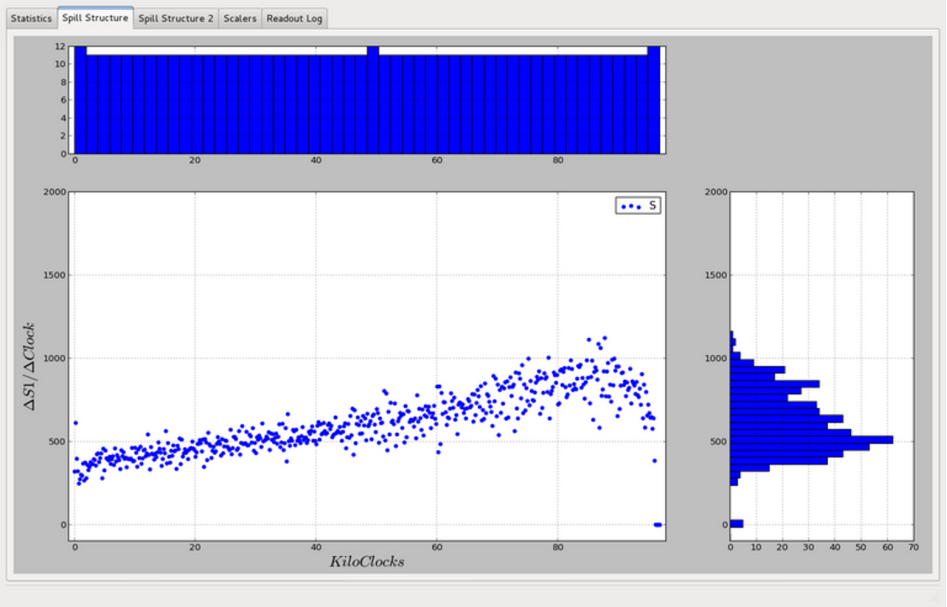
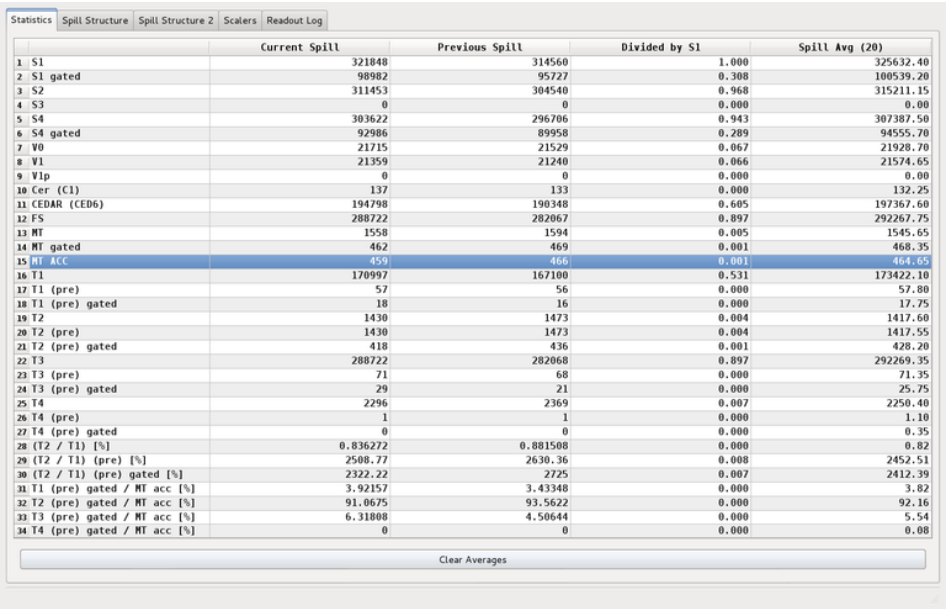


Figure 5.7: Trigger monitor application: spill structure tab. This tab displays information about time versus number of particles distribution.



	Current Spill	Previous Spill	Divided by S1	Spill Avg (20)
1 S1	321848	314560	1.000	325632.40
2 S1 gated	98982	95727	0.308	100539.20
3 S2	311453	304540	0.968	315211.15
4 S3	0	0	0.000	0.00
5 S4	303622	296706	0.943	307387.50
6 S4 gated	92986	89958	0.289	94555.70
7 V0	21715	21529	0.067	21928.70
8 V1	21359	21240	0.066	21574.65
9 V1p	0	0	0.000	0.00
10 Cer (C1)	137	133	0.000	132.25
11 CEDAR (CED6)	194798	190348	0.605	197367.60
12 FS	288722	282067	0.897	292267.75
13 MT	1558	1594	0.005	1545.65
14 MT gated	462	469	0.001	468.35
15 MT ACC	459	466	0.001	464.65
16 T1	170997	167100	0.531	173422.10
17 T1 (pre)	57	56	0.000	57.80
18 T1 (pre) gated	18	16	0.000	17.75
19 T2	1430	1473	0.004	1417.60
20 T2 (pre)	1430	1473	0.004	1417.55
21 T2 (pre) gated	418	436	0.001	428.20
22 T3	288722	282068	0.897	292269.35
23 T3 (pre)	71	68	0.000	71.35
24 T3 (pre) gated	29	21	0.000	25.75
25 T4	2296	2369	0.007	2250.40
26 T4 (pre)	1	1	0.000	1.10
27 T4 (pre) gated	0	0	0.000	0.35
28 (T2 / T1) [%]	0.836272	0.881508	0.000	0.82
29 (T2 / T1) (pre) [%]	2508.77	2630.36	0.008	2452.51
30 (T2 / T1) (pre) gated [%]	2322.22	2725	0.007	2412.39
31 T1 (pre) gated / MT acc [%]	3.92157	3.43348	0.000	3.82
32 T2 (pre) gated / MT acc [%]	91.0675	93.5622	0.000	92.16
33 T3 (pre) gated / MT acc [%]	6.31808	4.50644	0.000	5.54
34 T4 (pre) gated / MT acc [%]	0	0	0.000	0.08

Figure 5.8: Trigger monitor application: statistics tab. This tab displays statistical information gathered from all scaler modules.

5.4 Performance

The trigger system described in this chapter, have replaced trigger system inherited from the NA49 experiment[66], predecessor of the NA61/SHINE experiment. The previous experiment used a single trigger logic based on NIM coincidence modules [87].

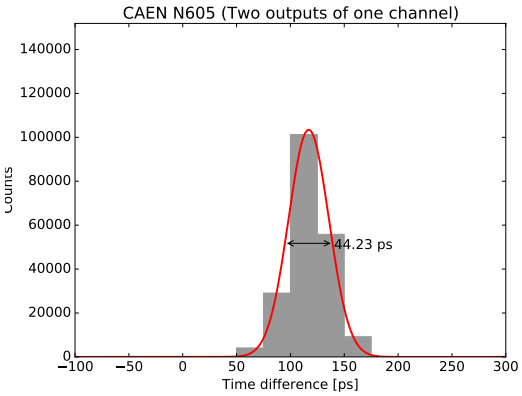
The new trigger system increased the time resolution from around 30ns to about 8.3ns, so it decreased a master trigger signal jitter and reduced the overall time to make a decision, if data should be collected.

Furthermore, the new system provides four independent trigger logics and an easy way to configure them, making data-taking more automatized. Previously, the triggers had to be modified manually for each trigger configuration each time. Because it was time consuming (loss of expensive beam time), switching between triggers was performed as rare as possible, what decreased quality of recorded data.

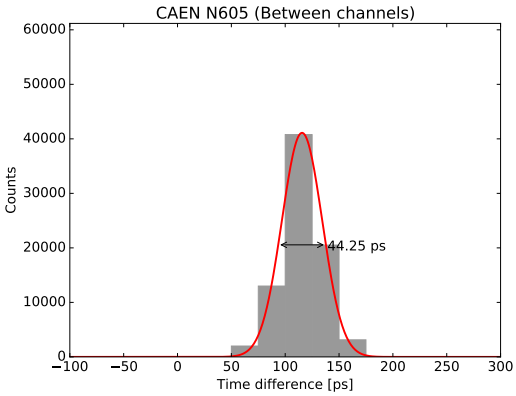
Over the time, hour by hour, some parameters of experimental environment fluctuate, changing continuously physical parameters. Those changes make physics analysis more difficult because, for example, data collected with an interaction trigger might be taken in different conditions than data collected without target for background measurements.

The new system, as mentioned previously, is able to generate up to four triggers, hundreds times per seconds, making the problem of changing conditions negligible.

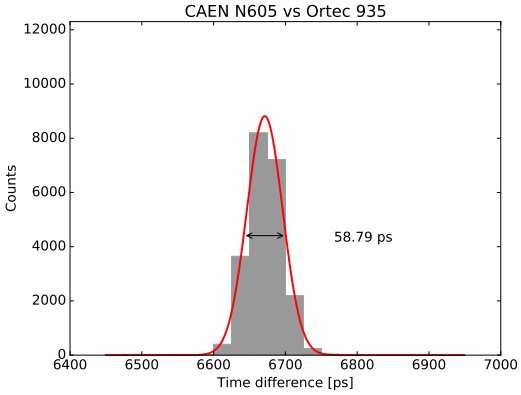
This trigger system together with DAQ system, proved to be reliable. Results of analysis of data collected using those systems have been reported in prestigious scientific journals [95, 96, 97, 98, 99, 100, 101, 102, 103].



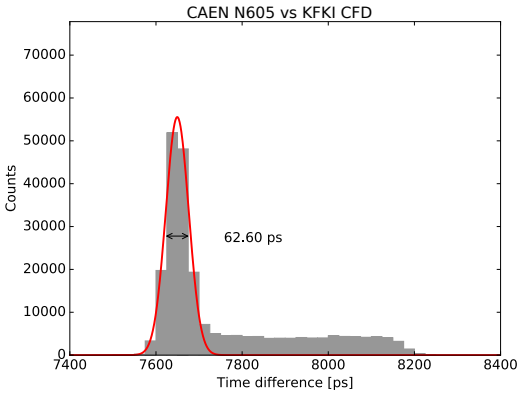
(a) Reference time difference between two outputs of the same channels (CAEN 605).



(b) Time difference between two independent channels of CAEN N605 module.



(c) Time difference between CAEN N605 and Ortec 935 channels.



(d) Time difference between CAEN N605 and KFKI CFD channels.

Figure 5.9: Comparison test of CFD modules produced by CAEN, Ortec and KFKI Central Research Institute for Physics (Hungarian Academy of Sciences). Description can be found in section 5.5.

5.5 Future Upgrades

The trigger system will require upgrades to cope with new DAQ system, which is being prepared for data-taking after year 2020 – after CERN's long shutdown 2 (scheduled from 2019-2020).

The most likely, CFD modules will have to be replaced with CAEN N605 CFD. The CAEN N605 module has been selected after performance study. Aim of the study was measurement of time resolution between currently used model produced by Central Research Institute for Physics of Hungarian Academy of Sciences (KFKI) and modules produced by CAEN and Ortec.

The time was measured using single crate VMEbus based DAQ equipped with:

- SBC model VP717
  produced by Concurrent Technologies
- CAEN V1290n Time to Digital Converter (TDC) module with 25ps time resolution

Although CAEN claims that N605 model has time resolution below 5ps, the V1290n TDC was sufficient to compare all modules and select the best one.

Final results of the study are presented on fig. 5.9. On the fig. 5.9a one can see a time difference between two inputs of the very same signal. It shows that the time resolution of the V1290n TDC module is around 44ps, whereas producer claims 25ps [107].

The V1290n module is using HPTDC chip [108] with 100ps time resolution. Thus, it is using a R-C delay lines in order to combine four 100ps channels into one 25 ps channel. The time resolution of 44ps indicates that R-C delay lines were not calibrated very well.

Time difference between two separate channels of N605 is presented on fig. 5.9b. The conclusion is that the time difference between channels is below time resolution

of TDC module.

Comparison between CAEN N605 and Ortec 935 can be found on fig. 5.9c. The time resolution is worse ($\approx 58ps$), so by comparison with fig. 5.9b we can conclude that Ortec 935 has worse time resolution that CAEN N605.

Similarly, comparison presented on fig. 5.9d reveals that currently used CFD module produced by KFKI has not only the worse time resolution but also it reveals signal instability (the long tail). Therefore, the KFKI module will be replaced the most likely by CAEN N605 module.

Last but not least, the trigger system will be moved from old CAMAC crate to more modern VMEbus crate, and current Busy logic will be integrated with the trigger system. This will increase flexibility and stability of DAQ system as it causes sometimes problems, due to current implementation, which uses around 30 NIM modules.

Chapter 6

Shine Offline Framework

This chapter is intended to provide introduction to the Shine software framework design. This framework was used to develop particle trajectory reconstruction algorithm, which is related to the main thesis of this work. Comprehension of this dissertation does not depend on this content. The author of this dissertation contributed to the software development, especially in the parts related to reconstruction, simulation and compiling process. The results of work presented in this chapter have been published [67, 68], additional is in preparation.

The data of the experimental apparatus is collected by the DAQ system, and subsequently it is stored in the CERN Advanced STORage manager (CASTOR) [93, 109]. The format of data, so called raw data format, is very technical and detector spectific, contains mostly ADC and TDC values for various detector channels in various order.

Before one could think of physics analysis of these data, it must be simplified: from the raw data the properties specific to the produced particles need to be restored. Normally, the first stage of this procedure is the calibration of the data. In this stage all the imperfections of detector apparatus are quantified and corrected,

and some calibration parameters are determined, for instance the drift velocity in the TPC chambers.

The second stage of the procedure is the actual reconstruction of the calibrated data. The reconstruction is the process of calculating essential physics parameters of the produced particles, such as the particle momentum, charge, energy loss, time-of-flight, required for further more specific analyses.

In order to facilitate flexible development of the calibration and reconstruction algorithms, a common set of tools have been written – the Shine Offline framework [68, 67].

The software in HEP is typically divided into two abstraction levels: the **online** and the **offline**. The former denotes pieces of software which are used in data taking process, whereas the latter represents the ones which do run on the already collected data, independently of data taking procedure.

In the NA61/SHINE experiment, the calibration and reconstruction is done on the offline level. Performing those tasks on the online level would require investment of substantial resources, both in terms of development and of computing. Additionally, it would increase the complexity of whole readout system unnecessarily. However, there are experiments which did implement online calibration [110] and real-time offline reconstruction [111], providing a counterexample for the common practice.

Shine offline framework is an adaptation of offline software framework originally developed by the Pierre Auger Observatory collaboration [112]. The main idea is that the framework allows the collaborators to contribute algorithms in an encapsulated manner by means of so-called modules.

With those modules, one can sequence up instructions in order to build up a large variety of applications. The framework provides a simple XML-based mechanism of module management, file handling as well as an access to run and

time-dependent detector description. Additionally, a rich set of helper utilities are delivered, including coordinate-free geometry package etc.

The Shine framework, as compared to its original form when imported from the Pierre Auger Observatory, has been substantially extended, due to more complex needs in its new environment. Furthermore, the DSPACK framework used by the predecessor, the NA49 experiment, has been assimilated in it, in order to profit from already existing and heavily used reconstruction and simulation algorithms.

This approach was necessary, because during the start-up phase of the NA61/SHINE experiment heavily relied on that calibration and reconstruction software. Thus, making a *tabula rasa* was not a viable option, since the NA61/SHINE collaboration had to deliver physical results right from the start, and breaking backward compatibility with data collected by the NA49 experiment was not an option.

Incorporating the DSPACK framework was a big challenge [67], as it was written in a mixture of languages, namely in C, C++ and proprietary The Portland Group, Inc. (PGI) FORTRAN. Despite of these great difficulties, the project ended with a great success and the NA61/SHINE uses it now as the official framework.

In the new Shine framework, C++ language is used exclusively instead of combining multiple programming languages. The standard Standard Template Library (STL) data containers are used in order to homogenize the data structures. Furthermore, the event data is streamed using ROOT [113] to ensure flexible and independent input/output system, as well as because ROOT file format is a standard at CERN and HEP community in general.

In the event stream, different levels of details are realized by selective omission of data fields. The Shine framework comprises five principal parts:

- processing modules – which can be sequenced through instructions provided by an XML file
- an event data model – through which modules relay data to one another,

storing all the simulation and reconstruction information of a collision event

- a detector description – provides a gateway to data describing the configuration and conditions in the NA61/SHINE detector as a function of time or run number. For example, if a module requests a value of drift velocity for a current event, the detector description will find and fetch correct value matching the time, when the event was collected.

- a collection of utilities – consists of the most fundamental algorithms and data structures, including: linear algebra, statistics, unit system, coordinate-free geometry etc. These are inherited from the offline software framework of the Pierre Auger Observatory [112].

- legacy support – it incorporates DSPACK [114] framework inherited from the predecessor experiment NA49, including reconstruction and simulation algorithms

These ingredients are described in the following sections.

6.1 Processing Modules and Run Control

All tasks, such as those related to simulation and reconstruction, are factorized into sequences of self-contained processing steps. Normally, these are written by the physicists, and are called modules in the Shine terminology. These modules are registered into the the framework via the `REGISTER_MODULE` macro.

The macro defines a factory function used by the framework to instantiate the module when requested. The registry provides the safety mechanism, preventing the calling of modules not known to the framework. All modules inherit a common interface, presented on listing 6.1, which declares processing methods.

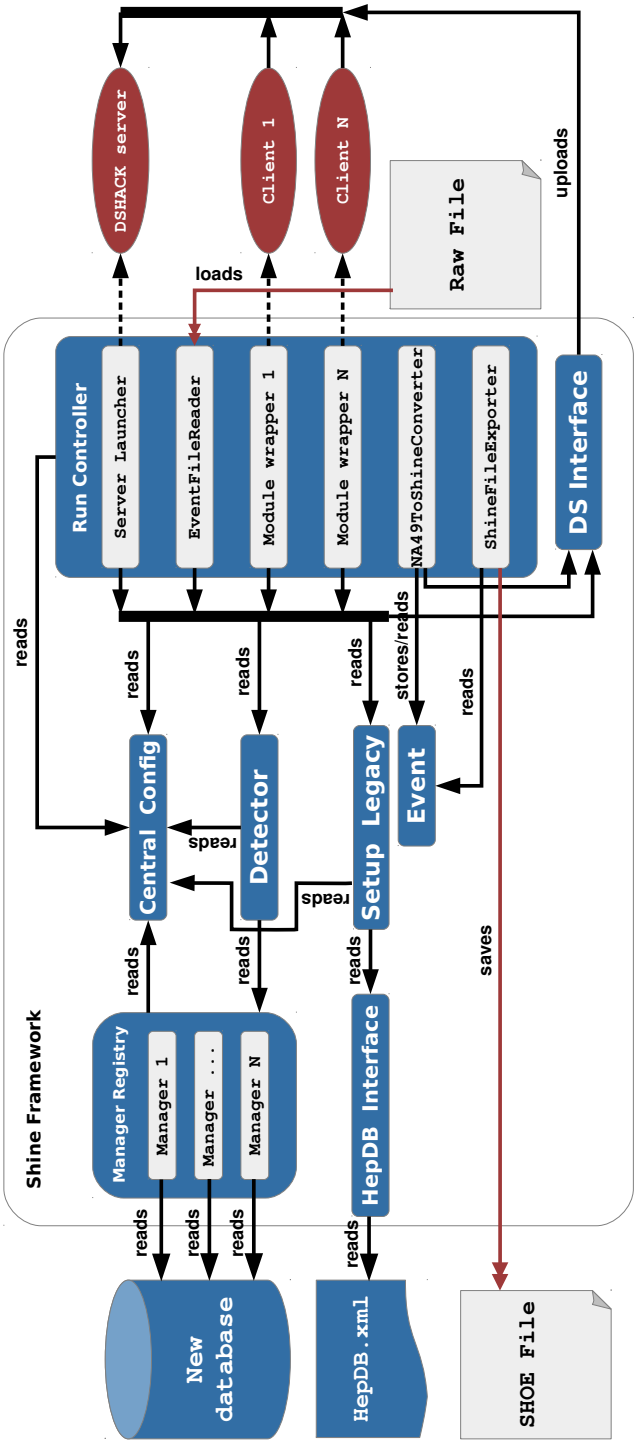


Figure 6.1: Pictorial diagram of communication between main objects within the Shine framework, including the legacy (NA49) support.

```
1  class FooBar : public fwk::VModule {
2    public:
3      FooBar();
4      ~FooBar();
5
6      fwk::VModule::EResultFlag Init();
7      fwk::VModule::EResultFlag Process(evt::Event& event,
           const utl::AttributeMap& attr);
8      fwk::VModule::EResultFlag Finish();
9
10   private:
11     REGISTER_MODULE("FooBar", FooBar,
12                     "Comment");
13 };
```

Listing 6.1: Class definition of an example module.

In principle, a module has to implement three methods:

- `Init` – initializes a module for the first run. It is called at the beginning of data processing.
- `Process` – defines processing algorithm. It is called once per event, and gets a handle to the actual event buffer.
- `Finish` – it is called to finalize processing. It is called at the end of data processing.

Each module is able to read information from the detector description and the event, and then to process the information and to write the results back into the event buffer.

Such modularity allows collaborators to exchange, test and compare algorithms easily, and to build up a wide variety of applications by combining modules in various sequences without substantial software modifications. An example sequence is defined by using an XML file as the one presented on listing 6.2.

```
<sequenceFile xsi:noNamespaceSchemaLocation=
  '[SCHEMAPATH]/ModuleSequence.xsd'>
  <moduleControl>
    <loop numTimes="unbounded">
      <module>  DSHACKServerLauncherSG </module>
      <module>  EventFileReaderSG      </module>
      <module>  ClientInitializerSG    </module>
      <try>
        <module config="VTPCs"> Dipt256NewModuleSG </module>
        <module config="MTPCs"> Dipt256NewModuleSG </module>
        <module config="GPC">   Dipt256NewModuleSG </module>
        ...
      </try>
      <module> V0FinderSG            </module>
      <module> PSDReconstructorSG    </module>
      <module> ProdQualityAssessment </module>
      <module> ShineFileExporterSG   </module>
    </loop>
  </moduleControl>
</sequenceFile>
```

Listing 6.2: Module sequence for an event reconstruction

An XML file contains only one element called `sequenceFile` with an `xsi:noNamespaceSchemaLocation` attribute, which specifies where the schema file resides. Schema files (XSD files) are used to validate XML, in other words, it

checks whether an XML file conforms to a certain format. The most important element contained by the `sequenceFile` element is `moduleControl`. It consist of the following elements:

- module - it defines a module to be called by Run Controller. Occurrence of this element is mandatory.
- loop - it defines number of iterations e.g. "5" over list of modules. Value "unbounded" causes that the loop is executed until a module returns break loop code e.g. due to the end of file. This element is not mandatory.
- try - if a module returns a failure code within this block, instead of termination, Run Controller continues to the next iteration. This element is optional.

So far, those elements were sufficient to implement every processing scenario required by the applications within the NA61/SHINE experiment.

6.2 Event Data Model

The most critical part of the Shine framework is a data structure of physics event. The structure has to preserve backward compatibility when slight modifications are introduced. It was a main principle when laying the standards of the Shine Offline Event (SHOE) file format. Large modifications, of course, still could break backward compatibility, and therefore, the Event class had to be modeled very carefully.

Simplified diagram of Event class is presented on fig. 6.2. The Event aggregates the following classes:

- EventHeader – holds information about type of the data, cardinal number of the conducted experiment, run number, time stamp etc.

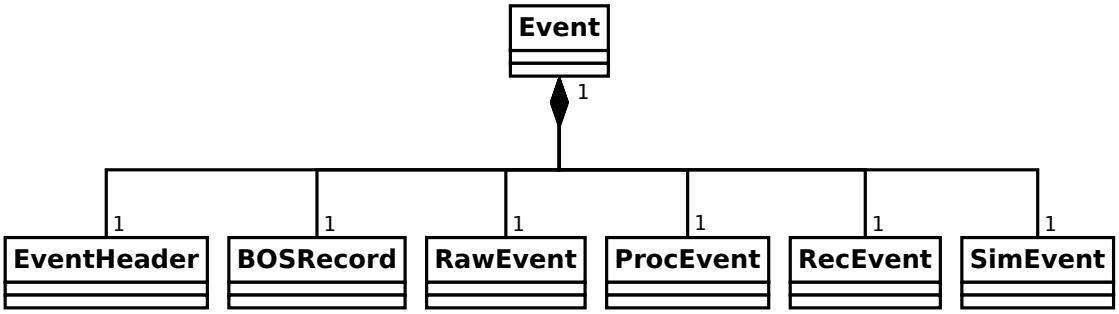


Figure 6.2: Simplified UML class diagram of the event model.

- BOSRecord – contains the raw binary data of an event in the form of so-called BOS banks, which literally contain unchanged data as it was received from the DAQ.
- RawEvent – common, more user friendly raw data format, to which BOSRecord is converted.
- ProcEvent – it is meant to be a place used by modules to store temporary data when processing an event.
- RecEvent – holds an event reconstructed by a sequence of modules.
- SimEvent – contains a simulated event. During analysis of simulated data, both RecEvent and SimEvent need to be present simultaneously.

Each module, as it is presented on listing 6.1, receives an Event to be processed. The typical case is that a module reads a part of RawEvent, processes and saves a result to RecEvent as well.

The fig. 6.1 illustrates an example sequence of modules. It can be seen that each module performs read access or write access on a single event. There are two modules dedicated to Input/Output operations, namely: EventFileReader and ShineFileExporter. The former one can read few formats, including the format of

former legacy (NA49) framework called DSPACK[114]. The latter one serializes the Event object only to the unified SHOE format, which was implemented using the ROOT [113] I/O possibilities.

6.3 Detector Description

The principal part of new Shine software framework is the detector description. It aims at modeling the experimental facility, providing a straightforward interface to all time-dependent or run dependent parameters of all detectors (fig. 6.3).

Therefore, its content (fig. 6.3) was implemented from scratch, since the NA61/SHINE experiment and the Pierre Auger Observatory apparatus differ substantially.

The detector related data consists of various parameters such as the setting of the magnetic field, configuration of the beam counters or the type and dimensions of the target. All these data are provided, upon a request, to any module via a singleton of the Detector class. The class is presented on fig. 6.3.

The Detector fetches the requested data using the Manager Registry. The Manager Registry handles set of so called Managers which can use various backends such as MySQL, Xerces or ROOT format.

When queried, the register iterates over the managers and requests the information from each one. When one of the managers returns the information, the register stops iterating. Each manager returns a flag to indicate success or failure in fulfilling the request. The overview of communication between objects within the Shine framework is illustrated on fig. 6.1.

The Detector queries the Central Config in order to get current configuration of the Manager Register, so that Detector can be initialized. Afterwards, the Manager Register is ready to serve data as a backend for the Detector.

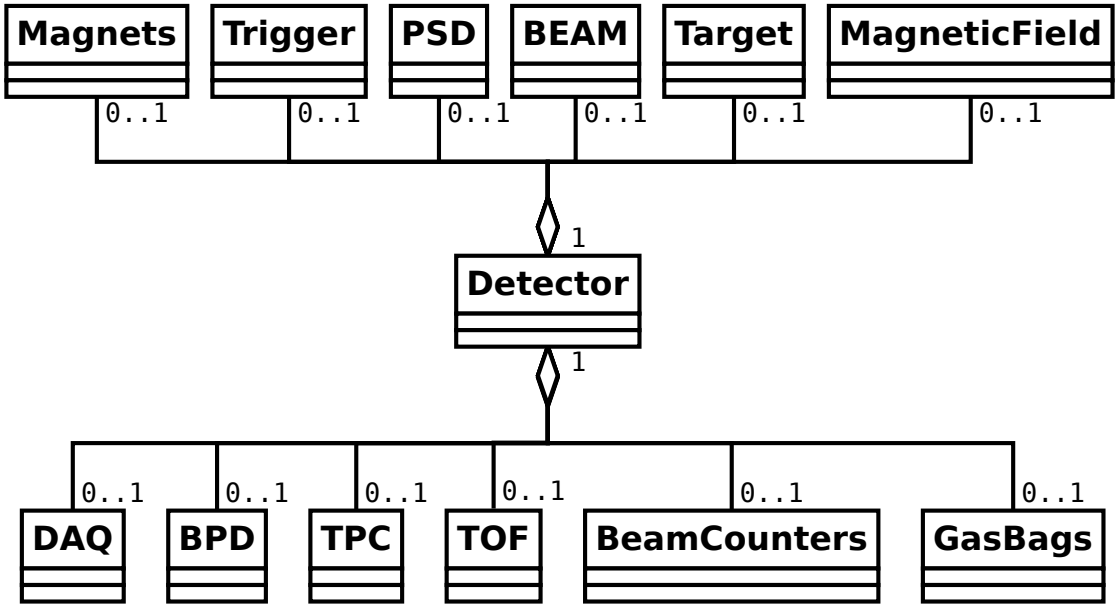


Figure 6.3: Simplified UML class diagram of the detector model.

6.4 Legacy Support

Parts of the NA61/SHINE offline software was inherited from the NA49 experiment. This was justified because the main tracking detector complex also comes from NA49. The core of the former software is a Persistent Object Manager called DSPACK [114, 115].

The DSPACK was developed in the early 1990s using a mixture of C and FORTRAN 77 language. Limitations of FORTRAN, such as variable name length (maximum 6 characters), polluted the whole software including parts written in C. Those limitations caused serious problems with code readability, and thus caused difficulties in maintaining and developing.

Another flaws of the inherited software are:

1. The reconstruction and simulation in the DSPACK framework were divided into so-called clients, which were stand-alone programs that read and wrote

data following a client-server model. Clients were managed by a large and complicated shell script, which used command-line arguments as well as environmental variables to pass settings to the clients. This approach caused considerable amount of human errors during reconstruction campaigns.

2. The legacy DSPACK based clients were written mostly in FORTRAN and C. The FORTRAN code was particularly difficult to maintain due to the non-standard dialect of the Portland Group's commercial compilers.

3. The data model in the legacy framework was highly rigid. Some of the inherited detectors, which are not used anymore, but they had to stay supported. Furthermore, introduction of new detectors was not foreseen during design of the DSPACK data structures and therefore some of the old data structures were used in order to accommodate new detector data, leading to large inconsistencies in the code.

4. Automatic self-tests were not used. All differences in the results were debugged manually, consuming abundant amounts of human resources. Also, without automated testing, the experiment was exposed to non-fatal but possibly significant bugs persisting in the code.

5. The documentation was scant and mostly out of date. Consequently, reverse engineering of the software was needed in order to extend the software with new algorithms.

In order to make the migration relatively smooth, module wrappers for legacy clients were developed to incorporate the old reconstruction clients within new framework. In a longer time perspective, those wrappers and clients are foreseen to be phased out by native Shine framework modules implementing new algorithms.

Besides the processing modules (clients) of the legacy software, the core of

DSPACK framework has also been assimilated into Shine legacy support, in order to profit from the possibility of reading in the data collected by predecessor – the NA49 experiment. A simple DSPACK interface was created to provide the communication medium between the old and the new software. This interface helps to convert event data easily from the old format into the new one, or vice verse.

Unfortunately, the effort to port FROTRAN code from PGI to GNU compiler in order to reuse the existing code, was significant. For example, in GNU FORTRAN there are no records and structures. Instead, nifty usage of type declaration can be used, namely `RECORD /item/` becomes `TYPE(item)` and `STRUCTURE /item/ ... END STRUCTURE` becomes `TYPE item ... END TYPE`. Furthermore, access operator `.` must be changed to `%`, e.g. `product.price = 3.15` to `product%price = 3.15`. In order to use pointers `-fcray-pointer` flag must be passed to a GNU compiler.

The next encountered problem was the impossibility to call `IARGC()` and `GETARG(i,s)` directly from the C code. Surprisingly, in GNU FORTRAN these functions were not implemented as library functions similarly to PGI FORTRAN, but as intrinsic functions. Two simple wrappers, presented on listing 6.3 and listing 6.4, have solved the problem.

```
integer function iargc_wrapper()
  implicit none
  intrinsic IARGC
  iargc_wrapper = IARGC()
  return
end
```

Listing 6.3: IARGC wrapper

```
subroutine getarg_wrapper( i, s )
  implicit none
  integer i
```

```
4     character(*) s
5     intrinsic GETARG
6     call GETARG(i,s)
7     return
8   end
```

Listing 6.4: GETARG wrapper

Noteworthy, neither FORTRAN nor C compilers do check for correct matching of the types of passed arguments. With the purpose of avoiding run-time errors, special care must be taken when arguments are handled.

Moreover, the detailed means of argument passing needs special attention. In FORTRAN, all parameters are passed by reference, whereas in C by value, if not indicated differently.

Furthermore, arrays passed as arguments provide additional problems since FORTRAN keeps arrays in column-major order and the C arrays are in row-major. On top of these nuisances, in FORTRAN array index starts with 1 and not with 0 as in C.

Passing strings is a particular problem, since strings in C are null terminated and in FORTRAN the length is declared and passed as a hidden argument which needs special handling on the C side of the code.

Additionally, GNU FORTRAN compiler turned out to be more restrictive than the PGI one. Features such as run-time array boundary check, revealed dozens of buffer overflow type bugs, which had to be fixed. The bug fixes had an impact on the data output, making further validation very complicated.

Unfortunately, beside above mentioned problems, The PGI FORTRAN also rounded floating point differently. Although the biggest calculated relative error for an algorithm used for trajectory reconstruction in DSPACK clients was equal to $5.9947e-08$ for double precision variables, some of them were ill-conditioned.

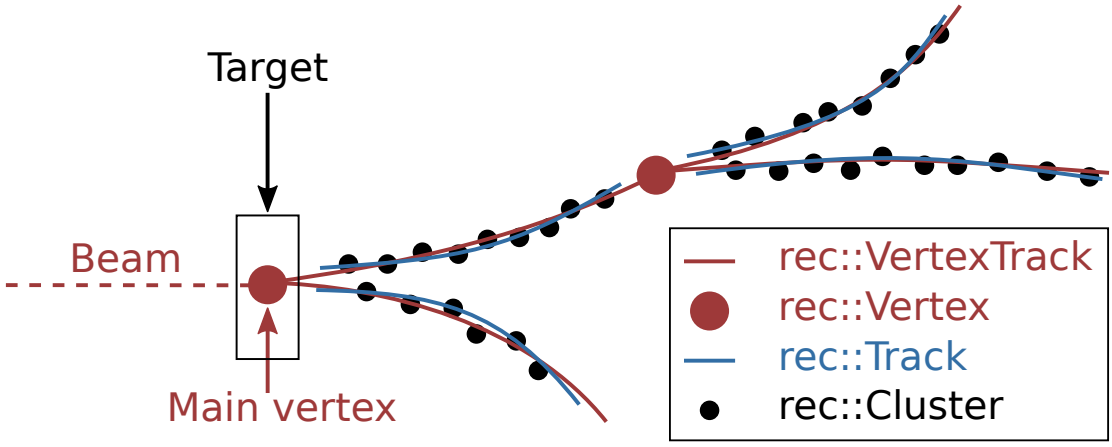


Figure 6.4: Relationship between classes aggregated into RecEvent.

Those algorithms were very sensitive to such small perturbation of the data, causing significant changes in physics results.

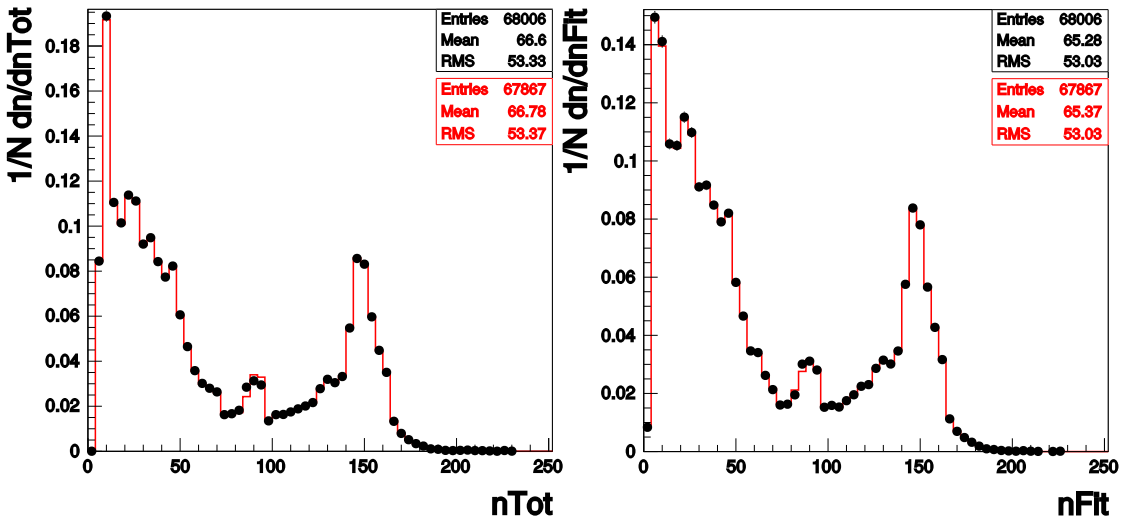
In the end, those algorithms were corrected and the physics data were validated. In the following section results of numerical validation are presented.

6.5 Numerical Validation

In order to determine if modification made to the code, during incorporating it to the Shine framework, had an impact on physics analysis, various control histograms have been created.

The most essential histograms comparing legacy software versus Shine framework (which runs ported and wrapped legacy algorithms), are those related to tracks and vertex tracks. A relationship between cluster, track, vertex and vertex track is depicted on fig. 6.4.

A cluster is an aggregation of measurements (ADC signals) from adjacent TPC readout pads. Using those measurements, the center of a cluster is calculated. The center is a tuple of space coordinates (X,Y,Z) used later as measurement point.



(a) Total number of measurement points assigned to a trajectory.

(b) Number of measurement points used for calculating trajectory parameters.

Figure 6.5: Normalized distributions of measurement points per trajectory between tracks reconstructed with legacy software (black dots) and Shine framework using wrapped clients (red line).

A track is a trajectory of a particle without its point of origin (vertex). A vertex track is a track which has been matched with a vertex. The vertex is a point in three dimensional space located typically in the target (called the main vertex) or located at the end of a track (called a secondary vertex, when a particle decays producing e.g. two offspring particles and consequently two offspring tracks).

The first comparison between legacy software and Shine framework is presented in fig. 6.5. First plot (fig. 6.5a) shows both distributions of number of clusters assigned to a track – there are no significant differences. Also no significant differences can be noticed in the number of clusters used in the track model fitting (fig. 6.5b).

Histograms presented on fig. 6.6 illustrate two essential parameters, namely momentum of a particle $p \equiv \sqrt{p_x^2 + p_y^2 + p_z^2}$ and its transverse momentum $p_T \equiv$

$\sqrt{p_x^2 + p_y^2}$ (i.e. the component of the momentum perpendicular to the beam axis). Momentum distributions depicted on fig. 6.6a and fig. 6.6c represent tracks and vertex tracks respectively. In fig. 6.6b and fig. 6.6d transverse momentum distributions for tracks and vertex tracks are shown respectively. It is seen that all distributions match perfectly.

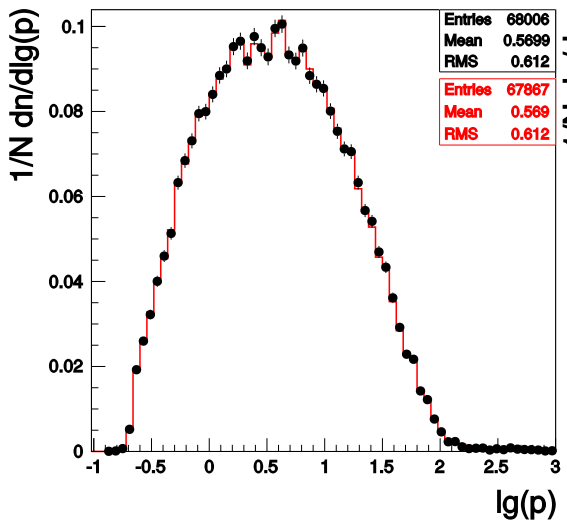
The last set of essential control histograms (fig. 6.7) is related to the main vertex. Plots fig. 6.7c and fig. 6.7d present histograms of X and Y components of main vertex position. The histograms presented on fig. 6.7a and fig. 6.7b illustrate X and Y components of difference between vertex position and a vertex track extrapolated to the vertex plane – it is called impact factor. In other words, impact factor is a distance at vertex plane, between a vertex track and its vertex. Those distribution imply that vertices were calculated correctly, simultaneously preserving the required match precision.

6.6 Summary

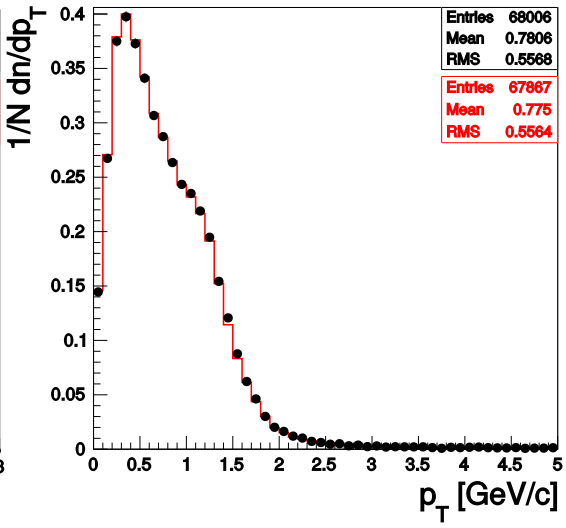
The enterprise of developing a new software framework for reconstruction, simulation and analysis purpose ended with great success. The final size of source code differentiated against the used languages in the project is presented in fig. 6.8. As it can be seen, development of the framework required knowledge of many languages.

All physics related elements were designed carefully. The NA61/SHINE detector has been modeled with the highest level of details, providing a handy tool to provide detector specific information such as geometry or calibration parameters at a given time. It provides an easy way to compensate detector effects without a need of expert knowledge.

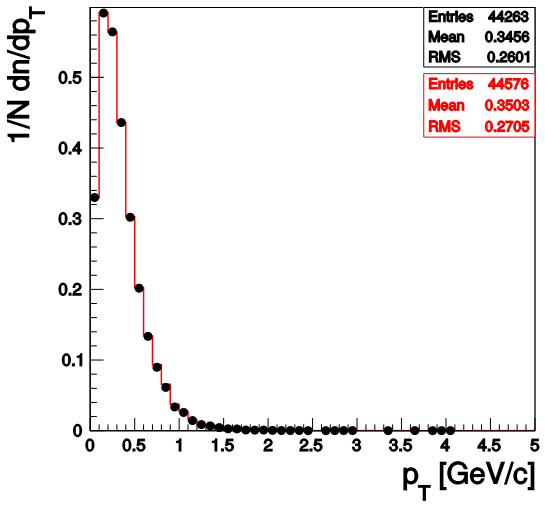
Additionally, an event structure has been designed, and together with the object



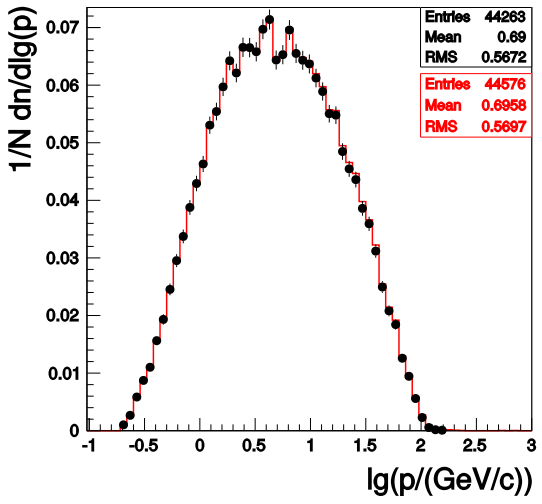
(a) Logarithmic momentum distribution of tracks.



(b) Transverse momentum distribution of tracks.

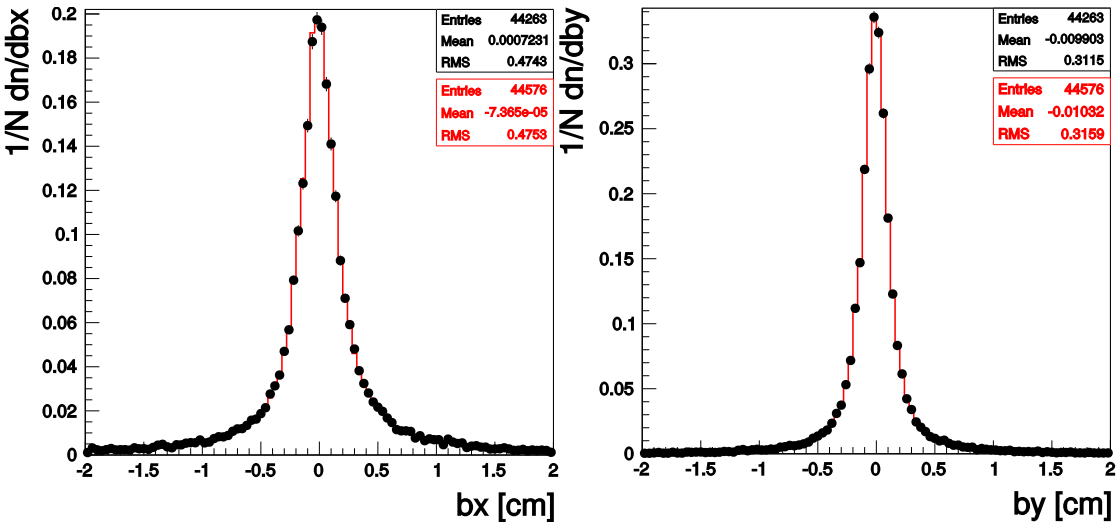


(c) Logarithmic momentum distribution of vertex tracks.



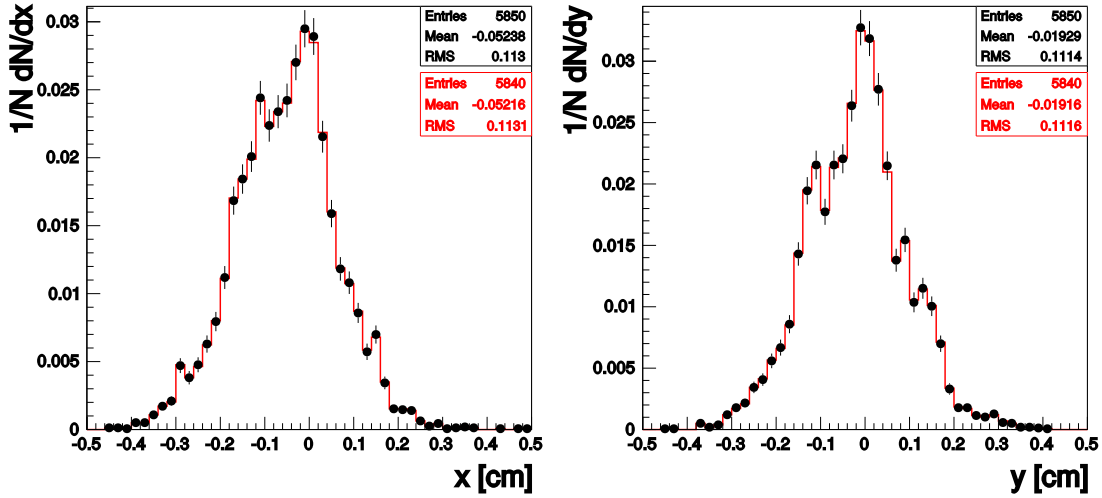
(d) Transverse momentum distribution of vertex tracks.

Figure 6.6: Normalized track and vertex track distributions. Black dots represent results obtained with legacy software and red line symbolize legacy algorithms ported into Shine framework.



(a) Distribution of impact factor: X component.

(b) Distribution of impact factor: Y component.



(c) Distribution of vertex position: X component.

(d) Distribution of vertex position: Y component.

Figure 6.7: Normalized distribution of vertex related parameters compared between results obtained with legacy software (black dots) and legacy algorithms incorporated into Shine framework (red line).

Language	Number of code lines
C++	114 796
XML	95 349
C	84 837
C/C++	46 553
FORTRAN	30 992
CMake	8 905
Total	381 432

(a) Number of lines of whole Shine framework source code, including legacy support.

Language	Number of code lines
C++	10 057
XML	1180
C	82 495
C/C++	16 685
FORTRAN	30 942
CMake	2 733
Total	144 092

(b) Number of lines of code related to the legacy support in Shine framework.

Figure 6.8: Number of code lines per programming/markup language.

streamer from the ROOT framework, it creates the SHOE data file format.

Moreover, the core as well as the most crucial components of the legacy software have been assimilated in a new framework, providing backward compatibility and access to old sparsely documented algorithms.

Creation of a new framework was a necessary step to develop new and more reliable event reconstruction algorithms. In the following chapters reconstruction algorithm using this framework is presented.

Chapter 7

CMA Evolution Strategy

This chapter is an introduction to Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which is a crucial component of the trajectory reconstruction algorithm, presented in the following chapter.

Evolution strategies are probabilistic optimization techniques and a special case of evolutionary algorithms that date back to the 1960s [116]. Evolution strategies are particularly well suited for nonlinear black-box optimization problems in continuous search spaces. Inspired by biological evolution, their original formulation is based on the application of recombination, mutation and selection in populations of candidate solutions. From the algorithmic point of view, evolution strategies are optimization methods that sample new candidate solutions stochastically, using for example a multivariate normal probability distribution. In general, all strategies follow an evolution loop, that consists of four steps:

- recombination – creates new solution vectors, also called offspring, from the parent population. Two major types of recombination, dominant and intermediate recombination, are typically distinguished. In dominant recombination, a property of a parent individual is inherited by the offspring, i.e.,

this property dominates the corresponding property of the other individuals. For intermediate recombination, the properties of all individuals are taken into account, such that, e.g., in the simplest case, their mean value is used.

- mutation – the mutation operator are parameterized, therefore it can change properties during optimization. This way, it provides the main source of variation of offspring in an evolution strategy. Based on sampling random variables, properties of individuals are modified and a new population is created.

- evaluation – the newly created individuals are then evaluated, i.e., their fitness values are calculated.

- selection – based on these fitness values, selection identifies a subset of individuals which form the new population which is used in the next iteration of the evolution loop.

The loop is terminated based on a termination criterion set by the user, such as reaching a maximum number of evaluations, reaching a target fitness value, or stagnation of the search process.

Novadays, many variations of Evolution Strategy (ES) exist, for example $(1+1)$-ES, $(1+\lambda)$-ES, $(\mu/\mu_w, \lambda)$-ES, $(\mu+\lambda)$-ES and more. Generally, the notation of variations are defined as follows:

$$(\mu/\rho, \kappa, \lambda) - ES \tag{7.1}$$

where: $\mu \in \mathbb{N}$ is a number of parent individuals; $\rho \in \mathbb{N}$,$\rho \leq \mu$ denotes the number of parents taken into account for generating of a single offspring by means of recombination; κ represents the largest age (number of generations) which can be reached by any individual in the population. Default value is 1.; λ denotes the

number of offspring individuals, size of next generation. The $(1+1)$-ES operates only on two vectors, the parent vector (current) and parent's mutation vector (child). Only if the mutant's fitness is at least as good as the parent one, it replaces the parent and becomes the parent of next generation. The notation $(\mu+\lambda)$-ES, means that not only one offspring is created per generation, but $\lambda >= 1$ descendants, and in order to keep the population size constant, the λ worst out of all $\mu+\lambda$ individuals are discarded. If the + sign is replaced by comma, (μ, λ)-ES, then the selection takes place among the λ offspring only. Their parents are discarded no matter how good their fitness was compared to that of the new generation. Obviously, this strategy relies on condition that $\lambda > \mu$.

The algorithm presented in this thesis is utilizing the ES called CMA-ES. The CMA-ES (**C**ovariance **M**atrix **A**daptation **E**volution **S**trategy) [8] [11] is a stochastic and derivative-free method for black-box optimization in continuous domain. It is considered as state-of-the-art in evolutionary computation and has been successfully used for difficult non-linear, non-convex and non-continues problems. The main steps can be distinguished in CMA-ES: Sampling; Mean Update; Covariance Matrix Update.

Mentioned three steps will be briefly described in the following sections. In this and consecutive chapters, the following notation is used: bold minuscule letter e.g. $\mathbf{v}$ denotes a column vector, bold majuscule letter e.g. $\mathbf{M}$ stands for matrix and a transposition is indicated by T upper index e.g. $\mathbf{v}^T$.

7.1 Sampling

This stage aims for creation of next population by generating the λ new solutions (offspring) by sampling a multivariate normal distribution according to mutation

operator. It is achieved using the following equation:

$$\mathbf{x}_i^{(g+1)} \sim \mathbf{m}^{(g)} + \sigma^{(g)}\mathcal{N}(\mathbf{0}, \mathbf{C}^{(g)}) \qquad \text{for } i = 1, \ldots, \lambda \tag{7.2}$$

where $\mathbf{x}_i^{(g+1)} \in \mathbb{R}^n$ denotes an i-th offspring of generation $g+1$, $\mathbf{m}^{(g)}$ and $\sigma^{(g)}$ are the mean value and step-size, respectively, and $\mathcal{N}(\mathbf{0}, \mathbf{C}^{(g)})$ is a multivariate normal distribution with zero mean and covariance matrix $\mathbf{C}^{(g)}$ at generation g. Typical recommendation of μ value is $\frac{\lambda}{4}$ [117]. The symbol $\sim$ denotes the same distribution on both sides. The next step is evaluation of their (offspring) fitness using a fitness function and to select μ candidates.

7.2 Mean Update

At this point, when a new generation is evaluated and μ candidates have been selected, all candidates undergo weighted recombination to became next parent (next mean value). The new mean $\mathbf{m}^{(g+1)}$ is calculated in the following manner:

$$\mathbf{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}^{(g+1)} \tag{7.3}$$

where $\mathbf{x}_{i:\lambda}^{(g+1)}$ stands for i-th best solution among generated offspring at $(g+1)$ generation, $\mu \leq \lambda$ denotes number of the best solutions to be used for mean estimation, and $w_{i=1\ldots\mu} \in \mathbb{R}_+$ are a positive weights and must satisfy the equation (7.4). The selection mechanism is implemented by choosing $\mu < \lambda$ as well as by different weights.

$$\sum_{i=1}^{\mu} w_i = 1, \qquad w_1 \geq \cdots \geq w_\mu \geq 0 \tag{7.4}$$

It is worth mentioning, that the equation (7.4) provides intermediate recombination feature whenever $\mu > 1$ and a selection mechanism by means of different weights assignment. However, work within this dissertation does not exploit assigning of differents weights, thus all weigths are equal $w_i = \frac{1}{\mu}$.

7.3 Covariance Matrix Update

The last step is the update of covariance matrix (7.5). In order to get reliable estimator for wide range of λ values, the combination of two adaptation methods are used: the rank-μ and the rank-one update. The former uses efficiently the information within large population, however it is constrained to one generation. The latter exploits correlation between generation steps by usage of evolution path, and allows lower number of offspring. Lower λ value may significantly increase a search speed, unfortunately the probability of finding a local (instead of global) minimum increases. The covariance matrix update step is defined as follows:

$$\begin{aligned}\mathbf{C}^{(g+1)} = (1 - c_1 - c_\mu)\mathbf{C}^{(g)} + c_1 \underbrace{\mathbf{p}_c^{(g+1)}(\mathbf{p}_c^{(g+1)})^T}_{\text{population rank-one update}} \\ + c_\mu \underbrace{\sum_{i=1}^{\mu} w_i \, \mathbf{y}_{i:\lambda}^{(g+1)}(\mathbf{y}_{i:\lambda}^{(g+1)})^T}_{\text{current generation rank-}\mu\text{ update}} ,\end{aligned} \tag{7.5}$$

where

$$\mathbf{y}_{i:\lambda}^{(g+1)} = \frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}$$

and $\mathbf{p}_c^{(g+1)}$ denotes evolution path at $(g+1)$ generation. The evolution path cumulates information between consecutive steps, the strategy takes over all generations. It is defined as a sum of steps with an exponential smoothing (7.6).

$$\mathbf{p}_c^{(g+1)} = (1 - c_c)\mathbf{p}_c^{(g)} + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}} \, \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{7.6}$$

where

$$\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} w_i^2 \right)^{-1}$$

μ_{eff} is an effective number of parents, sometimes paraphrased as variance effective selection mass. c_1, c_μ, c_c stand for learning rate of rank-one, rank-μ and evolution path, respectively. $1/c_c$ is the backward time horizon and contains roughly 63%

of the overall weight. In other words, it is used to fade out information from the older generations. Updated covariance matrix is ready for next sampling round.

7.4 Step Size Adaptation

The step size $\sigma^{(g)}$ provides an "overall scale" of the search distribution. It aims to make consecutive movements of the distribution mean in optimal direction. The step-size control effectively prevents premature convergence and ,in the same time, allows fast convergence to an optimum.

The covariance matrix adaptation increases or decreases the scale only in a single direction for each selected step or it decreases the scale by fading out old information. In order to retain optimal step length, the step-size update (7.7) is used:

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{(g+1)}\|}{\mathrm{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right) \tag{7.7}$$

The step length is controlled explicitly by dumping parameter d_σ on one hand, which scales change of magnitude of σ. On the other hand, the c_σ parameter ensures the optimal learning curve of the evolution path $\mathbf{p}_\sigma$. Therefore, Cumulative Step-size Adaptation (CSA) [118] is utilized (7.8). The CSA leads to nearly optimal step sizes improving convergence speed and global search capabilities at the same time [7]. Notice that if the length of path equals the expected value of Euclidean norm ($\mathrm{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|$), σ retains unchanged.

$$\mathbf{p}_\sigma^{(g+1)} = (1-c_\sigma)\mathbf{p}_\sigma^{(g)} + \sqrt{c_\sigma(2-c_\sigma)\mu_{\mathrm{eff}}}\,\mathbf{C}^{(g)^{-\frac{1}{2}}}\,\frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{7.8}$$

The evolution path (7.8) is similar to (7.6). However, in order to make step-size independent from direction, $\mathbf{C}^{(g)^{-\frac{1}{2}}}$ has been added. Therefore, in case of $\mathbf{p}_\sigma^{(g+1)} > \mathrm{E}\|\mathcal{N}(\mathbf{0},\mathbf{I}\|)$ the step size $\sigma^{(g+1)}$ increases (eq. (7.7)). Increasement of the

step size causes the elements evolution path vector $\mathbf{p}_c^{(g+1)}$ to decrease (eq. (7.6)). In the result a reduction of covariance matrix elements is performed (eq. (7.5)).

7.5 Time and Space Complexity

The space complexity of CMA-ES is straightforward, as it is dominated by covariance matrix. Thus the complexity is $\Theta(n^2)$, where n is number of model parameters to be optimized. On the other hand, the time complexity is a more complicated aspect. In this context, the most important parts of CMA-ES are:

- Factorization of Covariance matrix – the eigen decomposition of a covariance matrix is required in order to sample the multivariate normal distribution. This step has a complexity of $O(n^3)$. However, in many implementations (including the one used in this dissertation), this step is done every $n/10$ generation, thus the complexity becomes $\Theta(n^2)$ per generation. The disadvantage of this procedure is that a slightly outdated covariance matrix is used.

- Generation of candidate solutions (offspring, population) – sampling a multivariate normally distributed random vector has a complexity of $\Theta(n^2)$ because matrix vector multiplication must be performed.

- Update the search paths – involves matrix and vector multiplications as well, thus it is $\Theta(n^2)$.

- Update the covariance matrix – the complexity is $\Theta(n^2)$ as it is also dominated by matrix and vector multiplications.

In conclusion, the CMA-ES has several steps, which complexity is $\Theta(n^2)$. Noteworthy, when population size (λ) is greater than 20, the number of generations

needed to converge becomes practically independent of the population size[119]. Therefore, adaptation speed decreases in linear fashion when population size increases. In other words, the performance will not increase by simply increasing the population size.

Additionally, observed number of generations needed to solve several complex functions is proportional to n^2[119]. Nevertheless, it is crucial for time complexity of CMA-ES based methods to wisely choose an objective function, as it contributes greatly to overall performance.

It is worth to mention, that there exists a version of CMA-ES called sep-CMA-ES, which achieves linear time and space complexity for high dimensional objective functions[120]. However, it assumes that parameters (arguments of an objective function) are independent and an optimum can be obtained by optimizing each of n dimensions separately.

7.6 Summary

The CMA-ES is a state-of-the-art optimization routine, which combines classical deterministic concepts, such as Covariance Matrix Learning and statistical procedures (e.g. Principal Component Analysis (PCA)) with a stochastic method of ES. Although, there are some critics that the method is not a pure ES method due to incorporation of non-evolutionary components mentioned above, the CMA-ES is ranked very high [4, 5].

It is worth mentioning, that although the method is highly competitive Evolutionary Algorithm (EA) for local optimization, it also presents a good performance in searching for global optimum when restart mechanism is used[121]. Furthermore, the CMA-ES does not require a tedious parameter tuning for its application, what is so frequently needed when other methods are used.

Due to simplicity of CMA-ES method and wide range of problems it addresses, it should be considered as an alternative to quasi-Newton methods. Example problems solved by means of CMA-ES are presented in the following chapters.

More details about CMA-ES, can be found in [7] whereas a very comprehensive introduction is available as well [117].

Chapter 8

Event Reconstruction

In the previous chapters the main component of the experimental facility have been described so far. First component, the data acquisition system described in chapter 4, is responsible for digitization and collection of data. Due to enormous amount of data, a filtering system called trigger system (chapter 5) has been developed. Collected data is further processed by tools based on the Shine framework, which has been described in chapter 6. This chapter describes in details the subject of this thesis, namely a trajectory reconstruction algorithm. Before presenting the method, overview of event reconstruction process in the NA61/SHINE experiment will be outlined. It is necessary to comprehend the role of the algorithm in event reconstruction process. Furthermore, a brief introduction to the particle kinematics is given, as it is a formal model of a particle **trajectory** – also called a **track**. The results of performance studies are presented in chapter 9.

8.1 Overview

Processing of recorded data starts with calibration followed by event reconstruction process. The event reconstruction consists of the following stages: Clustering,

Local tracking, Global matching, Vertex finding. There are also additional stages, specific for a particular type of physics analysis. **Clustering** simply put, is a process of forming clusters by grouping in each pad-row all neighboring ADC signals (fig. 4.3) of values above a certain threshold. Afterwards, various parameters of these clusters are calculated, such as: center of gravity, average ADC value, total charge and other. Center of gravity is a 3-dimensional point in Euclidean space, **hereafter also called measurement, cluster position or just a point**. In the NA61/SHINE, the uncertainty of cluster position varies between 0.1 - 1.4 mm [122].

The aim of **Local tracking** is to find a group of clusters, which forms a trajectory. This is the most complicated task due to two-track resolution of about 1 cm in the NA61/SHINE detector[66]. In high multiplicity events, distance between trajectories can be shorter, resulting in merged clusters. Because of this reason, a project of developing local tracking software failed in the past. Therefore, currently used reconstruction software performs global tracking[1] with vertex position constrain for high multiplicity events [66, 123]. However, the algorithm presented in this thesis shows, that not only local tracking is possible in the NA61 TPCs but the efficiency of this process is very high (results presented in chapter 9).

In order to reconstruct whole trajectories one has to join local tracks form several TPCs in the process called **Global matching**. The process, extrapolates local tracks between two chosen TPCs chambers and then merges them properly into a single global trajectory, but without vertex yet.

After global tracks are reconstructed, a point of origin has to be calculated. This process is called **Vertex finding**. This is an important step, without which it would be difficult to determine if a trajectory belongs to a certain event or if

[1]Global tracking refers to reconstruction of trajectories using clusters from all TPC detectors simultaneously.

it is for example a trajectory of off-time particle[2]. This is especially crucial when using long target, which is typically a 1m long tube filled with gas.

The stages described above, are the base for **successive stages** which are different depending on a type of planned physics analysis, for example: calculation of dE/dx, finding decays and cascade decays.

8.2 Particle Motion

Recognition of particle trajectories in chambers without presence of magnetic field such as MTPCs, reduces the problem to finding straight lines. However, when magnetic field is applied (example field map is presented on fig. 8.1), the equation of motion has to incorporate the deflection caused by the Lorentz force F_l

$$F_l = \frac{d\mathbf{p}}{dt} = q \cdot (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \approx q \cdot \mathbf{v} \times \mathbf{B} \tag{8.1}$$

In case of the NA61/SHINE experiment, the strength of the electric field force ($\mathbf{E}$) in the TPCs is of the order of $\sim 2 \cdot 10^4 V/m$, whereas the magnetic field force ($\mathbf{B}$) is approximately $\sim 3 \cdot 10^8 V/m$. Therefore, neglecting electric force is justified. The elapsed time t of particle motion can be reparameterized using flight path length s as $dt = ds/v$, where $v = |\mathbf{v}|$ is the magnitude of velocity vector. Introducing $p = |\mathbf{p}|$ for the momentum magnitude, the unit vector $\mathbf{e} = \mathbf{v}/v = \mathbf{p}/p$ gives us the following equation

$$d\mathbf{e} = \frac{q}{p} \cdot \mathbf{e} \times \mathbf{B} \cdot ds \tag{8.2}$$

Afterwards, using a simple but tedious mathematical transformation, the path length s is reparameterized by detector coordinate z, see for instance [124]. This leads to a system of differential equations, where z is the running parameter. For

[2]particle which belongs to previous or next event

simplicity, the following extrapolation operator can be defined:

$$\mathcal{Q} : T \times \mathbb{R} \to T, \tag{8.3}$$

where T is a track parameters vector space. Thus, an outcome of the $\mathcal{Q}$, given a track parameter vector Θ_0 and destination z, results in new track parameters $\hat{\Theta}$:

$$\hat{\Theta} = \mathcal{Q}(\Theta_0, z) \tag{8.4}$$

Often instead of the above equation, an approximation is used, e.g. a helix equation. That is possible only in situations where the magnetic field is relatively homogeneous. However, to acquire a homogeneous magnetic field in practice is a challenging task, and therefore the helix equation may not always be a good approximation to the true solution of our differential equation. Nevertheless, the helix approximation may be handy for rough track parameter estimation.

8.3 Evolutionary Tracker

The evolutionary tracker is a recursive algorithm that uses a series of measurements and produces estimates of a trajectory parameters Θ. The measurement, also called cluster or simply point, are ordered according to an elapsing parameter such as time or z coordinate in our case.

The proposed algorithm consists of three principal components:

- Model of an event – a whole event is modeled using a discriminative model of a noise and a generative model of a particle trajectory. Competition between those models is defined in terms of decision rule, comparing posterior probabilities in order to classify a measurement.

- Trajectory creation – governs trajectory finding and initiation (using seeding technique) as well as measurement to track association. The measurement to

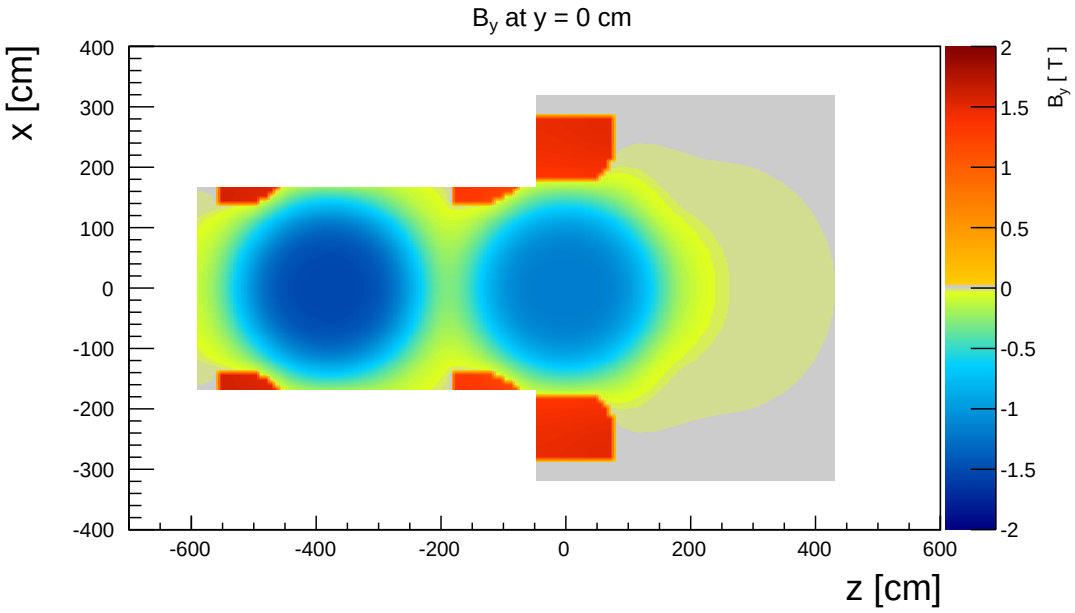


Figure 8.1: One of the magnetic field maps used in the NA61/SHINE experiment. The VTPCs chambers are localized withing Blue circular regions. Red spots denote reverse magnetic field vector and they are caused by support elements of superconductive magnets made of steel.

track association involves two stages: prediction and measurement update. A prediction is made by extrapolating a track parameter vector with equation (8.3) to a Z position of a next measurement, namely a cluster, in order to compare prediction with the measurement. If measurement fits prediction, it will be attached to a trajectory. Afterwards, during measurement update stage, the parameters of a track are re-estimated.

- Continuous track parameter optimization – after a cluster is attached to a track, the parameters are corrected by means of the CMA-ES.

8.3.1 Model of an Event

The event model consists of the two competing models, the generative model of a track and discriminative model of a noise. The aim of the competition is an efficient method for distributing valid clusters between tracks and noise. It compares calculated probabilities, using mentioned models, in order to make a decision. In the following section both models are described.

Generative Model of a Trajectory

The generative trajectory model, also called the Naïve Bayes, assumes all parameters to be independent and identically distributed. Interestingly, the Naïve Bayes demonstrates good performance in practice, also for problems which violate the assumption of statistical independence[125]. The explanation of this phenomenon can be found in [126], which shows that the optimality of Naïve Bayes approach does not depend on the independence attribute and the applicability is much greater than the original restrictive assumptions would suggest.

The main reason of choosing a generative model is the fact that uncertainty of a track cluster position is well known in the NA61/SHINE experiment, hence there is no need to learn likelihood from scarce labeled data. With constant arbitrary likelihood parameters, only learning a prior probability distribution is required.

Because of arbitrary likelihood function, the Naïve Bayes model was chosen although the the logistic regression is expected to overtake the performance of the Naïve Bayes method as the number of training samples increases over time [127]. The particle trajectory model is described as:

$$\overbrace{P(\hat{\boldsymbol{\Theta}}|\mathcal{S}(\mathbf{c}))}^{\text{Posterior probability}} \propto \underbrace{P(\hat{\boldsymbol{\Theta}})}_{\text{Prior probability}} \overbrace{P(\mathcal{S}(\mathbf{c})|\hat{\boldsymbol{\Theta}})}^{\text{Likelihood}} \tag{8.5}$$

where $\mathbf{c} = (x, y, z, s_1, \ldots, s_n)$ stands for a cluster with center of gravity located

at (x, y, z) along with charge deposition ADC signals $(s_1, \ldots, s_n)$ in the cluster. The symbol $\hat{\mathbf{\Theta}}$ denotes an estimation for a track parameter vector $\mathbf{\Theta} = p \oplus o$, where $p = (p_x, p_y, p_z)$ denotes the particle momentum vector at a starting point $o = (x, y, z)$ and $\oplus$ denotes concatenation operator $(dim(\mathbf{\Theta}) = dim(p) + dim(o))$. The operator $\mathcal{S} : T \cup C \to \mathbb{R}^3$ produces a three dimensional Euclidean space vector:

$$\mathcal{S}(\xi) = (\xi_x, \xi_y, \xi_z) \tag{8.6}$$

where C is a cluster parameter vector space and T is track parameter vector space. The model (8.5) consists of two components: likelihood function of a cluster, given a track; and an *a priori* probability of a particular track parameter vector. The likelihood is defined in the following manner:

$$P(\mathcal{S}(\mathbf{c})|\hat{\mathbf{\Theta}}) = P(\mathcal{S}(\mathbf{c})|\hat{\mathbf{\Theta}}, \Sigma) \sim \mathcal{N}(\hat{\mathbf{\Theta}}, \Sigma) \tag{8.7}$$

where " $\sim$ " denotes equality in distribution, $\hat{\mathbf{\Theta}}$ is an estimate of parameters $\mathbf{\Theta}$ being extrapolated by equation (8.3) to a position $\mathbf{c}_z = \hat{\mathbf{\Theta}}_z$ and $\mathbf{\Sigma}$ is a diagonal covariance matrix. The parameters of the likelihood function are contained in the diagonal covariance matrix

$$\mathbf{\Sigma} = diag\{\sigma_x^2, \sigma_y^2, \sigma_z^2\} \tag{8.8}$$

Note that the value of σ_z^2 is irrelevant as z position of a cluster is not a stochastic variable, but is determined by the padrow location.

The prior probability is an empirical distribution learned from simulated data (see section 9.1) using a maximum likelihood estimator. In order to avoid under-training, the Kolmogorov-Smirnov test was used on consecutive distribution updates until it reaches the significance level of $\alpha = 0.001$. Histograms were used in order to reduce the memory footprint of probability distributions. The empirical prior probability was chosen instead of a conjugate prior, as it gives great possibility to catch every nuance, what is hard to achieve otherwise.

The fact, that the training data are simulated provided an opportunity for a wide range of studies, such as selection of desired particle charge or tracks with desired momentum. The Bayesian approach makes it possible to consider some patterns more likely over others, and opens new possibilities for new techniques in physics analysis.

Discriminative Model of a Detector Noise

On the other side of the competition, the discriminative model is used to describe the background and detector noise. The discriminative model was chosen, because it is simple and intuitive, yet it tends to perform better than generative model as it was mentioned in the previous section. Furthermore, no prior knowledge about detector noise was available.

It is a simple one-dimensional model[3] which uses only one single feature of clusters, namely maximal value of ADC signal within a cluster. For the purpose of model training, a maximum likelihood estimator on labeled real data has been used.

The probability distribution can be observed on fig. 8.2. Note that cluster with maximum ADC equal to 255 is very likely to be a noise as it indicates a saturation in the electronics. Therefore, the final posterior probability is given

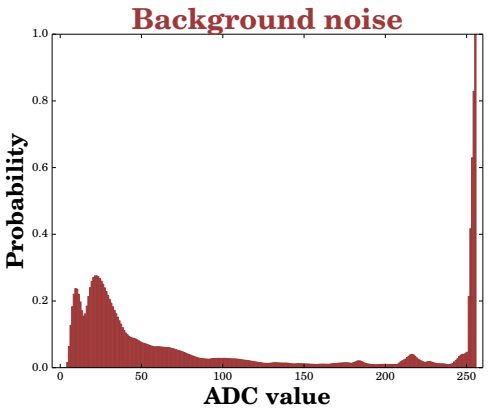


Figure 8.2: Probability distribution of background and detector noise P_N, using maximum ADC feature value of a cluster.

[3]Study of more precise model is planned

by

$$P(\mathbf{\Omega}|\mathbf{c}) = P_N(\max(\mathcal{R}(\mathbf{c}))), \tag{8.9}$$

with operator from cluster to ADC signal vector space $\mathcal{R} : C \to A$

$$\mathcal{R}(\mathbf{c}) = (\mathbf{c}_{s_1}, \ldots, \mathbf{c}_{s_n}) \tag{8.10}$$

which returns a vector of n ADC signals of a cluster of dimension n. The $\mathbf{\Omega}$ denotes a noise class and P_N is the probability distribution of maximum ADC value of a noise cluster. In order to avoid under-training, the Kolmogorov-Smirnov test was used on consecutive distribution updates until it reached the significance level of $\alpha = 0.001$. Despite of its simplicity, this noise model works very well as a competitor to track model.

The model reliably prevents attaching noise cluster to a track, that does not very likely belong to any track because of its unlikely signal amplitude. Therefore it reduces the number of outlier clusters on track candidates as it can be seen in the following chapter on fig. 9.5.

8.3.2 Trajectory Creation

The trajectory creation consists of two steps: seeding and data association. The first represents a greedy algorithm for searching and initializing tracks consisting of several clusters. In other words it determines if the group of clusters represents a trajectory or not. The second step of the trajectory creation governs competition between trajectory and noise model in order to classify a particular measurement as a part of trajectory or as a noise.

Trajectory seeding

The initial step of track creation is called seeding and a new track candidate object without estimated parameters is called a seed. During this stage, the seed

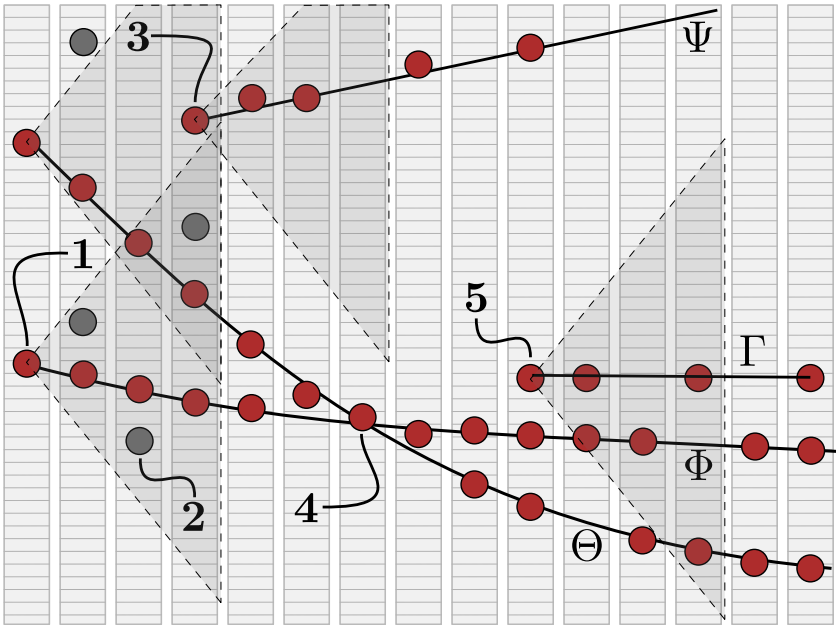


Figure 8.3: Illustration of the data association decisions made in a typical situation. The clusters (red circles) are processed from the left (downstream) to the right (upstream, towards target) pad-row by pad-row. There are five situation marked: 1) Failed to find a suitable track, thus a new seed is created before classification as a noise. 2) $P(\Omega|\mathbf{c}) < P(\Theta|\mathbf{c})$. It is not the most probable candidate. It will become a new seed and later classified as a noise. 3) $P(\Omega|\mathbf{c}) > P(\Theta|\mathbf{c})$, therefore being noise is more likely, nevertheless it gets a chance as a seed, to form a track. In the end it will become a track Φ. 4) Looks for the most probable trajectory, choosing between Φ and Θ. 5) Being noise wins, so a seed is created and it will be transformed to a track Γ.

collects all clusters within the search region (in this case a cone of 4 padrows height), which do not belong to an already existing track, but may share the cluster with other seeds. The overall mechanism of track competition is illustrated on fig. 8.3.

As we follow the particles of interest in the detector in the upstream direction (towards target – see fig. 2.3), it is possible to design a cone which will reduce the

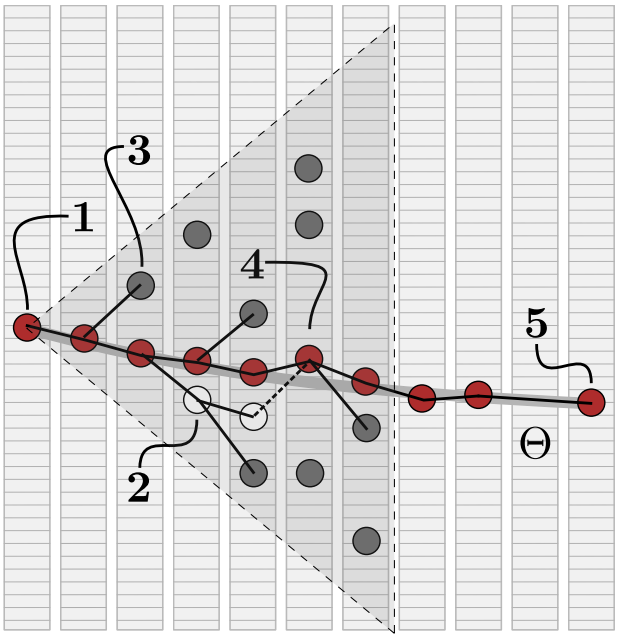


Figure 8.4: Illustration of the breadth first search performed on a seed (within the gray cone). 1) The root of a seed tree. 2) $P(\Omega|\mathbf{c}) < P(\Theta|\mathbf{c})$. It is not the most suitable candidate, as it is less probable. It will not became a seed, to avoid concurrent trajectory. However, it might be associated during second iteration. 3) $P(\Omega|\mathbf{c}) > P(\Theta|\mathbf{c})$. Classified as a noise, it becomes a seed. 4) Good track cluster. 5) The track is formed, thus a breath search is not performed. A new cluster is attached to the track.

measurement search space. In this case, the cone is defined as follows:

$$tg(\beta) \geq \frac{\sqrt{(\mathbf{c}_x^n - \mathbf{c}_x)^2 + (\mathbf{c}_y^n - \mathbf{c}_y)^2}}{|\mathbf{c}_z^n - \mathbf{c}_z|} \tag{8.11}$$

and

$$d \geq |\mathbf{c}_z^n - \mathbf{c}_z| \tag{8.12}$$

where β is a cone generatrix angle, $\mathbf{c}^n$ is a last (with highest z value) measurement of a track and d is a maximum acceptable gap within a track – distance in cm without cluster. Collected clusters within the cone can be seen as a tree as illustrated in fig. 8.4. The breadth first search algorithm along with a prior probability distribution is used to find the most likely branch of the tree, that is to become a new track with the parameters estimated using the clusters solely from this branch.

Algorithm 1 Probabilistic data association

```
1: procedure PDA(cluster)
2:     for all tracks do                          ▷ Including mature seeds
3:         probabilities[track] ← P(track | cluster)
4:     end for
5:     bestTrack ← MAX(probabilities)
6:     if probabilities[bestTrack] <= P(Ω | cluster) then
7:         RETURN(False)          ▷ Check a premature seed or create a new seed
8:     end if
9:     ASSOCIATEWITHTRACK(bestTrack, cluster)
10:    OPTIMIZEPARAMETERS(bestTrack)
11:    RETURN(True)
12: end procedure
```

The remaining clusters are then released. The seed becomes a track candidate when the estimation of its parameters becomes possible, namely, when the following equation is satisfied

$$h \geq dim(\mathbf{p}) \tag{8.13}$$

where h denotes level of a seed tree. Otherwise, without parameters, competition between tracks would not be possible. As mentioned above, the seed may accept and share all clusters which were not associated with a track.

In case when (8.13) is satisfied, the track parameters are estimated for a path from a leaf (the newest cluster) until the tree root. Afterwards the estimated parameters from the most probable branch is chosen. A detailed data association algorithm is provided in the form of pseudo-code (see algorithm 1).

In the end, a second iteration over clusters is performed for reassociation of

clusters improperly classified to the noise using tracks found in a previous trial. In this stage, new tracks are **not** created.

Measurement Association

Although all measurements are available at the same time, the algorithm proceeds using clusters ordered along the z coordinate which can be treated as a running time. To be precise, it processes the clusters in opposite directions to natural time elapse, namely towards origin – the target (see fig. 2.3).

This approach is justified by the fact that the density of tracks drops along the detectors in the downstream direction – away from the target. Therefore the algorithm starts from the regions of smallest density, increasing chances for success as the track separation is better.

The reason of the lower particle trajectory density in the downstream region is the initial opening angles between tracks, accompanied by the spreading effect of the magnetic field. The decision rule function $\delta(\mathbf{c})$, which governs measurement association to a particular track/seed or to noise, is defined in the following manner:

$$\delta(\mathbf{c}) = \begin{cases} \vartheta \in T, & \text{if } p(\mathbf{c}) \geq 1 \\ \boldsymbol{\Omega}, & \text{otherwise} \end{cases} \tag{8.14}$$

with the posterior probability ratio

$$p(\mathbf{c}) = \frac{\arg\max_{\vartheta \in T} P(\vartheta|\mathbf{c})}{P(\boldsymbol{\Omega}|\mathbf{c})} \tag{8.15}$$

where T denotes a set of track parameter vectors. Association to a noise implies a new track creation as the noise cluster may belong to another, yet undiscovered track.

8.3.3 Continuous Parameters Optimization

Association (see previous section section 8.3.2) of a new cluster naturally provides additional information, so a better trajectory estimate can be achieved. Therefore, every new assignment yields parameter optimization as shown at line 10 of algorithm 1. In order to produce a new estimate of Θ the CMA-ES[8, 11] algorithm has been chosen.

As it is mentioned in chapter 7, it is a stochastic and derivative-free evolutionary algorithm which allows the method to work whereas Quasi-Newton methods fail. The chosen approach is regarded to be robust for a non-convex functions which can comprise discontinuities, spikes or being even ill-conditioned.

Furthermore, the CMA-ES [8] in particularly is useful for solving "black box" scenarios, where the knowledge about the underlying function is limited or an algorithm should not depend on that knowledge. The latter feature is very useful when the model significantly changes: severe modification to the algorithm is not needed, because of this property. In brief (see chapter 7 for details), the CMA-ES consists of three main steps:

- offspring generation – new solutions (offspring) are generated by sampling a multivariate normal distribution;
- selection and recombination – for each offspring an objective function is evaluated;
- adapting a covariance matrix and a mean base on results of evaluation

repeating above listed steps should converge to an local or global optimum, depending on an objective function and CMA-ES parameters. For the algorithm to work, only an objective function $f : \mathbb{R}^n \to \mathbb{R}$ is required to be defined. The evolutionary

tracker uses an objective function, defined in the following manner:

$$f = \sum_{i=1}^{n} Z_i, \tag{8.16}$$

for a track with n clusters and with

$$Z_i = diag\{\mathbf{d}\}^2 \cdot \Sigma^{-1}, \tag{8.17}$$

where $\mathbf{\Sigma}$ is a diagonal covariance matrix representing cluster position uncertainty (eq. (8.8)) and distance vector $\mathbf{d}$ is defined as follows:

$$\mathbf{d} = \mathcal{S}(\mathcal{Q}(\hat{\Theta}, c_z^i)) - \mathcal{S}(c^i) \tag{8.18}$$

where c_z^i is a z position of i-th cluster (c^i) of a track $\hat{\Theta}$, $\mathcal{Q}(\cdot,\cdot)$ stands for an extrapolation operator (eq. (8.4)) and $\mathcal{S}(\cdot)$ operator transforms a cluster or track parameter vector into Euclidean space vector (eq. (8.6)).

In order to accelerate convergence, an improved method of CMA-ES called Active-CMA-ES (aCMA-ES) [9] was used. The aCMA-ES differs from the original method by exploiting information from unsuccessful offspring and not only from successful ones. The information is used in order to reduce variance in unpromising directions. The authors demonstrate that an algorithm is superior in performance compared to Original-CMA-ES [8]. The highest performance, more than 40%, can be observed in case when the eigenvalue of a particular objective function dominates the others.

In the following chapter, description of testing procedure along with performance result of the evolutionary tracker are presented.

8.4 Time and Space Complexity

The space complexity mostly depends on number of measurements (clusters), as an average number of clusters per trajectory is roughly constant, more clusters

means more trajectories. The change in number of noise clusters will just affect coefficient of space complexity, which for asymptotic analysis is not relevant. Furthermore, as the number of parameters is constant, the space complexity of CMA-ES method (section 7.5) is constant as well.

The analysis of time complexity must take into account the following parts:

- Seeding (track finding) – for many of clusters a seed will be created, thus the complexity is $O(n_c)$, where n_c is the number of clusters within an event. Track searching within a cone is neglected, as the cones do not change and the number of clusters does not vary significantly, thus it can be regarded as constant.

- Measurement assignment (track building) – the measurement association decision rule (eq. (8.14)) is performed for each clusters and each already initialized trajectory, thus it is $O(n_c n_t)$, where n_c is the number of clusters within an event and n_t is number of tracks already initialized. Because $\exists a : an_t = n_c$, the final complexity is $O(n_c^2)$

- Optimization (CMA-ES) – as explained in section 7.5, time and space complexity is $O(n^2)$, where n is number of track parameters. The optimization is done for almost all cluster of all tracks. Each cluster assigned to a trajectory triggers optimization procedure. Consequently, the overall complexity is $O(n_c n^2)$. However, as the number of parameters is constant, the final time complexity is reduced to $O(n_c)$, where n_c is the number of non-noise clusters (originating from a trajectory) within an event.

- Fitness/Error function – uses extrapolation function to extrapolate track parameters from the first cluster to other clusters belonging to the track and calculates the squared distance. This procedure is $O(n_{ct})$, where n_{ct} is a number of clusters belonging to a trajectory. The extrapolation function is

implemented using Runge-Kutta fourth-order (RK4) method, which complexity is $O(s)$ where s is number of steps ($s = (x - x_0)/h$, where x is destination point, x_0 is point of origin and h is step size of RK4 method). The number of steps between consecutive clusters is nearly constant and is approximately equal to the length of a padrow (fig. 2.4). Therefore the complexity of the extrapolation function is constant. However, the fitness function is called for every trajectory each time a cluster is attached. The number of clusters originating from a trajectory is proportional to all clusters within an event (it is a sum of trajectory clusters and noise clusters), thus the final complexity is just $O(n_c)$. Unfortunately, the complexity coefficient is of order of 10^3, what significantly affects overall performance performance.

Summarizing, in evolutionary tracker algorithm, the worst time complexity has the part related to measurement assignment. Therefore, the overall time complexity in respect of number of clusters is $O(n_c^2)$. However, because very high complexity coefficient of fitness function, execution time of this part dominates over measurement assignment roughly by a factor of 50-100 (it varies from event to event) for the high multiplicity events. Therefore, further studies of fitness function are planned.

Chapter 9

Validation and Performance Studies

In this chapter the results of performance studies are presented. Preparation of input test data as well as the quality assessment technique are defined. The algorithm has been tested on VTPCs which are immersed in magnetic field as the one presented on fig. 8.1.

9.1 Simulation of Test Data

In order to study the efficiency and correctness of the algorithm, simulated data where prepared. For event generating purpose, the flat phase space generator was used, which produces necessary input for the GEANT toolkit[128]. As a result of this process, an electronic response of the TPCs readout cards was simulated, resulting in raw data. Afterwards, real detector noise clusters were superimposed on the simulated raw data in order to imitate real conditions. Starting from this point, the algorithm can run on simulated data in the same manner as on real, experimental data. An example of simulated event with real detector noise is shown

on fig. 9.1.

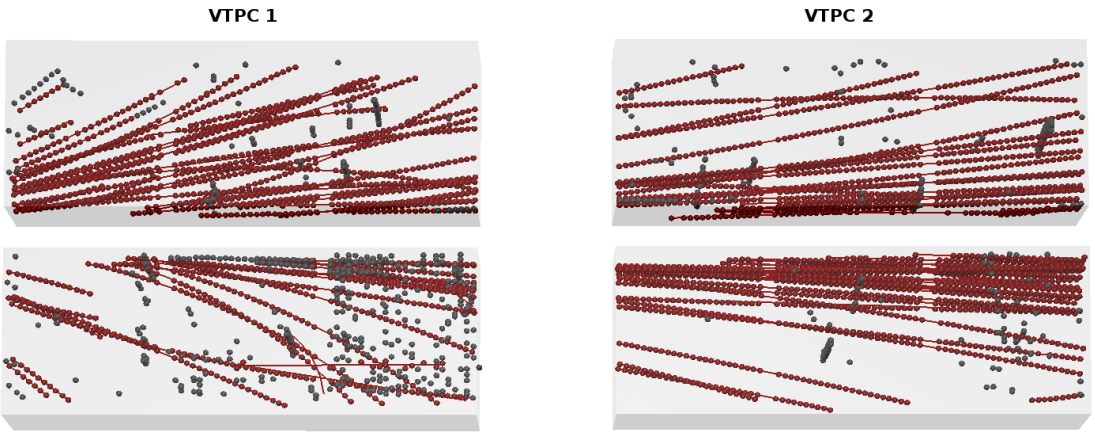


Figure 9.1: An example of simulated event of vertex TPCs (fig. 2.3 and fig. 2.4) with superimposed real detector noise, reconstructed using the algorithm presented later in this chapter. Black points denotes those classified as background.

9.2 Evaluation Procedure

The reconstruction efficiency of the new algorithm is assessed by comparing results between reconstructed tracks and simulated tracks from which they originate. The procedure is called matching and it is a standard method developed within NA61/SHINE collaboration. It searches for correlations between reconstructed and simulated track points. In order to make a decision on whether a reconstructed track point correspond to a simulated track point, a maximum distance is defined. Distance of 0.25 cm was used, which is large enough to ensure matching despite possible small distortions caused by detector effects. Furthermore, two tracks are considered to match, when ratio of matched to simulated points is greater than 0.8, but lower or equal to 1.0. In this procedure, multiple matches may occur. The ambiguities are handled by choosing the closest simulated point.

9.2.1 Reconstruction Performance

The results of the reconstruction efficiency study are shown on fig. 9.2, fig. 9.3 and fig. 9.4. The rapidity parameter denotes rapidity understood in the center of mass system with the assumption of pion mass for the produced particle.

As mentioned previously, this algorithm is applied only for the VTPC detectors, immersed in magnetic field. The shape of histograms is determined by the detector acceptance, therefore it contributes to a significant efficiency drop at the edges. Furthermore, for high multiplicity events efficiency losses occur due to the finite two-track resolution of the detector. These situations can result in merged tracks, which contribute to the fake track rate.

The fake track rates shown in table 9.1 as well as goodness of fit presented on fig. 9.5, indicate that the presented method is also robust in terms of forming fake tracks using noise clusters. Within the detector acceptance, the algorithm seems to perform remarkably well, especially that pure local tracking was considered to be unachievable in chambers with the highest track density [123].

Table 9.1: Summary of fake track rates [%]

TPC	Low multiplicity	Medium multiplicity	High multiplicity
VTPC 1	0.0084	0.0042	0.0018
VTPC 2	0.0077	0.0038	0.0033

9.3 Possible Improvements

The most complicated and error prone part of algorithm implementation for local tracking is seeding. It is used to determine number of tracks and to calculate initial parameters to be minimized, momentum of a particle and start position.

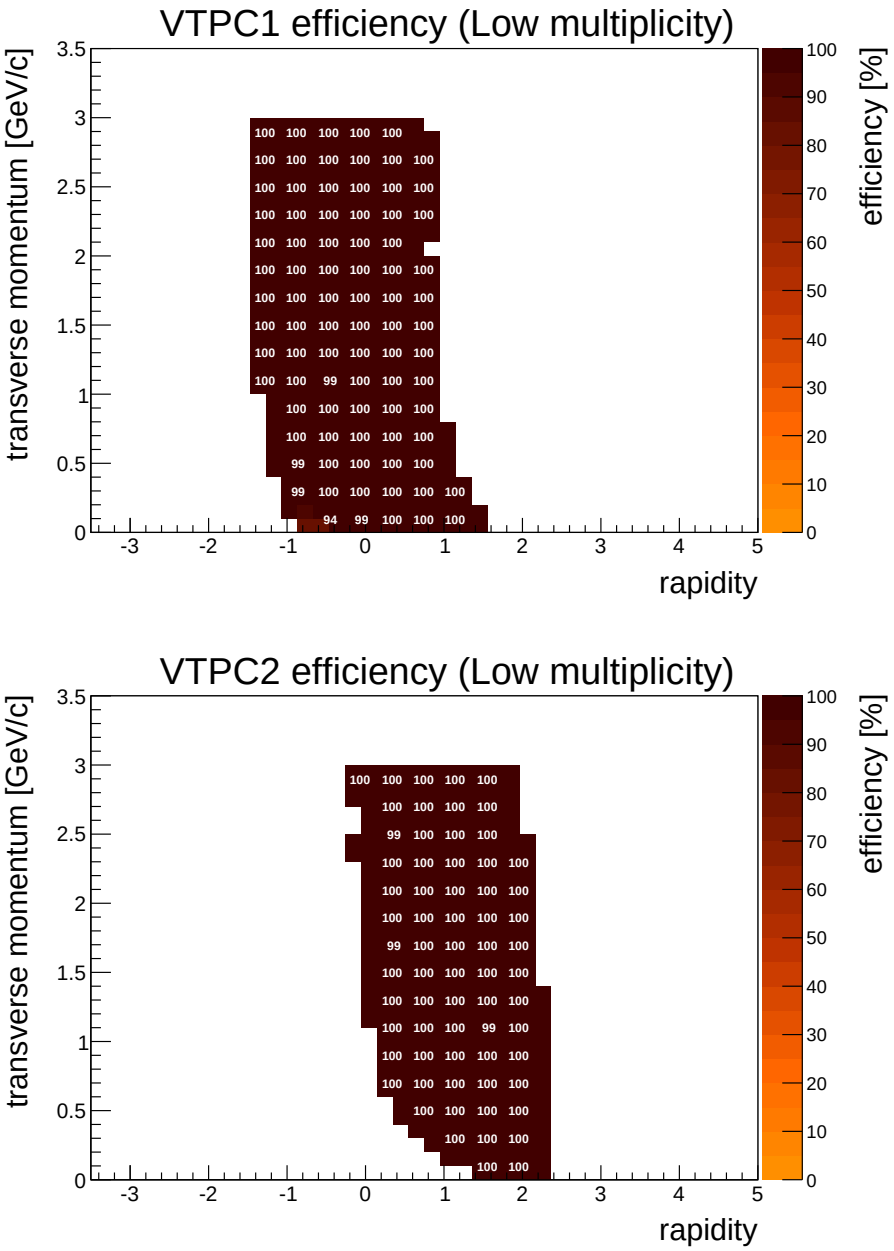


Figure 9.2: Efficiency for VTPC1 (top) and VTPC2 (bottom) chamber for low multiplicity events equal to 11.5 particles per event per unit rapidity.

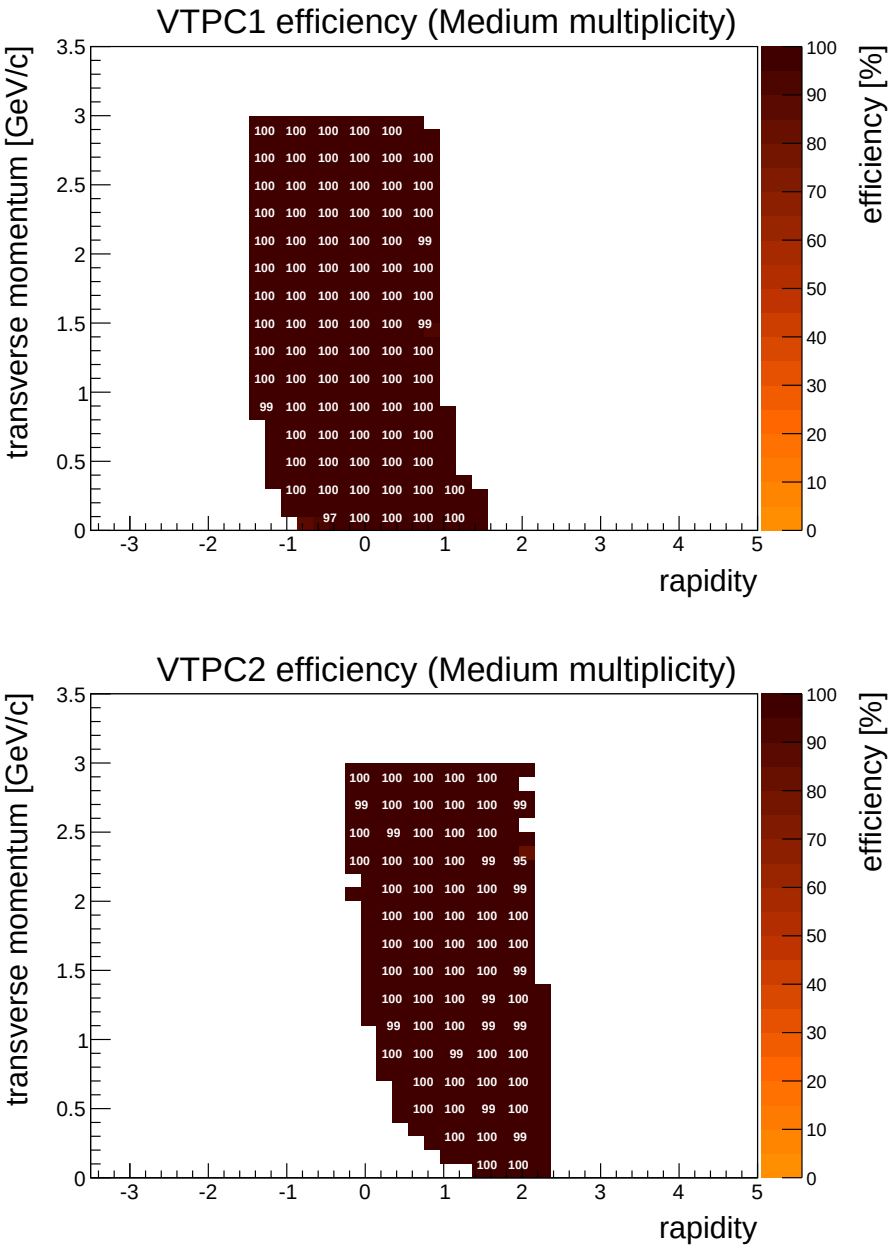


Figure 9.3: Efficiency for VTPC1 (top) and VTPC2 (bottom) chamber for medium multiplicity events equal to 115.6 particles per event per unit rapidity.

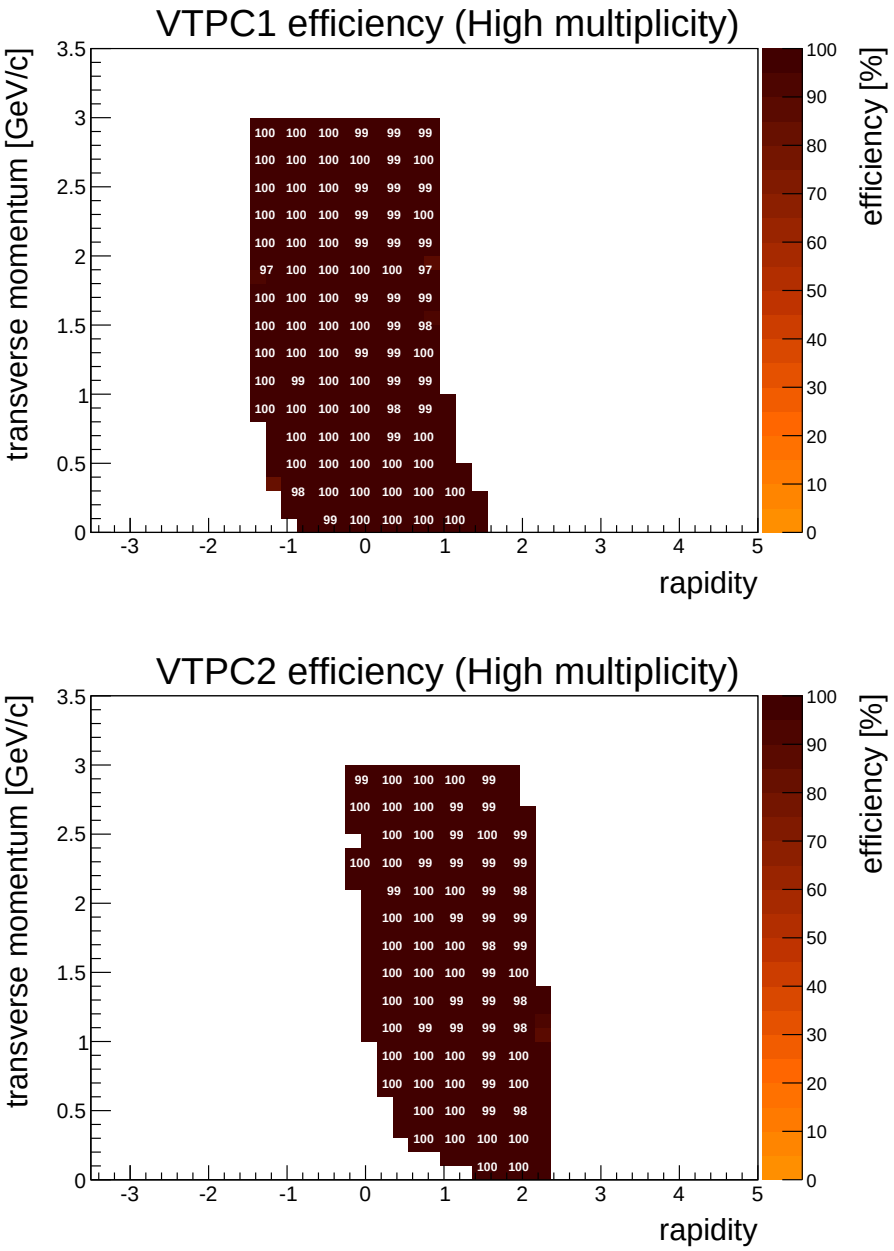


Figure 9.4: Efficiency for VTPC1 (top) and VTPC2 (bottom) chamber for high multiplicity events equal to 423.1 particles per event per unit rapidity.

A seed also is used to calculate parameters which stay constant such as charge (positive, negative or neutral)

The possible improvement is to replace seeding algorithm with more generic method. A method being under consideration is based on clasterization in a parameter space. The method working principle, illustrated on fig. 9.6 and fig. 9.6, goes as follow:

- For n closest clusters, parameters of a model are calculated. The variable n denotes number of model parameters e.g. for a model of straight line $n = 2$, for a circle $n = 3$ etc. The fig. 9.6a and fig. 9.6b present example input data for line and circle model respectively.

- For a set of parameter vectors obtained in previous step, a clasterization is performed using algorithm called Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [129]. The resulting clusters are depicted on fig. 9.6c and fig. 9.6d. Each cluster represents set of input data points pairs or triplets, which parameter vectors are similar.

- The clusters formed in parameter space are then used to select input points which are likely to belong to the same object (e.g. line, circle). The effect of this filtration can be observed on fig. 9.7a and fig. 9.7b. One can see that objects have been partially recognized, what is sufficient for a good initial parameter estimation.

- All points from the same clusters are used to calculate initial parameters of one track. The final results are presented on fig. 9.7c and fig. 9.7d.

This algorithm detects number of tracks and produces good estimates of parameters. Those tracks all together can be used to build a global model to consider all trajectories simultaneously within an event. Afterwards CMA-ES can be used to fine-tune parameters of the global model. Presented approach would eliminate seeding

part of the algorithm, as it is tailored to geometry of the NA61/SHINE detectors and therefore, adapting the algorithm to differently shaped TPC detectors would require adequate modifications. With the presented improvement, evolutionary tracker could became more versatile tool.

9.4 Summary

The results show that the proposed algorithm can reliably track an unknown number of particles traversing the detector, including those sharing parts of their trajectories. This was achieved using simple Generative and Discriminative models without incorporation of the Kalman Filter. Notably, the reconstruction performance is very high given the small amount of information provided.

The comparison of computational complexity between the evolutionary tracker and KF by measuring execution time is not accurate as it depends on factors such as code optimization techniques.

However, the comparison can be done regarding number of calls of extrapolation function. The trajectory extrapolation function implements differential equation of particle motion (described in section 8.2) using a Runge–Kutta method.

In case of KF, number of extrapolation function calls is linear to a number of points. Typically one call per point is sufficient. Whereas, number of calls for evolutionary tracker equals to the total number of CMA-ES offspring evaluations across all generations. Therefore, the evolutionary tracker is obviously much more computationally demanding.

However, the CMA-ES holds a great potential for parallel computing, much greater than KF. One of possible solution to improve execution time is using Graphics Processing Unit (GPU). In the article [13], it states that GPU implementation of CMA-ES, in the best scenario is over 800 times faster than Central Processing

Unit (CPU) implementation. Therefore, it might have a big impact on event reconstruction algorithm of the future.

Currently, evolutionary tracker requires around 1400 evaluation per point, in order to provide accurate results. However, that might be improved as the algorithm shows sensitivity to chosen parameters of CMA-ES(λ, μ and σ) in terms of computational time. This motivates further studies on algorithm performance, including exploiting parallel computing. Furthermore, track and background clusters should be analyzed in order to find computationally cheaper surrogate models.

Presented algorithm was developed as a part of the NA61/SHINE data reconstruction software, performing a sub-detector wise reconstruction.

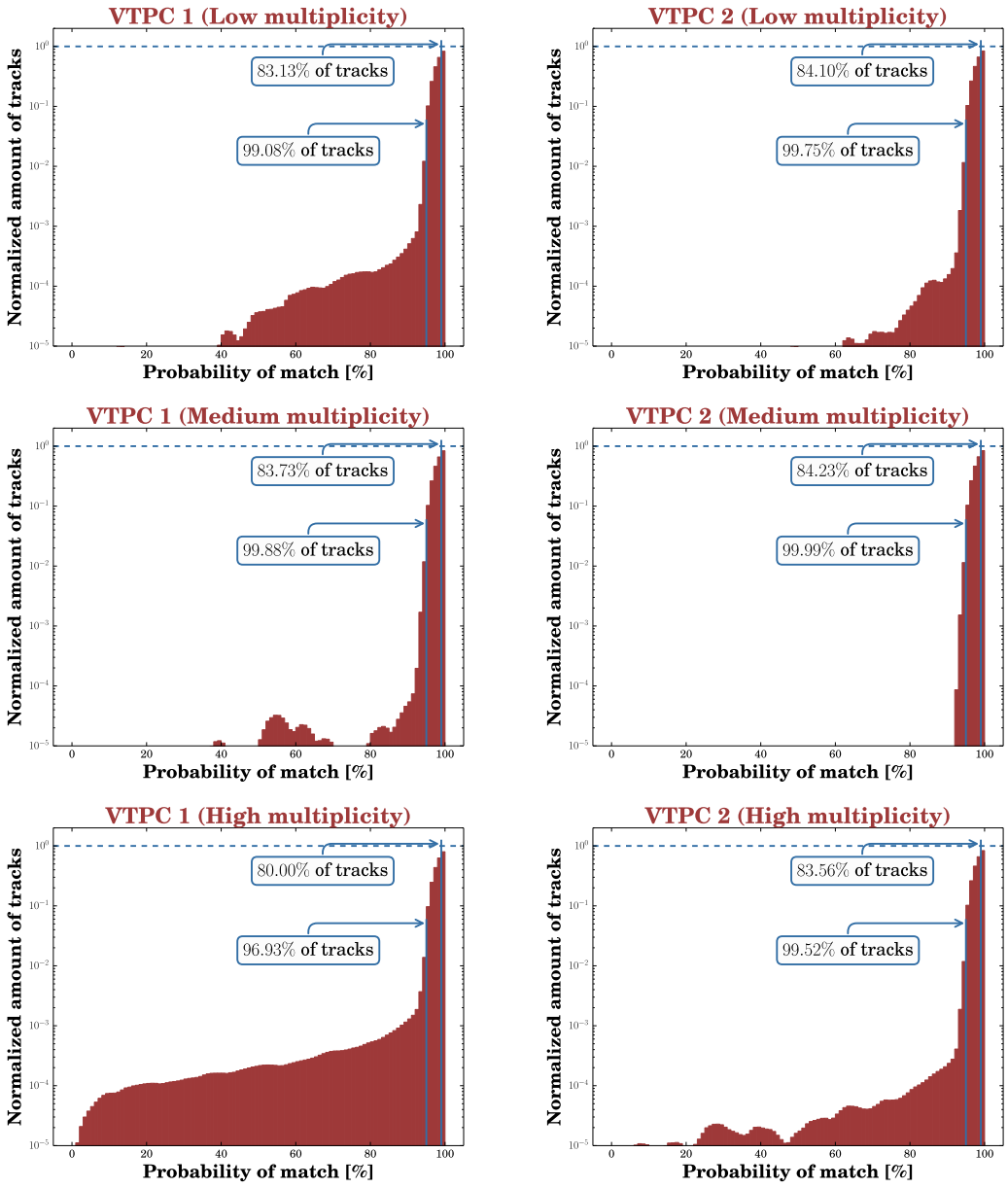
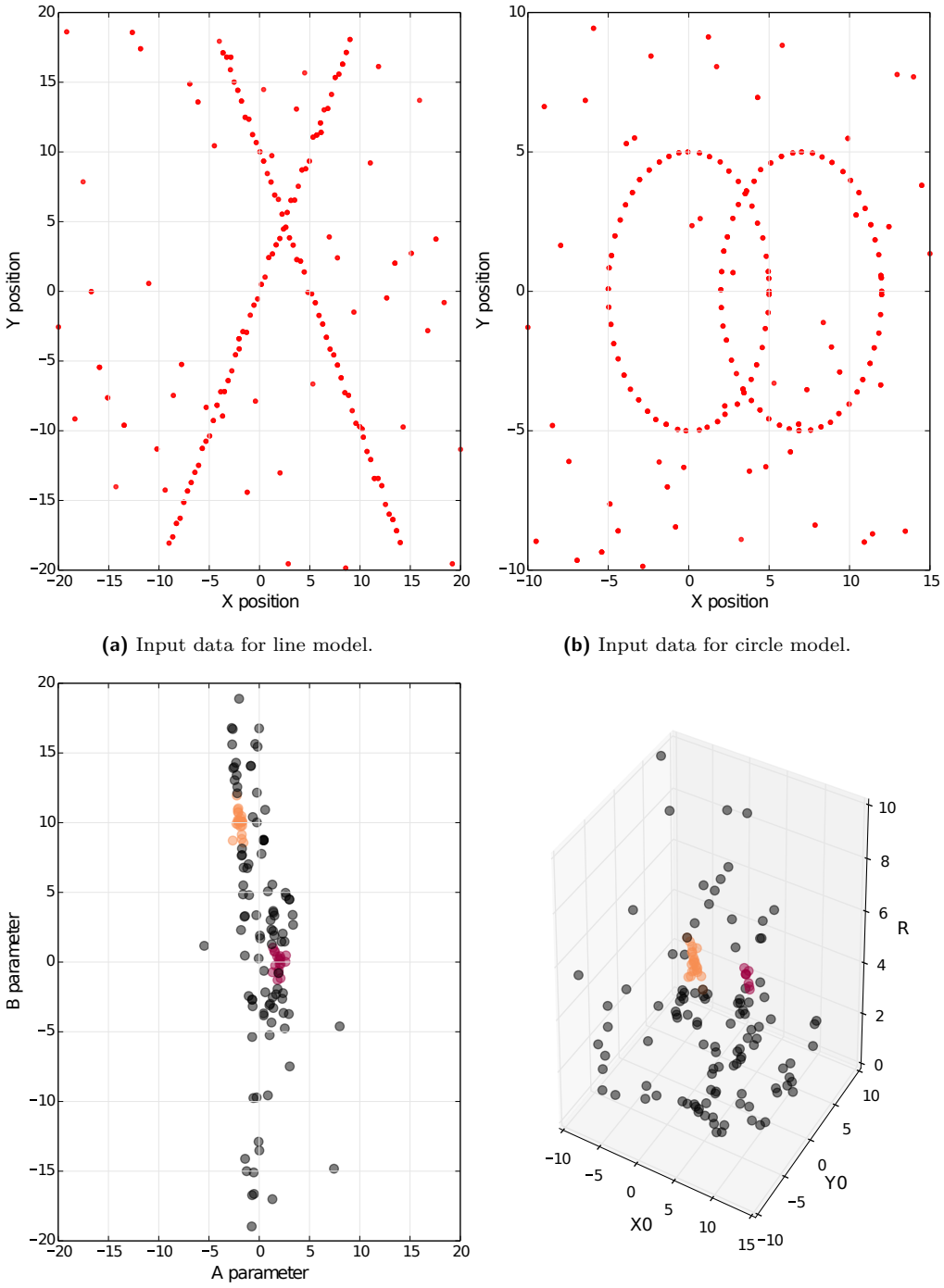


Figure 9.5: Goodness of fit using Pearson's χ^2 test for VTPC1 (top) and VTPC2 (bottom) chamber. Multiplicity denotes the number of particles per event per unit rapidity (often denoted by $\frac{dn}{dy}$). Low, medium and high multiplicity are respectively 11.5, 115.6, 423.1.



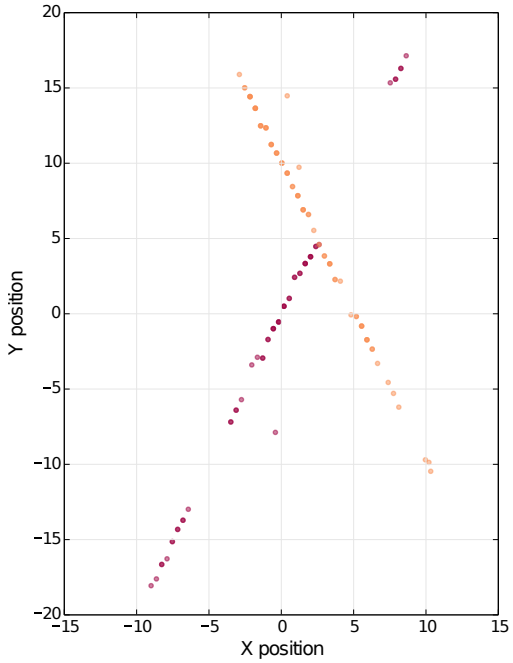
(a) Input data for line model.

(b) Input data for circle model.

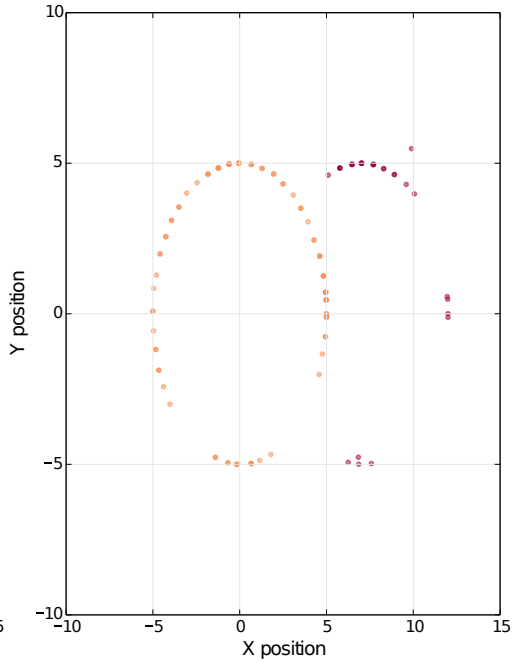
(c) Clusters in parameter space of line.

(d) Clusters in parameter space of circle.

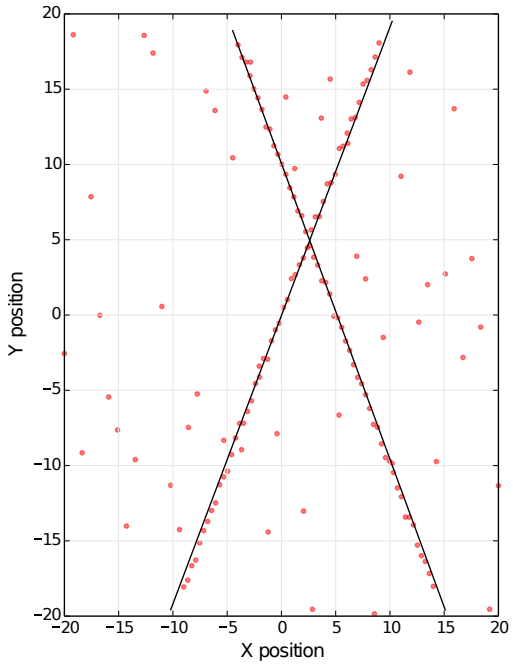
Figure 9.6: Input data and its representation in the parameter space.



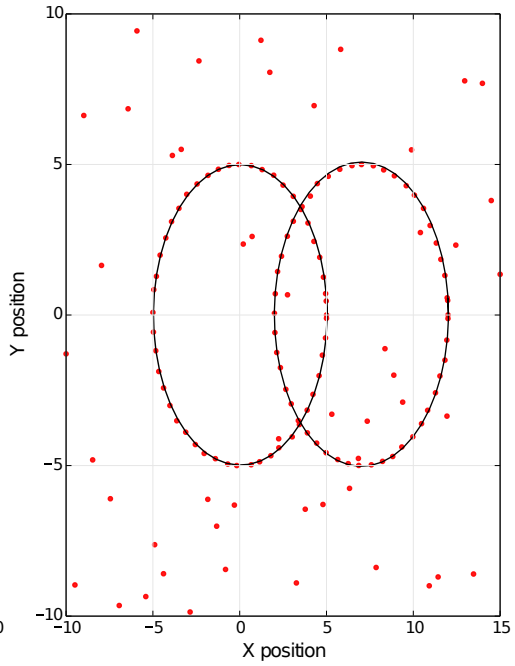
(a) Input data points which are in a cluster of line parameters.



(b) Input data points which are in a cluster of circle parameters.



(c) Initial parameters estimations based on filtered points of lines.



(d) Initial parameters estimations based on filtered points of circles.

Figure 9.7: Results of clusterization in the parameter space.

Chapter 10

Application in Nuclear Physics

This chapter presents a fully automated nuclei identification method. The method uses the CMA-ES in order to estimate parameters of the model, which afterwards is used to interpret the data.

In the following sections, the model as well as the data association and the classification algorithms are described. At the end, a test procedure and the results are presented together with short discussion on strong points as well as drawbacks.

Introduction to the topic is given in chapter 1

10.1 The Model

Correlations between measured energy losses in two successive detectors (ΔE - E) create a specific pattern, presented on fig. 10.1, which can be easily modeled. In order to build a model, the function proposed by L.Tassan-Got [57] has been used. For detectors delivering a linear response (e.g. silicon detectors), the functions take the following form:

$$\Delta E = t(E, g, \mu, \lambda, A, Z) = \left[(gE)^{\mu+1} + \left(\lambda Z^{\frac{2}{\mu+1}} A^{\frac{\mu}{\mu+1}} \right)^{\mu+1} \right]^{\frac{1}{\mu+1}} - gE \qquad (10.1)$$

where the parameters of interest are G, λ and μ. However it has already been noticed in the past [58, 130] that for a wider Z range, the above formula must be extended by additional parameters α, β, ν and ξ:

$$\begin{aligned}\Delta E &= t(E, g, \mu, \nu, \lambda, \alpha, \beta, \xi, A, Z) = \\ &= \left[(gE)^{\mu+\nu+1} + \left(\lambda Z^{\alpha} A^{\beta}\right)^{\mu+\nu+1} + \xi Z^2 A^{\mu} (gE)^{\nu}\right]^{\frac{1}{\mu+\nu+1}} - gE\end{aligned} \tag{10.2}$$

For a detector delivering a non linear response versus deposited energy this function needs to be corrected. The energy E in eq. (10.2) must now be expressed as a function of the light h emitted by the scintillator. The light response of CsI(Tl) crystals is deduced from the Birks formula [131] and allows to express the energy released in the CsI as a function of the emitted light :

$$E = \sqrt{h^2 + 2\rho h\left[1 + \ln\left(1 + \frac{h}{\rho}\right)\right]} \tag{10.3}$$

where $\rho = \eta Z^2 A$ is a new parameter.

The method presented in this chapter, uses the function for linear detectors (eq. (10.1)) to build our model in the following manner:

$$\text{model} \quad \begin{cases} t(E, g, \mu, \lambda, A_1, Z_1) \\ \dots \\ t(E, g, \mu, \lambda, A_n, Z_n) \end{cases} \qquad \text{where} \quad \{A_i, Z_i\} \in I \tag{10.4}$$

The set I contains mass number (A) and atomic number (Z) pairs for isotopes of interest, in the presented model from ${}^{1}_{1}\mathrm{H}$ up to ${}^{25}_{12}\mathrm{Mg}$. The model is easily adjustable with regards to the type of fragments by manipulating the parameters A and Z. Furthermore, as this is a generative model, it can be easily used to generate test data.

After the model is defined, the CMA-ES is used to estimate its parameters. When model is ready, it can be used for classification of an input data representing energy losses ΔE and E.

10.2 Data Classification

The model is iteratively compared with input data by means of a user defined fitness function. The function compares data and the model (eq. (10.4)) with given parameters in order to evaluate the correctness of model parameters. In the evolutionary strategy the definition of a fitness function is a key point, it is defined as follows:

$$f(D_{m,2}\,, g, \mu, \lambda) = \sum_{j=1}^{m} \arg\min_{i \in I} \left(D_{j,1} - t(D_{j,2}\,, g, \mu, \lambda, A_i, Z_i)\right)^2 \tag{10.5}$$

where

$$D_{m,2} = \begin{bmatrix} \Delta E_1 & E_1 \\ \dots & \dots \\ \Delta E_m & E_m \end{bmatrix} \tag{10.6}$$

denotes a matrix of m input data points (ΔE and E of fragments). In other words, the fitness function $f(\dots)$ quantifies difference between every data point $p_j = [D_{j,1}, D_{j,2}]$ and its closest function $t(\dots, A_i, Z_i)$ of the model.

With the model and fitness functions defined, the parameters g, μ, λ have to be estimated. In order to search the model parameter space, once again the CMA-ES has been used (chapter 7).

The method has been implemented using the R language[132] and C++ CMA-ES library[133] (the same library was used in evolutionary tracker) integrated by means of Rcpp package[134]. The decision to choose C++ implementation of CMA-ES has been made based on execution speed (it supports multi-threading) as well as due to advanced development stage of the project. Noteworthy, the

CMA-ES library is released under the GPLv3 license and offers rich set of options such as possibility of alternating values of CMA-ES parameters (e.g. μ, λ), multiple versions of the algorithms (e.g. aCMA-ES, BIPOP-aCMA-ES [135] etc.) as well as detailed monitoring features.

For solving this problem, as in evolutionary tracker, the Active-CMA-ES[9] has been used. After the aCMA-ES algorithm finds the minimum of objective function, the model parameters are used to classify the data. The classification is performed based on distance

$$d = |D_{j,1} - t(D_{j,2}\,, g, \mu, \lambda, A_i, Z_i)| \tag{10.7}$$

between a data point p_j and a function $t(\ldots, A_i, Z_i)$ of the model. For example, if a value of the function $t(E_j, \ldots, A_{25}, Z_{12})$ is the closest to a data point p_j, then the point p_j is classified as ${}^{25}_{12}$Mg.

In the following section, the identification efficiency of the method is presented.

10.3 Results

In order to test the algorithm, labeled data has been simulated with superimposed Gaussian noise as presented on the left panel of fig. 10.1. The numbers of particular isotopes (230k in total) as well as noise have been based on real telescope detector from the NIMROD array [54]. The data was generated using set of functions described by eq. (10.4), which define a generative model.

The parameters of the model g, μ and λ were 0.25, 0.7 and 84, respectively. As a first step, the re-classification of data to lines generated with these parameters was performed. Due to a noise application, some of the ΔE - E points have changed their position with respect to the original one so strongly, that their mass classification failed for 501 (0.22%) fragments with Z $>$ 4. This means, that if our evolutionary algorithm reconstructed the model function parameters with

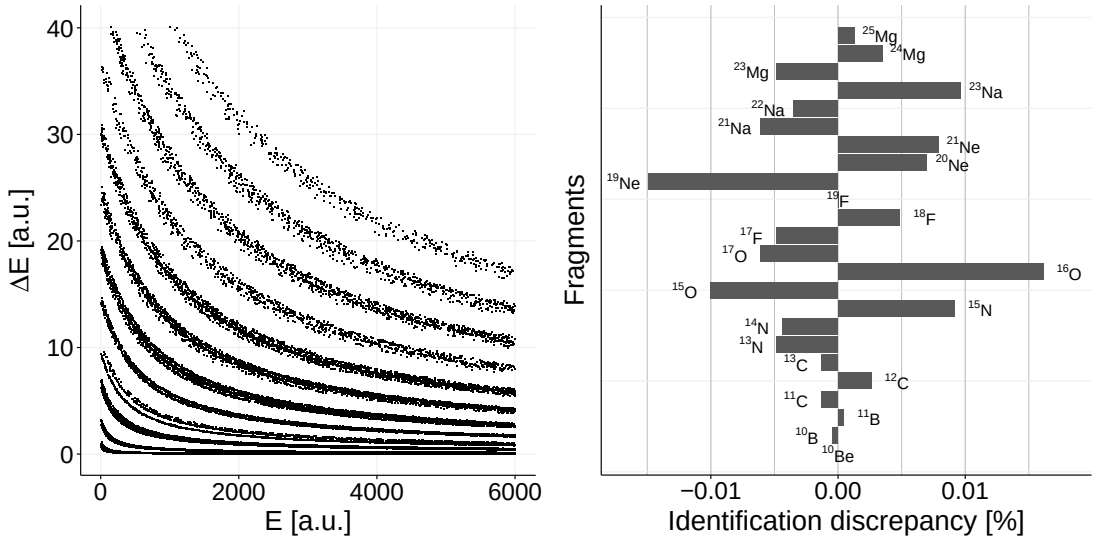


Figure 10.1: Left: ΔE - E points simulated using eq. (10.1) with Gaussian noise. Numbers of particular isotopes have been based on real telescope from NIMROD array. Right: Identification discrepancies of particular fragments after noise application. Negative values inform that numbers of fragments have been underestimated, positive - that they have been overestimated. Isotopes without identification discrepancy were not depicted.

100% accuracy, 501 fragments would be in any case not correctly classified in mass number. The atomic number has been assigned correctly to all studied fragments. In the right panel of fig. 10.1, the identification discrepancy for each fragment due to a noise application is presented. It is calculated as a difference between the originally simulated number of fragments of given A and Z and number of these fragments after noise application and re-classification, normalized to the total number of simulated fragments. Positive values indicate number of fragments that have been classified as a particular isotope, exceed original number of generated fragments of this isotope. The negative values represents opposite situation.

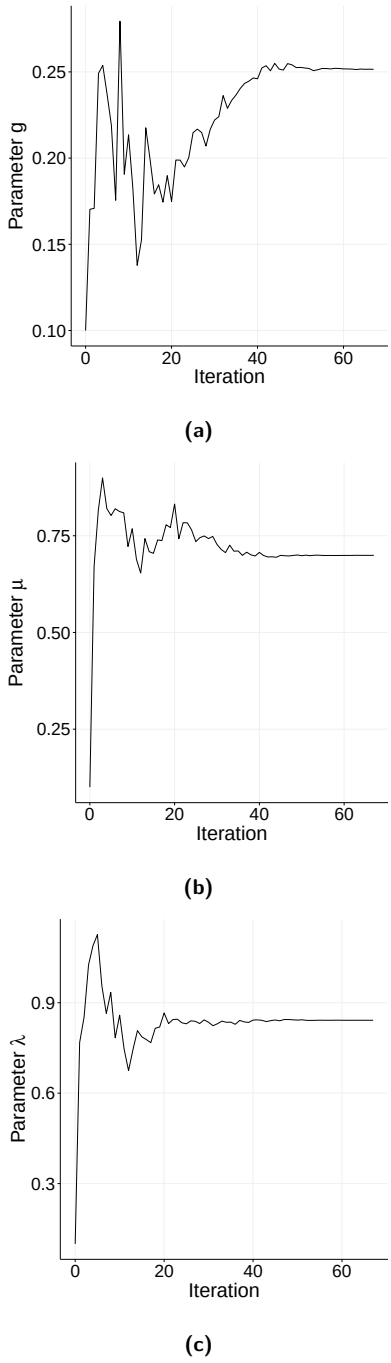


Figure 10.2: Model parameters evolution: (a) parameter g, (b) parameter μ, (c) parameter λ

The fragments with A and Z reassigned to simulated data after noise superimposition, will be from now on called the reference data.

The evolutionary algorithm has been run on an average class laptop equipment in 3rd generation of Intel's i5-3320m processor. It needed around 10 minutes and 68 iterations to complete the calculations. The initial values of parameters g, μ and λ have all been set to 0.1. Since the aCMA-ES works better when the parameters are the same order of magnitude, parameter λ has been scaled by 100. As a result of the algorithm, they have been estimated to 0.252, 0.699 and 0.842, respectively.

In fig. 10.2 the evolution of those parameters as a function of iteration number has been shown. The model function calculated with the initial parameters (a), after the 20th (b), 40th (c) and 68th (d) iteration, plotted on the top of simulated data is shown on fig. 10.3. It can be seen, that the model function with initial parameters certainly does not describe the simulated data, however after the 20th iteration it decently fits the data. The last two thirds of the iterations are used to fine-tune the parameters.

Parameters obtained as the result of the evolutionary algorithm have been used to a final

mass and charge classification. Fragments with A and Z assigned to simulated data after algorithm performance will be called operational data. The identification discrepancy for each fragment was again calculated. This time it has been calculated as a difference between the simulated data and the operational data. In fig. 10.4 these discrepancies are presented.

The function parameters have been reconstructed with high accuracy, it is not surprising therefore, that the data classification efficiency is as high as 99.76%. Fragments charge has been properly assigned to all fragments. 550 fragments, out of 228698, were misidentified in mass, all of them being a species with $Z > 4$. Compared to number of fragments misidentified due to noise superposition, this number rose about 49 and that is a 0.214‰ of total number of studied fragments.

The last test performed was to execute the algorithm on simulated data with missing the lightest or the heaviest isotopes with the model having number of isotope types not changed ($^{1}_{1}$H up to $^{25}_{12}$Mg). In the left hand side of fig. 10.5 the result of algorithm performance without isotopes of H and He is shown, in the right hand side of fig. 10.5 the result for simulated data without Ne, Na and Mg isotopes. It can be seen that the algorithm works very well for a such set of data.

The conclusion from that test is that in order to identify fragments produced in a given experimental reaction it is necessary to create a model with the maximum isotopes expected in the reaction. Although telescopes detect various numbers of isotopes usually smaller than the maximum, the algorithm should still fit the data properly.

10.4 Time and Space Complexity

The space complexity of this method depends only on data size. The complexity of CMA-ES, described in section 7.5, can be considered constant as the number

of parameters is constant. Therefore the overall space complexity is $\Theta(n_e)$, where n_e is size of input data represented by the number of rows of D matrix defined on eq. (10.6).

On the other hand, the time complexity is determined by complexity of two elements: fitness function and optimization. The fitness function compares each data point with the model (eq. (10.4)). Thus, the complexity is $\Theta(n)$. The time complexity of the optimization process, as described in section 7.5 is $\Theta(n^2)$, where n is an number of model parameters. However, because the number of parameters is constant, the over all complexity can be considered to be constant.

Therefore, the overall time complexity of the identification method presented in this chapter is $\Theta(n_e)$. Unfortunately, similarly to the trajectory reconstruction algorithm described in section 8.3, the complexity coefficient is of order of 10^3, what significantly decreases performance. The future studies will address this drawback.

10.5 Summary

The adaptation of the algorithm presented in this section is a proof of applicability of evolutionary strategy in ΔE - E identification procedure. The agreement between the simulated and operational data is 100% for Z (charge) identification. As presented in section 10.3, the efficiency of mass identification is 100% for fragments up to ^{10}Be. With increasing charge, the distinction between isotopes decreases, which led to lower mass classification efficiency. However, only 0.24% of isotopes were misidentified in mass, which is a significant result.

None of cited articles in chapter 1 provided evaluation of identification efficiency of described algorithms, since the authors have applied their methods directly on experimental, not labeled data. However, it is important to emphasize, that in this work it has been tested using model describing linear detectors, which has reduced

number of free parameters compared to extended version.

Furthermore, although the execution time is acceptable in most of the cases, currently identification takes around few weeks for Nimrod detector array, it is still substantial. For the Nimrod, which consists of 256 telescopes, the time required to calibrate it on a laptop would take $256 \times 10min = 2560min \approx 43hours$. Of course one can take advantage of running parallel processes, however, computing power should be used as efficiently as possible.

Therefore, the study improving convergence speed performance are planned. One of possible places for improvement is the fitness function. A fitness function which could exploit the fact that the lightest isotopes presents lower noise levels, could shorten execution time. An another way to improve the performance is using a variant of CMA-ES called BI-Population CMA-ES [135] which performed very well on most of the functions [4].

In the future work it is planed to test the method with various models, including the extended model of ΔE - E relation, (eq. (10.2)) for linear detectors and model with one nonlinear detector (eq. (10.2) with eq. (10.3) as a correction). Finally, application of the method on experimental data collected by several experiments with various detection arrays is scheduled.

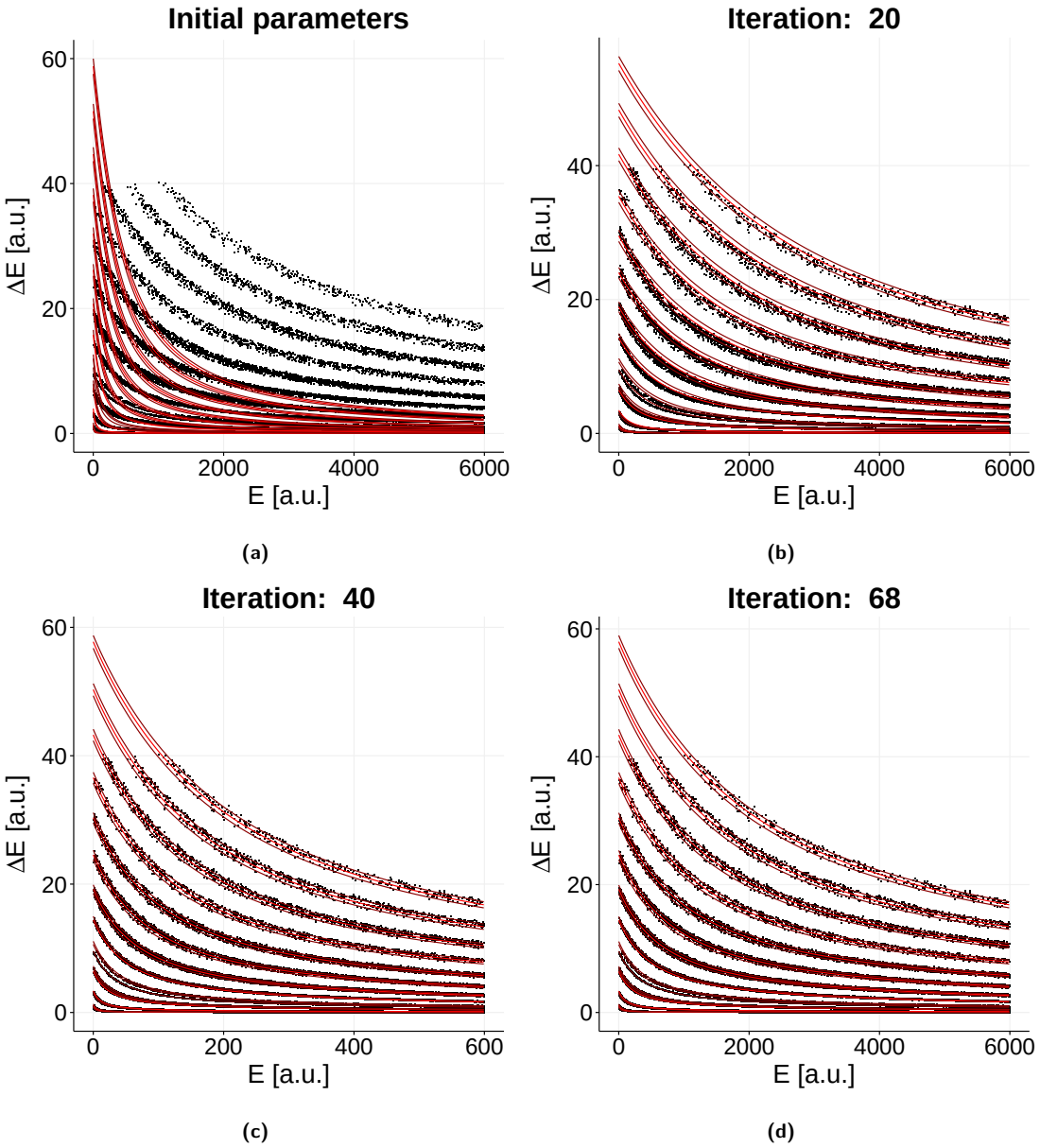


Figure 10.3: The model function (red lines) calculated with the initial parameters (a), after the 20th (b), 40th (c) and 68th (d) iteration, plotted on the top of simulated data (black points).

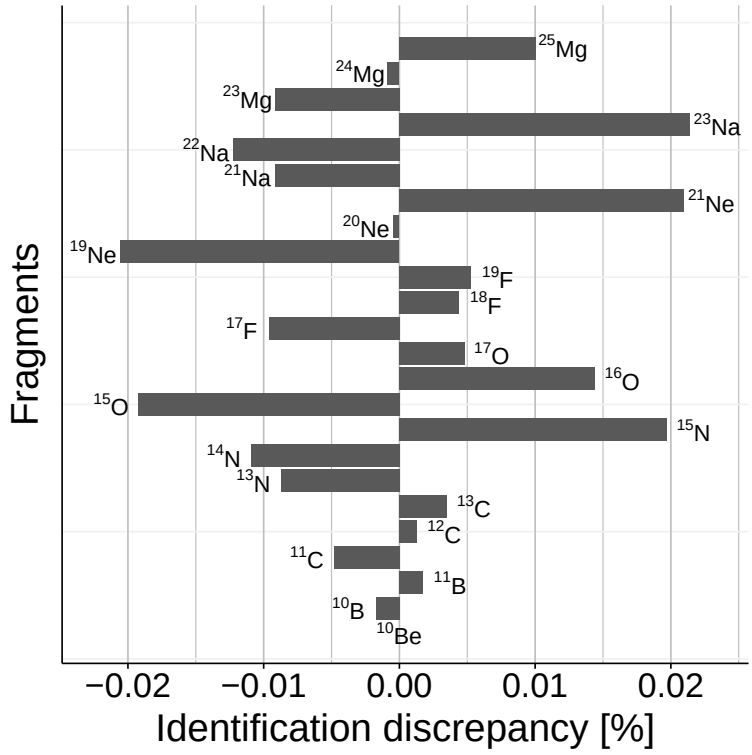


Figure 10.4: Identification discrepancies of particular fragments after final identification process. Negative values inform that numbers of fragments have been underestimated, positive - that they have been overestimated. Isotopes without identification discrepancy were not depicted.

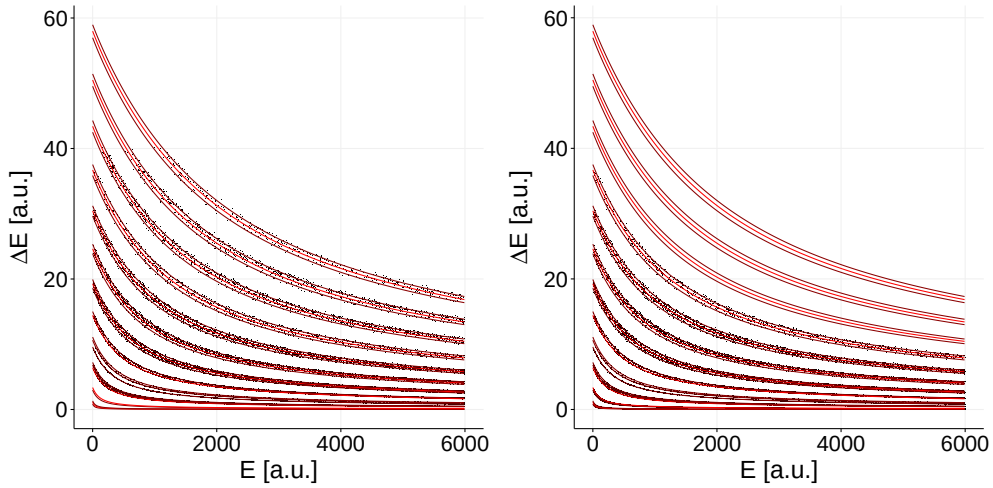


Figure 10.5: Algorithm performance on simulated data with missing isotopes. Left: missing H and He isotopes. Right: missing Ne, Na and Mg isotopes.

Chapter 11

Summary and Conclusions

In this thesis main aspects of conducting an experiment in high energy physics have been presented, starting with a general experiment and facility descriptions in chapter 2. Afterwards, in chapter 4 and chapter 3, the data acquisition system and modular electronics have been presented, respectively. Modular electronics are the most popular building blocks used for building data acquisition and trigger systems. The latter one is a system used for preselection of events and has been described in the chapter 5. The systems mentioned above, were necessary to collect data resulting from particle collisions and description of those systems offers more in depth knowledge about the field.

In order to perform reconstruction and analysis of data collected using above systems, a set of tools in form of software framework described in chapter 6 has been developed. The event reconstruction process has been presented in chapter 8. The main part of the process and at the same time, the main subject of this thesis, is reconstruction of particle trajectories which traversed detector apparatus. The reconstruction algorithm was designed using machine learning techniques and stochastic, derivative-free method called CMA-ES.

The algorithm has been tested on labeled data set using simulated particle

trajectories with superimposed real detector noise. The results shown in chapter 9, prove that the algorithm is an efficient solution to a problem of tracking particles in an inhomogeneous magnetic field, with reconstruction efficiency not lower than 94%. Furthermore, it reliably estimated number of trajectories and theirs parameters. Noteworthy, the efficiency was still very high for very high multiplicity events, where distance between trajectories was shorter that spatial resolution of the detector. Trajectory reconstruction in very high density regions was consider as not feasible[66] resulting in very complicated design of existing reconstruction software [123]. This achievement **proves that the first statement of this dissertation is valid**.

Although the presented method proved to be very efficient in term of reconstruction accuracy, it is not a flawless method. The main drawback is a time complexity making it not feasible for extensive data. In cases where development speed, accuracy as well as flexibility are the top priority, whereas execution time is of second importance, the method appears to be a good solution. The algorithm becomes even greater solution, if parallelism is exploited as described in section 9.4.

Therefore, continuation of this research will focus on improving execution time as well as on other possible improvements as it is described in section 9.3.

The idea of using stochastic and derivative-free CMA-ES method for nuclei identification in heavy ion collisions at intermediate energies was described in chapter 10. The algorithm has been tested using simulated data with addition of simulated Gaussian noise. The results presented in the same chapter, show very high identification efficiency as high as 99.76%. Therefore, the **second statement of this dissertation has been proved to be valid.**

The method has a great potential and it was already recognized by the community in the talk given during International Workshop on Multi facets of Eos and

Clustering (IWM-EC 2018) [136].

The further studies will focus on application of this algorithm on real data and overall fine-tuning for better performance and robustness.

Glossary

ADC Analog to Digital Converter. 60, 62, 83, 97

AMC Advanced Mezzanine Card. 14, 54, 55

ATCA Advanced Telecom Computing Architecture. 14, 54–56, 59

BEE Back-End Electronics. 57, 59

CAMAC Computer-Aided Measurement And Control. 14, 49, 50, 59, 70, 72, 75, 81

CASTOR CERN Advanced STORage manager. 83

CERN European Organization for Nuclear Research. 39

CFD Constant Fraction Discriminator. 15, 69, 79–81

CMA-ES Covariance Matrix Adaptation Evolution Strategy. 21, 22, 26, 30, 33–37, 103, 105, 109–111, 117, 126, 128, 138, 139, 143, 145, 146, 149, 151, 155, 156

CPU Central Processing Unit. 138

CSA Cumulative Step-size Adaptation. 108

DAQ Data Acquisition. 13, 36, 42, 57, 59, 64, 67, 74, 78, 80, 81, 83, 91

DBSCAN Density-Based Spatial Clustering of Applications with Noise. 137

DCS Distributed Control System. 76

DDL Detector Data Link. 63

EA Evolutionary Algorithm. 110

ECL Emitter-Coupled Logic. 49

EKF Extended Kalman Filter. 28, 29

ES Evolution Strategy. 104, 105, 110

FEE Front-End Electronics. 13, 43, 57, 59–61

FFT Fast Fourier Transform. 76

FPGA Field-programmable Gate Array. 14, 49, 61, 70, 72

FRU Field Replaceable Units. 55

GNU GNU's not Unix. 95, 96

GPU Graphics Processing Unit. 138

HEP High Energy Physics. 22, 25, 29, 36, 48, 84, 85

I2C Inter-Integrated Circuit. 55

IPMI Intelligent Platform Management Interface. 55

KF Kalman Filter. 28, 29, 138

LED Leading Edge Discriminator. 15, 68, 69

LHC Large Hadron Collider. 29, 39

LMPD Low Momentum Particle Detector. 41

LVDS Low-Voltage Differential Signaling. 49

MTPC Main Time Projection Chamber. 13, 44, 115

NASA National Aeronautics and Space Administration. 47

NFL No Free Lunch. 22

NIM Nuclear Instrumentation Module. 14, 48, 49, 59, 67, 68, 70, 78, 81

PCA Principal Component Analysis. 110

PGI The Portland Group, Inc.. 85, 95, 96

PICMG PCI Industrial Computers Manufacturing Group. 54

POSIX Portable Operating System Interface for Unix. 73, 75

PSD Projectile Spectator Detector. 13, 15, 41, 42, 44, 65, 67, 68, 73

SBC Single Board Computer. 14, 51, 80

SHINE SPS Heavy Ion and Neutrino Experiment. 39

SHOE Shine Offline Event. 90, 92

SPS Super Proton Synchrotron. 39

STL Standard Template Library. 85

TDC Time to Digital Converter. 80, 81, 83

ToF Time of Flight. 13, 41, 42, 59, 69

TPC Time Projection Chamber. 13, 15, 17, 40–43, 59, 61, 84, 97, 114, 132, 138

TTL Transistor–Transistor Logic. 49

UKF Unscented Kalman Filter. 29

VMEbus VERSAmodule Eurocard bus. 14, 50–53, 55, 56, 59, 80, 81

VTPC Vertex Time Projection Chamber. 16, 117, 131

VXS VMEBus Switched Serial. 53, 54

Bibliography

[1] James S Bergstra et al. "Algorithms for hyper-parameter optimization". In: *Advances in neural information processing systems*. 2011, pp. 2546–2554.

[2] Christian Igel. "Neuroevolution for reinforcement learning using evolution strategies". In: *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*. Vol. 4. IEEE. 2003, pp. 2588–2595.

[3] Ilya Loshchilov and Frank Hutter. "CMA-ES for hyperparameter optimization of deep neural networks". In: *arXiv preprint arXiv:1604.07269* (2016).

[4] Nikolaus Hansen et al. "Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009". In: *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*. ACM. 2010, pp. 1689–1696.

[5] Nikolaus Hansen and Raymond Ros. "Benchmarking a weighted negative covariance matrix update on the BBOB-2010 noiseless testbed". In: *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*. ACM. 2010, pp. 1673–1680.

[6] Petr Pošík, Waltraud Huyer, and László Pál. "A comparison of global search algorithms for continuous black box optimization". In: *Evolutionary computation* 20.4 (2012), pp. 509–541.

[7] N. Hansen. “The CMA evolution strategy: a comparing review”. In: *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*. Ed. by J.A. Lozano et al. Springer, 2006, pp. 75–102.

[8] N. Hansen and A. Ostermeier. “Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation.” In: *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*. IEEE, 1996, pp. 312–317.

[9] Grahame A. Jastrebski and Dirk V. Arnold. “Improving Evolution Strategies through Active Covariance Matrix Adaptation.” In: *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*. IEEE, 2006, pp. 2814–2821.

[10] C. Igel, N. Hansen, and S. Roth. “Covariance Matrix Adaptation for Multi-objective Optimization”. In: *Evolutionary Computation* 15.1 (2007), pp. 1–28.

[11] N. Hansen and A. Ostermeier. “Completely derandomized self-adaptation in evolution strategies”. In: *Evolutionary Computation* 9.2 (2001), pp. 159–195.

[12] David H Wolpert and William G Macready. “No free lunch theorems for optimization”. In: *IEEE transactions on evolutionary computation* 1.1 (1997), pp. 67–82.

[13] Aleksandar Tošic and Matjaž Šuber. “A GPGPU Implementation of CMA-ES”. In: ().

[14] Ruben Salvador et al. “Evolutionary design and optimization of wavelet transforms for image compression in embedded systems”. In: *Adaptive Hardware and Systems (AHS), 2010 NASA/ESA Conference on*. IEEE. 2010, pp. 171–178.

[15] M Joos et al. “The“Beamline for Schools”competition at CERN”. In: *Proceedings of the European Physical Society Conference on High Energy Physics. 5-12 July, 2017 Venice, Italy (EPS-HEP2017). Online at http://pos. sissa. it/cgi-bin/reader/conf. cgi? confid= 314, id. 557*. 2017.

[16] M Joos et al. “Detectors for the Beamline for Schools competition at CERN”. In: *PoS* (2018), p. 794.

[17] Georges Aad et al. “The ATLAS experiment at the CERN large hadron collider”. In: *Jinst* 3 (2008), S08003.

[18] Paul VC Hough. “Machine analysis of bubble chamber pictures”. In: *Conf. Proc.* Vol. 590914. 1959, pp. 554–558.

[19] Paul VC Hough. “A Method for Faster Analysis of Bubble Chamber Photographs (Hough and Powell)”. In: *Instrumentation for High-Energy Physics*. 1961, p. 242.

[20] Rudolph Emil Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Transactions of the ASME–Journal of Basic Engineering* 82.Series D (1960), pp. 35–45.

[21] Rudolph E Kalman and Richard S Bucy. “New results in linear filtering and prediction theory”. In: *Journal of basic engineering* 83.1 (1961), pp. 95–108.

[22] Brian DO Anderson and John B Moore. “Optimal Filtering. 1979”. In: *NY: Prentice Hall Google Scholar* (1979).

[23] Cornelius T Leondes. *Theory and applications of Kalman filtering*. Tech. rep. ADVISORY GROUP FOR AEROSPACE RESEARCH and DEVELOPMENT NEUILLY-SUR-SEINE (FRANCE), 1970.

[24] Jay H Lee and N Lawrence Ricker. “Extended Kalman filter based nonlinear model predictive control”. In: *Industrial & Engineering Chemistry Research* 33.6 (1994), pp. 1530–1541.

[25] S. J. Julier and J. K. Uhlmann. “Unscented filtering and nonlinear estimation”. In: *Proceedings of the IEEE* 92.3 (2004), pp. 401–422. ISSN: 0018-9219. DOI: 10.1109/JPROC.2003.823141.

[26] Simon J Julier and Jeffrey K Uhlmann. “A new extension of the Kalman filter to nonlinear systems”. In: *Int. symp. aerospace/defense sensing, simul. and controls*. Vol. 3. 26. Orlando, FL. 1997, pp. 182–193.

[27] J. Lacambre, M. Narozny, and J. Louge. “Limitations of the unscented Kalman filter for the attitude determination on an inertial navigation system”. In: *2013 IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*. 2013, pp. 187–192. DOI: 10.1109/DSP-SPE.2013.6642588.

[28] Pierre Billoir. “Track Fitting With Multiple Scattering: A New Method”. In: *Nucl.Instrum.Meth.* A225 (1984), pp. 352–366. DOI: 10.1016/0167-5087(84)90274-6.

[29] Pierre Billoir, R. Fruhwirth, and M. Regler. “TRACK ELEMENT MERGING STRATEGY AND VERTEX FITTING IN COMPLEX MODULAR DETECTORS”. In: *Nucl.Instrum.Meth.* A241 (1985), pp. 115–131. DOI: 10.1016/0168-9002(85)90523-6.

[30] Pierre Billoir. “Progressive track recognition with a Kalman like fitting procedure”. In: *Comput.Phys.Commun.* 57 (1989), pp. 390–394. DOI: 10.1016/0010-4655(89)90249-X.

[31] R. Fruhwirth. “Application of Kalman filtering to track and vertex fitting”. In: *Nucl.Instrum.Meth.* A262 (1987), pp. 444–450. DOI: 10.1016/0168-9002(87)90887-4.

[32] STEPHEN MYERS. “THE LARGE HADRON COLLIDER 2008–2013”. In: *International Journal of Modern Physics A* 28.25 (2013), p. 1330035. DOI: 10.1142/S0217751X13300354. eprint: http://www.worldscientific.com/doi/pdf/10.1142/S0217751X13300354. URL: http://www.worldscientific.com/doi/abs/10.1142/S0217751X13300354.

[33] Christiane Lefevre. *The CERN accelerator complex*. Tech. rep. 2008.

[34] Florian Hirsch, Atlas Collaboration, et al. “Tracking and vertexing with the ATLAS detector at the LHC”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 650.1 (2011), pp. 218–223.

[35] MJ Costa. “Vertex and track reconstruction in ATLAS”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 582.3 (2007), pp. 785–789.

[36] F.M. Palmonari. “CMS tracker performance”. In: *Nucl.Instrum.Meth.* A699 (2013), pp. 144–148. DOI: 10.1016/j.nima.2012.06.010.

[37] P. Merkel. “CMS tracker performance”. In: *Nucl.Instrum.Meth.* A718 (2013), pp. 339–341. DOI: 10.1016/j.nima.2012.10.008.

[38] CMS collaboration et al. “Description and performance of track and primary-vertex reconstruction with the CMS tracker”. In: *Journal of Instrumentation* 9.10 (2014), P10009.

[39] WD Hulsbergen. “The global covariance matrix of tracks fitted with a Kalman filter and an application in detector alignment”. In: *Nuclear Instru-*

ments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 600.2 (2009), pp. 471–477.

[40] E Rodrigues. "The LHCb Track Kalman Fit". In: *Note LHCb-2007-014* 164 (2007).

[41] A Badalà et al. "Tracking inside the ALICE Inner Tracking System". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 485.1-2 (2002), pp. 15–22.

[42] Are Strandlie and Rudolf Frühwirth. "Track and vertex reconstruction: From classical to adaptive methods". In: *Reviews of Modern Physics* 82.2 (2010), p. 1419.

[43] Rainer Mankel. "Pattern recognition and event reconstruction in particle physics experiments". In: *Reports on Progress in Physics* 67 (2004), pp. 553–622.

[44] M. Schiller. "Standalone track reconstruction for the Outer Tracker of the LHCb experiment using a cellular automaton." PhD thesis. Uni Heidelberg, 2007.

[45] Giovanni Passaleva. "A recurrent neural network for track reconstruction in the LHCb Muon System". In: *Nuclear Science Symposium Conference Record, 2008. NSS'08. IEEE*. IEEE. 2008, pp. 867–872.

[46] A. Pulvirenti et al. "Neural tracking in the ALICE Inner Tracking System". In: *Nucl.Instrum.Meth.* A533 (2004), pp. 543–559. DOI: 10.1016/j.nima.2004.06.176.

[47] A. Badala et al. "Combined tracking in the ALICE decetctor". In: *Nucl.Instrum.Meth.* A534 (2004), pp. 211–216. DOI: 10.1016/j.nima.2004.07.089.

[48] A Badala et al. “Neural tracking in ALICE”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 502.2 (2003), pp. 503–506.

[49] DW Stracener et al. “Dwarf Ball and Dwarf Wall: Design, instrumentation, and response characteristics of a 4π CsI (Tl) plastic phoswich multidetector system for light charged particle and intermediate mass fragment spectrometry”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 294.3 (1990), pp. 485–503.

[50] JOEL Pouthas et al. “INDRA, a 4π charged product detection array at GANIL”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 357.2-3 (1995), pp. 418–442.

[51] K Kwiatkowski et al. “The Indiana silicon sphere 4π charged-particle detector array”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 360.3 (1995), pp. 571–583.

[52] DG Sarantites et al. ““The microball” Design, instrumentation and response characteristics of a 4π-multidetector exit channel-selection device for spectroscopic and reaction mechanism studies with Gammasphere”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 381.2-3 (1996), pp. 418–432.

[53] A Pagano et al. “Fragmentation studies with the CHIMERA detector at LNS in Catania: recent progress”. In: *Nuclear Physics A* 734 (2004), pp. 504–511.

[54] S Wuenschel et al. “NIMROD–ISiS, a versatile tool for studying the isotopic degree of freedom in heavy ion collisions”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 604.3 (2009), pp. 578–583.

[55] R Bougault et al. “The FAZIA project in Europe: R&D phase”. In: (2014).

[56] J Łukasik et al. “KRATTA, a versatile triple telescope array for charged reaction products”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 709 (2013), pp. 120–128.

[57] L Tassan-Got. “A new functional for charge and mass identification in ΔE–E telescopes”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 194.4 (2002), pp. 503–512.

[58] GW Butler et al. “X. Tarrago For example”. In: *Nucl. Instr. and Meth* 89 (1970), p. 189.

[59] N Le Neindre et al. “Mass and charge identification of fragments detected with the Chimera Silicon–CsI (Tl) telescopes”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 490.1 (2002), pp. 251–262.

[60] PF Mastinu et al. “A procedure to calibrate a multi-modular telescope”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 371.3 (1996), pp. 510–513.

[61] J Dudouet et al. “Comparison of two analysis methods for nuclear reaction measurements of 12 C+ 12 C interactions at 95MeV/u for hadron therapy”. In: *Nuclear Instruments and Methods in Physics Research Section A: Ac-*

celerators, Spectrometers, Detectors and Associated Equipment 715 (2013), pp. 98–104.

[62] T Cap et al. "Detection and identification of large fragments from the partitioning of the 197Au+ 197Au system at 23A MeV". In: *Physica Scripta* 2013.T154 (2013), p. 014007.

[63] D Gruyer et al. "New semi-automatic method for reaction product charge and mass identification in heavy-ion collisions at Fermi energies". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* (2016).

[64] Oskar Wyszynski. "Trigger system of the NA61/SHINE experiment at the CERN SPS". In: (2014). DOI: 10.1109/RTC.2014.7097418.

[65] Andras Laszlo et al. "Design and Performance of the Data Acquisition System for the NA61/SHINE Experiment at CERN". In: *Nucl. Instrum. Methods Phys. Res., A* 798 (2015), 1. 14 p.

[66] S Afanasiev et al. "The NA49 large acceptance hadron detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 430.2 (1999), pp. 210–244.

[67] Oskar Wyszynski et al. "Legacy code: lessons from NA61/SHINE offline software upgrade adventure". In: *Journal of Physics: Conference Series*. Vol. 396. 5. IOP Publishing. 2012, p. 052076.

[68] Roland Sipos et al. "The Offline Software Framework of the NA61/SHINE Experiment". In: *Journal of Physics: Conference Series*. Vol. 396. 2. IOP Publishing. 2012, p. 022045.

[69] N Abgrall et al. "NA61/SHINE facility at the CERN SPS: beams and detector system". In: *Journal of Instrumentation* 9.06 (2014), P06005.

[70] András László, Na61/Shine Collaboration, et al. “The NA61/SHINE experiment at the CERN SPS”. In: *Nuclear Physics A* 830.1 (2009), pp. 559c–562c.

[71] John Bertram Adams and Edmund JN Wilson. *Design studies for a large proton synchrotron and its laboratory*. Tech. rep. Cern, 1970.

[72] SPS CERN. “Experimenters’ Handbook, ed”. In: *M. Reinharz, CERN* 198.1 (1981), p. 13.

[73] Krisztina Márton et al. “Low momentum particle detector for the NA61 experiment at CERN”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 763 (2014), pp. 372–382.

[74] Ward Horner and Steve Sabia. “Development of real-time software environments for NASA’s modern telemetry systems”. In: *International Telemetering Conference Proceedings*. International Foundation for Telemetering. 1989.

[75] S.A. Gygi and D.J. Kurfis. *VMEbus-UCDP interface module*. US Patent 5,129,062. 1992. URL: https://www.google.com/patents/US5129062.

[76] H. Liu, S. Zhang, and X. Wang. “Design of VME bus interface board based on FPGA for converter control”. In: *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*. Vol. 4. 2011, pp. 1718–1722. DOI: 10.1109/EMEIT.2011.6023434.

[77] C. Nair. “Modular test architectures for the aerospace industry”. In: *Proceedings, IEEE AUTOTESTCON*. 2002, pp. 241–247. DOI: 10.1109/AUTEST.2002.1047895.

[78] JH Trainor, CH Ehrmann, and TJ Kaminski. "CAMAC and NIM Systems in the Space Program". In: *IEEE Transactions on Nuclear Science* 22.1 (1975), pp. 521–525.

[79] Hiroya Kawasaki et al. "ATCA Server Systems for Telecommunications Services". In: *FUJITSU Sci. Tech. J* 47.2 (2011), pp. 215–221.

[80] NIM Committee. *Standard NIM instrumentation system*. 1990-05. DOI: 10.2172/7120327. URL: http://www.osti.gov/scitech/servlets/purl/7120327.

[81] A CAMAC. "Modular Instrumentation System for Data Handling". In: *EUR-4100* (1972).

[82] WK Dawson et al. "FASTBUS for the particle accelerator laboratories". In: *IEEE Transactions on Nuclear Science* 32.5 (1985), pp. 2089–2091.

[83] R. S. Larsen. "xTCA for physics standards roadmaps amp; SLAC initiatives". In: *2014 19th IEEE-NPSS Real Time Conference*. 2014, pp. 1–7. DOI: 10.1109/RTC.2014.7097432.

[84] Wade D Peterson. *The VMEbus Handbook*. Vita, 1997.

[85] ESONE Committee. *CAMAC Updated specifications, Volume 1*. v. 1. Commission of the European Communities, 1983. ISBN: 92-825-3597-5.

[86] LU Fangmin, LIAO Jianxing, and Ying Shi. *Advanced telecommunications computing architecture data exchange system, exchange board and data exchange method*. US Patent 8,811,577. 2014.

[87] Wolfgang Rauch, NA49 Collaboration, et al. "The NA49 data acquisition system". In: *IEEE Transactions on Nuclear Science (Institute of Electrical and Electronics Engineers);(United States)* 41.CONF-930640– (1994).

[88] WE Hearn and ME Wright. “Fully integrated 16 channel digitally trimmed pulse shaping amplifier”. In: *IEEE transactions on nuclear science* 41 (1994), pp. 1163–1168.

[89] Stuart Kleinfelder et al. “A 4096 cell switched capacitor analog waveform storage integrated circuit”. In: *Nuclear Science, IEEE Transactions on* 37.3 (1990), pp. 1230–1236.

[90] P Barale. *STAR SCA/ADC Datasheet*. 1994.

[91] G. Rubin et al. “The ALICE Detector Data Link”. In: *5th Conference on Electronics for LHC Experiments LEB 99*. 1999, pp. 493–8.

[92] György Rubin et al. *The ALICE detector data link*. Tech. rep. CERN, 1999.

[93] Jean-Philippe Baud et al. “CASTOR status and evolution”. In: *arXiv preprint cs/0305047* (2003).

[94] J.M. Nelson. Tech. rep.

[95] A Aduszkiewicz et al. “Production of\ Lambda-hyperons in inelastic p+ p interactions at 158 {\ mathrm {GeV}}\!/\! c”. In: *The European Physical Journal C* 76.4 (2016), pp. 1–18.

[96] N. Abgrall et al. “Measurements of cross sections and charged pion spectra in proton-carbon interactions at 31 GeV/c”. In: *Phys. Rev. C* 84 (3 2011), p. 034604. DOI: 10.1103/PhysRevC.84.034604. URL: http://link.aps.org/doi/10.1103/PhysRevC.84.034604.

[97] Nicolas Abgrall, ..., O Wyszyński, et al. “Measurements of\ piˆ{\ pm}, Kˆ{\ pm}, Kˆ 0_S,\ varLambda and proton production in proton–carbon interactions at 31 GeV/c with the NA61/SHINE spectrometer at the CERN SPS”. In: *The European Physical Journal C* 76.2 (2016), pp. 1–49.

[98] Nicolas Abgrall et al. “Measurements of $\pi\pm$ differential yields from the surface of the T2K replica target for incoming 31áGeV/c protons with the NA61/SHINE spectrometer at the CERN SPS”. In: *The European Physical Journal C* 76.11 (2016), p. 617.

[99] A Aduszkiewicz et al. “Multiplicity and transverse momentum fluctuations in inelastic proton–proton interactions at the CERN Super Proton Synchrotron”. In: *The European Physical Journal C* 76.11 (2016), p. 635.

[100] Nicolas Abgrall et al. “Measurement of negatively charged pion spectra in inelastic p+ p interactions at p_ {\ mathbf {lab}}= 20, 31, 40, 80 and 158 GeV/c”. In: *The European Physical Journal C* 74.3 (2014), p. 2794.

[101] N. Abgrall et al. “Measurements of production properties of K^0_S mesons and Λ hyperons in proton-carbon interactions at 31 GeV/c”. In: *Phys. Rev. C* 89 (2 2014), p. 025205. DOI: 10.1103/PhysRevC.89.025205. URL: http://link.aps.org/doi/10.1103/PhysRevC.89.025205.

[102] N Abgrall et al. “Pion emission from the T2K replica target: method, results and application”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 701 (2013), pp. 99–114.

[103] A. Aduszkiewicz, ..., O. Wyszyński, et al. “Two-particle correlations in azimuthal angle and pseudorapidity in inelastic p + p interactions at the CERN Super Proton Synchrotron”. In: *The European Physical Journal C* 77.2 (2017), p. 59. ISSN: 1434-6052. DOI: 10.1140/epjc/s10052-017-4599-x. URL: http://dx.doi.org/10.1140/epjc/s10052-017-4599-x.

[104] P Buncic, M Krzewicki, and P Vande Vyvre. *Technical Design Report for the Upgrade of the Online-Offline Computing System*. Tech. rep. 2015.

[105] F. Carena et al. “The ALICE data acquisition system”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 741 (2014), pp. 130 –162. ISSN: 0168-9002. DOI: http://dx.doi.org/10.1016/j.nima.2013.12.015. URL: http://www.sciencedirect.com/science/article/pii/S0168900213016987.

[106] *Django - The web framework*. URL: https://www.djangoproject.com.

[107] CAEN. “V1290 A/N, Technical information manual, Rev.11”. In: (2010).

[108] J Christiansen. *HPTDC, High Performance Time to Digital Converter version 2.2*. Tech. rep. CERN-EP/MIC, 2004.

[109] Xavier Espinal et al. “Disk storage at CERN: Handling LHC data and beyond”. In: *Journal of Physics: Conference Series*. Vol. 513. 4. IOP Publishing. 2014, p. 042017.

[110] S Chapeland et al. “Online processing in the ALICE DAQ The detector algorithms”. In: *Journal of Physics: Conference Series* 219.2 (2010), p. 022004. URL: http://stacks.iop.org/1742-6596/219/i=2/a=022004.

[111] Agnieszka Dziurda. “Full Offline Reconstruction in Real Time with the LHCb Detector”. In: *EPJ Web of Conferences*. Vol. 127. EDP Sciences. 2016, p. 00007.

[112] S. Argiro et al. “The Offline Software Framework of the Pierre Auger Observatory”. In: *Nucl. Instrum. Meth.* A580 (2007), pp. 1485–1496. DOI: 10.1016/j.nima.2007.07.010. arXiv: 0707.1652 [astro-ph].

[113] Rene Brun and Fons Rademakers. “ROOT—an object oriented data analysis framework”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 389.1-2 (1997), pp. 81–86.

[114] R Zybert and P Buncic. "DSPACK-Object Manager for High Energy Physics". In: *Computing in High Energy Physics: CHEP'95-Proceedings of the International Conference. Edited by SHELLARD RONALD ET AL. Published by World Scientific Publishing Co. Pte. Ltd., 1996. ISBN# 9789814447188, pp. 345-348*. 1996, pp. 345–348.

[115] P Buncic and R Zybert. *A persistent object manager for HEP*. Tech. rep. 1994.

[116] Hans-Georg Beyer and Hans-Paul Schwefel. "Evolution strategies–A comprehensive introduction". In: *Natural computing* 1.1 (2002), pp. 3–52.

[117] Nikolaus Hansen. "The CMA evolution strategy: A tutorial". In: *arXiv preprint arXiv:1604.00772* (2016).

[118] Alexandre Chotard, Anne Auger, and Nikolaus Hansen. "Cumulative step-size adaptation on linear functions". In: *International Conference on Parallel Problem Solving from Nature*. Springer. 2012, pp. 72–81.

[119] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)". In: *Evolutionary computation* 11.1 (2003), pp. 1–18.

[120] Raymond Ros and Nikolaus Hansen. "A simple modification in CMA-ES achieving linear time and space complexity". In: *International Conference on Parallel Problem Solving from Nature*. Springer. 2008, pp. 296–305.

[121] A. Auger and N. Hansen. "A restart CMA evolution strategy with increasing population size". In: *The 2005 IEEE International Congress on Evolutionary Computation (CEC'05)*. Ed. by B. McKay et al. Vol. 2. 2005, pp. 1769–1776.

[122] Antoni Aduszkiewicz. "Operation and performance of Time Projection Chambers of SHINE/NA61 experiment at CERN". Warsaw U., 2008.

[123] D Irmscher. “Philosophy and parts of the global tracking chain.” In: *NA49 Note number 131 (1997)* ().

[124] Sergey Gorbunov and Ivan Kisel. “Analytic formula for track extrapolation in non-homogeneous magnetic field”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 559.1 (2006), pp. 148–152.

[125] David J Hand and Keming Yu. “Idiot’s Bayes—not so stupid after all?” In: *International statistical review* 69.3 (2001), pp. 385–398.

[126] Pedro Domingos and Michael Pazzani. “Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier”. In: *Machine Learning*. Morgan Kaufmann, 1996, pp. 105–112.

[127] A Jordan. “On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes”. In: *Advances in neural information processing systems* 14 (2002), p. 841.

[128] S. et al. Agostinelli. “GEANT4: A Simulation toolkit”. In: *Nuclear Instruments and Methods in Physics Research* A506 (2003), pp. 250–303.

[129] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: AAAI Press, 1996, pp. 226–231.

[130] T Shimoda et al. “Simple ΔE- E particle identification with a wide dynamic range”. In: *Nuclear Instruments and Methods* 165.2 (1979), pp. 261–264.

[131] JB Birks and FWK Firk. “The theory and practice of scintillation counting”. In: *Physics Today* 18 (1965), p. 60.

[132] R Core Team. “R language definition”. In: *Vienna, Austria: R foundation for statistical computing* (2000).

[133] Emmanuel Benazera. *libcmaes: Multithreaded c++ 11 implementation of cma-es family for optimization of nonlinear non-convex blackbox functions*. 2014.

[134] Dirk Eddelbuettel et al. "Rcpp: Seamless R and C++ integration". In: *Journal of Statistical Software* 40.8 (2011), pp. 1–18.

[135] Nikolaus Hansen. "Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed". In: *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*. ACM. 2009, pp. 2389–2396.

[136] Diego Gruyer. *New trends in isotopic identification with telescope detectors*. `https://agenda.infn.it/getFile.py/access?resId=4&materialId=slides&confId=13852`. IWM-EC 2018, Catania, Italy.