A First Simulation Study for a Light Dark Matter Experiment at ELSA

von
Jan - Eric Heinrichs

Masterarbeit in Physik
angefertigt im Physikalischen Institut

vorgelegt der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität
Bonn

Mai 2020

I hereby declare that this thesis was formulated by myself and that no sources or tools other than those cited were used.

Bonn, ...
Date Signature

1. Gutachter: Prof. Dr. Klaus Desch
2. Gutachter: Prof. Dr. Florian Bernlochner

Contents

Chapter 1

Introduction

Modern cosmology allows us to understand the history of the universe on macroscopic scales (see [Ams]). It is also consistent with other observations, e.g. rotational curves of galaxies (see Section 2.2), which imply the existence of additional gravitationally interacting particles called Dark Matter. These particles however have so far not been observed directly. Thus, they are also not included in the Standard Model of particle physics. Consequently discovering the true nature of dark matter remains one of the most pressing challenges facing elementary particle physics to this date[Åke+18]. It also provides a large potential to expand our current knowledge of the laws of nature. Most experiments, which have been performed until today, are most sensitive to dark matter candidates in the GeV to TeV mass range[Åke+18]. The MeV to GeV mass range however has been and remains difficult to examine with current experiments[Åke+18]. Discoveries such as the recent observation of the ^{8}Be anomaly and possible explanations with new particles[Fen+17], only add to the interest in research in this field.

This thesis builds on the idea of a light dark matter experiment proposed in [Åke+18]. The goal of this experiment is to produce dark matter in fixed target collisions of an electron beam. The NA64 collaboration has already performed a first physics run with an electron beam fixed target experiment[Ban+17], but will not reach the required sensitivity due to luminosity limitations[Åke+18]. Thus, making the proposal in [Åke+18] interesting. Together with the future missing mass measurement from Belle II, this approach would largely provide the needed sensitivity to test most scenarios of dark matter freeze out via annihilation to Standard Model particles below a GeV. While the experiment is mainly focused on the discovery possibilities regarding dark photons, its experimental technique also provides sensitivity to other dark matter models.[Åke+18].

The basic design features an electron beam impinging on a thin target, creating dark matter particles through a “dark bremsstrahlung” process[Åke+18]. Most of the beam energy will be carried away by the produced particles. Trackers before and behind target in a magnetic field are used to reconstruct the kinematics, searching for low momentum electrons[Åke+18]. Additionally calorimetry is used to veto events[Åke+18].

The promising physics prospects and the compact design, make this experimental approach interesting to several accelerator facilities. In particular, the electron stretcher accelerator (ELSA) located at the “Physikalisches Institut” at the University of Bonn has

some features, which provide a suitable base for an experiment of this type. It is within the right energy range with an energy resolution of 0.8 ‰. Additionally the resonant extraction process can be used to extract electrons in a very controlled manner every 2 ns.

Before such an experiment can be realised however, the setup has to be simulated. This ensures proper design choices are made and is essential in the planning phase of an experiment. Additionally sensitivity studies can be made and the potential impact of the experiment can be studied.

This thesis aims to build upon the ExPlORA framework to establish a basic simulation for a light dark matter type experiment at ELSA. The main focus is on the simulation and building of the pixel detector trackers. It begins with an introduction into the theoretical framework of the Standard Model and dark matter, as well as a short introduction to ELSA (Chapter 2). Then the general idea behind the detector setup is explained (Chapter 3). Subsequently the general simulation setup is described (Chapter 4). This chapter also features an introduction into the ExPlORA framework. The process of simulating pixel detectors is explained in Chapter 5. Then the implementation of a new fitting routine is discussed in Chapter 6. Finally Chapter 7 features some interesting studies, which can be performed with the setup developed in this thesis, and Chapter 8 concludes and summarises the thesis.

Chapter 2

Theoretical Introduction

This chapter provides a brief introduction to the theoretical background needed for this thesis. Detailed information is omitted at appropriate places in favour of brevity. Additional information can be found in the referenced sources.

2.1 The Standard Model of Particle Physics

The Standard Model of particle physics (SM) is the currently used model to describe particles and their interactions. It allows us to correctly calculate cross sections and makes accurate predictions, which are verified by experimental observations. Notable examples of particles, which were predicted by the SM and then found in experiments, include the discovery of the top quark in 1995[Aba+95] and the Higgs boson in 2012 by the ATLAS and CMS collaborations[The12a; The12b; Hei17]

2.1.1 Particles and Interactions

An overview of the SM and its interactions can be seen in Fig. 2.1. The SM consists of six leptons (spin 1/2), six quarks (spin 1/2), four gauge bosons (spin 1) and the Higgs boson (spin 0). Particles with integer spin are called bosons and particle with half integer spin are called fermions. The gauge bosons act as force carriers and are responsible for the SM interactions. Quarks and leptons can be organised into three generations. The first generation consists of the lightest particles, which are stable. The second and third generation contain heavier particles, which eventually decay into particles of the first generation. Neutrinos are exempted from this rule.

The SM provides descriptions for the electromagnetic force mediated by photons, the strong force mediated by gluons and the weak force mediated by Z and W bosons. The Higgs boson does not carry a force, but is used to give particles their mass via the Higgs mechanism[BGB08]. Gravity is currently not a part of the SM and usually irrelevant for interactions in particle physics due to its relative weakness compared to the other forces.

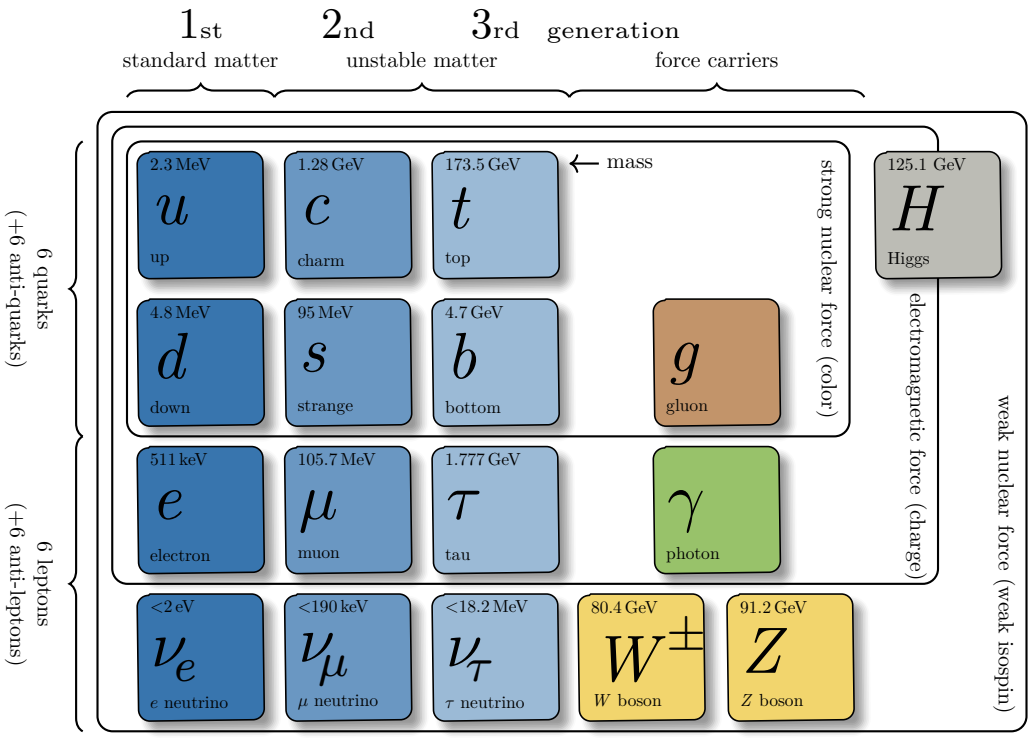


Figure 2.1: The Standard Model of particle physics with all included particles[Bur16]

2.1.2 Quantum Electrodynamics

The SM can be formulated as a quantum field theory with gauge group $\mathrm{SU(3)_C} \times \mathrm{SU(2)_L} \times \mathrm{U(1)_Y}$. This means, that the theory can be expressed by a Lagrangian and possible interactions and their cross sections are subsequently calculated using the Hamiltonian formalism[Sch17]. Particles can then be thought of as excitations in their respective quantum field.

This allows for the separation of the SM Lagrangian into several parts, which correspond to the fundamental interactions and their interacting particles described in Section 2.1.1. One of these interactions is quantum electrodynamics (QED). It describes the electromagnetic interaction between electrically charged particles.

The part of the SM Lagrangian $\mathcal{L}_{\mathrm{QED}}$, which describes QED can be written as[Sch17][1]:

$$\mathcal{L}_{\mathrm{QED}} = -\frac{1}{4} F^{\mu\nu} F_{\mu\nu} + i\bar{\psi} \not{D} \psi - m\bar{\psi}\psi \tag{2.1}$$

Here $F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu$ corresponds to the field strength tensor of the electromagnetic field, ψ to the Dirac field of the corresponding fermion, $\not{D}$ to the covariant derivative in

[1] This only shows the interaction for one type of fermion. This is however sufficient for the subsequent discussion.

Dirac slash notation[Sch17] and m to the mass of the respective fermion.

This Lagrangian determines all types of interactions, which can happen due to the electromagnetic force. One of these interactions is Bremsstrahlung.

Bremsstrahlung

The word bremsstrahlung comes from German *bremsen* "to break" and *Strahlung* "radiation". Consequently, it is a process in which a charged particle emits radiation and thus looses energy. A Feynman diagram is the usual representation of this process. The corresponding diagram can be found in Fig. 2.2.

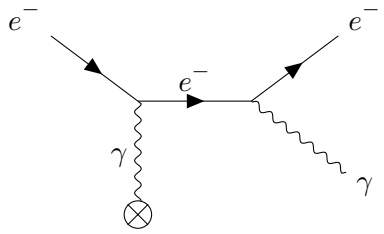


Figure 2.2: An exemplary Bremsstrahlung process with an electron as charged particle. Time runs on the x-axis from left to right.

The incoming electron of the initial state interacts with the atomic nucleus of some surrounding material and exchanges a photon. It then emits another photon, which leads to two particles in the final state with the outgoing electron having lower energy compared to the incoming one. The interaction with the nucleus is required to conserve energy and momentum.

2.2 The Need for Dark Matter

The term dark matter(DM) refers to some kind of matter, which so far has only been detected due to its gravitational interaction on cosmological scales. It has not been observed directly. Hence the "dark" attribute in its name[Sch06].

There are several observations in cosmology, which hint to the existence of DM. This sections purpose however, is to motivate the necessity of DM searches in elementary particle physics. A more detailed overview of the topic can be found in e.g. [Sch06].

One of the most compelling ways, the effects of DM can be seen is by studying the rotation curves of galaxies. The radial distribution of visible matter and Newton's law of gravity give a clear prediction on the rotational speed of stars at different distances from the galactic centre. An example of such a measurement can be seen in Fig. 2.3.

Other measurements imply, that both the stellar, and gas density fall off exponentially for large radii. This means, that the velocity should be $\propto 1/\sqrt{\text{radius}}$[Sch06]. The velocity however is nearly constant for large radii, implying that more matter is present in the galaxy.

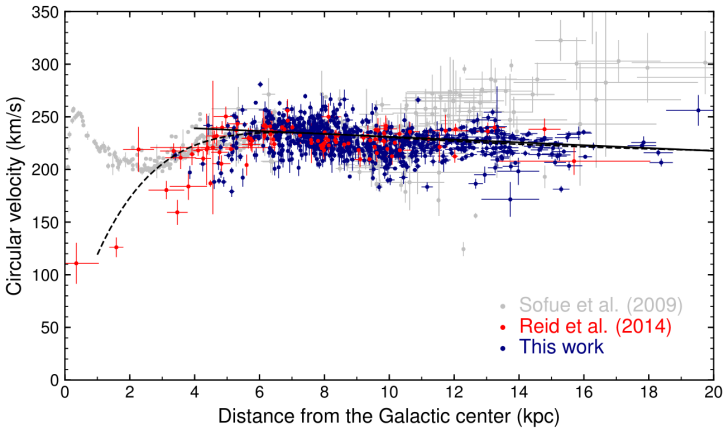


Figure 2.3: Rotational curve of the Milky Way. Taken from.[Mró+19]

However this measurement does not rule out some kind of astrophysical DM consisting of e.g. faint stars or black holes. Though the baryon abundance in the universe makes the concept of particle physics DM more viable[Sch06].

One possibility for such a particle physics approach is the existence of a weakly interacting massive particle (WIMP)[Sch06; Ams]. This refers to a particle with a mass of ≈ 0.1 to 10 TeV, which is weakly interacting[Åke+18]. This scenario has been the main focus of research so far, having produced powerful null results. The simplest scenarios can be ruled out by several orders of magnitude and the remaining parameter space can be covered by upcoming experiments[Åke+18].

The other possibility is to investigate a lower range of masses of ≈ 0.1 to 1 GeV, which is still consistent with the cosmological model of the early universe (see [Ams] for details.). Furthermore this so called light thermal dark matter (LDM) is motivated by the dark sector scenario(see Section 2.3). It thus makes for a compelling possibility, which encourages additional investigation.

2.3 The Dark Sector

Section 2.2 gave a brief overview over experiments and observations, which show the need for DM. The modularity of the SM's formulation as a quantum field theory allows for the addition of new terms to the Lagrangian. These terms can account for new particles and interactions. Constraints on these new terms are only given by the well established symmetries of the SM.

One possible solution to the introduction of DM is the addition of a dark sector, which consists of one or more DM and mediator particles. The coupling to the SM is achieved by a so called portal interaction. The nature of this portal interaction depends on the spin and parity of the corresponding mediator between the SM and dark sector[Ale+16]. Due to constraints from gauge and Lorentz symmetries of the SM, only four possible portal

interactions remain. An overview of them can be found in [Ale+16]. One of the most viable models for LDM is the introduction of a massive spin-1 mediator particle with gauge group $U(1)_D$ - the dark photon, along with additional DM particles.

The next sections give a brief introduction into the idea behind the theory and a possible implementation into the framework of the SM.

2.3.1 Implementation as a Quantum Field Theory

Dark Photons

One of simplest ways to introduce dark photons is via minimal kinetic mixing. The Lagrangian for the dark photon part of the theory then can look like Eq. (2.2)[Ale+16].

$$\mathcal{L}_{A'} = -\frac{1}{4}F'^{\mu\nu}F'_{\mu\nu} - \frac{1}{2}\frac{\epsilon}{\cos\theta_W}B^{\mu\nu}F'_{\mu\nu} + \frac{1}{2}m^2_{A'}A'^{\mu}A'_{\mu} \tag{2.2}$$

Here $F'_{\mu\nu}$ is the field strength tensor of the dark photon field, $m_{A'}$ is the dark photon mass, A'_μ is the dark photon vector potential, θ_W is the weak mixing angle and ϵ is a dimensionless coupling parameter. The hypercharge field strength tensor $B_{\mu\nu} = \partial_\mu B_\nu - \partial_\nu B_\mu$ is defined via the hypercharge field B_μ. B_μ is the gauge boson to the $U(1)_Y$ symmetry group of the SM.

The connection to QED is not immediately obvious due to the nature of the B_μ boson. It is used in the unification of the weak and electromagnetic forces and can be decomposed into a photon(A_μ) and a Z (Z_μ) part[Sch17]:

$$B_\mu = \cos\theta_W A_\mu - \sin\theta_W Z_\mu$$

Taking only the kinetic mixing with the photon field into account[2], the relevant term in Eq. (2.2) becomes:

$$-\frac{1}{2}F_{\mu\nu}F'^{\mu\nu}$$

This process changes the mass eigenstates and interactions of the involved vector bosons[CGR10]. They can be diagonalised and transformed to the new mass eigenstates. The process is not repeated here, but the corresponding calculation can be found in [CGR10]. The result of this transformation is the Lagrangian in Eq. (2.3)[Åke+18].

$$\mathcal{L}_{A'} = -\frac{1}{4}F'^{\mu\nu}F'_{\mu\nu} + \frac{1}{2}m^2_{A'}A'^{\mu}A'_{\mu} - \epsilon e A'_\mu J^\mu_{em} \tag{2.3}$$

Here J^μ_{em} is the electromagnetic current, which couples the dark photons to SM fermions. It can be written as $\sum_f Q_f \bar{\psi}_f \gamma^\mu \psi_f$, where Q_f is the charge of the respective fermion, ψ_f its spinor, γ^μ a Dirac matrix and the sum runs over all fermions.

[2] The part which couples to the Z boson will be neglected, because of the low cross section for interactions via the weak force.

Dark Matter Particles

To complete introduction of the dark sector, there is still the need for a DM particle. This can be accomplished by adding more terms to the Lagrangian. These can have the form given in Eq. (2.4)[Åke+18].

$$\mathcal{L}_{DM} = -g_D A'_\mu J^\mu_D + \text{corresponding mass term} \tag{2.4}$$

Here g_D refers to the coupling of the DM particle to the dark photon and J^μ_D is the dark current, which couples the dark photon to the dark matter particle. The exact form of J^μ_D depends on the type of particle to be introduced. An overview of options can be found in [Åke+18]. The mass term also depends on the introduced particle.

2.3.2 Secluded and Direct Annihilation

A Thermal DM freeze-out model is a simple and predictive explanation for the DM abundance observed in the universe[Ale+16]. It is thus highlighted in further detail. In this model, DM and ordinary matter are in thermal equilibrium in the early universe. Once temperatures cool down enough, the freeze-out occurs. The equilibrium is essentially broken, but the matter distribution remains[Ams].

This freeze-out model requires a thermally-averaged DM annihilation cross-section $\langle \sigma v \rangle$ of $\approx 3 \times 10^{-26}\,\mathrm{cm}^3\,\mathrm{s}^{-1}$[Ale+16]. This and other cosmological constraints(see Section 2.2), put strict limits on the theory.

There is an important distinction for all DM types and mediators, which has to be made: Secluded annihilation into SM particles or direct annihilation[Ale+16].

Secluded Annihilation

Let m_χ be the mass of the DM particle. It is then assumed, that $m_{A'} < m_\chi$. DM mainly decays into A' pairs as can be seen in Fig. 2.4(a). The annihilation rate for this process is proportional to[Ale+16]:

$$\langle \sigma v \rangle \propto \frac{g_D^4}{m_\chi^2}$$

It is important to point out, that there is no dependence on ϵ, which is the dark sector SM coupling. It is thus challenging to test this possibility in the laboratory[Ale+16]. Additionally, this option is constrained by data from the cosmic microwave background, which rules out DM masses below $\approx 10\,\mathrm{GeV}$[Åke+18].

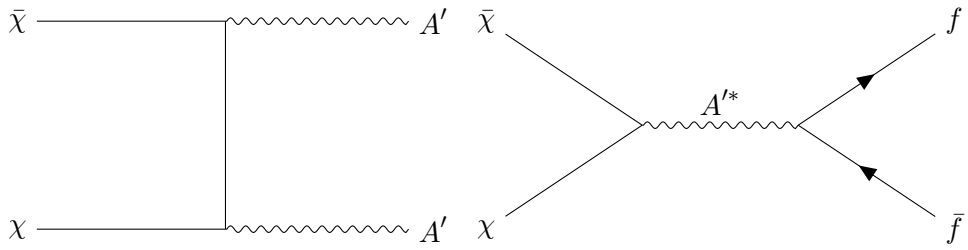
Direct Annihilation

Let m_χ again be the mass of the DM particle. The necessary assumption for this possibility is $m_{A'} > m_\chi$. The DM particle now decays via a virtual mediator into SM particles, as

can be seen in Fig. 2.4(b). The annihilation rate is now proportional to[Ale+16; Åke+18]:

$$\langle \sigma v \rangle \propto \frac{g_D^2 \epsilon^2 m_\chi^2}{m_{A'}^2} \propto \frac{y}{m_\chi^2} \quad , \quad y = \epsilon^2 \underbrace{\alpha_D}_{g_D^2/4\pi} \left(\frac{m_\chi}{m_{A'}} \right)^4 \tag{2.5}$$

This possibility offers a clear predictive target for discovery and falsifiability, as the dark coupling g_D and ratio $m_\chi / m_{A'}$ are at most $\mathcal{O}(1)$[Ale+16]. Consequently, there is a minimum ϵ in order to be consistent with the thermal history of the universe[Ale+16].



(a) Secluded Annihilation (b) Direct Annihilation

Figure 2.4: Secluded and direct annihilation of DM particles χ[Åke+18].

2.3.3 Possible Elementary Dark Matter Particles

It became clear in Section 2.3.2, that $m_{A'} > m_\chi$ in order for the theory to offer a predictive target and falsifiability. It is instructive to list possible choices for the DM particle χ and the corresponding J_D^μ in Eq. (2.4), whose relic density arises from direct annihilation.

- *Majorana Dark Matter*: The DM particle χ is a Majorana fermion in this case. It thus couples through an axial-vector current[Åke+18].

$$J_D^\mu = \frac{1}{2} \bar{\chi} \gamma^\mu \gamma^5 \chi$$

- *"Pseudo" Dirac Dark Matter*: In this case, χ can be a Dirac fermion. If the mass term for χ is $\mathrm{U(1)_D}$ preserving, this model is already constrained by cosmic microwave background measurements, unless there is a particle-antiparticle asymmetry[Åke+18]. If the mass term is $\mathrm{U(1)_D}$ breaking, analogous to $\mathrm{SU(2)_W}$ breaking mass terms in the SM, χ splits into to Majorana fermions(mass basis), which couple off-diagonally to the A'[Åke+18].

$$J_D^\mu = i \bar{\chi}_1 \gamma^\mu \chi_2$$

- *Scalar Elastic*: This scenario features a complex scalar χ with $\mathrm{U(1)_D}$ preserving mass terms[Åke+18].

$$J_D^\mu = i(\chi^* \partial^\mu \chi - \chi \partial^\mu \chi^*)$$

- *Scalar Inelastic*: In this scenario, χ is again a complex scalar. However the mass term is $U(1)_D$ breaking, analogous to $SU(2)_W$ breaking mass terms in the SM. Thus the coupling to A' is inelastic and χ must first transition to a heavier state in order to scatter through the current[Åke+18].

$$J_D^\mu = i(\chi_1^* \partial^\mu \chi_2 - \chi_2 \partial^\mu \chi_1^*)$$

Due to the different interaction with the dark photons, the parameter space looks different for each experiment. It is now possible to fix the ratio $m_\chi / m_{A'}$ and α_D and look at the resulting parameter space for a viable model for χ. This is shown in Fig. 2.5. The ratio and the value for α_D have been chosen to be disadvantageous to the parameter space[Åke+18]. Nevertheless it can be seen, that a LDM experiment can, combined with other upcoming experiments, cover the parameter space[Åke+18] and thus provides an integral part to research in this area.

2.3.4 Dark Bremsstrahlung

The signal process, which will be used to detect potential DM is called dark bremsstrahlung. It is analogous to the Bremsstrahlung process in Section 2.1.2. An incoming electron exchanges a photon with an atomic nucleus and then later emits a dark photon A'. The A' then decays into DM particles at a later time. Feynman diagrams for these processes can be seen in Fig. 2.6. This signal process creates a very clear signature in a possible detector[Åke+18]:

- The recoiling electron carries a small amount of the beam energy.
- The recoiling electron receives a s sizeable transverse momentum not present in the incoming beam
- Absence of any other particles than the recoiling electron in the final state, as the produced DM will likely have no further interaction with the detector

The detection of this process relies on the accurate measurements of the incoming and recoiling electron's momentum and is thus called missing momentum technique.

2.3.5 Background Processes

A detailed review of the background processes is needed, as the cross section for any DM process is rather small(see Section 2.3.2). Thus the rejection of non signal processes will have major influences on the detector design (see Section 3.1) An overview of the main background processes with their relative occurrence rate can be found in Fig. 2.7. Due to their influence on the detector design, it is instructive to review some of these background processes (for more information see [Åke+18]) in greater detail.

- *Incident low-energy particles*: at high enough integrated luminosity, there will be a sizeable number of low energy electrons. If their energies are consistent with energies

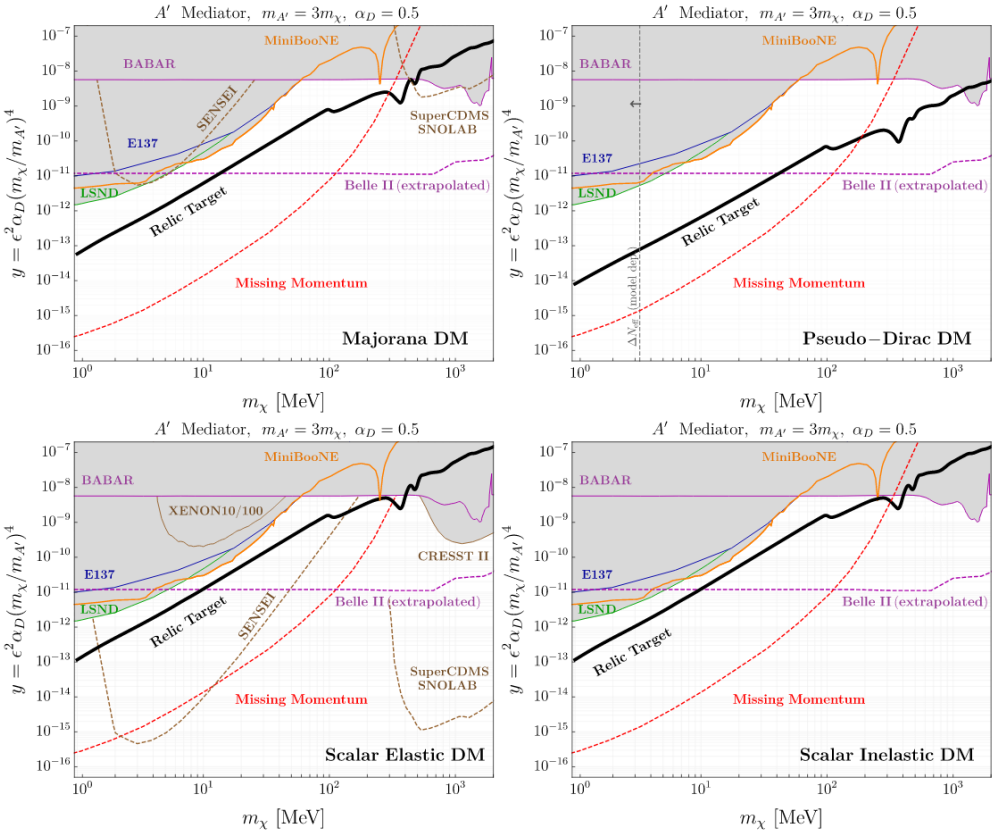
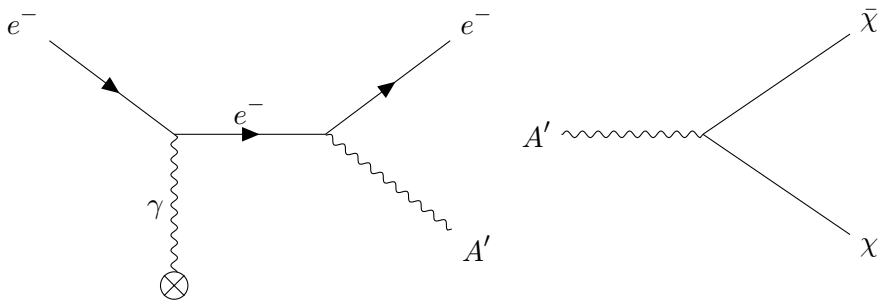


Figure 2.5: The parameter space for a LDM experiment for different models of χ. The relic target represents the set of parameters, which are consistent with the thermal history of our universe. The capabilities of other experiments are shown. "Missing Momentum" refers to potential capabilities of a LDM type experiment as it is described in this thesis. Taken from[Åke+18].



(a) Dark Bremsstrahlung (b) Dark photon decay

Figure 2.6: Dark bremsstrahlung and the following decay process of dark photons. Adapted from [Åke+18].

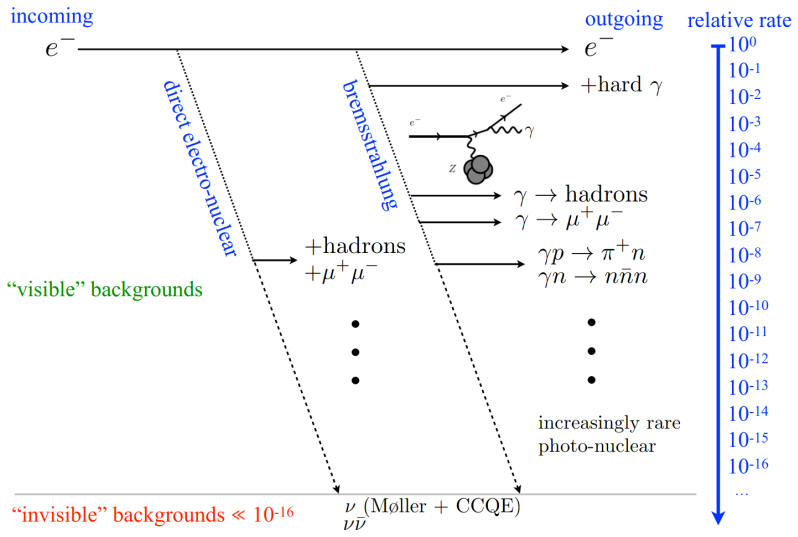


Figure 2.7: Background processes to dark bremsstrahlung with their relative rate for a linear accelerator. Taken from[Åke+18].

from recoiling signal electrons, they cannot be distinguished from signal recoiling electrons[Åke+18]. Thus [Åke+18] proposes a tagging tracker, which measures the momentum of incoming electrons.

- *Hard bremsstrahlung*: In events like this, the incoming electron loses a significant fraction of its energy through an ordinary bremsstrahlung event. If the recoiling electron's energy is consistent with signal event energies, it can be misidentified as a dark bremsstrahlung event. It is thus proposed to employ an electromagnetic calorimeter at the end of the beam line to veto this type of events(see Section 3.1)[Åke+18].

- *Interactions with hadron production*: If hadrons are produced in the target or anywhere else along the beam line, the incoming electrons can loose a significant amount of their energy which imitates signal events. [Åke+18] therefore proposes to introduce a hadronic calorimeter to measure the total energy in the final state together with the electromagnetic calorimeter. This energy measurement can then be used to veto such events.

2.4 The Accelerator - ELSA

The Electron Stretcher Accelerator(ELSA) is an electron accelerator located at the "Physikalisches Intitut" of the University of Bonn. The purpose of this thesis is to conduct a first simulation study of LDM type experiment at this accelerator. It is thus useful to point out its main features. A general overview of the experimental hall can be found in Fig. 2.8.

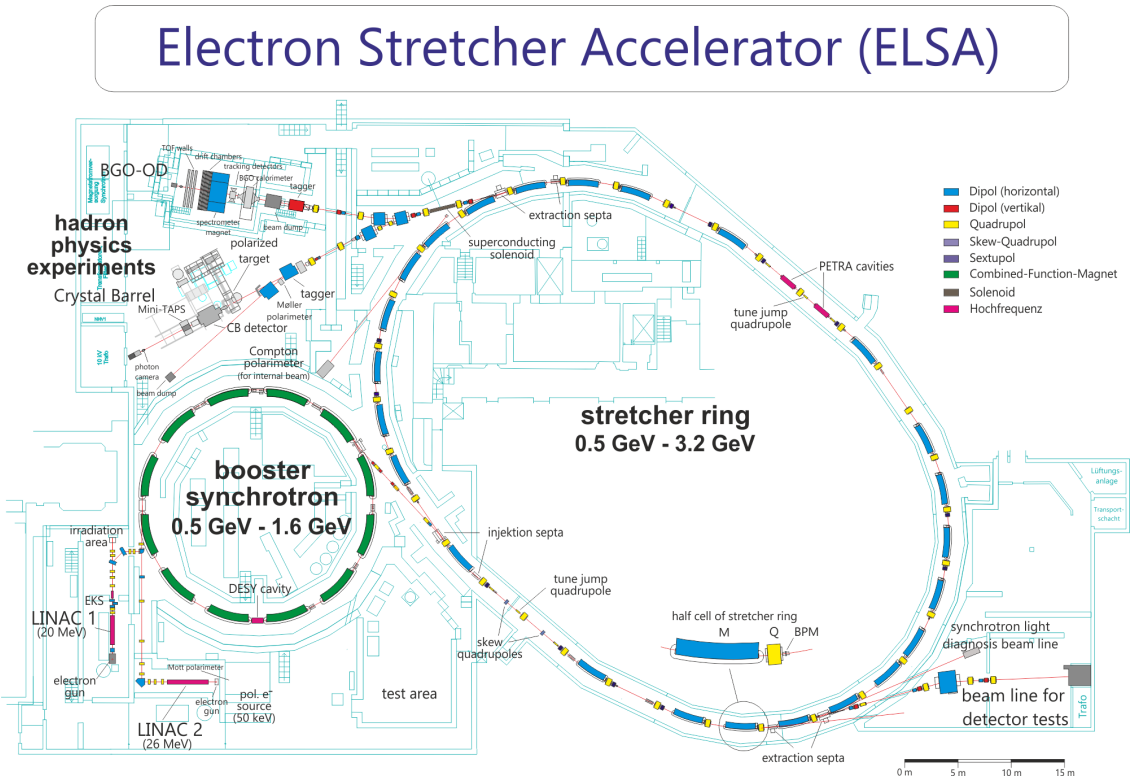


Figure 2.8: Layout of the ELSA experimental hall with pre accelerators and current experiments. Taken from [Fro19].

Its main features relevant for a LDM experiment include a well controlled electron beam with a maximum energy of 3.2 GeV. A possibly significant advantage compared to the setup proposed in [Åke+18] is the considerably better energy spread of the incoming beam of 0.8 ‰.[3] This might reduce the background from slow electrons (see Section 2.3.5) and thus improve overall performance. Another possible background reducing factor is the ability to extract very few electrons per event with a spacing of 2 ns. This makes it much easier to track the incoming and recoiling electrons, resulting in a possibly better momentum resolution.

These advantages exist due to the resonant extraction process used at ELSA.

2.4.1 Resonant Extraction

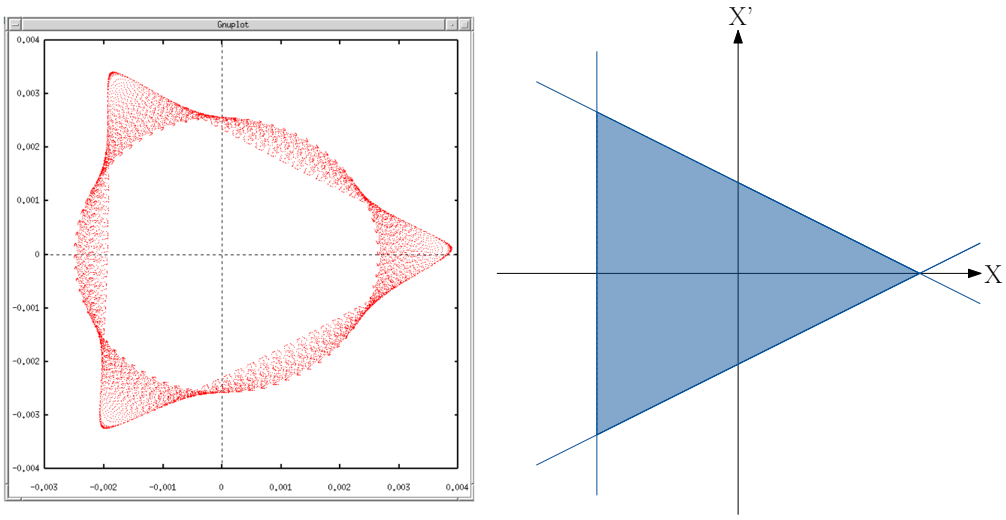
This section provides a brief overview into the physics behind resonant extraction. For a detailed discussion see [Pul99; Wil05].

This section will use the normalised coordinate system for accelerators. X refers to

[3] This value was obtained by personal communication with Dr. Frank Frommberger, who is working at the facility.

the horizontal coordinate perpendicular to the beam axis, Y is the corresponding vertical coordinate and μ the phase along the direction of the beamline[Pul99]. For extraction ELSA is driven at a horizontal betatron frequency Q close to 14/3[Fro19], which increases μ by $2\pi Q$ for one revolution. Primed coordinates refer to their derivative with respect to μ. Extraction happens in the horizontal X plane. This coordinate will thus be the focus of further discussion.

If a particle's position in phase space is mapped in a stable trajectory within the accelerator, the result is a circle in e.g. the X and X' coordinate[Pul99]. Once a sextupole field is introduced the trajectory in phase space changes and traces out a triangular shape at the sextupole position. Fig. 2.9(a) shows a particles trajectory during a slow(compared to the revolution time of the accelerator) increase of the sextupole field and outlines the transition between both shapes nicely.



(a) Particle trajectory in increasing sextupole field. Taken from [Pul99]

(b) Visualisation of separatrices

Figure 2.9: Particle trajectory in changing sextupole field and visualisation of separatrices.

The triangular shape appears, when particles are close to a resonance with 3 in its denominator. The shape is outlined by the so called separatrices, shown in Fig. 2.9(b). It is possible to show, that particles within the separatrices(shaded region in Fig. 2.9(b)) are on stable trajectories, while particles outside this region will move away from the main beam and gain distance with every revolution[Pul99]. The size of the stable region can be finely tuned by changing the sextupole field strength or putting particles closer to the corresponding betatron frequency. This allows for the selection of just a few particles with well defined momentum.

The separated particles can then be extracted by a septum magnet. This process is

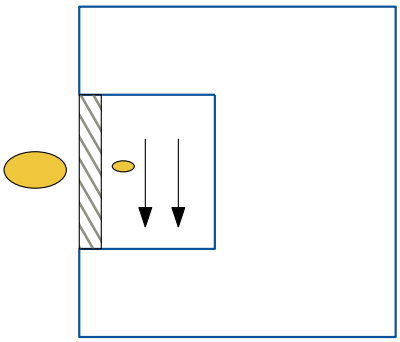


Figure 2.10: Visualisation of extraction with septum magnet. Arrows indicate the magnetic field. The yellow ellipses represent the main and extracted beam. Graphic inspired by [Wil05].

visualised in Fig. 2.10. If the distance between the main beam and the particles to be extracted becomes large enough, the extraction beam moves into the magnetic field of the septum magnet. A magnetic shield is used(black dashed section in Fig. 2.10) to keep the influence of the septum magnet on the main beam minimal. The extraction beam is deflected due to the field and can thus be transferred to a separate beamline and used for experiments.

Chapter 3

Detector Concept and Experimental Setup

This chapter gives a review of the experimental concept proposed for a light dark matter experiment (LDMX) at SLAC[Åke+18]. The main focus will be put on the tracker, as this comprises the main part of this thesis. Additionally adaptions for ELSA are discussed and described.

3.1 LDMX Detector Concept and Setup

The main design focus for the LDMX detector is on suppressing the most occurring backgrounds described in Section 2.3.5. A schematic overview of the proposed design can be found in Fig. 3.1. The experimental setup is mostly guided by the requirement to

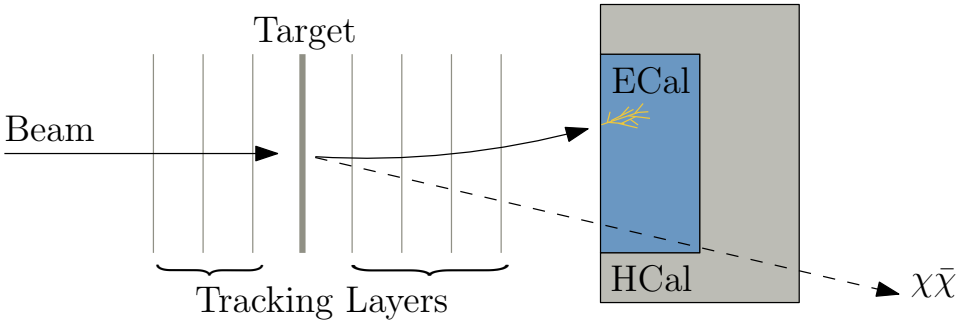


Figure 3.1: Schematic of the detector concept proposed in [Åke+18] with all relevant components. Figure adapted from [Åke+18]

precisely track every single incoming electron due to the rarity of signal events. This puts strict requirements on the detector as faster readout speeds and signal processing allow for more electrons on target.

The LDMX setup can loosely be divided into six main parts. The incoming electron beam, the tagging tracker in front of the target, the target, the recoil tracker behind the target and the electromagnetic and hadronic calorimeters.

Fig. 3.2 gives an overview of the entire setup in software. The 88.9 cm magnet gives a sense of scale and shows the way different components are arranged.

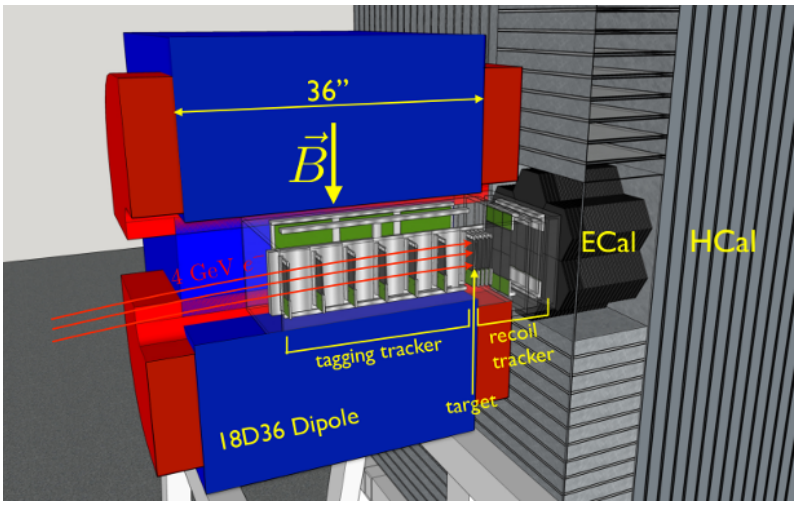


Figure 3.2: The LDMX setup proposed in [Åke+18] in simulation software. This provides an overview of the entire setup as a whole. Taken from [Åke+18]

3.1.1 Incoming Beam

The incoming electrons have a momentum of 4 GeV. The exact beam parameters are determined by the respective accelerator. The beam spread can also be adjusted to change the amount of electrons per extraction as a higher spread allows for the tracking of more electrons.

3.1.2 Tagging and Recoil Trackers

The tagging tracker consists of seven layers of silicon strip detectors and is placed into the central field of a dipole magnet with a field of 1.5 T. Its main purpose is to measure the momentum of incoming 4 GeV beam electrons. Thus it is used to provide a veto against slow electrons from the beamline[Åke+18].

The recoil tracker is similar in construction to the tagging tracker and consists of six layers of silicon strip detectors. It is placed in the fringe field of the 1.5 T dipole magnet. This tracker can cope with a weaker magnetic field, because it should track the low momentum electrons (50 to 1 200 MeV).

3.1.3 Target

The target can theoretically be any kind of material. The signal dark bremsstrahlung events will occur here. Reference [Åke+18] proposes a 10 % radiation length ($0.1X_0$) sheet of tungsten. Other combinations are possible and might be even more viable. However the target thickness is a compromise of signal event rate and transverse momentum resolution[Åke+18].

3.1.4 Calorimeters

The electromagnetic calorimeter is a high granularity Si-W sampling calorimeter, whose design is based on the high granularity calorimeter for the upgrade of the CMS experiment[Con+15]. Its main purpose is the detection of photons and electrons. This allows for the rejection of hard bremsstrahlung events by reconstructing the energy of recoiled electron and photon, which, if summed up, should be close to the incoming electron momentum[Åke+18].

The hadronic calorimeter is a scintillator based sampling calorimeter. It is used to contain showers, which escape the electromagnetic calorimeter. Furthermore it is able to detect neutral hadrons. It therefore provides a veto for hadronic background events.

3.2 Proposed Adaptions and Design Considerations for ELSA

The goal is to make a first simulation of a LDMX type experiment at ELSA. Consequently the design is inspired and similar to the one described in [Åke+18]. However there are some adaptions and changes to be made, in order to adapt this missing momentum approach to a different accelerator (see Section 2.4).

3.2.1 Beam Parameters

Due to the fact, the electron beam is produced at a different accelerator, the beam parameters are different. The most important one being the lower energy of 3.2 GeV and the possibility of extracting few electrons per event by using resonant extraction (see Section 2.4.1). For details about other parameters such as opening angles see Section 4.4.6.

3.2.2 Magnetic Field

The ELSA experimental hall already contains a large open dipole magnet (OD magnet) with a maximum field of more than 0.5 T for the BGO-OD experiment[Fre17]. This can probably be increased to about 1 T[1]. The OD magnet has a vertical opening of 84 cm and a horizontal opening of 150 cm respectively[Fre17]. Its larger interior might offer enough space for both tagging and recoil tracker to be placed inside the homogeneous part of the magnetic field. This will however depend on final number of detector layers and their placement. Furthermore there already exists a complete map of its magnetic field. This is a useful starting point for the simulation as no rough approximation for the magnetic field has to be made.

Even though it is possible to use a different magnet, the fact that a large enough dipole magnet already exists, offers an interesting opportunity to test the achievable performance with already existing equipment.

[1] This value was obtained in personal correspondence with Dr. Frank Frommberger

3.2.3 Tracker Adaptions

As already stated in the beginning of this chapter, the main focus of this thesis is the simulation of the tracker. Instead of silicon strip detectors, pixel detectors are used. As the requirements for the detector components are demanding, the model for pixel detectors in this thesis in terms of size and geometry is influenced by considerations for upgraded pixel detectors at the High-Luminosity LHC[Mig16].

This is a viable template, due to similar considerations and requirements for the pixel detectors. These include high granularity for good momentum resolution, the need to transfer large data rates and minimising the material budget[Mig16]. Even though these parameters are desirable for most applications in which pixel detectors are used, the design in [Mig16] shows what is possible with current technology.

The main features relevant for this thesis are[Mig16]:

- $2\,500\,\mu m^2$ area per pixel. With dimensions of $50\,\mu m \times 50\,\mu m$ or $25\,\mu m \times 100\,\mu m$
- Silicon thickness of 100 to $200\,\mu m$
- Total material budget of $2.5\,\%$ of a radiation length
- Detection threshold of $1\,200\ e$ per pixel

It is also worth noting, that the installation of pixel detectors is simplified due to the geometry compared to the layered design in CMS. Since the pixel layers for this experiment are stacked along the beam axis instead of layered radially around the beam, the space is not as constrained. This makes it less complicated to deliver services such as cooling and read out to each layer.

The significantly better energy spread at ELSA compared to a linear accelerator (see Section 2.4) could eliminate the need for as many layers in the tagging tracker. Thus reducing the overall material budget which is desirable. This however, has to be tested in simulation.

3.2.4 Calorimeters

The design of the calorimeters can probably be mostly left unchanged as they fulfil the same tasks as in [Åke+18].

A limitation, which might have to be considered in the future are spatial constraints towards the back of the experimental setup. Assuming the experiment is placed around the already existing OD magnet, the space behind the target is limited (see upper left of Fig. 2.8 with the BGO-OD experiment. This section also shows the location of the OD magnet). This might have implications on the calorimeter design. Furthermore due to the proximity of institutes at the university, the entire radiation has to be contained in the experiment and can not be dumped into the wall.

Chapter 4

Simulation Setup and ExPlORA

This chapter describes required software packages and the setup of different parts of the simulation. It also serves as a general overview and explains basic and necessary steps. Special emphasis will be put on the simulation framework ExPlORA (Extended Pluggable Object-oriented ROOTified Analysis), which is used to perform simulations for this thesis. ExPlORA uses GEANT4 for physics simulation. It is thus useful to start with a short explanation on this software package.

4.1 GEANT4

GEANT4 is a toolkit for the simulation of the passage of particles through matter[Gea]. It provides a complete set of tools for all areas of detector simulation. It thus provides the user with the possibility to simulate an entire detector. As such it is a large and complex software package and a complete and thorough description is beyond the scope of this thesis. Detailed information and user support can be found on the GEANT4 support website: [Gea].

The GEANT4 toolkit uses Monte Carlo methods to simulate the passage of particles through matter. It makes extensive use of object-oriented programming, such that different aspects of the simulation can be grouped into individual parts of the toolkit. This includes detector geometry, materials, fundamental particles, response of sensitive detector components, elementary physics processes etc. Particularly noteworthy for the context of this thesis is the implementation of elementary physics processes[Gea].

The toolkit provides several physics lists, which describe elementary particle interactions. Due to necessary approximations, there is no perfectly fitting list for every use case and it is up to the user to choose the list best suited to his setup. It is also possible to edit and add new processes by changing or implementing a new physics list. This is probably the cleanest way to implement a dark bremsstrahlung process. Due to the large effort involved in this, a different way was chosen for this thesis, which is explained in Section 4.4.8.

4.2 ExPlORA

The software framework ExPlORA has been developed for the CBELSA/TAPS collaboration for use at the CBELSA/TAPS experiment in Bonn[Fre17] and is written in C++. The version used in this thesis was forked from the original version in 2011 for the BGO-OD experiment. It provides a working simulation for the BGO-OD experiment at ELSA. Hence large amounts of code can be reused and it is not needed to implement every aspect of the detector simulation from scratch. Thus, some parts from this experiment are mentioned and used throughout this thesis (e.g the OD magnet).

4.2.1 Idea and General Structure

The basic idea behind ExPlORA is to have a single framework for simulation and physics analysis, which can be configured via a markup language. The advantage of this is, that the program does not have to be recompiled after a change to the physics analysis is made. Furthermore, event by event simulation is done using GEANT4, while the physics analysis can be done with the ROOT software framework. Thus combining two powerful toolkits.

Plugins

ExPlORA makes extensive use of the object-oriented nature of the C++ language. Every functional part of the simulation is written as its own plugin and can be invoked at any time. The necessary flexibility is achieved by configuring each plugin with a XML file. An example of such a configuration can be seen in Listing 4.1.

```
<BTDetectorGeometry
  debug="0"
  name="magnetgeo"
  detector="magnet"
  origin="(0.0, 0.0, 276.5)"
  />
```

Listing 4.1: Example of a XML configuration file in ExPlORA. This example shows the geometry configuration for the OD magnet

It shows the configuration which describes the placement of the OD magnet. The entire content of Listing 4.1 is referred to as element. The tag *BTDetectorGeometry* shows the plugin, which is called by this element. The contents of the tag e.g. *debug* and *origin* are called attributes[Fre17]. The value for each attribute can be accessed by this instance of the BTDetectorGeometry plugin. In this case it would mean, that the magnet is placed at $(0, 0, 276.5)$ in the global coordinate system, the debug level is set to 0 and the plugin name is ”magnetgeo”. The debug level is used internally to determine the amount of additional debug information displayed on the console. The *detector* attribute is used internally, if other plugins might want to access the geometry class associated with the OD magnet.

It is easy to see how the ability to pass parameters over this system gives the necessary flexibility for generically written plugins to be used for different purposes. Another addition to the flexibility of this approach is that XML files can be included by other XML files,

resulting in a master XML file. Only this file is actually given as a parameter to ExPlORA at program start. This master file then includes all other XML files (this happens over several levels), which completely describe the simulation and subsequent analysis by configuring their plugins.

Exchanging Data between Plugins

In order to perform any kind of analysis, data needs to be exchanged between plugins. This is done by the use of containers. Some plugins can be set up to receive input containers and output data by setting a corresponding output container. The data within such a container can then be accessed by all other plugins via its name. The content of a container is pretty much arbitrary. It is possible to write custom C++ classes, which can be used as data object and consequently be written into and read from containers. These data objects can contain various amounts of information.

An example for such a custom data class might be a hit in some detector. It can contain information such as the position of the detector volume and the amount of energy deposited in it. The requirements for such a generic hit class will most likely be very similar for different detectors such as the calorimeter and a pixel detector. This shows another advantage of object-oriented programming, because this hit class can be used as base for both detectors with minor necessary changes being made by inheriting from this base and having custom hit classes for both detectors.

Making these data classes generic is also important for the analysis of data. Ideally the data objects from simulation and measurements should be the same, such that the same plugins can be used in the analysis. For this reason there are separated classes to describe the geometry and setup of the detector. This way the simulation with GEANT4 and the analysis can access needed information about the geometry.

4.3 Building Detector Components

A crucial part for this thesis is building custom detector components. Thus, it is instructive to review the process used to implement these components. It was already mentioned in Section 4.2.1, that the geometry for detector components is separate from any logic associated with it. This section explains how to implement the corresponding geometry. For the implementation of the logic needed to complete the simulation, see Chapter 5.

There are two necessary classes or plugins, which need to be implemented in order to place a detector component. A detector geometry plugin and a detector geometry builder. The idea behind this structure is the following: The geometry plugin holds all the information needed to place and construct the respective detector component. It also provides some basic methods to return this information to other plugins. The geometry builder takes the information present in the geometry class and then constructs the relevant detector component according to the specifications given in its code.

This structure is chosen in order to make the geometry data easily accessible to other plugins. The number of pixels in a detector or its position are typical pieces of information

provided by the geometry, which might be needed for plugins and parts of the simulation.

Section 4.4 shows how this is done for every detector component in the current setup.

4.4 Current Setup for Simulation

This section gives an overview over all detector components, which are used in the current state of the simulation and shows how the geometry and geometry builder classes are used to place detector components in the simulation.

4.4.1 Target

The easiest way would be to start with the target. This detector component is basically a solid box filled with a defined material. This can be seen nicely by looking at the geometry XML file in Listing 4.2.

```
<BTSolidBoxGeometry
  debug="0"
  name="Box"
  detector="target"
  boxlength="15"
  boxwidth="15"
  boxheight="0.03504"
  origin="(0,0,250.0)"
  material="Tungsten"
  />
```

Listing 4.2: The XML file for the target geometry with all relevant attributes

Most of the attributes are self explanatory. It should however be noted, that ExPlORA measures all distances in cm. Thus all dimensions of the box and the position of its geometric centre (the *origin* attribute) have to be given in cm. Consequently the resulting tungsten box will have dimensions of 15 cm × 15 cm × 0.03504 cm (0.03504 cm correspond to 10 % X_0 for tungsten[PW]). All geometry classes also provide the ability to rotate detector components with Euler angles. This has not been used in this case and thus does not show up in the XML file. The *name* and *detector* attribute are not as intuitive. The value given in *name* is used to refer to this plugin in debug messages, while the value in *detector* is the program internal name for the detector component. This usage becomes clear by looking at the XML configuration for the geometry builder in Listing 4.3

```
<BTSolidBoxGeometryBuilder
  detector="target"
  />
```

Listing 4.3: The XML file for the target geometry builder with all relevant attributes

It can be seen, that the geometry builder only takes the *detector* attribute and then constructs a solid box of Tungsten with the corresponding parameters. The final result can be seen in Fig. 4.1 and is, as one might expect, a solid box.

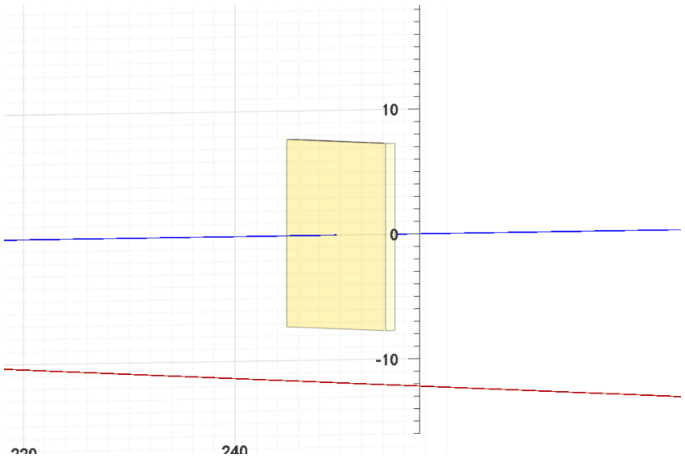


Figure 4.1: The finished target geometry in simulation. The coloured lines indicate the position of the axes and can be ignored for merely observing the target.

4.4.2 Magnet and Magnetic Field

It was already mentioned in Section 3.2.2, that it would be ideal to use the already existing OD magnet. This magnet is easy to implement into the simulation as it already has a working geometry, geometry builder and field map. The magnet is thus included similarly to the target. Listing 4.4 shows the respective XML file for both the geometry and geometry builder. The attributes are already known from Listing 4.2 and Listing 4.3. It should however be noted, that the magnetic field map does not appear here and is included at a different part in the master XML file. The field map plugin also uses the *detector* attribute in order to be assigned to the correct magnet.

```
<BTDetectorGeometry
  debug="0"
  name="magnetgeo"
  detector="magnet"
  origin="(0.0, 0.0, 276.5)"
  />


<BTMagnetDetectorGeometryBuilder
  detector="magnet"/>
```

Listing 4.4: The XML snippet for the OD magnet geometry and geometry builder

The end result of including the magnet with corresponding field lines can be seen in Fig. 4.2.

4.4.3 Pixel Detectors

The pixel detectors are arguably the most important part of the detector and the focus of this thesis. However this section will only describe the geometry and their setup. For all

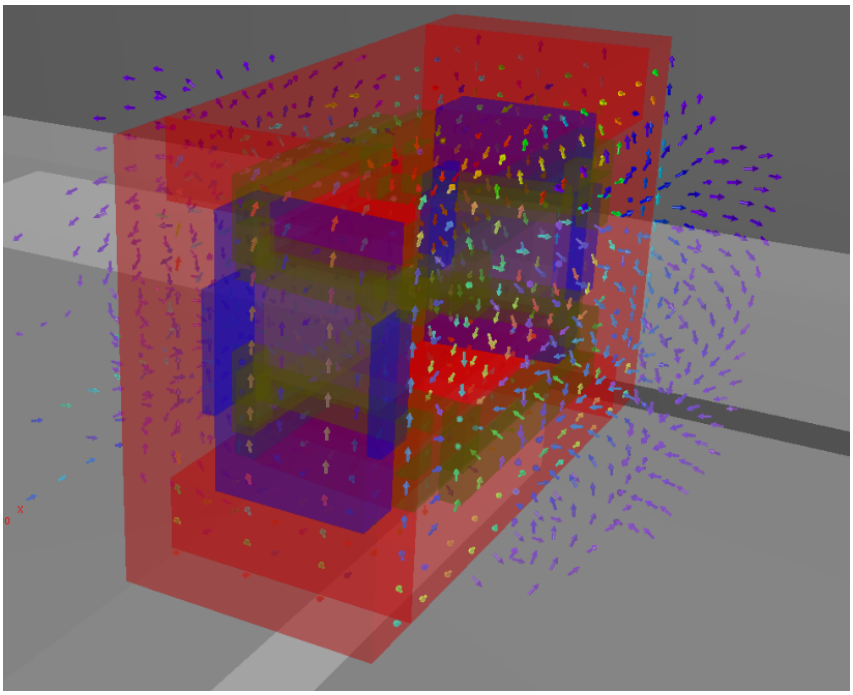


Figure 4.2: OD magnet with corresponding field lines in the graphic event display. The colour indicates the strength of the magnetic field at the respective position.

the necessary logic, see Chapter 5.

The approach for silicon pixel detectors in this thesis is highly simplified. The box geometry from the target is taken as a starting point. This means, that a layer of pixels is approximated by layer of silicon as sensitive volume, followed by a layer of some generic polymer. The silicon thickness is chosen according to the values given in Section 3.2.3. Consequently the polymer thickness is chosen in order to be consistent with the total thickness in radiation lengths of one pixel layer.

It is important to note, that several aspects one normally has to consider are not investigated in this approximation. All services (cooling, power, data transfer, bump bonding to readout chips etc.) and support structures are neglected and approximated by the polymer layer. Particularly noteworthy is the negligence of timing and data transfer, as these are a challenging part of this experiment. However, this is justified as this serves as a first and rough simulation of the experiment, which is limited by the scope of a master thesis.

Listing 4.5 gives an overview of all the attributes, which can be used to configure and place a pixel layer. Some attributes are already known from the placement of magnet and target. The attributes *nrows* and *ncols* determine the number of rows (y-axis) and columns (x-axis) in which the pixels are arranged. Consequently the *elements* attribute is the product of the two and needed for the digitizer (see Section 5.1). The *pixellength*, *pixelwidth* and *pixelthickness* determine the dimensions of the sensitive layer in the x, y and

z-axis respectively. The size of the polymer layer is determined by the dimensions of the sensitive layer in the x-y plane and by the attribute *fillerthickness* in the z direction. The materials for the sensitive and polymer layer are determined by *material* and *fillermaterial* respectively.

```
<BTDetectorGeometryCrudePixelDetector
 debug="0"
 name="SiTest1"
 detector="silicon_0_A_h"
 elements="2250000"
 nrows="1500"
 ncols="1500"
 pixelthickness="0.015"
 fillerthickness="1.17"
 pixelwidth="0.005"
 pixellength="0.005"
 origin="(-4,0.0,249.0)"
 material="Silicon"
 fillermaterial="Polyethylene"
 />

<BTCrudePixelDetectorGeometryBuilder
  detector="silicon_0_A_h"
  />
```

Listing 4.5: The XML snippet for a pixel layer geometry and geometry builder

The finished pixel geometry can be seen in Fig. 4.3. There is a similarity between this and Fig. 4.1, because the pixel detector geometry is based on the box used for the target. However, the two different layers (silicon and polymer) are clearly distinguishable by colour (yellow and green respectively). This shows, that this geometry plugin is working the way it is intended.

4.4.4 Calorimeter

The hadronic calorimeter is not implemented in this thesis. The electromagnetic calorimeter is modelled after the proposal in [Åke+18]. It is a sampling calorimeter and features hexagonal cells. The corresponding geometry configuration file is given in Listing 4.6. Note the unusual value for the *detector* attribute. Values for this attribute have to be chosen from an already existing names list in ExPlORA. Therefore a more descriptive name was unfortunately not available.

The calorimeter consists of hexagonal cells of side length *sidelength* and made of silicon with a thickness of *siliconthickness*. These cells are arranged radially into a larger module, such that the module consists of *numaxiallayers* layers. Each module is capped with a hexagonal W base plate of thickness *absorberthickness*. Seven of these modules are arranged as shown in Fig. 4.4 to create one layer of the calorimeter. Finally *numlayers* layers are stacked to create the completed electromagnetic calorimeter.

The calorimeter also has a working digitizer. See Section 5.1 for details on how digitising is done. Due to the complex geometry of the calorimeter, the position of each hexagonal

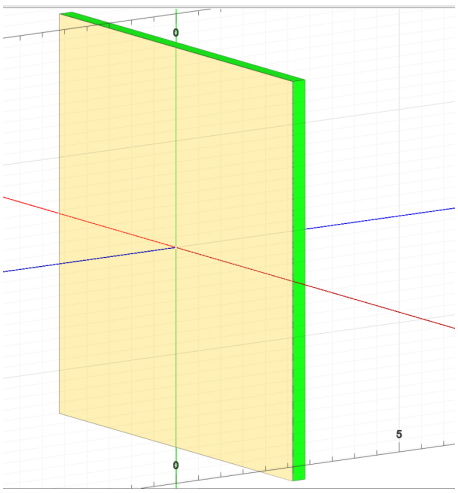


Figure 4.3: The finished pixel detector geometry in simulation. The two different layers (silicon and polymer) are clearly distinguishable by colour.

```
<BTDetectorGeometryECal
  name="ECal"
  detector="moeller1"
  origin="(0.0,0.0,296)"
  sidelength="0.447379"
  numaxiallayers="7"
  numlayers="32"
  siliconthickness="0.05"
  absorberthickness="0.646438"
  elements="28448"
/>
```

Listing 4.6: XML configuration for the electromagnetic calorimeter geometry

cell is saved in an array at build time. This allows the digitizer to access the position of each hexagonal cell by its index. The corresponding plot showing that this idea works in principle can be found in Fig. A.1.

Due to the high density and thickness of the calorimeter, a lot of interactions happen within its volume. This impacts the simulation speed significantly. For this reason, parts of the detector will be simulated separately. In particular, the calorimeter is only included when explicitly stated.

4.4.5 Complete Simulation Setup Overview

All parts of the geometry have been described so far. It is thus instructive to view all of them at once with several pixel detector layers as overview and important debug step. If mistakes were made in setting up different parts of the geometry or in naming conventions,

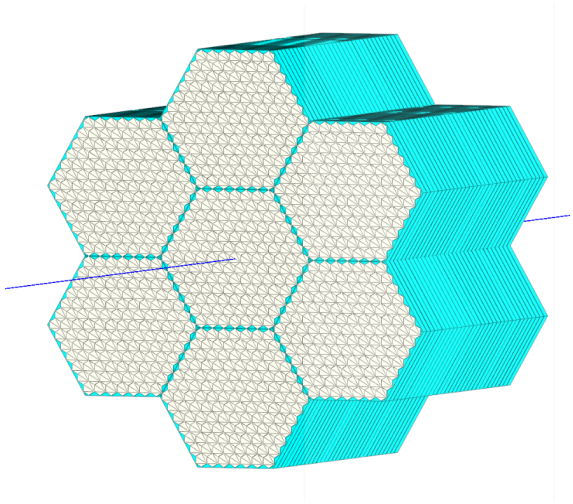


Figure 4.4: The finished electromagnetic calorimeter geometry in simulation.

these would now occur visibly. The complete setup can be seen in Fig. 4.5

It should be noted, that the number and placement of the pixel detectors is not final in this example and it only serves overview and debug purposes for the finished geometry.

4.4.6 Particle Emission

Particle emission is an essential part to completing the simulation. There was already an existing plugin in ExPlORA, which could generate electrons with a given momentum. It is however best to explain its properties by looking at the corresponding XML configuration file for the `BTVirtuaMCElectronBeamGenerator` plugin in Listing 4.7.

```
<BTVirtualMCElectronBeamGenerator
  electronradiatorposition="(-0.0,0,-600.0)"
  electronbeamdivergencex=".0171 * TMath::DegToRad() * 1000*1"
  electronbeamdivergencey="0.00458 * TMath::DegToRad() * 1000*1"
  generatedparticlecontainer="generatedparticles">
  <BTNumberGeneratorFormula
    name="Energy"
    formula="0.95/(2.5066*${EnergyDeviation})*exp(-(x-${ElectronBeamEnergy})^2/(2*(${
        EnergyDeviation})^2))+0.05*(x &lt; 3222)/(3.142)*(${EnergyDeviation})/((x-${
        ElectronBeamEnergy})^2+(${EnergyDeviation})^2)"
    min="${ElectronBeamEnergy}-40*(${EnergyDeviation})"
    max="${ElectronBeamEnergy}+40*(${EnergyDeviation})"
    />
</BTVirtualMCElectronBeamGenerator>
```

Listing 4.7: The XML configuration file for electron emission from a predefined source

The attribute *electronradiatorposition* defined the creation point for all created electrons. The opening angle for the electron beam can be defined with *electronbeamdivergencex* and *electronbeamdivergencey*. Due to the need to exchange data between plugins, the

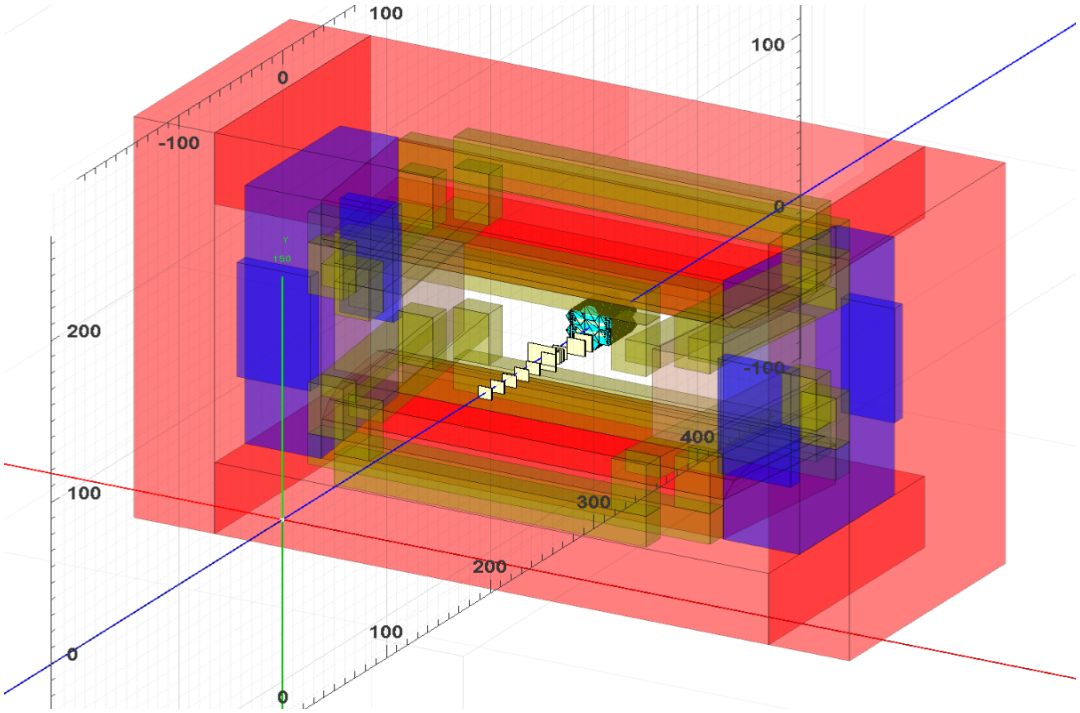


Figure 4.5: The entire simulation setup geometry in simulation with several pixel detectors layers, target, electromagnetic calorimeter and magnet.

created electrons are saved in a container with a name given by *generatedparticles*. In order to simulate the energy distribution of the accelerator, the generated momentum is chosen randomly according to a fully configurable probability density function (PDF). The `BTNumberGeneratorFormula` plugin is used for this. The *formula* attribute makes it possible to choose any PDF. The PDF for this simulation are given by a combination of a Gaussian with the energy and corresponding spread given in Section 2.4. Both values can be set in the master XML file. The other PDF in the combination is a 5 % contribution from a Cauchy distribution with the same width as the Gaussian. This has no effect for energy values larger than 3 222 MeV and is used to parametrize potentially longer tails not captured by the Gaussian. The *min* and *max* attributes are used to set the minimum and maximum possible generated momentum. A histogram with generated momenta can be found in Fig. 4.6.

4.4.7 Background Event Generation

Background events are generated by just running the `QGSP_BERT` physics list in GEANT4. Fig. 4.7 shows the total and transverse momentum of recoil electrons simulated with this physics list. They are obtained by plotting the Monte Carlo truth level information present in the simulation. It is a good consistency check to compare these histograms with

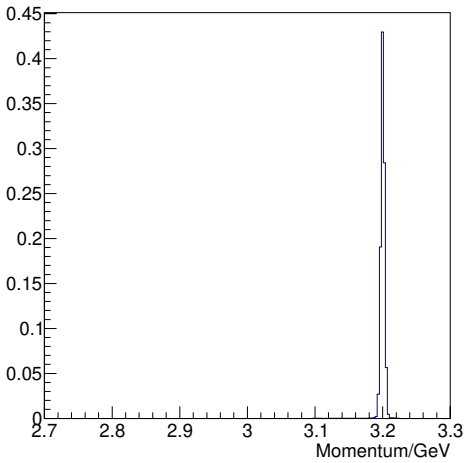


Figure 4.6: Distribution of generated electron momenta according to the PDF given in Listing 4.7. The histogram is normalised.

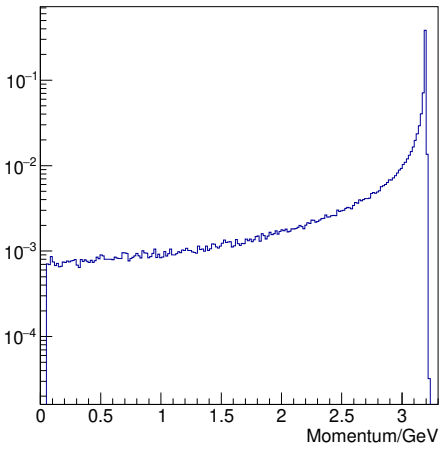
histograms for the same kinematic variables in [Åke+18]. For convenience they are given in Fig. 4.8.

In order to make a sensible comparison, it is required to look at the *Inclusive Single* e^- *Background* in the plots in Fig. 4.8. The transverse momentum is hard to compare due to the scale of Fig. 4.8, it is nevertheless included in Fig. 4.7. This consistency check is important, because it verifies the physics list and entire setup. It is thus possible to conclude, that this simulation setup works in principle and is able to produce meaningful results.

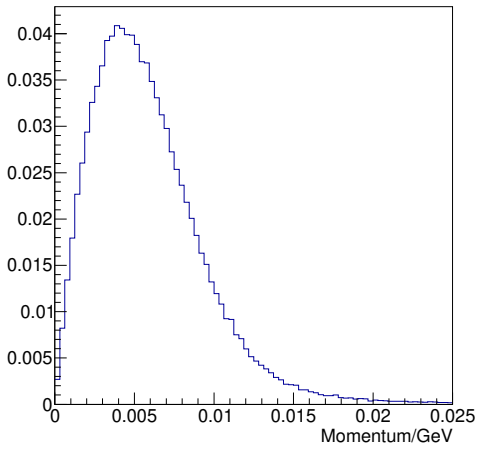
However, there is a disadvantage to this process of generating background events. It is desirable to increase cross sections of some background processes. This would make it possible to determine and simulate detector performance and response to these specific processes. An adaption to the physics list would be required to use this option. Since this is outside the scope of this thesis, no alterations have been done to the physics list. Consequently the likelihood of all background processes is determined by the default `QGSP_BERT` physics list.

4.4.8 Signal Event Generation

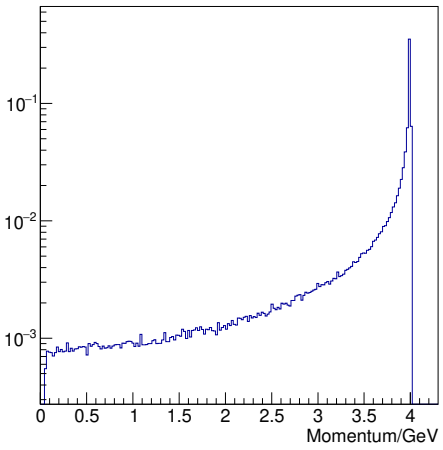
The most intuitive and cleanest way to introduce DM signal events into the simulation would be to change the underlying physics list used by the simulation. This would cleanly integrate signal events into the detector setup and is the simplest way to set up the simulation once integrated. However since no alterations to the physics list have been done, a different method is required.



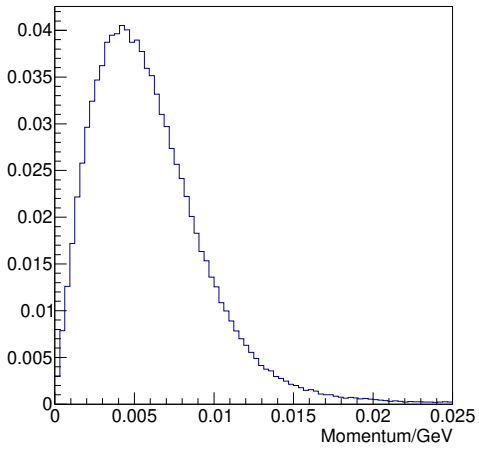
(a) Total recoil momentum of 3.2 GeV incoming electrons



(b) Transverse momentum of 3.2 GeV incoming electrons



(c) Total recoil momentum of 4 GeV incoming electrons



(d) Transverse momentum of 4 GeV incoming electrons

Figure 4.7: Total and transverse momentum of recoiling electrons, simulated with the standard `QGSP_BERT` physics list in GEANT4. The histograms are normalised.

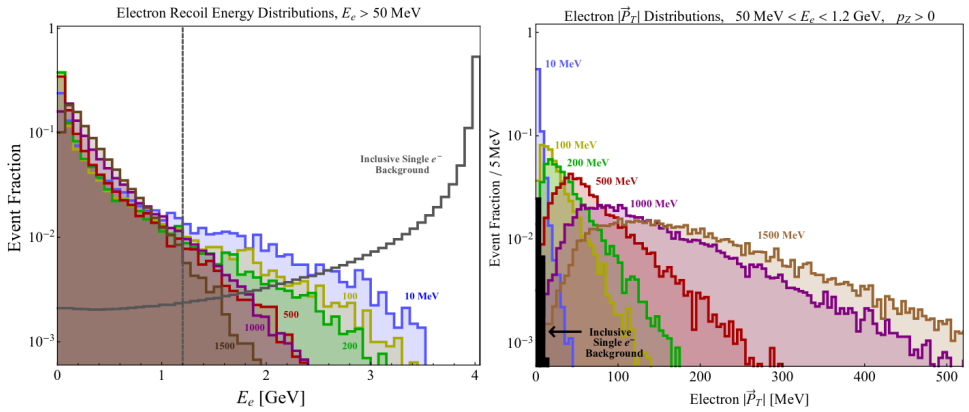


Figure 4.8: Total and transverse momentum of recoiling electrons from [Åke+18]. Applied cuts are given in the top. The horizontal line at 1.2 GeV represents a cut, which is used in subsequent analysis step in [Åke+18].

Courtesy of Ruth Pöttgen, it was possible to obtain the original code used in the simulation of signal events in [Åke+18]. For this reason it is instructive to give a synopsis of the signal generation process in [Åke+18]:

The dedicated event generator MadGraph/MadEvent4 is used to generate signal events via the $e^- W \to e^- W A', A' \to \chi\bar{\chi}$ reaction, where W represents a tungsten nucleus. The generated recoil electrons are then included in the simulation by letting their tracks start within the target. Appropriate smearing is applied to the starting position around the nominal centre of the beam. From this point the simulation with GEANT4 continues and takes this track as starting point, essentially not simulating all interactions in front of the target.

The code, which was obtained, contains a shell script to start the event generation as well as several configuration files. These files can be used to set simulation parameters before starting the event generation. This includes number of simulated events, beam energy, initial cuts on kinematic variables and more. Especially important is the ability to change the beam energy, in order to set it to the 3.2 GeV possible with ELSA.

The output of the simulation is a Les-Houches event file (LHEF). The format of these files is explained in [Alw+07; Boo+01] and summarised here. The files contain events in a XML style format. The beginning of each file contains information about the current run applied cuts, set energy and more. An example of a single signal event in this format is given in Listing 4.8.

```
<event>
 5   0  0.7108941E+06 0.9118800E+02 0.7818608E-02 0.1180000E+00
      11  -1   0   0   0   0  0.00000000000E+00 0.00000000000E+00 0.31999999592E+01
          0.32000000000E+01 0.51109989000E-03 0. 1.
     -623  -1   0   0   0   0  0.00000000000E+00 0.00000000000E+00 0.15732304348E-10
         0.17130000000E+03 0.17130000000E+03 0. 1.
      11   1   1   2   0   0 -0.17553389318E-01 0.29143206290E-01 0.37795374439E-02
          0.34234399013E-01 0.51109989000E-03 0. 1.
     -623   1   1   2   0   0 -0.17508839209E-01 -0.50844695272E-01 0.32314799632E-01
         0.17130001149E+03 0.17130000000E+03 0. 1.
      622   1   1   2   0   0  0.35062228527E-01 0.21701488982E-01 0.31639056221E+01
          0.31657541124E+01 0.10000000000E+00 0. 1.
</event>
```

Listing 4.8: Excerpt from a LHE event file showing a single signal event with all corresponding particles.

It is instructive to break down the contents of this element in order to understand the subsequent steps. The first line after the `event` opening tag contains general event information. Each column contains the following information (columns are described from left to right)[Alw+07; Boo+01]:

- `NUP`: Number of particle entries in this event
- `IDPRUP`: ID of the process for this event
- `XWGTUP`: event weight (The script generates unweighted events. Consequently this is the same for every entry)
- `SCALUP`: scale of the event in GeV, as used for calculation of PDFs
- `AWEDUP`: QED coupling constant for this event
- `AQCDUP`: QCD coupling constant for this event

The next five lines correspond to the kinematic variables of each particle involved in the reaction. Their meaning can be inferred from [Alw+07; Boo+01] and is repeated here (columns are described from left to right):

- `IDUP(I)`: Particle ID according to Particle Data Group convention
- `ISTUP(I)`: Status code (-1 for incoming, +1 for outgoing particles)
- `MOTHUP(1,I)`: Index of first mother particle
- `MOTHUP(2,I)`: Index of second mother particle
- `ICOLUP(1,I)`: Integer tag for the colour flow line passing through the colour of the particle
- `ICOLUP(2,I)`:Integer tag for the colour flow line passing through the anti-colour of the particle

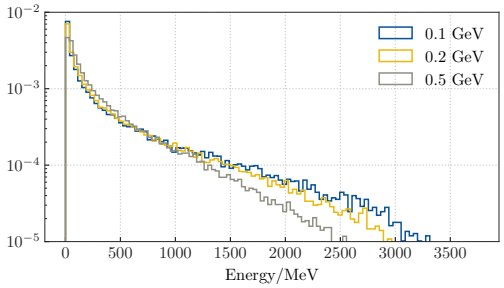
- `PUP(1,I)`: Momentum in x-direction in the lab frame in GeV
- `PUP(2,I)`: Momentum in y-direction in the lab frame in GeV
- `PUP(3,I)`: Momentum in z-direction in the lab frame in GeV
- `PUP(4,I)`: Energy in the lab frame in GeV
- `PUP(5,I)`: Rest mass of particle in GeV
- `VTIMUP(I)`: Invariant lifetime $c\tau$ in mm
- `SPINUP(I)`: Cosine of the angle between the spin-vector of particle I and the 3-momentum of the decaying particle, specified in the lab frame

This LHEF is then processed by a Python script. This has the advantage of being able to easily plot kinematic variables of signal events for consistency checks. Furthermore, it is convenient for filtering out unnecessary information for the implementation of signal events into the simulation. This becomes obvious when considering that only the kinematic variables of the recoil electron are required to start the simulation beginning at the target. Consequently, the Python script writes a new file, which only includes the information about the recoil electron. The output of this script would only be the third line of all event elements in the input file. This line corresponds to the outgoing scattered electron.

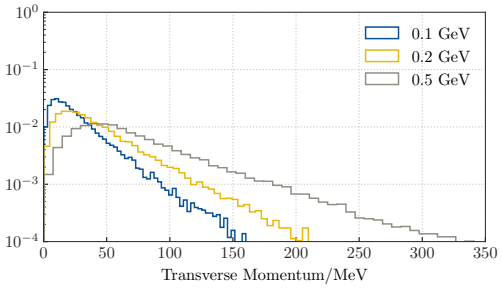
Fig. 4.9 shows a selection of kinematic variables of signal events produced in the way mentioned above. The beam energy was set to 4 GeV and the same cuts as in [Åke+18] have been applied. This allows for a comparison of the corresponding distributions (see e.g. Fig. 4.8). It can also be used as reassurance, that the code runs correctly on the setup in Bonn and the relevant kinematic parameters could be successfully extracted. For signal events at ELSA, the beam energy is set to 3.2 GeV. The analogous distributions of kinematic variables can be found in Appendix A.2.

The processed file can then be used as input to a similar signal generator as described in Section 4.4.6. The idea is that the already existing `BTVirtualMCElectronBeamGenerator` plugin can almost be completely reused. It already provides the ability to place an electron with a desired momentum anywhere in the simulation and can also apply smearing in every spatial dimension. The plugin is modified where the momentum is assigned to the generated particle. Instead of choosing it randomly according to the specified PDF, the already processed event file is read and the electron 3-momentum is set to the value given in this file. The electrons are placed with the correct uncertainties according to the values obtained in Section 7.1.

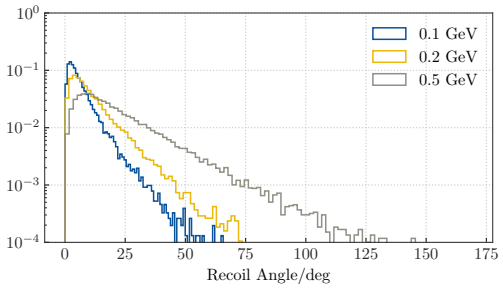
It is again advisable to do a consistency check and see, if the plugin fulfils its given task. For this reason Fig. 4.10 is included. It shows a signal event in the event display mode of the simulation. It can be seen, that an electron with significant transverse momentum (the straight line shows the direction of the initial momentum) and low energy (strong curvature in the magnetic field) is created in the area of the target. This is exactly, what one would expect. Thus, it is possible to conclude, that this plugin works and can be used to produce further results.



(a) Energy of recoiling electron



(b) Transverse momentum of recoiling electron



(c) Scattering angle

Figure 4.9: Shown are selected kinematic variables of signal events generated wit a beam energy of 4 GeV. The same cuts as in[Åke+18] have been applied, in order to make them comparable with [Åke+18]. See e.g. Fig. 4.8. Histograms have been normalised with their integral.

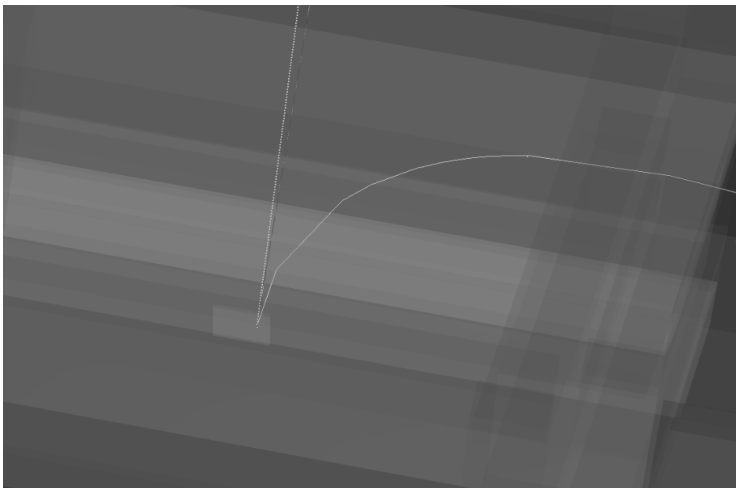


Figure 4.10: A signal event shown in event display mode of the simulation. It was generated, saved in a LHEF, processed with a Python script and then put into the GEANT4 simulation. The straight dashed line shows the direction of the generated momentum. The original colour has been desaturated to increase visibility in the printed version of the thesis.

Chapter 5

Simulating Pixel Detectors

5.1 Digitizer

The first step to analysing the simulated data is digitising the energy deposits simulated by GEANT4. This is done by calling the `BTVirtualMCDigitizer` from ExPlORA. It is easiest to explain its working principle by looking at the corresponding XML configuration file in Listing 5.1. The *name* and *debug* attribute are already known from other plugins in Chapter 4. All other relevant attributes, which are useful to illustrate the working principle are explained in the following.

```
<BTVirtualMCDigitizer
    folder="VMCDebugHistos/digitizer2"
    name="testdetectordigitizer2"
    debug="8"
    container="silicon_1_A_h"
    tdcdatatype="MTDC"
    adcdatatype="ADC"
    detector="silicon_1_A_h"
    format="silicon_1_A_h_%d"
    energythreshold="0.00432"
    adcgain="0.0001"
    tdcgain="1"
    pixeldetectorwithoneelement="true"
    />
```

Listing 5.1: The XML file for the digitizer plugin corresponding to the detector `silicon_1_A_h`.

If a sensitive detector volume registers an energy deposit, it calls upon the digitise method in the digitizer. This method has access to the Monte Carlo energy deposit and an identifying linearised index. The index can be inferred from the volume name, which is called upon the digitizer. When building the pixel detector, it is important to give each volume, which comprises the pixel detector, a name according to the *format* attribute. This might look like this: Consider a detector layer with 10×10 pixels. Assuming there is an energy deposit in the second row and first column, the corresponding pixel volume would have the name `silicon_1_A_h_11`. Contrary to the default in many programming

languages, indexing in ExPlORA starts at one. Since the detector name is given in the *detector* attribute, it is possible to infer the correct index.

The digitise method then creates a sensitive detector object (see Section 5.2), which is put into a container for later use in other plugins. This container is named by using the *container* attribute appended by *_SensDet* to avoid ambiguities. If the energy deposited is below the value given in *energythreshold*, the sensitive detector object will not be created. The value of of 0.00432 corresponds to the energy required to create 1 200 e in silicon in MeV.

The corresponding sensitive detector object will have access to the index mentioned before. However, due to ExPlORA aiming to provide a seamless transition between simulated data and actual measurements, the energy is not directly stored in this sensitive detector object. The digitizer is meant to output hardware data, which is the same as the data obtained by an actual experiment. This might seem unnecessarily complicated for a thesis based on simulation. Nevertheless it was more convenient to use this already existing code base and build upon it, rather than developing a completely new digitizer. Due to this, the energy is stored as ADC value in the sensitive detector object. This value is computed by dividing the deposited energy in MeV by the value given by the *adcgain* attribute. It is important to choose a sufficiently low number for this gain, because the ADC value is stored as an integer. Energy deposits of less than 1 MeV would thus be stored as zero and consequently not be measured.

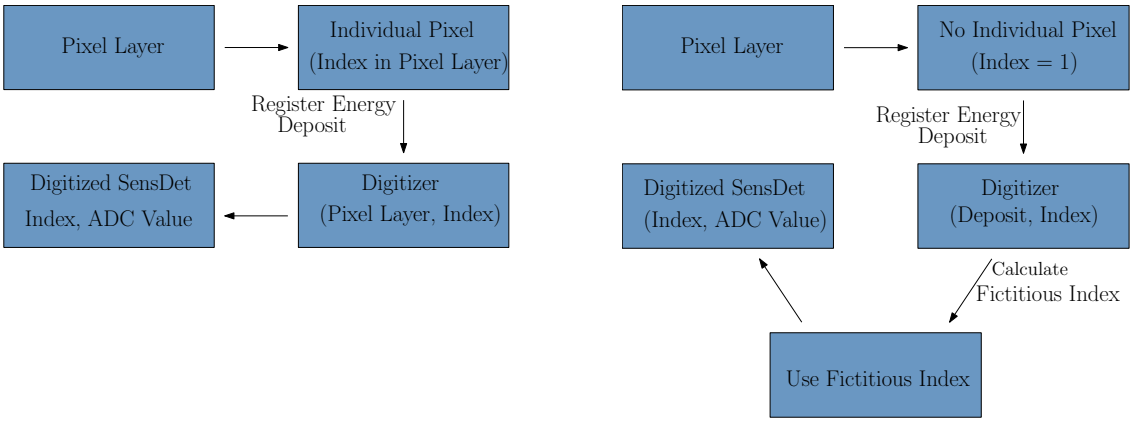
5.1.1 Digitising Highly Granular Pixel Detectors

Based on the current description in this section, it is reasonable to think that every pixel in a detector layer is its own volume. While this is intuitive and easy to understand in the framework which has been set up above, it has some crucial limitations. Every volume occupies space in an array in the simulation. This takes significant amounts of RAM and slows down the startup of the simulation. Furthermore it has an impact on the overall performance, because there is overhead with every volume introduced. Overall this means that having a single volume for each pixel is not feasible.

The solution is to make the sensitive area of the pixel detector layer out of one solid block of silicon[1]. This reduces the number of required volumes to a minimum. It does however introduce the problem of effectively having only one pixel for each detector layer. This is avoided by modifying the digitizer. The process is summarised as a flow chart in Fig. 5.1(b) in comparison with a flow chart describing the basic digitizer in Fig. 5.1(a). It also illustrates the point, where the modification is made.

The index, which is usually used to infer an energy deposits position, would not work with this setup. However, the digitizer has access to the Monte Carlo energy deposit. The deposit contains its current location inside the local volume coordinate system. With the local volume being the sensitive layer of silicon, this position can be used to infer an index. This fictitious index serves the same purpose as the index a single dedicated pixel volume

[1] Courtesy of David-Leon Pohl, who gave the initial input to this idea.



(a) Flow chart for the digitisation process as described in Section 5.1

(b) Flow chart for the digitisation process as described in Section 5.1.1

Figure 5.1: Flow charts describing the digitisation process as comparison between Section 5.1 and Section 5.1.1. The abbreviation *SensDet* for sensitive detector was used.

would have at the same position. In order to calculate the fictitious index from the deposit position, a custom method from the geometry class is used. This is the easiest way to obtain the index, because all the information about the pixel layout is stored there.

Once this fictitious index is obtained, it can then be set as the actual index for the pixel containing the energy deposit. The created sensitive detector object will thus be given the fictitious index and every consequent step in the analysis can be left unchanged. In order to keep the digitizer working for other plugins, the additional attribute *pixeldetectorwithoneelement* was added. It can be used to switch off the overwriting of indices.

It should be mentioned that there are more memory conserving ways to do this. One way to see this is by considering, that usually every channel in a detector has to be calibrated. Thus, ExPlORA creates a calibration file for every pixel. The number of pixels is read from the *elements* attribute in the geometry (see Listing 4.5). This can cause a RAM need of several 10 GB, depending on the number of pixels. However the approach described above was chosen due to minimal changes required to an already existing code base and relative effectiveness. The introduction of fictitious indices allows for the simulation of virtually any amount of pixels with reasonable performance.

5.2 From Sensitive Detectors to Hits

The Digitizer creates sensitive detector objects. These objects have all the relevant hardware information. At this point it might be counter intuitive to split up the process of hit creation and have the digitizer create objects with hardware information, instead of

hits directly. The reason for this in the general application of the ExPlORA framework. Calibration for every pixel might be different and different parts of the detector might be rearranged over time. It is thus useful to have another plugin, which takes the hardware information and transforms this into physically relevant quantities.

This process is quite simple for this thesis. The channel numbers map to the detector indices one to one and the deposited energy can be directly inferred from the ADC value by multiplying with the corresponding gain. The hit position can be determined by using its corresponding index. Thus this step can be thought of as just changing the data type of the digitizer output. It is however important to distinguish these steps.

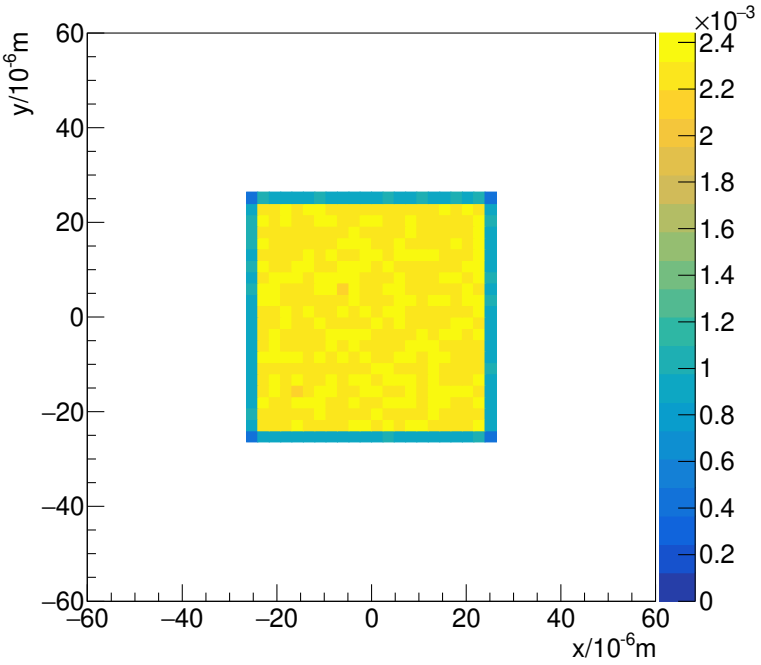


Figure 5.2: Difference of digitised hit position to true energy deposit position for one layer of the pixel detector with a pixel size of 50 µm × 50 µm. All pixels in the detector layer are summed up for this histogram. The histogram is normalised with its integral. The distribution is uniform in both axes. The edge has a lower count overall due to binning effects.

At this point physical information about the incoming beam is present. It is thus important to make a consistency check and show that the method, which is described above, actually works. Fig. 5.2 shows the difference of the actual energy deposit position and the digitised hit position in the x-y plane. This is done for one layer of the pixel detector and for all pixels within this layer. The pixels have a size of 50 µm × 50 µm. One

would expect a uniform distribution in both axes with the width of the pixel, because the hit is always digitised in the middle of the corresponding pixel. This is exactly what can be seen. The corresponding standard deviations of 14.43 µm and 14.42 µm are also consistent with this expectation. ($50\,\mu m/\sqrt{12} = 14.43\,\mu m$). It can thus be concluded, that the simulation has a working digitisation process.

5.3 Clustering

The next step to having a complete simulation of the pixel detector is writing a clustering algorithm. This will put appropriately spaced hits together into one cluster. The reason for this is that a single passing electron might deposit energy in multiple pixels of the detector. However, these multiple deposits in this case really correspond only to one physical hit.

The algorithm used in this thesis is relatively simple: Hits in adjacent pixels are grouped together into one cluster. It is again instructive to look at the corresponding XML file in Listing 5.2. The plugin is provided with information about input and output containers and the corresponding detector.

```
<BTClusterFinderPixel
  detector="silicon_1_A_h"
  name="clustering_silicon_1_A_h"
  prefix="clustering_silicon_1_A_h"
  srccontainer="silicon_1_A_h_hits"
  dstcontainer="silicon_1_A_h_clusters"
  folder="Pixel/Clustering/silicon_1_A_h"
  debug="0"
/>
```

Listing 5.2: The XML file for the clustering plugin corresponding to the detector `silicon_1_A_h`.

The plugin treats every hit in the input container as potential seed for a new cluster. Hits are grouped together if they fulfil a custom criterion. In this case it was chosen in such a way that hits from adjacent pixels are grouped into one cluster. Additionally a choice on the computation of the cluster position has to be made. One sensible solution is to calculate the energy weighted average of all hit positions in one cluster. Due to the different requirements of the BGO-OD experiment, ExPlORA did not have the capability to create clusters this way.

The solution to this problem was found with trying to keep alterations to the existing code base to a minimum. A new cluster class `BTClusterPixel` was added, which modified the method to calculate a clusters position upon its creation. All other features from the base class `BTCluster` remained unchanged. This ensures compatibility with other already existing plugins.

5.4 Assigning Clusters to Tracks

In order to use the clusters to fit the track momentum, it is required to group clusters together in tracks. This can be understood by considering an electron passing multiple detector layers and leaving clusters in each of them. All of these clusters would belong to the electron track. This step is especially important behind the target, where multiple particles can pass through several detector layers. Categorising which clusters comprise a track, is a complicated task and corresponding algorithms can become very sophisticated. Examples are Hough transformations and neural networks, as demonstrated in [Hei13] and [Far+17] respectively.

Due to the scope of this thesis, a simpler approach was chosen. The assignment is done using the available Monte Carlo truth level information. Obviously, this is not present in a real experiment and the cluster assignment will have to be changed. However as this is a first simulation study, this simplification seems to be justified. Nevertheless this part of the simulation has to be considered critically when interpreting the results. It is best to explain the concept of this assignment by looking at the corresponding configuration XML file in Listing 5.3.

```
<BTCombinePixelClustersToTracks
 inputvmctracks="VMCTracks"
 outputcontainer="PixelRecoTracks"
 outputcontainerforward="PixelRecoTracksForward"
 allowedposdiff="0.01"
 minimumnumberofclusters="3"
 inputclusters = "silicon_0_A_h_clusters,silicon_1_A_h_clusters,silicon_2_A_h_clusters,
     silicon_3_A_h_clusters,silicon_4_A_h_clusters,silicon_5_A_h_clusters,
     silicon_0_A_v_clusters,silicon_1_A_v_clusters,silicon_2_A_v_clusters,
     silicon_3_A_v_clusters,silicon_4_A_v_clusters,silicon_5_A_v_clusters,sciri_clusters"
 debug="9"
 />
```

Listing 5.3: The XML file for the assignment of clusters.

It is easiest to start the explanation with the *inputvmctracks* attribute. It holds the information on the Monte Carlo simulated tracks. The Monte Carlo track object contains all energy deposits, which were made by the respective particle. The two attributes *outputcontainer* and *outputcontainerforward* both describe the output container name and will contain the same objects after the assignment is done. There are two attributes for the same purpose due to compatibility reasons with other already existing plugins. The *inputclusters* attribute is a list of all cluster containers to be considered for the assignment of tracks. As can be seen, this list encompasses the clusters of the deployed pixel detector layers.

The plugin starts by going through every Monte Carlo track and the energy deposits in this track. Within this loop, every cluster from the list given by *inputclusters* is compared to the current Monte Carlo energy deposit. If the distance in the x-y plane is smaller than the given value in *allowedposdiff*, this cluster is added to a track and subsequently saved in an output container. If the output container does not have a track corresponding to

the Monte Carlo track in the current loop, the cluster can not be added. In this case a new track is created in the output container and the cluster is added to it. Added clusters are removed from the list for later iterations of the loop, ensuring that each cluster is only assigned to one track. The attribute *minimumnumberofclusters* provides an early level of filtering events. A created track in the output container must have at least three clusters in three different detector layers in this case. Otherwise the track will be deleted from the container and can not be used in later stages of the analysis. This can be useful, because a track of just two clusters can not be fit sensibly.

At this point, it is important to make a consistency check. This will show that both, clustering and their assignment to tracks, are working as intended. Fig. 5.3 shows the difference between cluster position and the associated energy deposit, which was used for the track assignment, for a single detector layer. Again the individual pixel can be seen, indicating that most clusters consist of just one hit. The circular halo reflects the *allowedposdiff* attribute in Listing 5.3. Fig. 5.4 shows the rate of different cluster sizes for one detector layer. It also verifies the assumption that the vast majority of clusters contain a single hit.

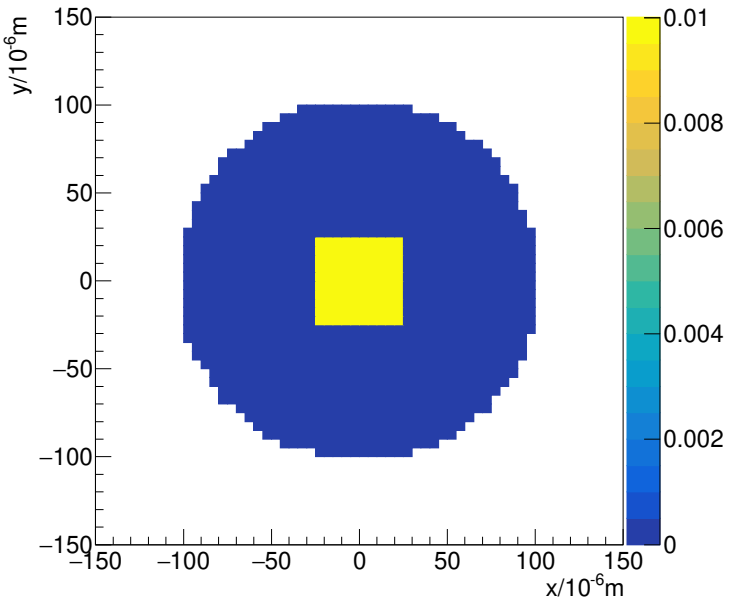


Figure 5.3: Difference between cluster position and energy deposit position for all pixel layers in the x-y plane. The histogram is normalised by its integral. The contour of an individual pixel can be clearly seen. The circular halo surrounding the pixel is due to the *allowedposdiff* attribute in Listing 5.3.

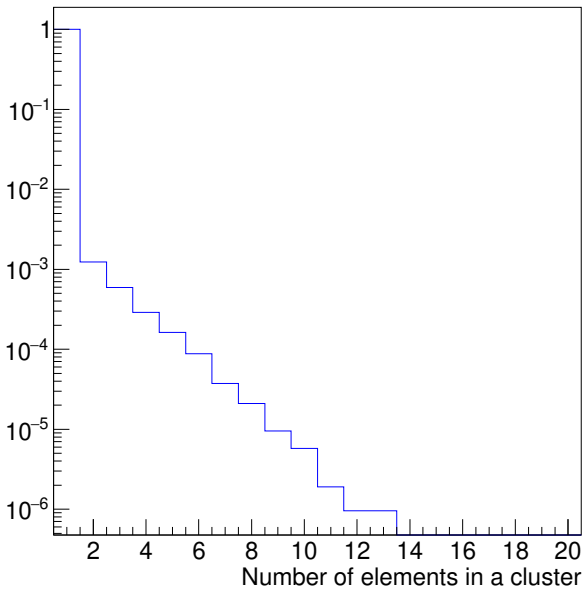


Figure 5.4: This histogram shows the rate of different cluster sizes for one detector layer with 1 046 531 simulated events. It is normalised by the total number of entries. The vast majority of cluster contain only a single hit.

The small cluster size can be explained by the fact that the digitizer (see Section 5.1) does not take the charge sharing with other pixels into account. Due to the drift of charged particles in the field of a silicon pixels, charge will also be collected in neighbouring pixels of the initially hit one. This would lead to a generally bigger cluster size. It would also improve the potential resolution, because the position calculation of the cluster is weighted with the individual hit energy. The process is described in [WSH20]. However, it is not implemented here due to the scope of this thesis. Consequently the resolution of each cluster is limited by the size of the individual pixels. Thus, the pixel resolution in this thesis provides a conservative estimate and higher resolutions can be achieved by using charge sharing.

Chapter 6

Implementing a Fitting Algorithm

The previous chapters explained and described necessary steps in a detector simulation to be able to fit the measured tracks. This chapter gives an overview over the already existent fitting algorithm and points out why it is not suitable for this detector setup. Additionally, the implementation of a new fitting routine, the Kalman filter is explained.

6.1 ExPlORAs Fitting Algorithm

ExPlORA already has an implemented fitting routine, which was used in the BGO-OD experiment. This section describes its general principle and points out its issues for a LDMX type experimental setup.

6.1.1 Idea and Principle

In order to explain the idea behind the existing fitting algorithm, it is convenient to start with the setup of the corresponding plugin. First, the entire detector geometry is crudely recreated in slices along the z axis into so called tracking regions. This has the advantage of being able to step through them during the fitting and having access to the geometry of the current component. It also gives the possibility to switch between tracking regions without associated clusters and those with. This can be used to change the subroutine used for propagating the particle within the tracking region volume (see Section 6.3.2). Furthermore, having access to the material of the current detector component allows for the computation of energy loss and multiple scattering.

The entire plugin loops over each track in its input container. In each iteration of the loop, the clusters within the tracks are added to their corresponding tracking regions. This way the fitting routine has a copy of the detector geometry (just in slices along the z axis) and the hits associated with the current track in their corresponding detector components via the tracking regions. The next steps are best explained with the help of Fig. 6.1.

The tracking regions are separated by vertical lines, which correspond to pixel detector layers. Technically each detector layer consists of two tracking regions due to the way they are built (see Section 4.4.3). However, this detail is neglected in the figure. The yellow dots represent the clusters belonging to the track shown as solid black line. The

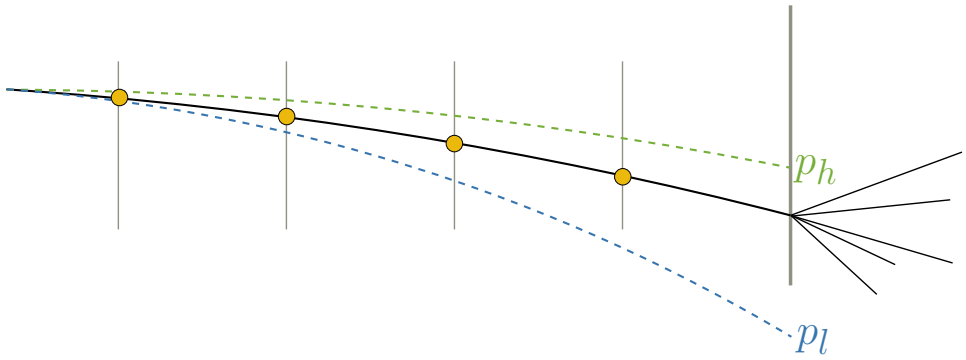


Figure 6.1: Schematic showing the existing fitting algorithm inExPlORA. Clusters are indicated by yellow dots, detector layers by vertical lines. The last vertical line represents the target. Thus this only shows a hypothetical setup in front of the target

last vertical line represents the target. Hence, this schematic only shows the experimental setup in front of the target. Nonetheless, this will be sufficient to explain the principle behind the fitting algorithm.

The first step is to guess the momentum's direction. Another plugin, which is run before the fitting routine, was written for this reason. Sensible choices are a momentum pointing into just the z direction or the difference vector between the first two detector layers. The user can choose a momentum range bounded by p_h and p_l. Next, the plugin tests this range by propagating the electron through all tracking regions. If a tracking region has an associated cluster, the χ^2 summand is calculated. A final χ^2 value can be obtained by adding the summands from all tracking regions with clusters.

The propagation is done by assuming a locally homogeneous field, which leads to a circular path. The electron is propagated along this arc. The magnetic field is updated and a new arc used for subsequent steps. The entire propagated track is thus a sequence of connected helical segments. The step size, i.e. the arc length can be chosen by the user and has to be sufficiently small in inhomogeneous fields.

When the track has been propagated through all tracking regions and tested for all eligible momenta, there will be one momentum value with the lowest χ^2. This is then used as the fitted momentum value. The process described above is inspired by a classical χ^2 fit.

6.1.2 Problems of the Old Fitting Algorithm

Even though parts of the code for this fitting algorithm had to be adapted in order to work with the setup in this thesis, it is not suitable and a different fitting routine had to be used. A description of appearing problems is given in the following.

Having to guess the initial momentum direction can be problematic, because the fitting routine does not allow for a change in the direction apart from propagation in the magnetic field. This is especially problematic behind the target, where the initial momentum

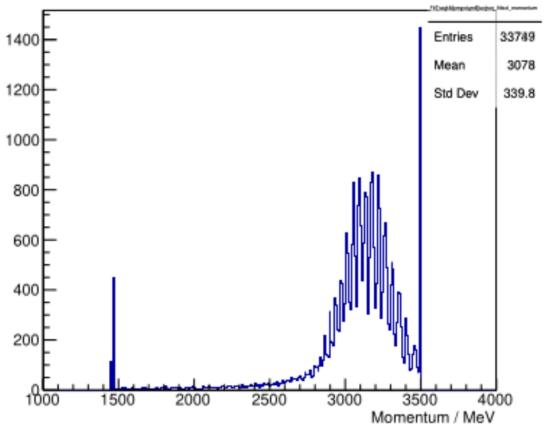


Figure 6.2: Reconstructed momentum with ExPlORA's fitting routine in front of the target using electrons generated according to Section 4.4.6.

direction is essentially unknown. Furthermore the magnetic field is inhomogeneous before the electron beam can hit the target (see Fig. 4.2). This means, that for guessing the initial momentum direction, there needs to be at least one detector layer before the area influenced by the magnetic field. This increases the amount of multiple scattering. Additionally, this layer can not be used for measuring the actual particle momentum, because it is not located within the magnetic field. This leads to an increase in total material budget.

In order to make this point clearer, Fig. 6.2 is included. It shows the reconstructed momentum in front of the target obtained by employing ExPlORA's old fitting routine. Even though the track parameters are well known at this point, the fit is unstable. It can be seen, that it is peaked around 3 200 MeV, but the distribution has a large deviation. Additionally, the algorithm predicts one of the two boundaries for p_l and p_h, which have been set to 1 500 MeV and 3 500 MeV respectively, frequently. This leads to the conclusion, that the previous algorithm is not suitable for the setup in this thesis and would work even worse for tracks behind the target, as the initial momentum guess cannot be made as accurately.

6.2 Kalman Filter

The new fitting algorithm, which was chosen for this thesis, is a Kalman filter. This section summarises its working principle. Furthermore special features in the application of the Kalman filter to the setup in this thesis are highlighted.

A Kalman filter can be used whenever there exists a prediction and a measurement for a physical system. The usefulness can be seen by investigating the following example: Let there be some physical quantity x, whose initial value is measured with some uncertainty. Now this system evolves in time according to some known physical law. At some point,

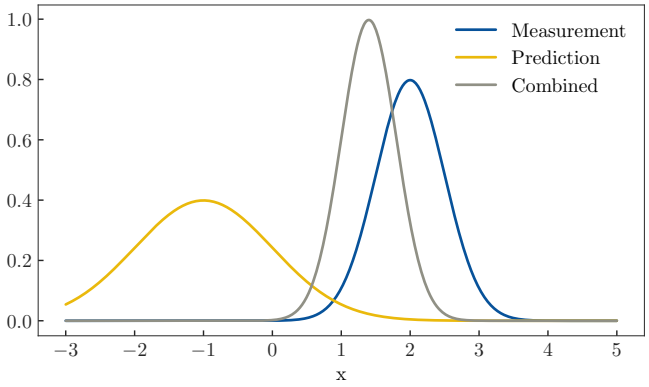


Figure 6.3: The basic principle of a Kalman filter with just one variable is shown. The labels 'Measurement' and 'Prediction' show the PDF for the new measurement and the propagated prediction from the initial measurement respectively. The label 'Combined' shows the PDF of the estimate by combining the two pieces of information according to a Kalman filter.

another measurement with uncertainty is performed and can be used together with the prediction from the initial measurement to form an even better estimate of the quantity x. This process is shown in Fig. 6.3(The measurements and their uncertainties are represented by Gaussian PDFs).

It shows the PDFs for both, the prediction from the initial measurement and the new measurement. It can also be seen, that a better estimate can be formed by combining these two pieces of information. The usefulness of a Kalman filter comes from the fact ,that it is possible to use it in multidimensional linear systems, where variables depend on each other. It can also be shown that this is the statistically optimal way to combine the information[Kre12]. This makes it very versatile and suitable for a wide range of applications. A notable mention would be its application in the Apollo program[MS85].

In order to see, why these properties might be useful for tracking particles, it is helpful to look at Fig. 6.4. Similarly to Fig. 6.1 it shows a schematic overview of a few layers of pixel detectors in front of the target. However, the estimate of the particle's position is indicated by the blue shaded region. The algorithm starts off with the initially measured position and some momentum guess, which is not accurate in the beginning. The particles trajectory is propagated to the next layer and the uncertainty is indicated by the size of the shaded region. Here another measurement is made and the track parameters are updated with the Kalman filter. It can be seen, that the estimate becomes better with every detector layer. It also shows the advantage over the old algorithm, because the position measurement can affect the momentum parameters of the track. Hence, this algorithm is more stable towards the initial momentum guess.

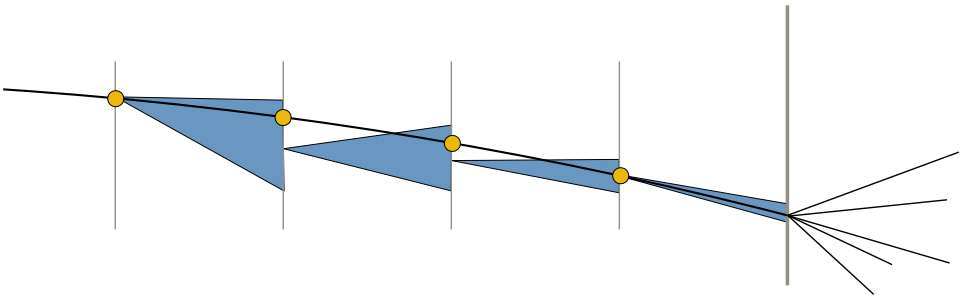


Figure 6.4: This schematic shows the general working principle of a Kalman Filter applied to track fitting. Similarly to Fig. 6.1, the yellow dots represent clusters, pixel layers are vertical lines and the track is indicated in black. The blue shaded region indicates the prediction from each step to the next. Inspired by [KW16]

6.2.1 Mathematical Formulation of the Kalman Filter

The basic idea behind the Kalman filter has been explained above. It is also important to provide a mathematical formulation for the implementation into ExPlORA. This overview is based upon the descriptions in [Kre12; LFG14]. Further information can be found there.

Assume a state at time k, denoted by $\mathbf{x}_k$ with corresponding covariance matrix C_k. Also assume the time evolution of the state is known due to some physical law, such that the evolution can be written as[LFG14]:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, k-1) + \mathbf{w}_{k-1} \tag{6.1}$$

Here $f(\mathbf{x}_{k-1}, k-1)$ refers to the state propagator, which is smooth and deterministic. The term $\mathbf{w}_{k-1}$ is called process noise and used to parametrise random disturbances. It is assumed to be unbiased and has a covariance matrix given by $Q_k \equiv \mathbf{w}_k \mathbf{w}_k^T$[Kre12; LFG14].

Next, assume measurements of some observables are performed at specific points. The observables $\mathbf{m}_k$ are arbitrary and do not need to have the same dimension as the state. Furthermore, since the state completely describes the system, there needs to be a relation between the observables and the state vector given in Eq. (6.2)[Kre12].

$$\mathbf{m}_k = h(\mathbf{x}_k, k) + \mathbf{v_k} \tag{6.2}$$

Here $h(\mathbf{x}_k, k)$ is a projection operator, which gives the measurement vector if no uncertainties were present. Consequently, $\mathbf{v_k}$ describes the measurement uncertainty, whose covariance matrix is given by $V_k \equiv \mathbf{v}_k \mathbf{v}_k^T$[Kre12; LFG14].

Eqs. (6.1) and (6.2) describe the basic requirements, which have to be fulfilled for the Kalman filter. However, the filter only works optimally for linear systems. If $f_k(\mathbf{x}_k)$ and $h_k(\mathbf{x}_k)$ are not linear, it is possible to use the so called extended Kalman filter. This is no longer optimal. It does provide good results, if the system is approximately linear over short ranges[Kre12]. The required step is to linearise the operators by using the

corresponding Jacobi matrices[Kre12; LFG14]:

$$F_k = \frac{\partial f}{\partial \mathbf{x}_k} \tag{6.3}$$

$$H_k = \frac{\partial h}{\partial \mathbf{x}_k} \tag{6.4}$$

From now on the linearised formulation will be used for this section. This also means that Eqs. (6.1) and (6.2) will be rewritten into Eqs. (6.5) and (6.6) respectively.

$$\mathbf{x}_k = F_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{6.5}$$

$$\mathbf{m}_k = H_k\mathbf{x}_k + \mathbf{v_k} \tag{6.6}$$

It is now important to differentiate between the propagation and filtering step. In the propagation step, the state is simply propagated according to Eq. (6.5). This step also requires to update the covariance matrix of the state[LFG14]:

$$C_k = F_{k-1}C_{k-1}F_{k-1}^T + Q_{k-1} \tag{6.7}$$

The filtering step is done, when a measurement exists at the current position. It is hence used to update the state vector with the new information provided by the measurement. The so called Kalman gain K_k is defined first in Eq. (6.8), as it will be used in subsequent steps[Kre12]

$$K_k = C_k H_k^T (H_k C_k H_k^T + V_k)^{-1} \tag{6.8}$$

With the Kalman gain, the filtered state $\mathbf{x}_k'$ can be calculated, which is described in Eq. (6.9). The filtered state is then used as base for subsequent propagation and filtering steps[LFG14].

$$\mathbf{x}_k' = \mathbf{x}_k + K_k(\mathbf{m}_k - H_k\mathbf{x}_k) \tag{6.9}$$

The filtering step also alters the covariance matrix. Similarly to $\mathbf{x}'$ the covariance matrix after filtering C_k' will also be used for subsequent propagation and filtering steps[Kre12].

$$C_k' = (I - K_k H_k)C_k \tag{6.10}$$

The I in Eq. (6.10) is the identity matrix with appropriate dimensions.

6.3 Implementing the Kalman Filter into ExPlORA

6.3.1 Mathematical Prerequisites

The first challenge, when implementing a Kalman filter is to find a suitable parametrisation for the state and hence the propagation matrix F_k. Originally, it was tried to reuse the propagation used in ExPlORA. However, in every propagation step a local rotated coordinate system is created in such a way, that the particle is always incoming on the

x axis. This requires a rotation in every propagation step and therefore complicates the Jacobian. It can however be done as shown in [LFG14]. Instead the parametrisation and propagation from [Kre12] is chosen and summarised in the following.

The state vector has the five entries given in Eq. (6.11)[Kre12].

$$\mathbf{x} = (x, y, \underbrace{p_x/p_z}_{t_x}, \underbrace{p_y/p_z}_{t_y}, q/p)^T \tag{6.11}$$

Here x and y refer to the coordinate in the respective axis, $p_{x/y/z}$ is the momentum projected on the $x/y/z$ axis, q is the particle charge and p the total momentum. These five parameters are sufficient to describe all track properties. The z axis is used as stepping dimension. This means that the k subscript in e.g. Eq. (6.10) refers to some z coordinate.

The basis for propagation through the magnetic field is the Lorentz force, which describes the force acting on a moving charged particle in a magnetic field $\mathbf{B}$. This allows to express the change in momentum with respect to the path length[Kre12]:

$$d\mathbf{p} = \kappa \cdot q \cdot \frac{\mathbf{p}}{p} \times \mathbf{B} ds, \quad \kappa = 2.99792458\,\mathrm{MeV\,cm^{-1}\,T^{-1}}\,, \text{(with } c = 1) \tag{6.12}$$

Since the z coordinate is the stepping parameter, this fixes the propagation function $f(\mathbf{x}_z)$[Kre12]:

$$f(\mathbf{x}_z) = (x', y', t_x', t_y', (q/p)' = 0)^T \tag{6.13}$$

The primed variables in Eq. (6.13) refer to derivatives with respect to z. Consequently, $(q/p)' = 0$, because the Lorentz force does not change the total momentum of the particle.

In order to solve Eq. (6.12), a Runge-Kutta method of fourth order is used. This is a standard way to solve differential equations and usually gives good results and performance. It is convenient to first define the starting point z_0 and the step size s for the subsequent propagation step. Four intermediary z coordinates are also defined[Kre12]:

$$z_i = \left\{ z_0, z_0 + \frac{s}{2}, z_0 + \frac{s}{2}, z_0 + s \right\}$$

Where $i \in [1, 4]$. Four intermediary states are also defined[Kre12]:

$$\begin{aligned}
\Delta\mathbf{x}_1 &= s \cdot f(\mathbf{x}_{z_1}, z_1) \\
\Delta\mathbf{x}_2 &= s \cdot f(\mathbf{x}_{z_1} + \frac{\Delta\mathbf{x}_1}{2}, z_2) \\
\Delta\mathbf{x}_3 &= s \cdot f(\mathbf{x}_{z_1} + \frac{\Delta\mathbf{x}_2}{2}, z_3) \\
\Delta\mathbf{x}_4 &= s \cdot f(\mathbf{x}_{z_1} + \Delta\mathbf{x}_3, z_4) \\
\mathbf{x}_{z_i} &= \mathbf{x}_{z_0} + \frac{\Delta\mathbf{x}_i}{s}(z_i - z_0)
\end{aligned}$$

The propagator can then be extrapolated by the following matrix[Kre12]:

$$F_z = I + \frac{\tilde{F}_1}{6} + \frac{\tilde{F}_2}{3} + \frac{\tilde{F}_3}{3} + \frac{\tilde{F}_4}{6}$$

The intermediate $\tilde{F}_i$ can be calculated with the following recursion formula[Kre12]:

$$\tilde{F}_1 = s \cdot \frac{df(\mathbf{x}_{z_i}, z_i)}{d\mathbf{x}_{z_i}} \left(I + \tilde{F}_{i-1} \frac{z_i - z_0}{s} \right) \tag{6.14}$$

This leads to the Jacobian of the propagator in Eq. (6.15)[Kre12]. The full formulation of the derivatives is lengthy and does not support the explanation at this point. Hence, the non abbreviated and fully calculated form of the derivatives in Eqs. (6.13) and (6.15) can be found in Appendix A.3.

$$F_z = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \partial t'_x/\partial t_x & \partial t'_x/\partial t_y & \partial t'_x/\partial(\frac{q}{p}) \\ 0 & 0 & \partial t'_y/\partial t_x & \partial t'_y/\partial t_y & \partial t'_y/\partial(\frac{q}{p}) \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.15}$$

The projection or measurement matrix is also needed. Since each detector layer measures the coordinate in the x-y plane, this is merely a projection matrix[Kre12]:

$$H_z = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

6.3.2 Programming the Kalman Filter

Once the parametrisation and propagation matrices are known, the Kalman filter can actually be implemented in ExPlORA. The Kalman filter was implemented into the plugin, which already performed the old fitting routine. This was done to save programming work, as several convenience and compatibility features are present in these.

The concept of tracking regions was kept. However, in order to accommodate the needs of the Kalman filter, their attributes were extended. Additionally to position, material minimum step size and access to the magnetic field, they also contain pointers to matrices for propagation, measurement and their corresponding covariances. This has several implications. The propagation methods are called from within the tracking regions. Due to the scope[1] of the program in this case, it is useful to have the matrices accessible. Additionally, things like the measurement matrix and its corresponding covariance matrix might change over tracking regions. Hence, it makes sense to have these matrices dedicated to each tracking region. This allows for the change of matrices in between regions. Also having the state covariance matrix given to the regions by pointer, allows them to share

[1] Note: Here scope refers to the accessibility of certain variables within functions and objects.

the matrix during fitting. This is useful, because fitting the track is done sequentially. Sharing the covariance matrix eliminates the need to pass it between tracking regions, when a region transition has to be made.

In order to conserve compatibility with other plugins, the extension described above is not added to the already existing tracking regions. The extension is applied to custom regions, which inherit from the existing ones. Modifications to the code base can cause the code to not be compiled, because other plugins might depend on the one which is changed.

The main challenge in implementing the Kalman filter into the existing ExPlORA plugin was the fact, that the existing plugin tracked through the entire geometry before changing the momentum. Also the track direction and position was isolated in the tracking routine and not designed to be altered from a different scope. This complicated the implementation as the flow of the plugin had to be changed without breaking the compatibility. This was solved by changing them to pointers and have them declared in the setup of the plugin, before the tracking begins. This way it is possible to access their values from different scopes. In order to preserve the compatibility, the polymorphism ability of the C++ language was used, essentially leading to several functions having a pointer and non pointer version.

The individual clusters are still added to their respective tracking region and the plugin will loop through every tracking region for each track in the input container. The state is initialised with the momentum guess explained above and its position in the first detector layer. The covariance matrix is initialised by the initial measurement error from the cluster and the other uncertainties can be configured by the user. For simplicity only electrons are propagated. Hence, their charge is known and all initial parameters are given at this point.

The track is then propagated according to the method given in Section 6.3.1. If the propagation takes place in a tracking region with associated clusters and the clusters have not been used, it is checked, whether the next step would propagate the track into the same x-y plane as given by the cluster's position. In case this check is true, the next propagation step is performed, such that it results in this plane. Once the correct plane is reached, the filtering is performed. The Kalman gain and the updated state and its covariance matrix are computed according to Eqs. (6.8) to (6.10). This is where having a common state covariance matrix is useful. Changes to it are immediately effective in other tracking regions. Also the measurement covariance matrix is adjusted to the uncertainties of the cluster position before the filter step is performed. After the filtering, the propagation is continued as described above, until all tracking regions have been considered. The fit results are written as additional parameters to the track and saved in an output container specified by the user. In order to clarify the way this plugin and fitting routine works, key steps have been summarised in Fig. 6.5.

There is a distinction to be made between tracking in front of and behind the target. Tracking before the target works as described above. A dedicated plugin is responsible for tracking behind the target. It inherits all its functionality from the base tracking plugin. However the setup of regions is of course adapted to the geometry behind the target. It also uses the output of the initial fit as starting point for its fit. Splitting this process

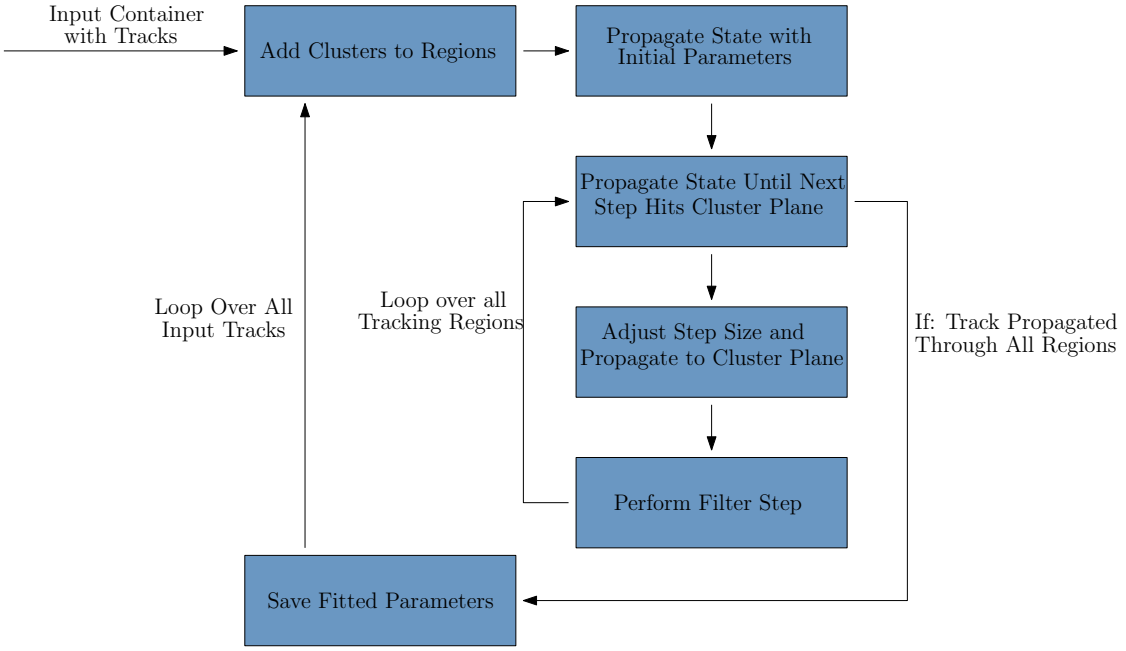


Figure 6.5: This schematic shows the Kalman filter fitting routine with basic and key steps.

allows for an easier comparison of the fit performance.

It is also useful to try if the algorithm described above works in principle. For this reason the same algorithm was implemented into a small Python script, before it was put to use in ExPlORA. This is a very simplified version, because apart from the magnetic field and its interaction with a charged particle, no physics processes were implemented. However, it allows to test more easily and quickly than a full implementation into ExPlORA would allow. Fig. 6.6 shows one sample track in an inhomogeneous magnetic field. The measurements are located in layers along the z axis (pointing up). Even though the initial momentum guess is not optimal, the algorithm is able to fit the momentum with a difference of 1.79 %. This shows the chosen Kalman filter algorithm can in principle work with the setup in this thesis.

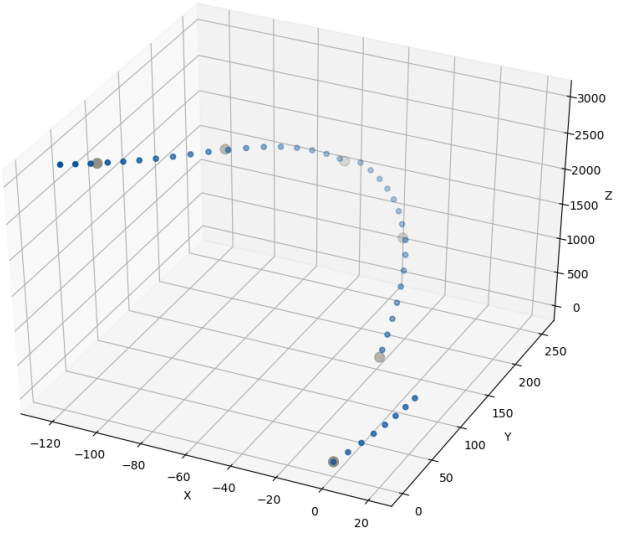


Figure 6.6: Shown is a sample track, with the Kalman filter fitting routine from Section 6.3 applied. Points in blue are individual propagation steps. Points in grey are actual measurement positions. Units are arbitrary, because this plot serves as proof of concept

Chapter 7

Results

7.1 Impact Point on Target

It is useful to know, where the beam will hit the target. This allows for considerations of pixel layer and target size. It can also be used to determine the starting point for including signal events (see Section 4.4.8). Fig. 7.1 shows a histogram of impact points at the target position.

The resulting square shape of the histogram entries is noticeable. It can be attributed to the way this histogram is generated. In order to access Monte Carlo data in ExPlORA,

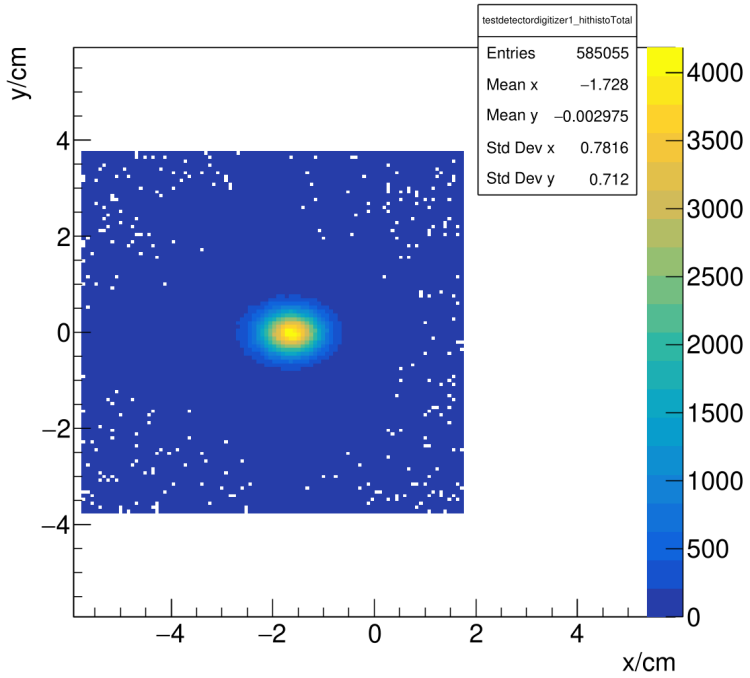


Figure 7.1: Histogram of recorded impact point in the x-y plane at the target position (see Table 7.1 for coordinates).

it is necessary to look at simulated energy deposits. However, energy deposits are only created in sensitive materials such as silicon. For this reason a silicon pixel detector was put in the place of the target and its energy deposits were used to create this histogram. The resulting square shape corresponds to the shape of the used pixel detector.

7.2 Number of Clusters in Different Detector Layers

In the case of an actual realised detector, it is important to know the data rates, the experiment needs to cope with. Especially the pixel detectors are limited in the number of hits in a given time frame. The setup developed during the course of this thesis is able to simulate this. Table 7.1 shows the positioning of different detector components for this simulation.

Component	Spatial Coordinates		
	x	y	z
silicon_0_A_h	−2.0	0	249.0
silicon_1_A_h	−2.0	0	235.0
silicon_2_A_h	−2.0	0	225.0
silicon_3_A_h	−2.0	0	215.0
silicon_4_A_h	−1.0	0	205.0
silicon_5_A_h	0.0	0	195.0
Target	0.0	0	250.0
silicon_0_A_v	−2.0	0	253.0
silicon_1_A_v	−4.0	0	255.0
silicon_2_A_v	−4.0	0	257.0
silicon_3_A_v	−4.0	0	260.0
silicon_4_A_v	−5.0	0	270.0
silicon_5_A_v	−5.0	0	277.0
Electromagnetic Calorimeter	0.0	0	280.0
OD-Magnet	0.0	0	276.5

Table 7.1: Shown are positions for different detector components for simulating the number of clusters in different components. The shown coordinate refers to the centre of the silicon sheet for pixel detectors, the centre of the tungsten sheet for the target, the centre of the OD-Magnet and the centre of the silicon layer of the first central hexagonal cell for the calorimeter.

It can be seen, that the pixel detector positions shift along the x-axis to accommodate the electrons curvature in the magnetic field. This array of pixel detectors makes it possible to observe the number of hits at different positions along the setup. No signal events have been used for this run, because the majority will be standard model background anyway. This also allows to observe the number of clusters per event along the entire setup, because signal events can only be generated at the target (see Section 4.4.8). In total 62 000 events

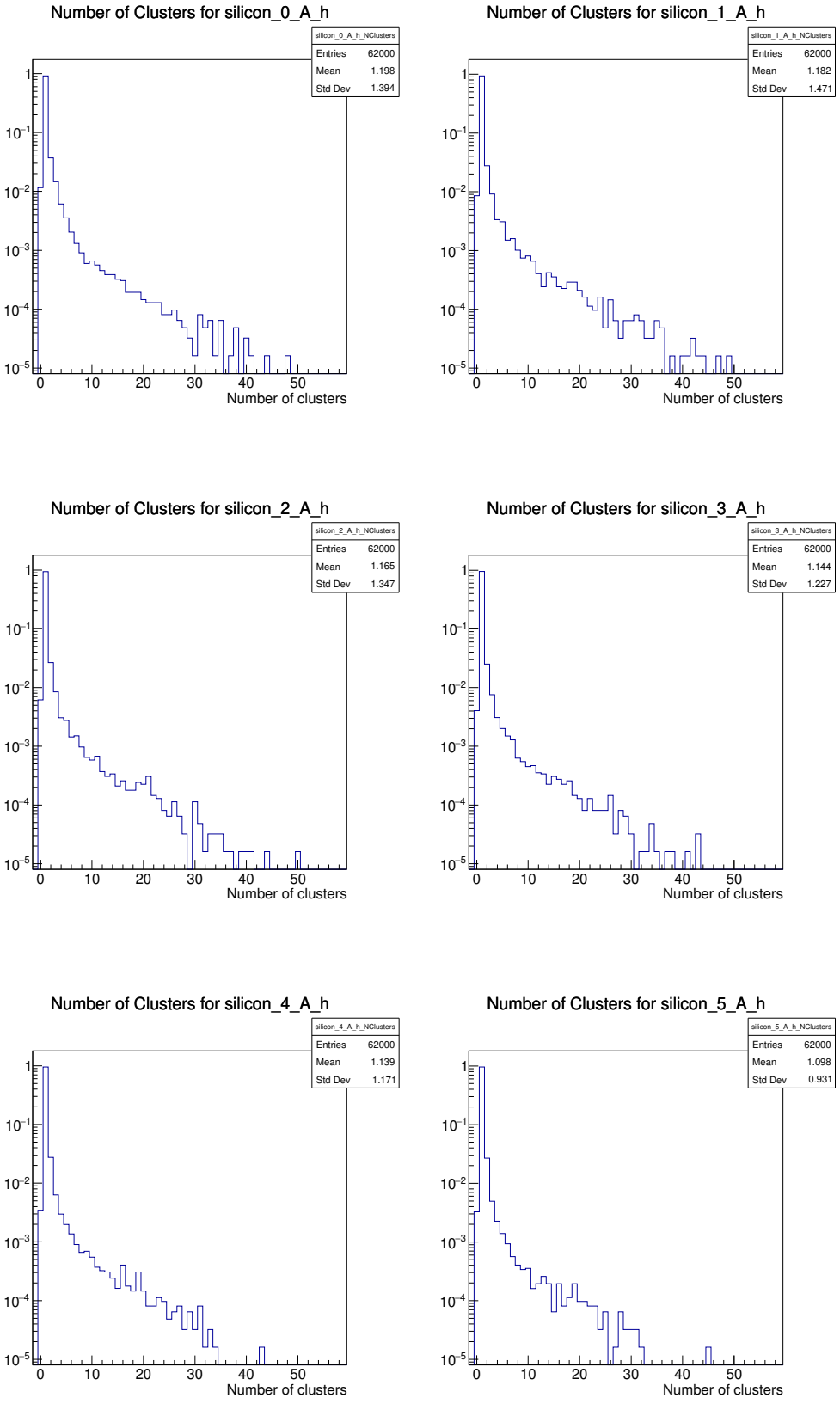


Figure 7.2: Number of clusters for different detector layers in front of the target.

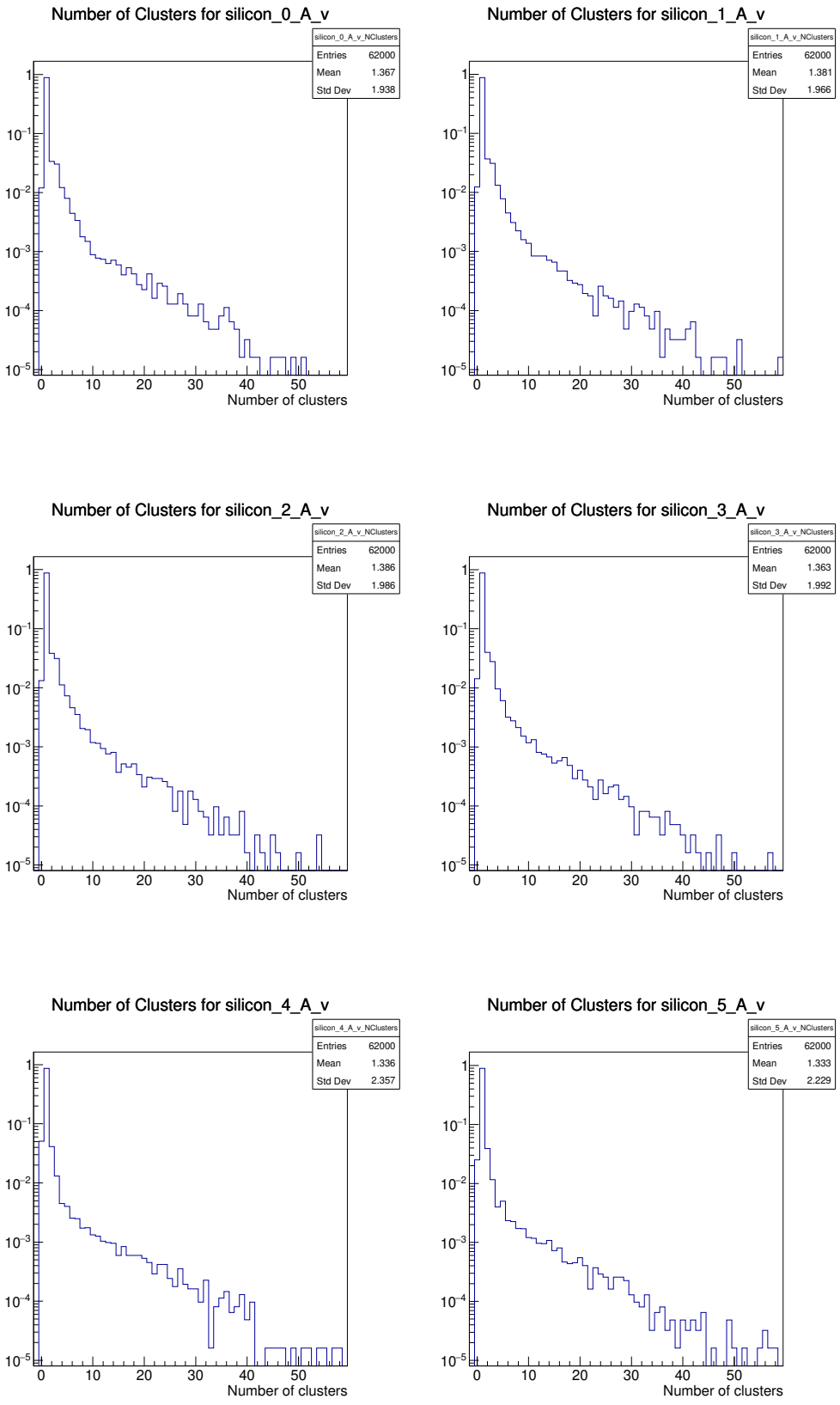


Figure 7.3: Number of clusters for different detector layers behind the target.

were generated and Figs. 7.2 and 7.3 show the number of clusters in each pixel detector layer per event.

As would be expected, the detector layers behind the target have a higher average cluster count and a greater deviation from the mean. This can be explained by the greater number of tracks behind the target and the back-scattering at the calorimeter.

7.3 Kalman Filter Performance

It is useful to study the performance of the Kalman filter fitting routine. This can give an indication, whether this approach to fitting is feasible and should be continued in later stages of the design of a potential experiment. For this reason the algorithm's ability to reconstruct an electrons momentum in front of and behind the target is examined. This method does not cover all aspects of the fit as other parameters such as missing mass(see Section 7.4), can potentially become vital parameters in the analysis.

The detector setup is left unchanged and the values from Table 7.1 are still valid(calorimeter removed). The starting values and initial uncertainties for the Kalman filter are given in Table 7.2. It should be noted, that these have to be optimised in order to get the best performance. These values represent the first iteration in this process. They do however show the general ability of the Kalman filter to be used as a fitting routine. It can can be seen, that the uncertainties behind the target are comparatively large. This is due to the low amount of information about the track at this point and ensures, that a wide range of momenta can be fitted.

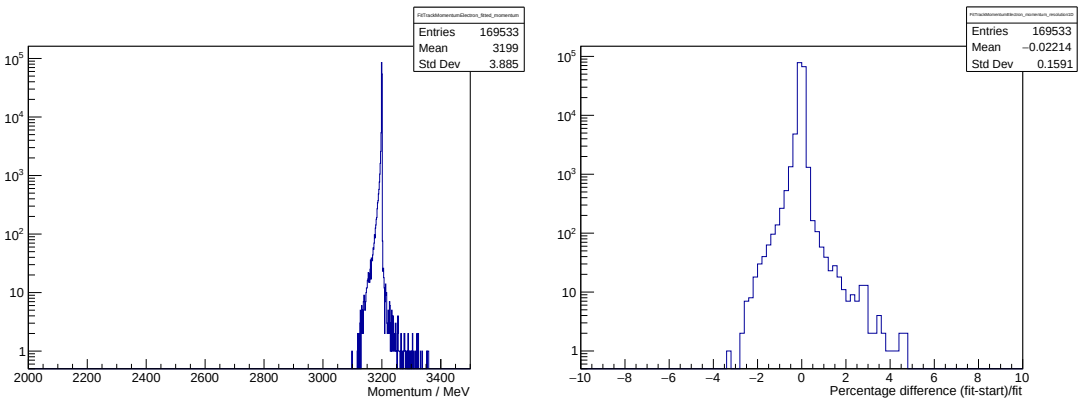
Fig. 7.4 shows the performance of the Kalman filter in front of the target measuring the momentum. It can be seen, that the Kalman filter works in principle and the momentum is reconstructed with deviations given in the figure. Comparison with Fig. 6.2 shows, that this approach works better than the old fitting routine. There is an asymmetry in the relative deviation from the true value. This skews the position of the mean and introduces a bias to reconstruct lower than generated momenta.

The corresponding fit performance behind the target is shown in Fig. 7.5. At this point it has to be noted, that the true momentum distribution looks different to the one in Fig. 4.7(a). This can be attributed to mainly two facts. First, there are several detector layers between the target and the point of measurement, contrary to no detector layers for Fig. 4.7(a). Second, there are some energy deposits counted to belong to the original track, which do not come from this track. This can be explained by ExPlORA having the tendency to have simulated data analysed the same way as measured data. Because of this, there is no direct way to determine, which fitted track energy deposits belong to the actual Monte Carlo tracks. Tracks are assigned based on their distance, which introduces some energy deposits being counted to the wrong track. This effect however is at most of the order of 5 000 events, because this run was simulated with 150 000 events. While this presents a problem, it does not change the general meaning of the following discussion.

It can be seen, that the Kalman filter performs worse than in front of the target. This makes sense as the amount of information after scattering has occurred in the target

In front of the target	
Parameter	Initial Value
Start position	Given by first measurement in pixel detector
Start position uncertainty	Uncertainty on cluster determining start position
Initial direction	Difference of first two measured positions
t_x uncertainty	0.05
t_y uncertainty	0.05
Initial momentum	3 200 MeV
Initial momentum uncertainty	3 MeV
Behind the target	
Parameter	Initial Value
Start position	Predicted by Kalman filter in front of the target or signal generator emission position respectively
Start position uncertainty	5 cm in x and y respectively
Initial direction	parallel to z axis
t_x uncertainty	0.8
t_y uncertainty	0.8
Initial momentum	800 MeV
Initial momentum uncertainty	3 000 MeV

Table 7.2: Initial values for the Kalman filter including the necessary uncertainties to set up the covariance matrix. Values have to be optimised in order to get the best results with the Kalman filter.



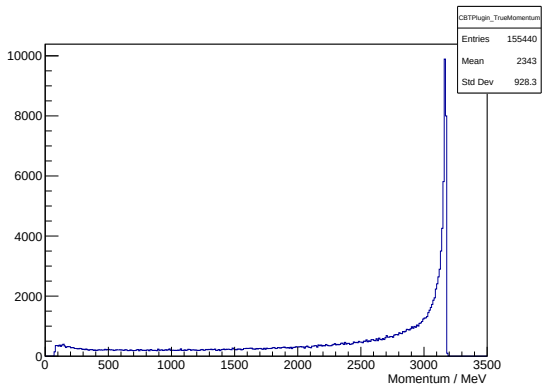
(a) Reconstructed Momentum

(b) Relative Difference to true momentum

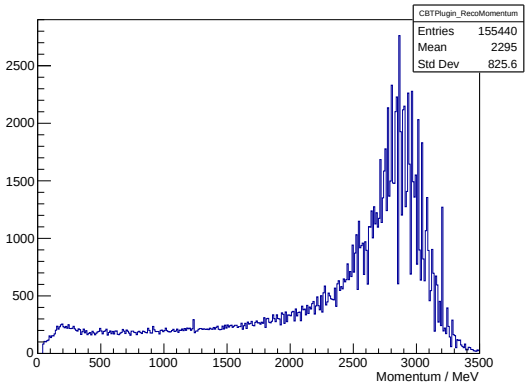
Figure 7.4: Fit performance of the Kalman filter for electrons generated according to Section 4.4.6.

is significantly lower, which impacts the fit performance. The distribution showing the reconstructed momentum has the general shape of the actual momentum distribution, but there is a significant difference.

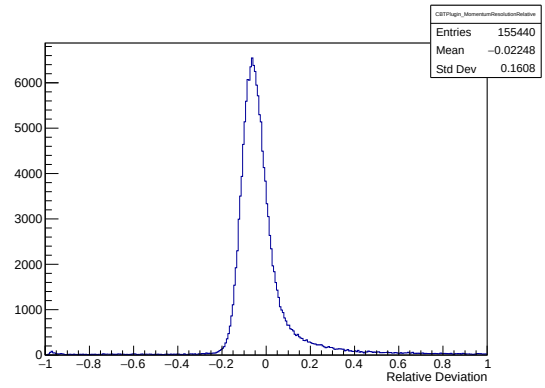
The asymmetry in the distribution of relative deviations is more pronounced in this case. The mean implies, that there is a tendency to reconstruct lower momenta, than what would be correct. However there is a tail towards reconstructing higher than correct momenta. Fig. 7.5(d) shows that this long tail comes from the inability to reconstruct low momentum tracks correctly.



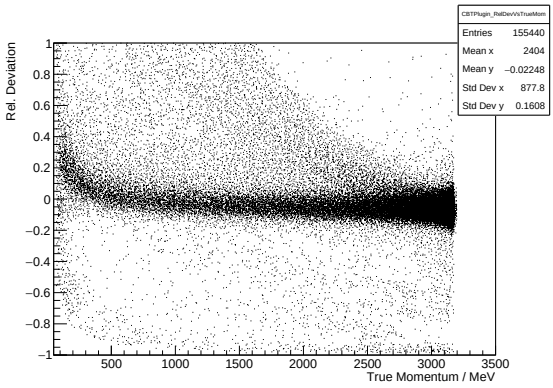
(a) True momentum



(b) Reconstructed momentum



(c) Relative deviation of true and reconstructed momentum

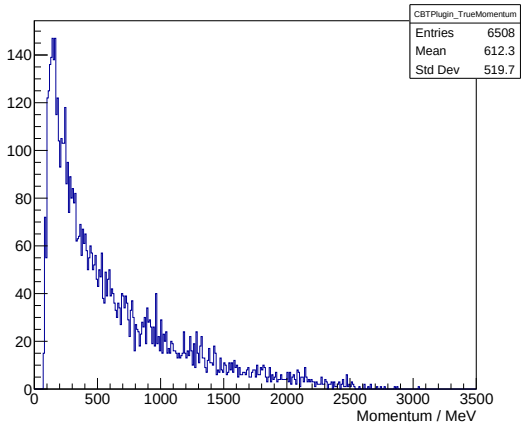


(d) Relative deviation of true and reconstructed momenta for different true momenta

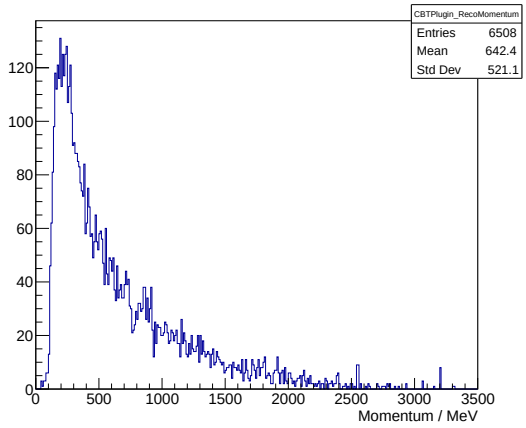
Figure 7.5: Fit performance of the Kalman filter behind the target with electrons produced according to Section 4.4.6

Since signal events generated by dark bremsstrahlung have been incorporated into the simulation in Section 4.4.8, it is possible to look at the ability of the Kalman filter to reconstruct the momentum of signal electrons. The corresponding plots are given in

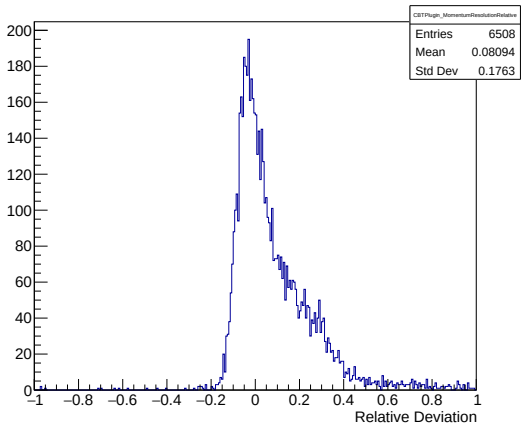
Fig. 7.6. The overall results are similar to the ones presented in Fig. 7.5. There is a long tail to reconstruct momenta which are higher than the actual momentum of the particle. This tail also stems from not being able to correctly reconstruct low momentum events.



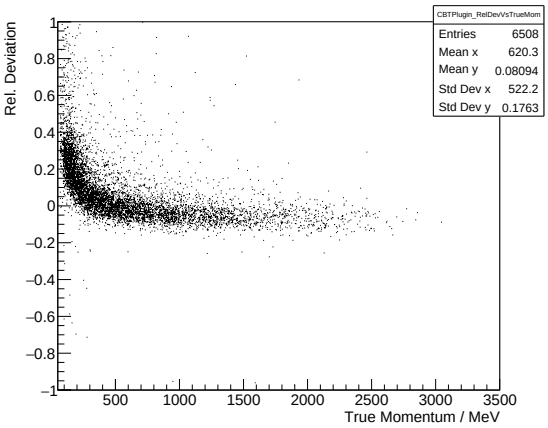
(a) True momentum



(b) Reconstructed momentum



(c) Relative deviation of true and reconstructed momentum



(d) Relative deviation of true and reconstructed momentum for different true momenta

Figure 7.6: Fit performance of the Kalman filter behind the target with 100 MeV dark photon signal events

Overall it can be said, that the implemented Kalman filter works in principle and seems to be promising for this setup. However, there are effects, which are not included yet. These have an impact on the overall performance of the Kalman filter. This becomes especially noticeable behind the target, where less information is known about the track. The lower momentum of particles behind the target also means, that interactions within the

detector layers become more significant. The Kalman filter can be improved by correctly modelling multiple scattering and energy losses due to interactions in the detector layers.

7.3.1 Predicting the Impact Point on the Calorimeter

Another test showing the ability of the Kalman filter to correctly predict a particle's position at different points in the detector setup, is predicting the potential impact point on the calorimeter. The setup from Table 7.1 is reused. However due to the proximity of the calorimeter to the last detector layer, the calorimeter is moved 20 cm back to a z position of 297.0.

Fig. 7.7 shows the difference of predicted impact point on this hypothetical calorimeter position and the actual impact point. It can be seen, that the distribution peaks around 0 and most hits can be localised well within a cm. The spread in the x direction is larger than in the y direction, which is to be expected. Considering essentially no information about the track is known behind the target, this shows the Kalman filter is working in principle.

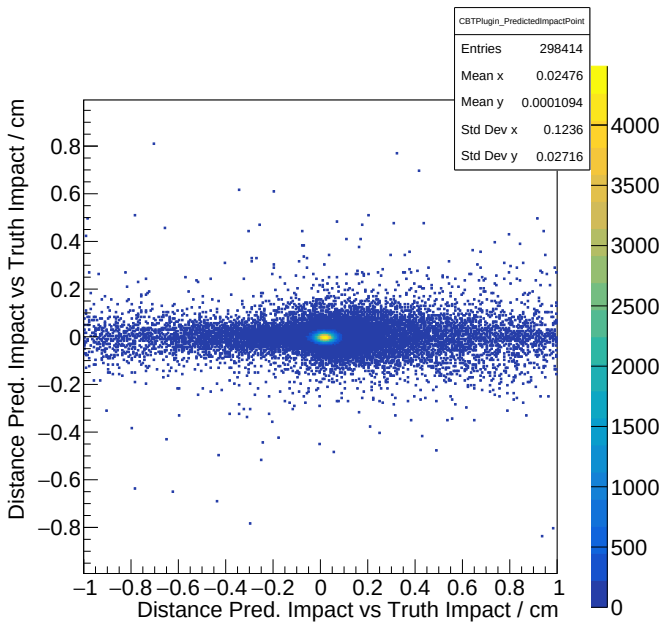
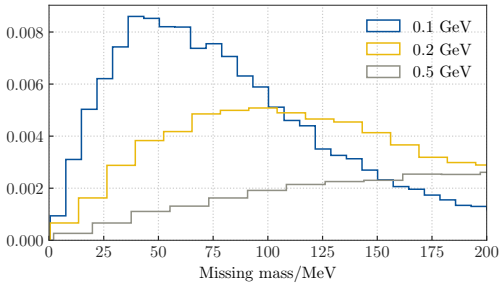


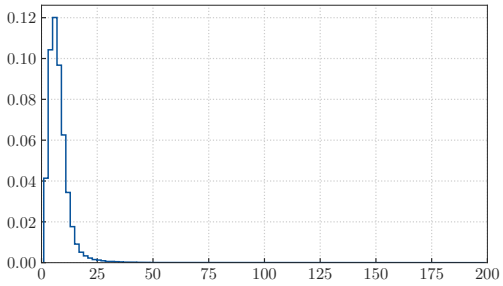
Figure 7.7: Difference of predicted and actual impact point on the electromagnetic calorimeter in the x-y plane. The calorimeter is placed 20 cm behind the last pixel detector layer.

7.4 Missing Mass Distributions

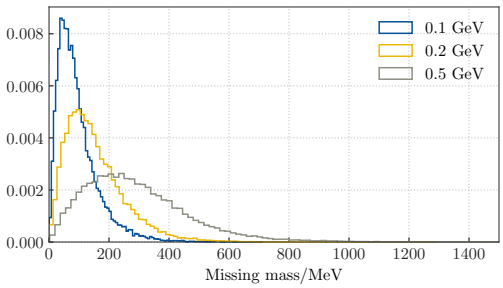
It is interesting to look for variables, which could discriminate between signal and Standard Model background events. One of these variables is missing mass. It is calculated by subtracting the Lorentz vectors of the incoming and recoiling electron from eachother and taking the square root of the resulting vectors scalar product in Minkowski space. Fig. 7.8 shows the corresponding distributions for Standard Model background events and signal events simulated with different dark photon masses.



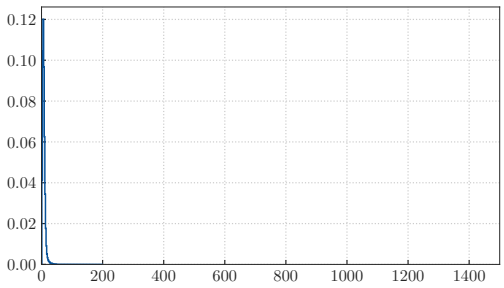
(a) Signal events for different dark photon masses



(b) Standard Model background



(c) Signal events for different dark photon masses



(d) Standard Model background

Figure 7.8: Missing mass distributions for Standard Model background and signal events for different dark photon masses. Shown are two different momentum ranges to view the distributions more easily. Histograms are normalised with their area.

It can be seen, that there is a strong difference in these distributions regardless of dark photon mass. This difference makes visualisation difficult and the version in Fig. 7.8 was chosen as a compromise between comparability and visibility. It seems promising to use this variable as additional information to the other kinematic quantities in the analysis of an actual experiment. However, it's actual feasibility has to be tested together with the ability to reconstruct this variable with the fitting algorithm.

Chapter 8

Conclusion and Summary

This thesis features the gradual construction of detector components in simulation, such as target, pixel detectors and the electromagnetic calorimeter for a light dark matter experiment at ELSA. It was possible to integrate the first iteration of a pixel detector with configurable pixel size, thickness and position. A target of adjustable material and dimensions was also integrated. An electromagnetic calorimeter with hexagonal cell structure and configurable geometry was built and equipped with a working digitizer.

The simulation of non dark bremsstrahlung Standard Model background events was achieved by putting electrons with configurable energy position and beam angle into the simulation. The general feasibility of the used physics list and geometry was verified by comparing certain kinematic variables to distributions given in [Åke+18].

Dark bremsstrahlung signal electrons were included courtesy of Ruth Pöttgen, who provided the MadGraph/MadEvent4 code for signal generation. The output Les-Houches event file is read by a custom parser, which feeds the kinematic information of the signal electron into the electron generator. Furthermore the feasibility of this approach was verified by comparing kinematic quantities with [Åke+18].

The constructed pixel detectors did not have any logic, which would allow to analyse the simulated data. For this reason a digitizer was employed. It mainly built upon the already existing digitizer in ExPlORA, but uses a trick involving fictitious indices. This allows for the simulation of highly granular pixel detectors, which would not be possible otherwise due to performance issues.

It was possible to assign clusters to tracks by using available Monte Carlo data. This was done to save time and due to the limited scope of this thesis. Hence, this module of the simulation will have to be reworked in later iterations and improvements to the simulation. It can however be seen as working proof of concept.

ExPlORA already featured an existing fitting routine, which was adapted to work with the setup for a light dark matter experiment. Due to the strong dependency on the initial momentum measurement and the sparse amount of information about the track behind the target, this routine had to be discarded and would not be feasible for this experiment.

A new fitting routine based on a simple Kalman filter was implemented using ideas such as tracking regions from the old fitting algorithm. Its performance was tested by investigating how well the momentum in front of, behind the target, and with signal

electrons could be reconstructed. While this approach seems to be promising, there are definitely effects which are not accounted for, such as multiple scattering and energy loss. This will have to be addressed in later iterations of a simulation software.

Overall it has to be said, that simplifications have been made for various parts of this thesis (e.g. the pixel detectors without support structures or track assignment using Monte Carlo data). However this has to be expected, as this is the first iteration of the simulation and bounds are set by the scope of a master's thesis. It was possible to set up the simulation and include necessary logic into the detector components to an extent, which allows track fitting with a simple Kalman filter. This setup already allows for basic simulation studies, such as the Kalman filter performance or the amount of hits per event in different detector layers. This shows, that even though many components currently are just a proof of concept, the basics of most of the detector setup have been implemented and are functioning. The work done in this thesis can be used as a foundation to expand on the existing simulation and build upon it. Eventually leading to a more sophisticated and accurate depiction of a real detector setup.

Bibliography

[Ams] Amsterdam Cosmology Group, *Lecture Notes on Cosmology*, URL: http://cosmology.amsterdam/education/cosmology/ (visited on 20/02/2020) (cit. on pp. 1, 6, 8).

[Åke+18] T. Åkesson et al., *Light Dark Matter eXperiment (LDMX)*, 2018, arXiv: 1808.05219 [hep-ex] (cit. on pp. 1, 6–13, 17–20, 27, 31, 33, 35, 36, 71, 78).

[Fen+17] J. L. Feng et al., *Particle physics models for the 17 MeV anomaly in beryllium nuclear decays*, Physical Review D **95** (2017), ISSN: 2470-0029, URL: http://dx.doi.org/10.1103/PhysRevD.95.035017 (cit. on p. 1).

[Ban+17] D. Banerjee et al., *Search for Invisible Decays of Sub-GeV Dark Photons in Missing-Energy Events at the CERN SPS*, Phys. Rev. Lett. **118** (1 2017) 011802, URL: https://link.aps.org/doi/10.1103/PhysRevLett.118.011802 (cit. on p. 1).

[Aba+95] S. Abachi et al., *Observation of the top quark*, Phys. Rev. Lett. **74** (1995), arXiv: hep-ex/9503003 [hep-ex] (cit. on p. 3).

[The12a] The ATLAS Collaboration, *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, Phys. Lett. **B716** (2012), arXiv: 1207.7214 [hep-ex] (cit. on p. 3).

[The12b] The CMS Collaboration, *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, Phys. Lett. **B716** (2012), arXiv: 1207.7235 [hep-ex] (cit. on p. 3).

[Hei17] J. Heinrichs, *Feasibility of Using Boosted Trees to Measure Higgs $\mathcal{CP}$ in $H \to \tau\tau$ Decays at the LHC*, 2017 (cit. on p. 3).

[BGB08] V. A. Bednyakov, N. D. Giokaris and A. V. Bednyakov, *On Higgs mass generation mechanism in the Standard Model*, Phys. Part. Nucl. **39** (2008), arXiv: hep-ph/0703280 [hep-ph] (cit. on p. 3).

[Bur16] C. Burgard, *Texample: Standard model of physics*, Slightly adapted from original, 2016, URL: http://www.texample.net/tikz/examples/model-physics/ (visited on 20/02/2020) (cit. on p. 4).

[Sch17] M. D. Schwartz,
QUANTUM FIELD THEORY and the STANDARD MODEL,
Cambridge University Press, 2017, ISBN: 978-1-107-03473-0
(cit. on pp. 4, 5, 7).

[Sch06] P. Schneider, *Extragalactic Astronomy and Cosmology - An Introduction*,
Springer, 2006, ISBN: 9783540331742 (cit. on pp. 5, 6).

[Mró+19] P. Mróz et al., *Rotation Curve of the Milky Way from Classical Cepheids*,
The Astrophysical Journal **870** (2019) L10, ISSN: 2041-8213,
URL: http://dx.doi.org/10.3847/2041-8213/aaf73f (cit. on p. 6).

[Ale+16] J. Alexander et al., *Dark Sectors 2016 Workshop: Community Report*, 2016,
arXiv: 1608.08632 [hep-ph] (cit. on pp. 6–9).

[CGR10] S. Cassel, D. Ghilencea and G. Ross,
Electroweak and dark matter constraints on a in models with a hidden valley,
Nuclear Physics B **827** (2010) 256, ISSN: 0550-3213,
URL: http://dx.doi.org/10.1016/j.nuclphysb.2009.10.029
(cit. on p. 7).

[Fro19] F. Frommberger, *ELSA Homepage*, 2019,
URL: https://www-elsa.physik.uni-bonn.de/index_en.html (visited on
26/02/2020) (cit. on pp. 13, 14).

[Pul99] M. Pullia, *Slow extraction dynamics and its influence on transfer lines design*,
1999, URL: https://inspirehep.net/record/511446?ln=en
(cit. on pp. 13, 14).

[Wil05] K. Wille, *The Physics of Particle Accelerators - An Introduction*,
Oxford University Press, 2005, ISBN: 9780198505495 (cit. on pp. 13, 15).

[Con+15] D. Contardo et al.,
Technical Proposal for the Phase-II Upgrade of the CMS Detector,
tech. rep. CERN-LHCC-2015-010. LHCC-P-008. CMS-TDR-15-02,
Upgrade Project Leader Deputies: Lucia Silvestris (INFN-Bari), Jeremy Mans
(University of Minnesota) Additional contacts: Lucia.Silvestris@cern.ch,
Jeremy.Mans@cern.ch, 2015, URL: https://cds.cern.ch/record/2020886
(cit. on p. 19).

[Fre17] O. Freyermuth,
Studies of ω Photoproduction off Proton at the BGO-OD Experiment,
PhD thesis: Universität Bonn, 2017,
URL: http://hss.ulb.uni-bonn.de/2017/4839/4839.pdf
(cit. on pp. 19, 22).

[Mig16] E. Migliore, *CMS Pixel Detector design for HL-LHC*,
Journal of Instrumentation **11** (2016) C12061,
URL: https://doi.org/10.1088%2F1748-0221%2F11%2F12%2Fc12061
(cit. on p. 20).

[Gea] Geant4 Collaboration, *The GEANT4 support website*, URL: https://geant4.web.cern.ch/support (visited on 24/03/2020) (cit. on p. 21).

[PW] Particle Data Group, *Atomic and nuclear properties of tungsten*, 2018, URL: http://pdg.lbl.gov/2019/AtomicNuclearProperties/HTML/tungsten_W.html (visited on 03/04/2020) (cit. on p. 24).

[Alw+07] J. Alwall et al., *A standard format for Les Houches Event Files*, Computer Physics Communications **176** (2007) 3002013304, ISSN: 0010-4655, URL: http://dx.doi.org/10.1016/j.cpc.2006.11.010 (cit. on pp. 33, 34).

[Boo+01] E. Boos et al., *Generic User Process Interface for Event Generators*, 2001, arXiv: hep-ph/0109068 [hep-ph] (cit. on pp. 33, 34).

[Hei13] I. Heinze, *Development of a Hough Transformation Track Finder for Time Projection Chambers*, PhD thesis: Universität Hamburg, 2013, URL: https://bib-pubdb1.desy.de/record/165831 (cit. on p. 44).

[Far+17] Farrell, Steven et al., *The HEP.TrkX Project: deep neural networks for HL-LHC online and offline tracking*, EPJ Web Conf. **150** (2017) 00003, URL: https://doi.org/10.1051/epjconf/201715000003 (cit. on p. 44).

[WSH20] K. Wolters, S. Spannagel and D. Hynds, Allpix^2 *User Manual*, 2020, URL: https://project-allpix-squared.web.cern.ch/project-allpix-squared/usermanual/allpix-manual.pdf (visited on 16/04/2020) (cit. on p. 46).

[Kre12] E. Krebs, *Application of a Kalman filter and a Deterministic Annealing filter for track reconstruction in the HADES experiment*, 2012 (cit. on pp. 50–54, 79).

[MS85] L. A. McGee and S. F. Schmidt, *Discovery of the Kalman Filter as a Practical Tool for Aerospace and Industry*, 1985, URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19860003843.pdf (visited on 19/04/2020) (cit. on p. 50).

[KW16] H. Kolanoski and N. Wermes, *Teilchendetektoren - Grundlagen und Anwendungen*, Springer, 2016, ISBN: 978-3-662-45349-0 (cit. on p. 51).

[LFG14] B. Li, K. Fujii and Y. Gao, *Kalman-filter-based track fitting in non-uniform magnetic field with segment-wise helical track model*, Computer Physics Communications **185** (2014) 754, ISSN: 0010-4655, URL: http://dx.doi.org/10.1016/j.cpc.2013.11.003 (cit. on pp. 51–53).

Appendix

A.1 Digitizer Plot for the Electromagnetic Calorimeter

Fig. A.1 shows the difference of a digitised hit in the electromagnetic calorimeter and the actual energy deposit position in the x-y plane. One would expect a uniform distribution over the individual hexagonal cell, analogous to Fig. 5.2. This requirement is, apart from binning effects at the edges, fulfilled.

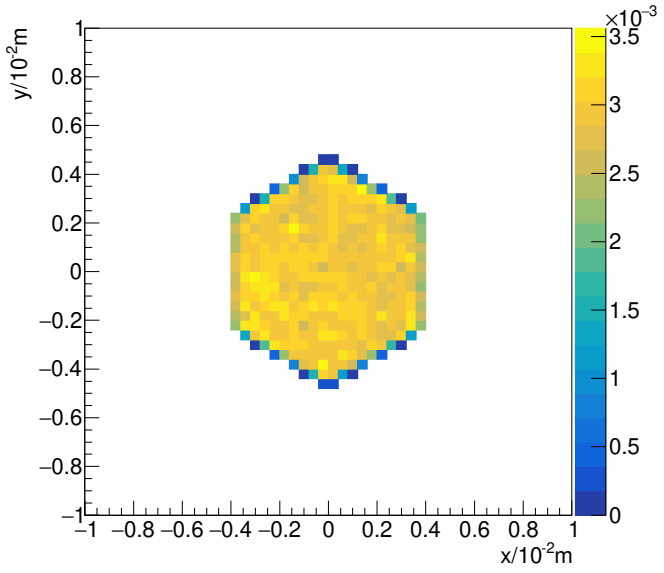
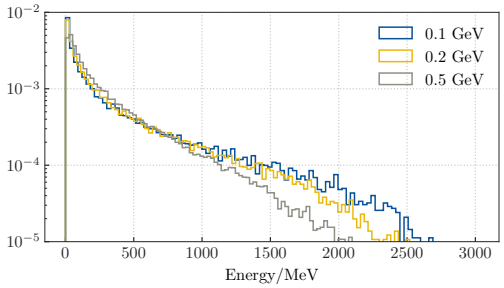


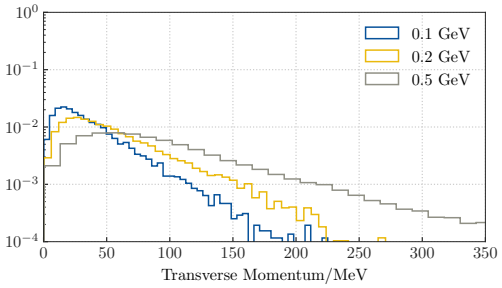
Figure A.1: This normalised histogram shows the difference between a digitised hit in the electromagnetic calorimeter and the actual energy deposit position in the x-y plane for all cells in the calorimeter. The colour indicates bin content. In total this histogram has 556 179 entries

A.2 Kinematic Variables for Signal Events Generated with a 3.2 GeV Beam

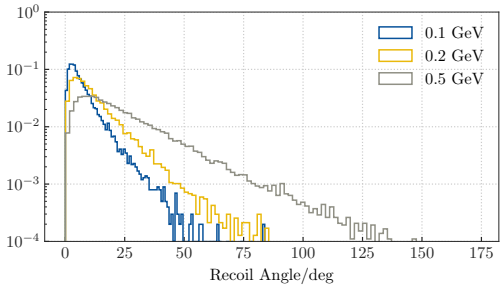
Fig. A.2 shows the same kinematic variables as Fig. 4.9. However, they are generated using a 3.2 GeV beam and give an impression of the properties of potential signal events at ELSA.



(a) Energy of recoiling electron



(b) Transverse momentum of recoiling electron



(c) Scattering angle

Figure A.2: Shown are selected kinematic variables of signal events generated wit a beam energy of 3.2 GeV. The same cuts as in[Åke+18] have been applied. Histograms have been normalised with their integral.

A.3 Fully Calculated Derivatives For The Kalman Filter Propagation Matrix

This section contains the fully calculated derivatives used in Eqs. (6.13) and (6.15).
Starting with Eq. (6.13)[Kre12]:

$$
\begin{aligned}
x' &= t_x \\
y' &= t_y \\
t_x' &= \kappa \cdot \frac{q}{p} \cdot \sqrt{t_x^2 + t_y^2 + 1} \cdot (t_y B_z - (1 + t_x^2) B_y + t_x t_y B_x) \\
t_y' &= \kappa \cdot \frac{q}{p} \cdot \sqrt{t_x^2 + t_y^2 + 1} \cdot (-t_x B_z - (1 + t_y^2) B_x - t_x t_y B_y) \\
\left(\frac{q}{p}\right)' &= 0
\end{aligned}
$$

Where the subscripts on the magnetic field $\mathbf{B}$ denote the respective coordinate.
Next are the derivatives used in the matrix in Eq. (6.15):

$$
\begin{aligned}
\frac{\partial t_x'}{\partial t_x} &= \kappa \cdot \frac{q}{p} \cdot \frac{B_x(t_y^3 + 2t_x^2 t_y + t_y) - t_x B_y(2t_y^2 + 3t_x^2 + 3) + B_z t_x t_y}{\sqrt{t_x^2 + t_y^2 + 1}} \\
\frac{\partial t_x'}{\partial t_y} &= \kappa \cdot \frac{q}{p} \cdot \frac{B_x t_x(2t_y^2 + t_x^2 + 1) - B_y t_y(t_x^2 + 1) - B_z(-2t_y^2 - t_x^2 - 1)}{\sqrt{t_x^2 + t_y^2 + 1}} \\
\frac{\partial t_x'}{\partial (\frac{q}{p})} &= \kappa \cdot \sqrt{t_x^2 + t_y^2 + 1} \cdot (t_y B_z - (1 + t_x^2) B_y + t_x t_y B_x) \\
\frac{\partial t_y'}{\partial t_x} &= \kappa \cdot \frac{q}{p} \cdot \frac{-B_x t_x(t_y^2 + 1) - B_y t_y(2t_x^2 + t_y^2 + 1) - B_z(2t_x^2 + t_y^2 + 1)}{\sqrt{t_x^2 + t_y^2 + 1}} \\
\frac{\partial t_y'}{\partial t_y} &= -\kappa \cdot \frac{q}{p} \cdot \frac{B_x(2t_x^2 t_y + 3t_y^3 + 3t_y) + B_y(t_x^3 + 2t_x t_y^2 + t_x) + B_z t_x t_y}{\sqrt{t_x^2 + t_y^2 + 1}} \\
\frac{\partial t_y'}{\partial (\frac{q}{p})} &= \kappa \cdot \sqrt{t_x^2 + t_y^2 + 1} \cdot (-t_x B_z - (1 + t_y^2) B_x - t_x t_y B_y)
\end{aligned}
$$

List of Figures

List of Tables

Listings

Acknowledgements

First, I would like to thank Klaus Desch for proposing this interesting topic, when I approached him for potential thesis topics. His advice on physical matters was helpful throughout the thesis. I am also grateful for the cordial integration into the Desch group by all its members.

Physical advice by Philip Bechtle was always helpful and I want to thank him for having been available and helping out with problems during the course of this thesis.

I am very grateful for the support I received by Oliver Freyermuth. He was very patient concerning program specific and IT related questions and problems. He helped out, whenever there was need for advice.

Thanks also go to Patrick Schwäbig, Nassim Ainouz, Dominik Stamen and Philip Bechtle for offering to read parts of the thesis and giving advice on improvements.

Lastly, many thanks go to my family and friends for their emotional support during the last year. This is especially helpful during a global pandemic, when working conditions are not ideal and opportunities for recreational activities are limited.