Optimisation of the Data Reduction for the Belle II Pixel Detector and Development of a new Track Finding Algorithm using the Belle II Vertex Detector

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

von
Christian Wessel
aus
Ostercappeln

Bonn, 21.05.2022

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter:	Prof. Dr. Jochen Dingfelder
2. Gutachter:	Prof. Dr. Florian Bernlochner
Tag der Promotion:	27.06.2022
Erscheinungsjahr:	2023

Contents

CHAPTER 1

Introduction

Particle physics experiments aim for ever higher event rates and detectors with higher granularity and increasing number of readout channels. Especially silicon-based detectors in close proximity to the interaction region nowadays provide millions of channels. The combination of both increased event rates and number of pixels and strips leads to high data rates that need to be transmitted and the data need to be stored. However, often only a small fraction of the information is of interest for physics analysis, while most of the hits are from background processes.

The Belle II experiment [1], located in Tsukuba, Japan, utilises a pixel detector with nearly 8 million pixels at radii of only 14 mm and 22 mm from the interaction point. In the final phase, when the experiment will be operated with its target instantaneous luminosity of $6 \times 10^{35}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ and trigger rate of 30 kHz, the expected data rate of the Pixel Detector (PXD) is 256 Gbps at 3 % occupancy. Only a small fraction of the hits are from decay products of $\Upsilon(4S)$ mesons and other interesting collision events. On average, 11 tracks are created in these events, resulting in about 25 hits on the PXD, while in total more than 40000 hits are recorded per readout cycle of the PXD. Thus, more than 99.9 % of the hits on the PXD are from background, and it is neither feasible nor useful to store all of the information, and an online data reduction for the PXD is employed. To accomplish this, the data of the tracking detectors surrounding the PXD are used to reconstruct tracks and extrapolate them to the PXD sensors to find intercepts. Around each intercept a Region of Interest (ROI) is created, and only pixel information inside these ROI are stored. With this approach, a data reduction by a factor of around 10 is targeted. However, this value only is necessary for an occupancy on the PXD of 3 %, for lower occupancies lower data reduction rates are acceptable, too.

Two systems are employed for the task, the High Level Trigger (HLT)and the Data Acquisition Tracking and Concentrator Online Node (DATCON). While the HLT used the same track finding algorithms that are also used for offline track reconstruction based on hits from the Silicon strip Vertex Detector (SVD) and the Central Drift Chamber (CDC), DATCON makes use of the Hough transformation with information from the SVD alone to find patterns in the data. DATCON uses Field Programmable Gate Arrays (FPGA) boards to ensure a fast and predictable execution time in order to provide ROI faster than the trigger rate of 30 kHz. Previous developments of DATCON are reported in [2–4]. In this thesis, the algorithms are improved and adapted to represent the FPGA implementation described in [4] in C++ in the Belle II software framework. They are tested and optimised with simulated $\Upsilon(4S)$ events with simulated beam backgrounds, all of which is presented in Chapter 4.

Due to the increasing event rates and higher number of hits, fast reconstruction algorithms that can

cope with these conditions become more and more important. The track reconstruction algorithms belong to the most important and time consuming algorithms in the full event reconstruction. While the Belle II experiment already successfully records data and uses the existing track finding algorithms, improvements in the track finding efficiency and the reduction of the number of wrongly reconstructed tracks are always possible. Based on the Hough transformation, a new SVD standalone track finding algorithm is developed. Although the Hough transformation is widely used for pattern recognition also in particle physics experiments, no algorithm for track reconstruction in the SVD based on this concept was available to Belle II before this thesis. To cope with random combinations of hits found as track candidates by the Hough transformation, additional algorithm are developed to distinguish good and bad tracks. As for DATCON, these algorithms are validated on simulated events, as well as on actual data recorded by Belle II. The development of this new algorithm is described in Chapter 5.

Before describing the new developments for DATCON and the new track finding algorithm, the general concepts of particle physics, the Belle II experiment, and its tracking detectors are introduced in Chapter 2. This is followed by an introduction to the concepts of track finding and fitting and performance metrics for track finding in Chapter 3.

CHAPTER 2

Experimental Setup

The Belle II Experiment offers unique possibilities to study very rare processes and to solve shortcomings of the currently best tested theory of particle physics. This chapter will provide an overview over the Standard Model (SM) of particle physics, the Belle II Experiment and the SuperKEKB accelerator. In addition, a brief introduction of the interactions of (charged) particles with matter, the background processes, the readout chain of the Belle II Experiment, as well as the Belle Analysis Software Framework 2 (basf2) will be given.

2.1 The Standard Model of particle physics

The foundations of the SM of particle physics were laid in the early 20th century with the discovery and development of the theory of quantum mechanics [5–7]. It was extended over time with the introduction of Quantum Electro Dynamics (QED) [8–11], the electroweak unification [12–14], Quantum Chromo Dynamics (QCD) [15–17] and the Higgs mechanism [18, 19]. Although the SM is a very successful theory that correctly predicted and still predicts many phenomena and particles, it has some shortcomings. One of these is that neutrinos are massless in the SM, while measurements of neutrino oscillations proved that neutrinos indeed have a very small but non-zero mass [20–25]. In addition, the measured parity and Charge-Parity (CP) violation [26] in several processes like oscillations of neutral K mesons [27] or B mesons [28, 29] is not enough to explain the observed matter-antimatter asymmetry in the universe. At the time of the big bang there was one additional matter particle for one billion matter and antimatter particles each, which did not annihilate and remained. These are the foundation for everything we see in the universe today, including ourselves. Observations of rotation curves of galaxies show that the measured velocity distribution as a function of the distance to the centre of the galaxies cannot be explained by the visible matter alone [30, 31]. This is a clear hint for the existence of Dark Matter (DM), which also is not part of the SM. There are several extensions of the SM which introduce new particles as DM candidates, but so far none of these particles is proven to exist.

In the SM 17 particles [32] (excluding anti-particles) are known which can be sorted in four categories: *quarks*, *leptons*, *gauge bosons*, and the *Higgs boson*. They are depicted in Figure 2.1. The six quarks and leptons are themselves grouped in three *generations* each. In each of the three quark generations there is an *up-type* quark with a charge of $+2/3$ (u, c, t quark) and a *down-type* quark with a charge of $-1/3$ (d, s, b quark). The mass increases with each generation from (u, d)

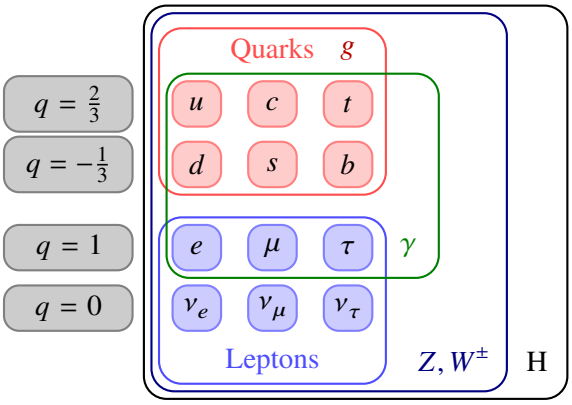


Figure 2.1: Particles of the SM. Each of the two fermion types is grouped by color (quarks, leptons) and in the three generations. The charges of the fermions are depicted in the grey boxes on the left. While the gluon only interacts with the quarks, the photon γ interacts with all charged fermions plus the two $W^\pm$ bosons. Finally, the weak force, which is mediated by the Z and $W^\pm$ bosons, acts on all fermions, and the Higgs boson H which interacts with all massive particles, i.e all particles except for photon and gluon.

over (c, s) to (t, b) with the top quark (t) having a mass similar to a tungsten atom. In addition to their electric charge the quarks also carry a strong or colour charge (*red*, *blue*, and *green*) and they all participate in the strong interaction with the *gluon* (g) as mediator. In contrast, each lepton generation consists of a charged lepton with a charge of -1 (e, μ, τ) and a neutral and nearly massless *neutrino* (ν_e, ν_μ, ν_τ), again with increasing mass over generations for the charged leptons. All particles carrying an electrical charge participate in the electromagnetic interaction with the *photon* (γ) as the corresponding mediator. Finally, all quarks and leptons interact weakly with the Z boson and the W^+ / W^- bosons as mediator. For each of the 12 quarks and leptons an anti particle exists, with the exact same properties except for inverse charges.

While the leptons can exist on their own, all quarks except for the top quark form bound states, called hadrons. The top quark is too short-lived and decays weakly before being able to form a bound state with a lighter quark. The known hadrons either consist of three quarks (baryons), or a quark and an anti-quark (meson). All hadrons need to be colour neutral by either containing one quark of each of the three colours (baryons) or a quark and an anti-quark having the corresponding colour and anti-colour. In principle also particles with a higher number of quarks are possible, as long as they are colour neutral. Only the neutrinos, electrons, and protons, which belong to the baryons, are stable on their own. Electrons and protons together with neutrons form the stable universe. While neutrons on their own are not stable and decay within 15 minutes on average [32], they are stable inside atomic nuclei.

2.1.1 Motivation for B physics

While the ATLAS, CMS, and LHCb experiments at the LHC at CERN near Geneva, Switzerland, are particle physics experiments at the *energy frontier*, trying to discover new particles by producing them directly at higher and higher energies of up to 14 TeV, Belle II is a *intensity frontier* experiment. By exploiting a large data set, like its predecessor the Belle experiment, and the precise knowledge of the initial state kinematics, Belle II aims to measure the nature of particles and their interactions with

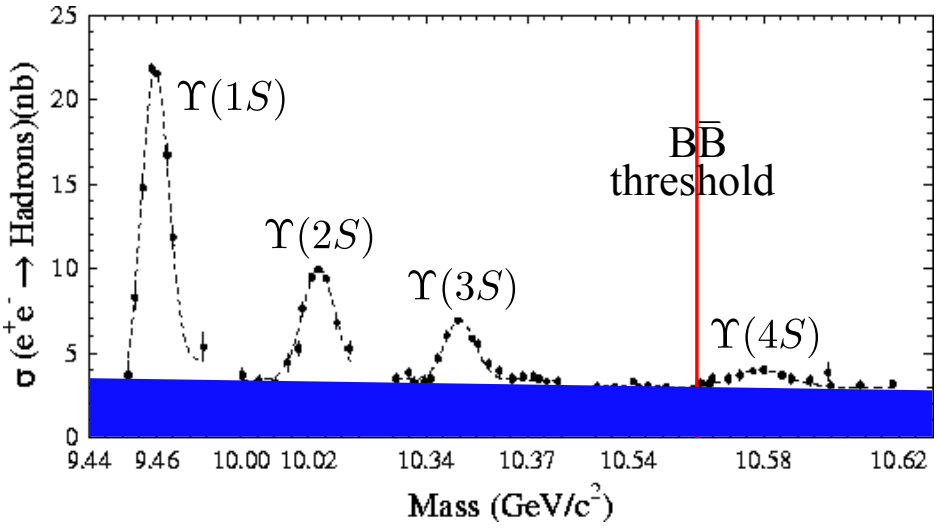


Figure 2.2: Spectrum of the first four $\Upsilon(nS)$ resonances. The vertical red line indicates the threshold for the production of two B mesons in decay, and the blue region at the bottom indicates the continuum floor of lighter quarks (u, d, s, c). Only about one in five events at the $\Upsilon(4S)$ energy are actually $B\bar{B}$ events, all other events only contain lighter quarks. Original in [33], adapted from [34].

unprecedented precision, reducing the uncertainties of the measured quantities by up to an order of magnitude [35].

As mentioned above, the top quark is too short-lived to form bound states. Thus, the heaviest quark known to form bound states is the *bottom* or *beauty* quark. Not only can it bind to lighter quarks, but also to itself, forming a *bottomonium*. The lightest bottommonia that can be created from an e^+e^- initial state without additional particles are the Υ resonances. The lightest four of which are shown in Figure 2.2 with masses ranging from 9.4603 GeV for the $\Upsilon(1S)$ to 10.5794 GeV for the $\Upsilon(4S)$ [32]. Of these four, the $\Upsilon(4S)$ has a mass that is just above the threshold to create two B mesons, which is indicated by the vertical red line. The brownish band at the bottom of marks the continuum production which consists of pairs of all lighter quarks (u, d, s, c) that are created in addition to the Υ states over full range, and hadronise afterwards. With SuperKEKB, continuum events are approximately four times as abundant as $\Upsilon(4S)$ production and thus B meson pairs. B mesons have a mass of $m_B = 5.279\,\text{GeV}$ and consist of a bottom quark and a u or d quark. As the heaviest quarks forming bound states, b quarks offer unique features to study peculiarities of the SM because of their rich spectrum of decay modes.

In particular, they allow studying the weak transition between the third generation and the two lighter generations, investigation of particle flavour oscillations and CP violations. In addition, very rare higher-order processes and the search for processes of physics beyond the SM are possible, which require large data sets of intensity frontier experiments. While heavy particles like new Higgs bosons can be directly discovered at the LHC experiments, they can show themselves in higher order corrections or loop Feynman diagrams of SM processes, where they can hint to beyond SM physics. In addition, the direct discoveries with the LHC experiments are limited to the energy the collider can provide, while indirect searches with Belle II can probe the mass range of up to 100 TeV via contributions in higher order processes.

The key advantage of Belle II, and the last generation *B-factories* Belle and BaBar, compared to the

dedicated B physics experiment LHCb at the LHC is that the initial state is precisely known as the $\Upsilon(4S)$ with known momentum. In contrast, far more B mesons are created at LHCb as they are created by strong interaction processes which cross section is orders of magnitude higher compared to the electroweak production in Belle II. However, the initial state is not precisely known at LHCb, and the reconstruction of neutral particles is worse, among with other difficulties in the event reconstruction. In Belle II it is easy to conclude that neutrinos are involved in a decay chain from the total reconstructed energy and momentum distribution of the final state particles in the event, which is not possible for LHCb.

In addition, Belle II also aims to find or further constrain *beyond SM* processes. Among these are lepton number violating processes like $\tau \rightarrow \mu$ or $\mu \rightarrow e$ without any neutrinos which require a clean experimental environment and large data sets, as well as *dark sector* searches where DM particles leave the experiment undetected, or decaying into several SM particles.

2.2 The SuperKEKB accelerator

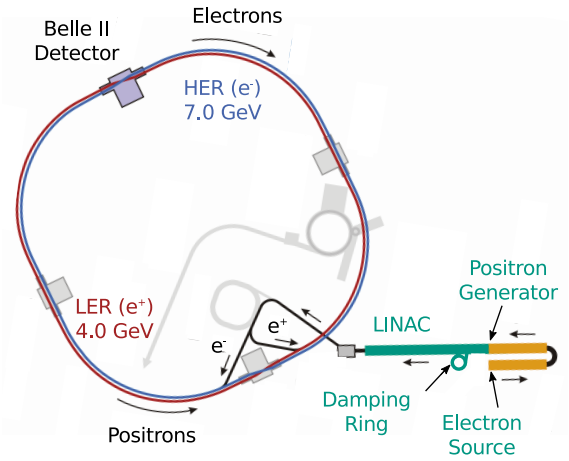


Figure 2.3: SuperKEKB accelerator and collider complex (taken from [36]).

The SuperKEKB collider has a circumference of 3 016 m and located at the kō-enerugī kasokuki kenkyū kikō (High Energy Accelerator Research Organization) (KEK) near Tsukuba, Japan. It is an asymmetric electron positron collider accelerating electrons to an energy of 7 GeV in the High Energy Ring (HER), and positrons to 4 GeV in the Low Energy Ring (LER). This results in a centre of mass energy of $\sqrt{s} = 10.58\,\text{GeV}$, which is the energy of the $\Upsilon(4S)$ resonance. The $\Upsilon(4S)$ is an excited bottomonium state with a mass slightly higher than that of two B mesons, allowing it to decay into exactly two B mesons in more than 96 % of all cases [32]. For this reason SuperKEKB is called a *B-factory*. Both mesons are nearly at rest in the Center of Mass System (CMS), moving along the z-direction of the lab system due to the asymmetric beam energies. A boost of $\beta\gamma = 0.28$ along the z-axis allows for (time dependent) CP violation measurements, as the B mesons fly a few hundred µm before decaying.

An overview of the SuperKEKB accelerator complex is given in Figure 2.3. Electrons are emitted in bunches from a photocathode radio frequency gun at the beginning of the Linear Accelerator (LINAC)

in which they are accelerated to 4 GeV. Half of the electron bunches is brought to 7 GeV right away and injected into the HER. The other half of the electron bunches hits a tungsten target to produce positrons. These are accelerated to 1 GeV first before entering a damping ring to shrink the high emittance of the beam, improving the beam quality. Afterwards they are accelerated to 4 GeV in the remaining part of the LINAC and injected into the LER.

Each ring can store up to 2500 bunches which collide in the Interaction Point (IP) every 4 ns. Both beam's directions deviate slightly from the z-axis with an angle of 41.5 mrad each in the x-z-plane, resulting in a full crossing angle of 83 mrad. This large crossing angle is necessary to squeeze the beams in vertical direction for the *nano beam scheme* which requires new and more complex final focusing magnets. It was initially proposed for the Super B factory in Italy [37]. While the beam size at the IP will finally be as small as $\sigma_x^* = 7.75\,\mu\text{m}$ in horizontal direction, it will be squeezed to an even smaller vertical beam size of $\sigma_y^* = 59\,\text{nm}$. This is necessary to achieve a target luminosity of $\mathscr{L} = 6.5 \times 10^{35}\,\text{cm}^{-2}\,\text{s}^{-1}$ which will set a new world record and be a factor of 30 higher compared to the KEKB accelerator. At the time of writing, a record setting instantaneous luminosity of $\mathscr{L} = 3.8 \times 10^{34}\,\text{cm}^{-2}\,\text{s}^{-1}$ has been achieved.

2.3 The Belle II Experiment

The Belle II detector surrounds the IP and is depicted in Figure 2.4. As a general purpose detector it is designed to measure the properties of the particles produced during the collision and the subsequent decays, except for the undetectable neutrinos. Particles that leave the interaction region and produce detectable signals in the detector are called *final state particles.* The trajectories of charged particles and the vertex positions are measured with the Vertex Detector (VXD) and the CDC. The measurement of the full momentum is enabled by a superconducting solenoid magnet providing a nearly homogeneous field of 1.5 T along the z-axis bending the trajectory in the x-y-plane and allowing for the measurement of the transverse momentum p_T

$$p_\text{T}/\text{GeV/c} = 0.3 \cdot R/\text{m} \cdot B/\text{T}$$

with R as the track's radius in metres and B the solenoid's magnetic field strength in Tesla. As this work focuses on the development of tracking algorithms, the tracking detectors will be explained in more detail in Sections 2.4 to 2.5.

To distinguish charged particles, πs and Ks in particular, the Time Of Propagation (TOP) detector surrounds the CDC in the barrel region, while the Aerogel Ring Imaging Cherenkov (ARICH) is located in the forward direction. Both detectors use Cherenkov radiation to measure particle properties. While the TOP counts single photons and their propagation time inside the quartz crystals, ARICH consists of two aerogel layers with different refractive indices to focus the cones of Cherenkov light onto the readout plane. In both cases the particles β is measured from the opening angle of the Cherenkov cone.

Tracking detectors can only measure the momentum of charged particles, but can't measure neutral particles like photons nor the energy of the charged particles. Energy measurements are performed by the Electromagnetic Calorimeter (ECL) which surrounds all the previously mentioned detectors. It consists of 8 736 CsI(Tl) crystals in which photons, electrons, and positrons create electromagnetic showers and are fully absorbed. From these showers the energy of the particles can be estimated. Because the ECL stops all photons, electrons, and positrons, and also reduces the energy of other

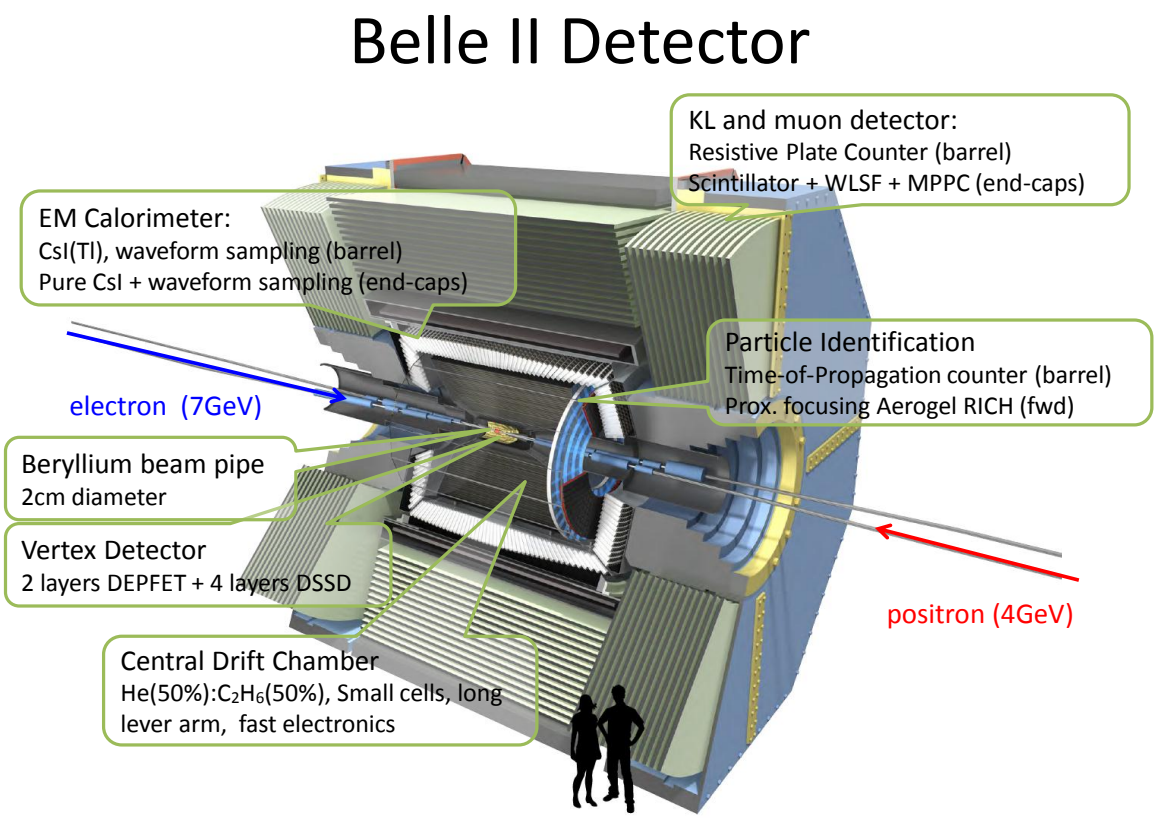


Figure 2.4: The Belle II detector in its final form [38]. The single components are described in the main text.

charged particles, it needs to be positioned outside of the tracking and Particle Identification (PID) detectors. As π^0 decay into two photons in 98.82 % of all cases and $e^+e^-\gamma$ in 1.17 % [32], they can be reconstructed with the ECL information. Additionally, electrons and positrons can radiate photons bremsstrahlung in the inner detectors, which are also measured and can later be attributed to their original particle to correct the momentum.

Heavier charged particles like πs or Ks only lose a fraction of their energy in the ECL and enter the K-Long-Muon detector (KLM) which is the outermost detector. It consists of scintillator strips (end-caps) and glass-electrode resistive plate chambers (barrel) that are installed inside the iron solenoid return yoke that also serves as a shower generator and is used to identify μ and K_L. These particles only rarely interact with the material of the inner detectors, and have a rather long lifetime. In addition, other hadrons like charged π and K that are not completely stopped in the ECL often create measurable signals in the KLM.

In addition to TOP, ARICH, ECL, and KLM, PID information are also provided by CDC and SVD. Both can measure a particle's energy loss per unit length $\left\langle \frac{dE}{dx} \right\rangle$.[1] Especially for charged hadrons with low (transverse) momentum, and those not in the acceptance region of ARICH and TOP, these measurements can be the only reliable PID information.

[1] An introduction to the interaction of particles with matter is given in Section 2.7.

In addition, ECL and CDC data are used to calculate and provide the trigger signal for Belle II, and a TOP trigger is in preparation at the time of writing this thesis. Thus a reliable and fast readout of these detectors and a fast online reconstruction of their signals is essential for Belle II. All trigger calculations and decisions are conducted on FPGA. The CDC trigger reconstructs tracks on FPGAs[2], employing neural networks to estimate track properties. As a result they provide information on how many tracks are found, the angle between tracks in case more than one track was found, the z-coordinate of the vertex of the tracks, and more. The ECL trigger searches for specific patterns in the clusters created by the particles, and in addition to the cluster size and energy it also takes into account the angular separation of the clusters. In addition, the information from both CDC and ECL trigger are combined to either decide whether a valid combination of a CDC track and a ECL cluster was found, and / or whether this likely belongs to a background process (bhabha veto). Lastly, the measurements from SVD, ECL, CDC, and TOP are used to extract precise information about the original event time t_0. However, for the level 1 (L1) trigger, only CDC and ECL hit information are used.

2.4 Central Drift Chamber

The CDC is the main tracking detector of Belle II. It is a gaseous detector with 14 336 *sense wires* and 42 240 *field wires* covering a radial range between 16 cm and 111 cm. Charged particles traversing the CDC ionise the gas, which is mixture of 50 % helium and 50 % ethane with a very small admixture of water vapour. Field wires are necessary to create a nearly radial symmetric electrical field around the sense wires, which is distorted by the magnetic field of the solenoid. While the ions drift along the electrical field lines towards the field wires, the electrons drift towards the sense wires, which are on a potential of about 2 kV relative to the field wires, where they are accelerated on the last few µm creating an avalanche which is easier to measure.

The sense wires are arranged in 56 layers which are grouped in 9 Super Layer (SL). Figure 2.5 shows a section of the x-y-projection of the CDC in the top part, and in the r-z-projection in the bottom part. All SLs except for the innermost consist of 6 layers (SLs 1 to 8), while the innermost SL 0 consists of 8 layers of wires. The SLs with an even number contain *axial* wires that are parallel to the z-axis (denoted as A, coloured in blue in Figure 2.5), while the SLs with odd numbers are *stereo* layers, with the wires being slightly skewed compared to the z-axis with positive angles (SL 1 and SL 5, denoted as U, red) and negative angles (SL 3 and SL 7, V, green), respectively. This AUAVAUAVA pattern of layers enables the CDC to measure the dip angle $\lambda = \pi/2 - \theta$, with θ being the polar angle, and subsequently p_z and the full momentum, which would not be possible otherwise.

The length of the wires is very different between the innermost and the outermost layers, ranging from 70 cm to 230 cm. While the angular coverage ranges from 17° to 150° in θ in the innermost layers, it is much smaller for particles that traverse all layers. The 55 layers of field wires are located between the sense wire layers to create one *drift cell* per sense wire.

2.4.1 CDC hit reconstruction

The momentum of particles traversing the CDC and their dip angle λ determines the path length per drift cell and thus the number of primary electrons created by ionisation. As all electrons created in a

[2] Information on FPGA are and how they work are provided later in Chapter 4.

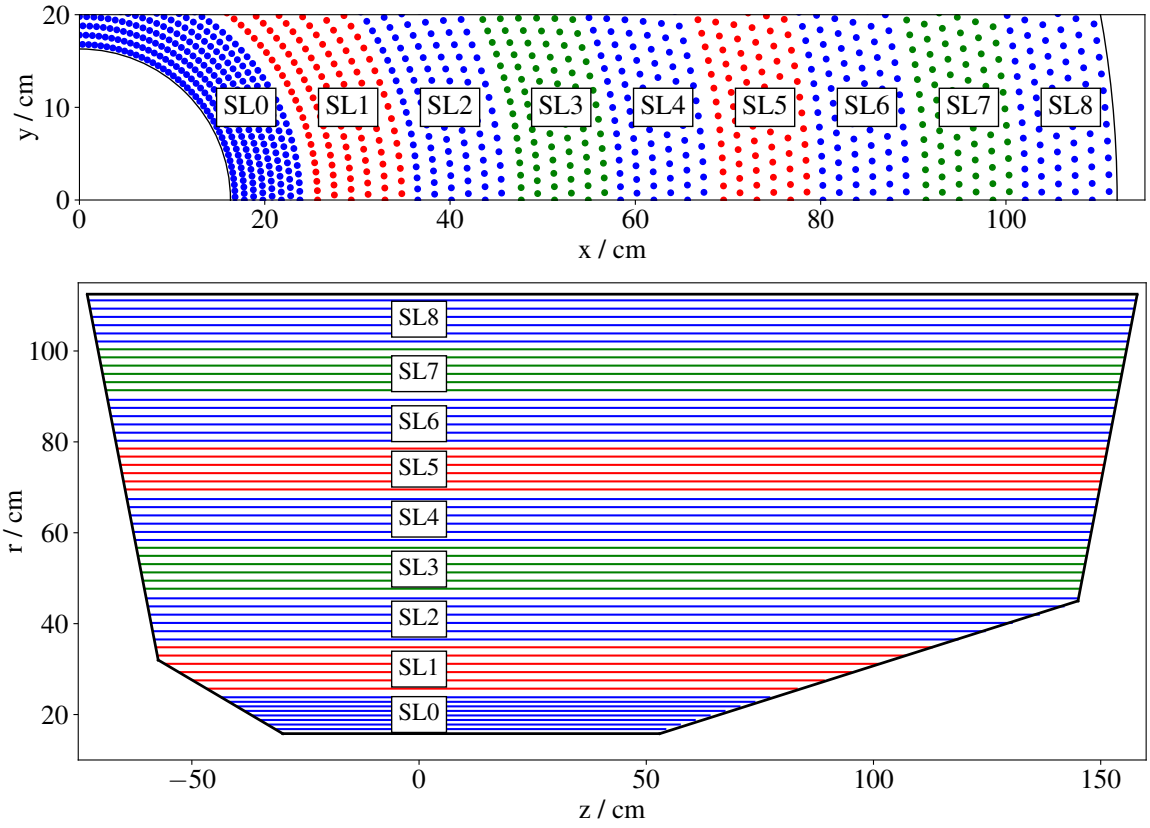


Figure 2.5: Cross section view of the CDC in the x-y-plane (top) and in the r-z-plane (bottom). The wires of the axial SL are coloured in blue, and those of the stereo SLs are coloured in red (green) with positive (negative) skew angle. Since the stereo SLs don't have fixed x-y-coordinates, their x-y-coordinates at $z = 0$ are shown.

drift cell follow the electrical field to the sense wire, the total number of electrons per track can be estimated from the total charge collected at each wire. This can be used to calculate the average energy loss per unit length $\left\langle \frac{dE}{dx} \right\rangle$ of each track to use in PID, as the $\left\langle \frac{dE}{dx} \right\rangle$ depends on the particle type and its momentum. Especially in the low momentum region ($p < 0.5 - 1.0$ GeV/c) particles can clearly be distinguished from one another, while in the higher momentum region most of the single particle bands overlap. The data of the wires are read out with a frequency of 1.02 GHz at the backward side only in order to keep the amount of material between the IP and ARICH at a minimum. Thus, charge sharing between the forward and backward ends of the wires can not be utilised for position estimation along the wire for single hits (= z position of the hits). Instead, the time between the trigger signal and the arrival of the charges at each sense wire is measured, allowing the calculation of the *drift distance* which is used for precise track reconstruction. After sampling in the Front End Electronics (FEE) the hit data are sent to the CDC trigger system. While the energy measurement at each wire, represented by the Analog-to-Digital Converter (ADC) and Time Digital Converter (TDC) values, are used for PID in combination with the wire positions, the wire positions and the drift distance are the CDC input for track reconstruction.

Table 2.1: Specifications of the Belle II PXD and SVD [39, 41].

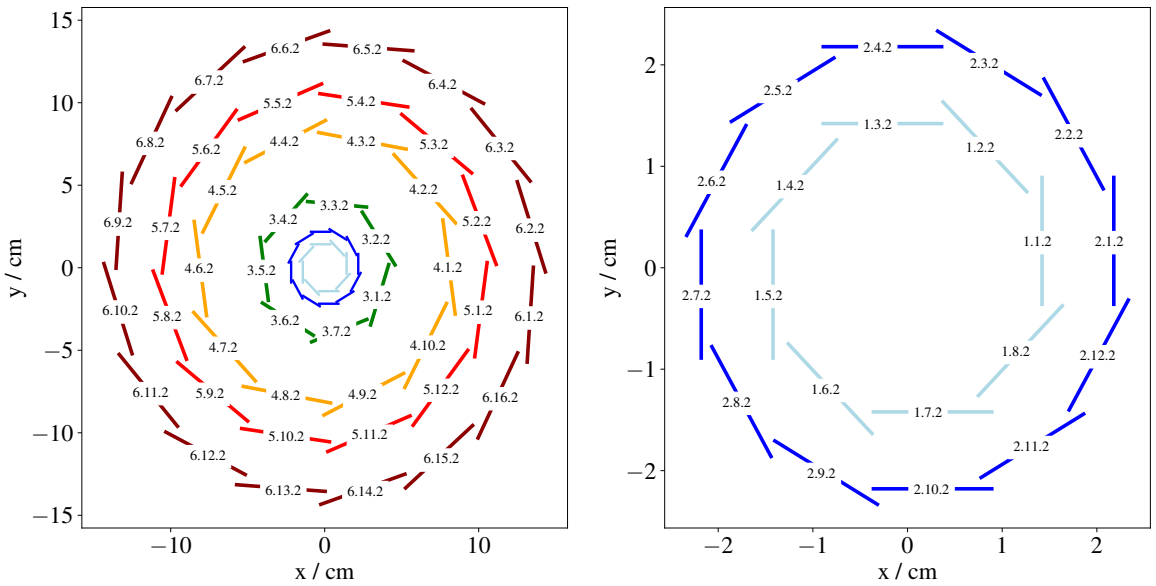
	PXD	SVD
Layer radii in mm	14, 22	39, 80, 104, 135
Number of channels	7 680 000	132 096 (u) + 91 648 (v)
Number of sensors	40	172
Pitch (u / v) in µm	50 / 55 to 85	50 to 75 / 160 to 240
Radiation length per layer	0.19 %	0.6 %
Thickness in active area in µm	75	300
Integration time	20 µs	300 ns
Expected occupancy	max. 3 %	max. 8 %
Acceptance range θ	17° to 150°	17° to 150°

2.5 Vertex Detectors

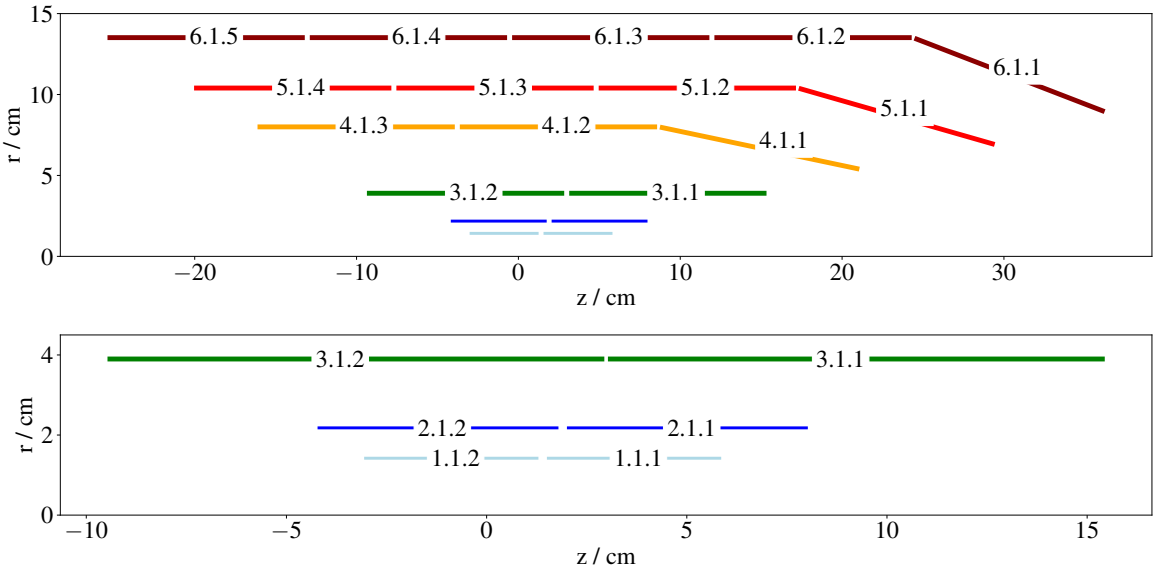
For precise measurements of the vertex properties of the particles created in the collision and the subsequent decays, a high spatial density of measurement points with high resolution is required. As it is difficult for gas based detectors to provide very precise position measurements in a timely manner, silicon based detectors are the technology of choice for this purpose in all current High Energy Particle Physics (HEP) experiments. In Belle II the VXD is employed for this task. It consists of a two layer pixel detector employing the DEpleted P-channel Field Effect Transistor (DEPFET) technology, called PXD, and four layers of Double-Sided Silicon Strip Detector (DSSD) sensors, called SVD. Each layer is comprised of equal ladders which are located at different azimuthal angles.

2.5.1 The SVD

The SVD covers a radial range of 39 mm to 135 mm in the barrel region. To reduce the amount of material particles have to traverse, the forward sensors of layers 4 to 6 are slanted with respect to the barrel sensors, and have a trapezoidal shape, as shown in Figure 2.6. The strips on the two sides of the DSSD sensors are orientated perpendicular to each other. The p-doped strips are parallel to the z-axis and enable the measurements in the r-φ-direction, or in the u-direction in local coordinates. In contrast, the n-doped strips are orthogonal to the p-doped strips and oriented along the r-φ-direction, enabling measurements in the v-direction in local coordinates. These coordinates are used to describe the position of a particle hit or of a strip or pixel on the sensor with the origin of the local coordinate system in the centre of the active area of a sensor (for both SVD and PXD). All sensors contain 768 strips in the u-direction. While the layer 3 sensors contain 768 strips in the v-direction, layers 4 to 6 contain only 512 v-strips, resulting in 132 096 (91 648) u-strips (v-strips). For more precise measurements, the strip pitch on layer 3 is 50 µm (160 µm) in u-direction (v-direction), while it is 75 µm (240 µm) in u-direction (v-direction) on the barrel sensors of the other layers. The slanted sensors are different, as their pitch in u-direction varies from 75 µm (back) to 50 µm (front), while the pitch in v-direction is the same as on the barrel sensors. Table 2.1 summarizes the most important geometrical quantities of the SVD.



(a) x-y-view of the VXD showing the full cross-section of the VXD (left) and a zoom to the PXD layers (right).



(b) Simplified r-z-view of the VXD with the full VXD in the top part and a zoom to the PXD layers in the bottom part.

Figure 2.6: Sketch of the geometry of the VXD. Each line corresponds to one sensor (a) or ladder (b), the thickness is exaggerated for visibility. Values for the figure taken from [39], and the Belle II software basf2 [40], design inspired by [36].

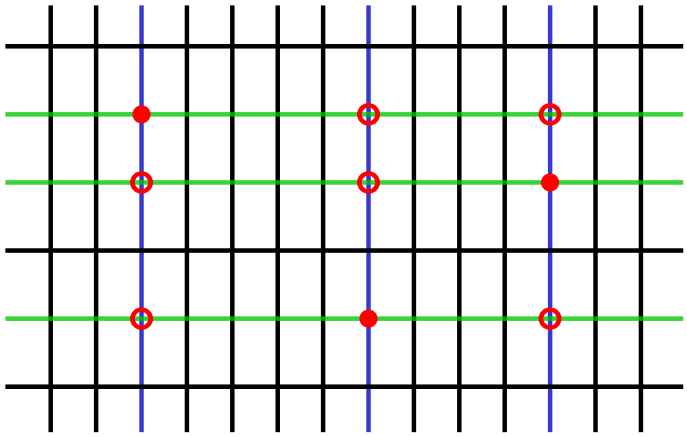


Figure 2.7: Illustration of ghost hits in the SVD. Only three actual hits are present, indicated by the red points, and the blue and green lines for the two strip directions. In addition to the three real hits, six ghost hits are created, indicated by the red circles, by combining all blue with all green strips. During reconstruction of SVD hits the creation of ghost hits needs to be avoided when possible, e.g. by using hit time and energy measurements.

SVD hit reconstruction

Since the SVD strips are up to 12.88 cm long, measurements of individual strips are not particularly valuable for track and vertex finding. Instead, 3D hits, called *space points*, are reconstructed from the strip information. First, each hit is reconstructed individually by performing a fit to the wave form of the Analogue Pipeline Voltage (APV) and checking its Signal-to-Noise-Ratio (SNR). Only if the SNR is high enough (usually SNR ≥ 5), and if the fit is successful and provides a decent fit quality, the hit is accepted and not discarded as noise. Afterwards, all accepted strips on each sensor and each side (p and n) are clustered. Adjacent strips are combined into one hit, for which the overall SNR is calculated again. For each cluster the cluster position and time are calculated, which are used in tracking, as well as the deposited energy (for PID). More detailed information on the reconstruction of the hit position and hit time are provided in [39].

However, because the strip and cluster measurements are only one dimensional on a 2D detector surface, they need to be combined to the aforementioned space points as 3D hits. First, for each sensor each cluster from either side is combined into a 2D measurement if their hit time and charge are similar enough. But this introduces a problem: two hits on either side can be combined in four different ways, three hits on either side can be combined in nine different ways, and so on. While the actual combination of strips crossing at the point of traversal of a track is part of the set of combinations, additional hits, so called *ghost hits* are created. This is illustrated in Figure 2.7. Creating ghost hits can be partially avoided by carefully optimising the way the single-side hits are combined into 2D hits, but their creation is in general unavoidable. Finally, the 2D measurements on each sensor are converted to true 3D space points using the information about the location of each sensor in space.

2.5.2 The PXD

The main task of the PXD is to provide precise measurements close to the IP to enable high precision estimations of particle decay vertices. To achieve this, the pixels have a small size of 50 µm in the u-direction, which is the local coordinate pointing in circumference or r-φ-direction, and pixel sizes of 55 to 85 µm in v-direction, the local coordinate along the global z-direction. To place the PXD as close to the Interaction Region (IR) as possible, the radii of the two layers are 14 mm and 22 mm. The most important quantities of the PXD are summarized in Table 2.1. In total the PXD consists of 8 ladders in layer 1 and 12 ladders in layer 2, each containing two sensors, aggregating in 40 sensors in total. However, in the first years of operation, layer 2 is only installed partially. Due to difficulties at assembly, only the ladders 4 and 5 of layer 2 are installed, at azimuth angles of φ= 120° and φ= 150°, respectively. The two sensors of each ladder are glued together in the middle.

Detectors based on DEPFET technology, which was first proposed by J. Kemmer and G. Lutz in 1987 in [42], are able to operate with a low power consumption in the pixel area itself while providing a signal with a high SNR. On the PXD the readout electronics (Drain Current Digitizer (DCD) and Data Handling Processor (DHP)) are located in the very forward or backward end of each chip, outside of the acceptance, where they can easily be cooled. Each sensor carries four DCD and four DHP, generating 1 W each, thus 8 W in total per sensor. In addition, six switcher chips are located at one long side of the sensor, inside the acceptance, each generating 0.5 W, which can be cooled with forced air flow. Because of the location of the DCD and DHP the PXD is operated in a *rolling shutter* mode. All rows in each sensor are read out sequentially, with four rows being read out at a time. Thus 192 readout operations are performed for each readout cycle, each lasting 100 ns, resulting in a total integration time of 19.2 µs. This is much slower than all other Belle II sub-detectors and renders the PXD unusable for utilisation in the trigger decision and leading to up to 3 % of occupancy in each readout cycle. The occupancy is the fraction of pixels (or strips in case of the SVD) with a signal above threshold compared to the total number of pixels (strips) in the PXD (SVD). In case the occupancy exceeds 3 % for several events in series, the internal buffers are not big enough to store all data and data loss occurs on the chip itself. While in most cases only one interesting event is recorded during one integration cycle with ten tracks on average, most of the hits accumulated are from (beam) background processes. At nominal luminosity far more than 99 % of all hits recorded by the PXD per readout are from background. Details on the different beam background sources are described in Section 2.6.

The overwhelming number of beam background hits on the PXD in each readout cycle poses several problems. First of all, a high number of hits in the PXD means an increased likelihood to assign wrong PXD hits to tracks. To avoid this, the track properties need to be known very precisely. Second, the vast majority of hits is of no interest at all for analysis. In addition, the high number of hits implies a high data rate from the PXD to the storage system ($\mathcal{O}$(1 MB) per event), and a lot of wasted storage. Thus, when designing the PXD, an online data reduction of the number of hits by a factor of about ten was foreseen. Two independent systems and methods were planned to be employed for this task. In both cases the data of the surrounding tracking sub-detectors are used to reconstruct tracks and calculate ROI on the PXD to only store hits inside the ROI.

The first system is the Belle II HLT running on a server farm next to the experiment itself. While it was initially planned for the HLT to only use tracks based on the data provided by the SVD, during development of the algorithms in basf2 it was decided that the HLT should perform the full tracking chain including CDC tracking to minimise systematic uncertainties between the online tracking on

HLT and the offline tracking during reconstruction for analysis. Thus tracks based on both SVD and CDC hits are reconstructed in the HLT and extrapolated to the PXD on which ROI are calculated.

However, this is not the case for Monte Carlo (MC) data production. During simulation and creation of MC data the trigger as well as the PXD data reduction are simulated before performing the full reconstruction including tracking. Since the CDC tracking is very time consuming, it is not executed for ROI creation during MC data production, but only SVD tracking with the Vertex Detector Track Finder 2 (VXDTF2) is performed as input for ROI creation.

The second system used for data reduction is the DATCON, which is described in more detail in Chapter 4. The first implementation of DATCON is described in [2], containing optimisations described in [43]. Further developments and improvements of DATCON are described in [3]. DATCON is an FPGA based online system. In this context, the term online refers to all systems that are part of the readout and data acquisition system, which need to have a short processing time per event, whereas offline refers to the algorithms that are used for calibration and event reconstruction for later analysis. In Belle II, large parts of the reconstruction software is the same online and offline, while FPGA systems are only part of the online system. As the HLT it receives all SVD hits with a sufficiently high SNR for each event with a positive L1 trigger decision and calculates ROI on the PXD using these information.

2.5.3 Data Acquisition and Triggering

Because of the dependence of the PXD Data Acquisition (DAQ) and data reduction of the global DAQ and the trigger scheme and the HLT, the Belle II DAQ chain is briefly introduced in this section. A simplified sketch of the Belle II DAQ system is shown in Figure 2.8.

While electron and positron bunches cross at the interaction point every 4 ns, in most cases nothing of interest happens. In case a physics process of interest takes place the signals of CDC, ECL, and TOP are evaluated by the hardware *L1 trigger*, which includes the subdetector trigger and the global decision logic. The L1 trigger consists of several independent FPGA boards which analyse the data in a very short time of only a few µs. If the L1 trigger finds patterns of interest in the data and approves an event, a trigger signal with a unique event number is sent to all subsystems except the PXD.

For each positive L1 trigger decision the data of all detectors except PXD are then sent from their FEE to the Event Builder 1 (EB1), which merges the raw data and sends it to the HLT. On HLT the full reconstruction is performed on the raw data. First the data are unpacked and the individual hit objects in each sub-detector are constructed using basf2. Based on these the full track reconstruction (c.f. Section 3.1) is conducted. All information available from tracking, ECL, and KLM are considered for the HLT decision at the time of this work.

Since the expected L1 trigger is 20 kHz on average, with a maximum of 30 kHz, based on the expected event rate, the HLT has to cope with the L1 rate as input rate. Although the HLT consists of several worker nodes with a total of nearly 10 000 Central Processing Unit (CPU) cores, one of the key requirements is that it does not take more than 5 s on average to process each event sent by the L1 trigger. Otherwise the queue of unprocessed events in HLT and ONSEN fills up until a maximum is reached, which causes an error and a stop of the Belle II data taking. Thus, the reconstruction software on the HLT needs to run as fast as possible.

In case the event is accepted by the HLT, the found tracks are fitted and extrapolated to the PXD to calculate ROI which are sent to ONSEN. Additionally all raw data and the ROI information are transferred to the EB2.

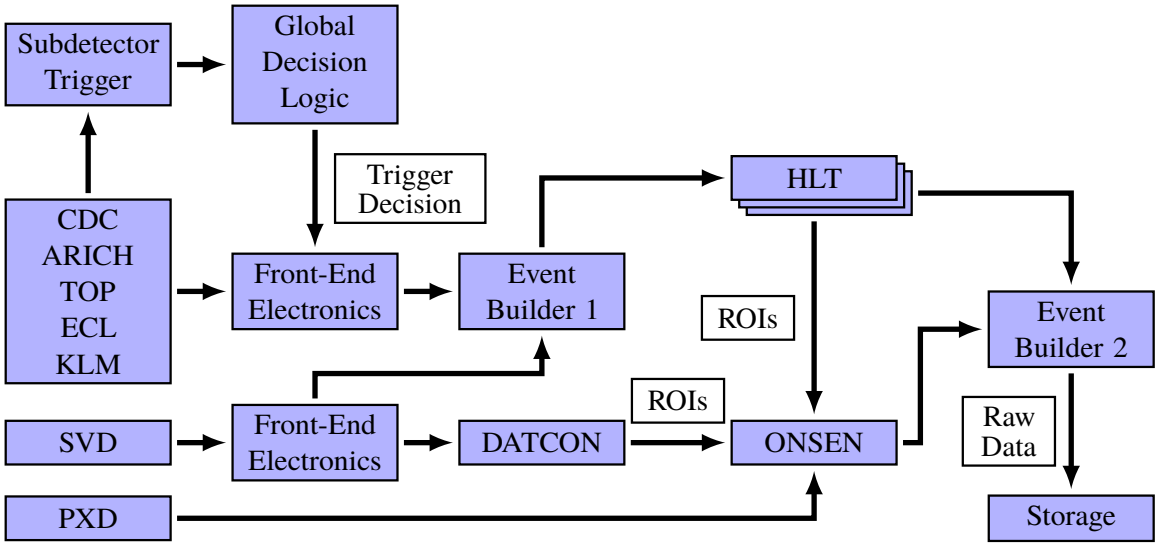


Figure 2.8: Simplified overview of the Belle II DAQ system. For each event accepted by the level 1 trigger, data from all sub-detectors except PXD are sent to EB1 and from there to the HLT where a full reconstruction of the event is conducted, and finally ROIs on the PXD are created. Meanwhile, data from SVD are sent to DATCON where ROIs are calculated. All ROIs are subsequently transferred to ONline SElector Node (ONSEN) where the PXD hits inside the ROIs are selected. Afterwards all data are combined in the Event Builder 2 (EB2) and sent to mass storage.

On the second path, DATCON receives data from the SVD for each event accepted by the L1 trigger. DATCON calculates ROI as described in Chapter 4 and sends them to ONSEN as well. The selection inside ONSEN is a mixture of a logical ***and*** and a logical ***or***. While any active PXD pixel contained in an ROI from HLT or DATCON is saved, a signal to accept an event from HLT is required to issue the selection. This HLT signal can be ROI or an empty event where no ROI are found, but the HLT still marks the event as interesting for keeping, like a long lived neutral particle decaying outside of the PXD.

Usually the HLT ROI calculation usually takes longer than the ROI calculation on DATCON, thus ONSEN conducts the hit selection for an event as soon as the HLT ROI arrive regardless of HLT ROIarriving earlier than DATCONROI or not. Since the HLT ROI calculation usually takes longer than the ROI calculation on DATCON, ONSEN conducts the hit selection for an event as soon as the HLT ROI arrive without waiting for DATCON to send ROIs. In case DATCON takes longer than the HLT for its reconstruction and the DATCON ROIsarrive at later than the HLT ROIs for an event, the DATCON ROIsare discarded. Furthermore are the DATCON ROIs discarded by ONSEN in case the HLT rejects an event, or if it does not receive any signal from HLT. However, this should not happen, as the HLT must take a decision to keep an event in every case.

While the online hit selection provides the benefit of a potentially high data reduction, it also introduces a few problems. In case of an inefficient track reconstruction in both HLT and DATCON, which result in missing tracks and thus missing ROIs, all PXD hits that are not contained in an ROI are inevitably lost without any chance of recovery. Thus a high ROI finding efficiency has a higher priority compared to a data reduction by a factor of about 10 to keep as many PXD hits of physics tracks hits as possible. Additionally, very busy events with a large number of hits in both SVD and CDC can

lead to large reconstruction times in the HLT. The internal buffers inside ONSEN are designed to store data for up to 5 s, which was defined as longest time the HLT reconstruction and ROI calculation would take per event before data taking started during the design phase of the detector and the DAQ system. Experience has shown that the HLT sometimes needs much longer, even at the low luminosity and data rates at the time of writing. At the time of writing this thesis, no hit selection is performed by ONSEN, this did not yet have any consequences like loss of data.

PXD hit reconstruction

Compared to the SVD, the hit reconstruction and space point creation is much simpler. Again, the single hits of each pixel need to be reconstructed. Because DEPFET detectors have a very low noise by design, nearly all pixel hits are valid, except those that are known to be noisy or defective from calibration. All others are clustered, but in contrast to the SVD not in one dimension, but in two dimensions. Since contrary to the SVD there is no ambiguity of potentially combining hits from both local directions that should not be combined, each valid cluster can directly be converted into a space point, which is then used in vertex reconstruction.

2.6 Beam induced backgrounds

At electron positron colliders there are several background processes that produce signals in the detector, Belle II in this case. Due to the high instantaneous luminosity and the small beam sizes at the IP a large number of background hits is expected in the different Belle II sub-detectors. The VXD and the CDC will suffer most from these backgrounds as the rates for most backgrounds fall off quickly with increasing distance from the IP. Before the full Belle II detector was installed beam induced backgrounds were investigated using the Beam Exorcism for A STable experiment (BEAST) experiment surrounding the beam pipe, called *phase 1*. In 2018 most of Belle II was installed, except for the full VXD. Instead, a slice of the VXD was installed, with one ladder of each layer in the positive x-direction. The remaining VXD volume was filled with the *phase 2* BEAST detectors which consisted of several detectors designated to study the beam induced background. In 2019 the VXD replaced most of the BEAST detectors in the VXD volume. However, only two PXD ladders of the second layer were installed, as described in the Section 2.5. A small set of detectors dedicated for measurements of beam backgrounds remained installed to monitor the backgrounds and estimate the radiation dose deposited in the VXD detectors, as well as to allow for an abort of operation in case the background conditions get out of control. The main beam induced background sources in Belle II, and in the VXD in particular, are described in the following.

Two photon process The two photon process is by far the largest beam background source in the VXD and the CDC. It can be described by QED. From the initial e^+e^- pair an additional e^+e^- pair is produced via two photons in the process

$$e^+e^- \;\rightarrow\; e^+e^-\gamma\gamma \;\rightarrow\; e^+e^-f\bar{f}$$

as shown in Figure 2.9. The newly created fermion pair $\bar{f}f$ in most cases is an e^+e^- pair, too, but it can also be a $\mu^+\mu^-$ pair, a $\tau^+\tau^-$ pair, or two quarks that hadronise. In most cases the

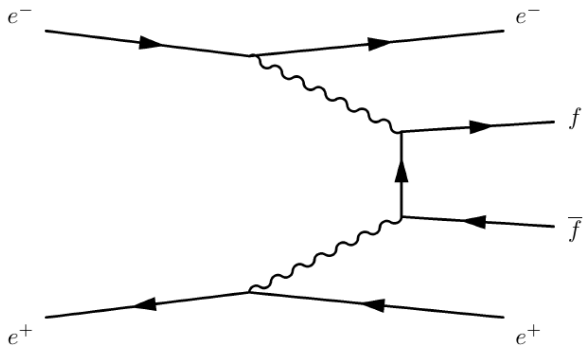


Figure 2.9: Two photon process.

original e^+e^- pair retains a high momentum with a small scattering angle, i.e. the particles propagate in nearly the same direction as they would without scattering, while the newly created particles usually have a very low momentum and thus mainly create hits in the VXD and the innermost CDC SLs.

Touschek effect The particles in a bunch have the same charge and thus repel each other. This process of intra-bunch scattering is called Touschek effect. During such a scattering process both the longitudinal and transverse momentum and energy of the involved particles are changed. This can cause them to leave the beam and to hit the vacuum chamber, beam pipe, or the magnets, producing particle showers which can hit the detector. Touschek background scales with the bunch current squared, the number of bunches, the inverse beam size and the third power of the inverse beam energy E^{-3}, so the increased beam energy of the LER compared to KEKB and the Belle experiment is a first countermeasure against LER Touscheck background. Nonetheless, since the beam size in Belle II at the IPis very small due to the nano beam scheme, the amount of Touscheck background is expected to be high in Belle II. The main contribution here comes from the LER due to the high beam current and the lower beam energy, while the contribution from the HER have a smaller impact because of the higher energy. To reduce the fraction of Touschek background, collimators for the horizontal and the vertical axis are installed along the beam pipe.

Beam-gas scattering Due to imperfect vacuum residual gas will always be left in the beam pipe. Beam particles undergo elastic (bremsstrahlung) and inelastic (Coulomb) scattering with these gas molecules. Some of the scattered particles then hit the vacuum chamber and the magnets and produce particle showers. The rate of this background process is proportional to the beam current, and the vacuum pressure. Thus the main mitigation effort is to conduct *vacuum scrubbing* to remove as much residual gas as possible to improve the vacuum pressure.

Radiative Bhabha scattering Electrons and positrons can produce photons which then propagate along the beam pipe and interact with the magnets or directly hit the detectors. The interaction cross section for photons with nuclei is huge and this process frees neutrons from the nuclei. These neutrons are the main background for the outermost detector KLM, and they are also produced in the beam pipe outside of Belle II, hitting the KLM endcaps from outside. Radiative Bhabha scattering scales linearly with the luminosity, thus it is expected to be higher by a factor

of 30 compared to KEKB. On the other hand this background contribution can be reduced by a different magnet arrangement, and by installing additional shielding around the beam pipe and tunnel to protect the outermost KLM layers.

Synchrotron Radiation Synchrotron radiation is produced when the beam is bent by magnetic fields. It consists of γ-rays in the lower keV range, its rate is proportional to the beam current, the square of the beam energy and the square of the magnetic field, thus the contribution of the HER dominates. Additionally, secondary synchrotron is seen in the PXD. Photons produced in the final focusing magnets hit the beam pipe, from where they get back scattered onto the PXD, or the atoms in the beam pipe material are excited, and the relaxation photons then hit the PXD layer 1. At the time of writing, synchrotron radiation is not included in the beam background simulation and thus also not in the background samples used throughout the studies presented in this work. Thus the rate and amount of beam induced backgrounds in the simulated background samples is very likely to be underestimated. However, it is contained in random trigger beam background data, which are not used in this work.

Injection Backgrounds If bunches are newly injected into the two rings, not all particles do have the desired momentum and energy. These bunches need to *cool down*, such that the new particles behave like all other particles in the beams. Particles with the largest momentum and energy offset are more likely to leave the bunch, especially when experiencing the strong magnetic fields of the final focusing magnets. Due to these imperfections, the background rates are very high after each injection due to these imperfections, which is referred to as *injection background*. Injection background can create high occupancies in all Belle II sub-detectors, but are most prominent in the tracking detectors. Because of its long integration time of nearly 20 µs, the PXD has a special mode of operation where the pixel matrix is blinded while a newly injected noisy bunch crosses the IP. The cooling phase usually takes two full circulations in the rings, thus the injection background rate is highest and often dominant in the first 20 ms after injection. It is the worst understood beam induced background component. As for Synchrotron radiation, injection backgrounds are currently not simulated and thus also not included in the background samples used for all the studies in this thesis.

2.7 Interaction of particles with matter

All particles traversing matter interact with the nuclei and electrons in that matter via different processes and in a probabilistic way. While neutrinos only interact weakly, making them practically undetectable, photons can interact with matter in four different ways. Elastic scattering like Thomson or Rayleigh scattering [44] only change the photon's direction, while in inelastic Compton scattering [45] an electron is removed from the scattering material and potentially creates an additional track in the detector in addition to changing the photon's direction. Additionally, a photon can ionise an atom, losing some of its energy to the electron. If an electron from Compton scattering or ionisation has enough energy it can itself produce secondary ionisation of the material. A third interaction of photons with matter is pair production in which usually a e^+e^- pair is created. Pair production only is possible if a photon's energy is at least as large as the mass of the to-be created particle pair, e.g. 1.022 MeV to create a e^+e^- pair. If pair production occurs in the tracking volume, two additional tracks are created. In the ECL, detected photons deposit energy mainly through pair production as soon as they arrive in

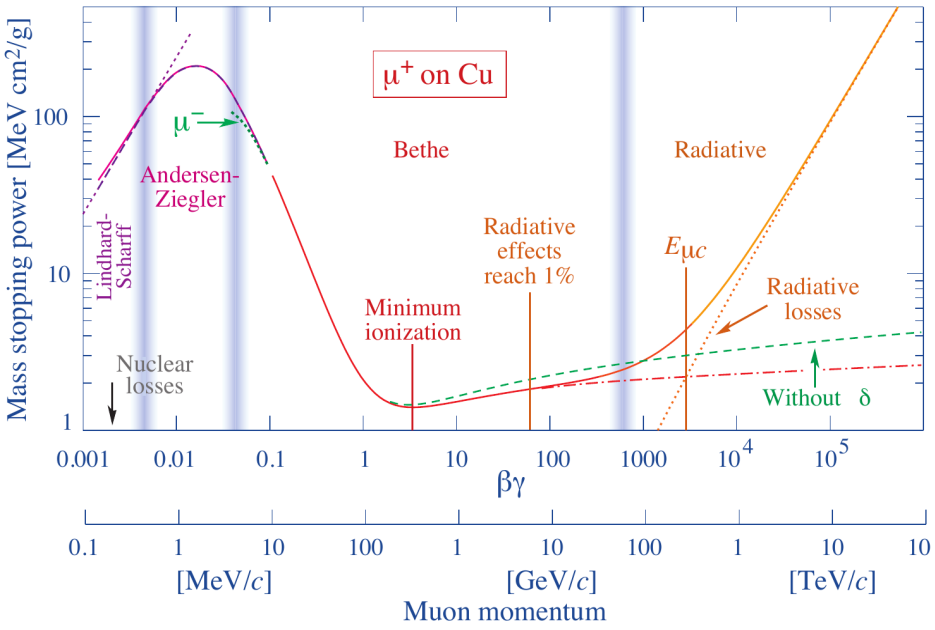


Figure 2.10: Energy loss per unit length following the Bethe-Bloch-Formula, also called mass stopping power [32].

the crystal. The created electrons and positrons themselves radiate off new photons and thus lose part of their energy, and these new photons create a new e^+e^- pair, forming a cascade until no photon can produce additional e^+e^- pairs and until all particles are eventually stopped in the ECL crystals.

The energy loss of charged particles, except for electrons and positrons, is described by the Bethe-Bloch Formula (BBF)

$$-\frac{1}{\rho}\left\langle\frac{\mathrm{d}E}{\mathrm{d}x}\right\rangle = \frac{4\pi\hbar^2\alpha^2}{m_e\beta^2}\frac{z^2Z}{Au}\left(\frac{1}{2}\ln\left(2m_e\beta^2\gamma^2c^2\frac{T_{\max}}{I^2}\right)-\beta^2\right). \tag{2.1}$$

The BBF describes the mean energy loss $\mathrm{d}E$ per length $\mathrm{d}x$, where ρ is the mass density of the traversed material, α the fine structure constant, $\hbar$ the reduced Planck constant, m_e the electron mass, z the charge number of the traversing particle, Z and A the atomic number and the relative atomic mass of the traversed material, u the atomic mass unit, $T_{\max}$ the maximum energy transfer, and I the mean excitation energy of the material. It is not valid for electrons and positrons because of their small mass, and because of the indistinguishability of the traversing electrons with the electrons in the material. However, with additional corrections it is also possible to describe the energy loss of electrons and positrons with the BBF in Equation (2.1). Figure 2.10 shows the graph described by the BBF for a muon in copper over a very wide range of muon momentum. In general particles have their lowest ionisation potential at $\beta\gamma$ values between 2 and 4, which does not increase much for larger values of $\beta\gamma$ until $\beta\gamma$ values of $O(10^3)$ where radiative losses begin to dominate. Thus a particle in the region $1 \lesssim \beta\gamma \lesssim O(10^3)$ is called a Minimum Ionizing Particle (MIP). For MIPs the mean energy loss is $\frac{1}{\rho}\left\langle\frac{\mathrm{d}E}{\mathrm{d}x}\right\rangle \approx 1.5\,\mathrm{MeVcm^2g^{-1}}$, saturating at about $4\,\mathrm{MeVcm^2g^{-1}}$ for larger values of $\beta\gamma$.

Furthermore, the BBF also allows for the description of Multiple Scattering (MS), in which a charged particle is repeatedly elastically scattered while traversing a material. These scattering

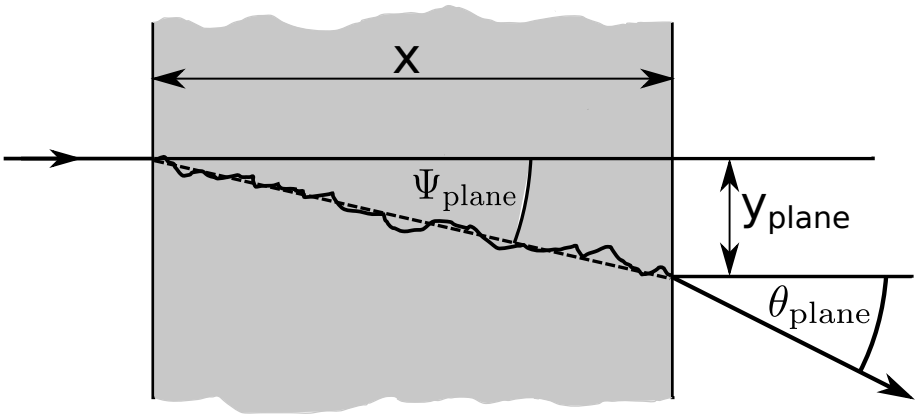


Figure 2.11: Multiple scattering

processes occur often and on very short distances, deflecting the particle randomly by a small angle dθ each time following a Gauss distribution. Figure 2.11 sketches the scattering of a particle while traversing material with a thickness of x. The overall deflection in the material is described by the angle Ψ_{plane}, resulting in a deflection offset of y_{plane} when leaving the material. However, the final scattering angle θ_{plane} determines the trajectory of the particle after leaving the material, and it not necessarily the same as Ψ_{plane}.

The mean scattering angle θ_0 is defined by

$$\theta_0 = \frac{13.6\,\text{MeV}}{\beta c p} z \sqrt{\frac{x}{X_0}} \left(1 + 0.038 \ln\left(\frac{x}{X_0}\frac{z^2}{\beta^2}\right)\right) \tag{2.2}$$

where X_0 is the radiation length,[3] z the charge of the traversing particle, x the flight length in the material and p the total momentum of the traversing particle. Due to the Gaussian nature of this process, the final scattering angle $\Psi^{\text{rms}}_{\text{plane}}$ and offset $y^{\text{rms}}_{\text{plane}}$ are

$$\Psi^{\text{rms}}_{\text{plane}} = \frac{1}{\sqrt{3}}\,\theta^{\text{rms}}_{\text{plane}} = \frac{1}{\sqrt{3}}\,\theta_0 \tag{2.3}$$

$$y^{\text{rms}}_{\text{plane}} = \frac{1}{\sqrt{3}}\,x\,\theta^{\text{rms}}_{\text{plane}} = \frac{1}{\sqrt{3}}\,x\,\theta_0. \tag{2.4}$$

Since MS potentially adds an offset to the trajectory of a particle while traversing matter, as well as deflecting the particle, it deteriorates the track finding and the resolution of the track parameters.

In contrast to MS, which is an undesired interaction of charged particles with matter, there also desired processes used to create signals that are subsequently read out.

First there is ionisation. A charged particle traversing the detector material either ionises a gas, or creates electron-hole-pairs in a semiconductor. The positive and negative charges are then further separated by an applied electrical field and collected in readout electrodes, after which they usually

[3] The radiation length is a material property. It is defined as the the mean path length of a particle needed to decrease the particles energy to $1/e$ while traversing that material.

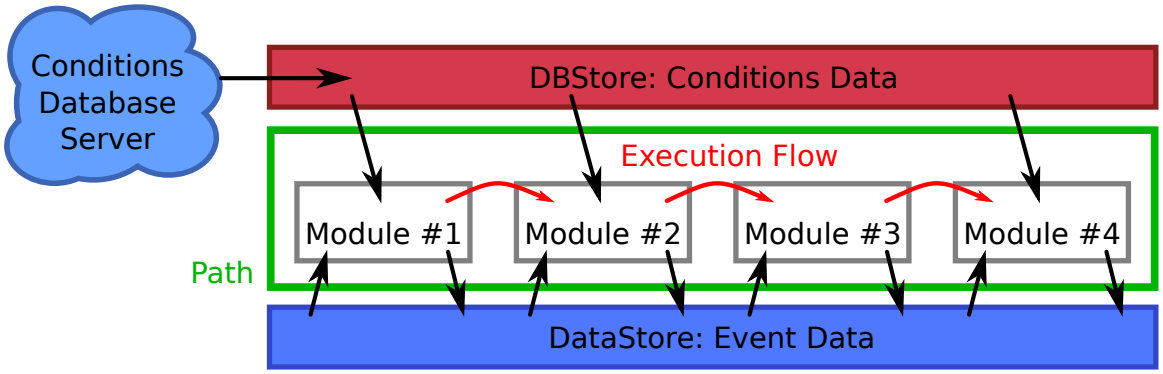


Figure 2.12: Example path (green) with four modules that are executed in order. Each module can request data from the conditions database (DBStore) used for calibration or general settings. In addition, it can retrieve data from the DataStore, and write processed or new data to it.

are amplified and shaped by electronic circuits. For both ionisation and electron-hole-pair creation the average energy loss per length is described by the BBF. However, both processes are random processes, and the freed electrons can have a relatively high energy. In this case they are called δ *electrons* or δ *rays*, and they can themselves travel significant distances in the detector, creating secondary ionisation. Due to this, the energy loss cannot be described by a Gaussian distribution, but it is described by a *Landau distribution*. These δ rays can be a significant background source in a detector.

A second desired process is the Cherenkov effect. If a charged particle travels through a material faster than the speed of light in that material, light is emitted on a conical surface. The opening angle of the cone depends on the $\beta\gamma$ value of the particle, and the emitted light usually is of high frequency, and thus has the highest visibility in blue part of the visible spectrum. Third, charged particles and photons traversing a material can excite the atoms and molecules in that material and is called *scintillation*. When transitioning back to their ground state, they emit light, often in the ultraviolet or visible spectrum. The individual photons can be collected by photomultipliers, which create a measurable signal e.g. via an avalanche effect. Counting all photons yields a measurement of the total amount of energy deposited in the scintillator material.

This of course is only a very brief overview of the interaction of particles with matter, and only the effects relevant in Belle II are mentioned.

2.8 The Belle II Analysis Software Framework

Belle II relies on fast and well tested software to operate the detector and analyse the data online and offline. The software used with and developed for the Belle II experiment is basf2 [40, 46]. It is mostly written in C++ [47] and Python [48]. The steering of basf2 is done in Python, too, and the interface between C++ and Python is implemented using the boost library [49]. basf2 is both used on the HLT to analyse the data taken online and to provide trigger information, as well as for offline analysis. In addition, it is used to simulate MC events for different processes, providing a full simulation of the complete Belle II detector. A schematic overview of the functionality of basf2 is shown in Figure 2.12. During simulation the same *DataObjects* are created that are also obtained

from the reconstruction of raw data. These DataObjects are the same that are reconstructed from raw data in the HLT or during offline analysis. For exchange of data between the single building blocks of the software, called *modules*, the DataObjects are stored in the *DataStore*. Each modules executes one task at hand and is optimised for this specific task, and can fetch data from and write data to the DataStore. Different modules are collected in a *path* which is executed in the order the modules are added to it. This modular approach makes it easy to develop and test new features within an existing module, or to develop a new module that replaces an existing one in the path. The higher level information reconstructed from raw data or simulated hits are then used for instance to find the trajectories of charged particles in the detector, or to measure the energy deposited in the ECL, which themselves are saved in the DataStore.

2.8.1 Monte Carlo simulation

basf2 uses various external software libraries. The foundation of basf2 is ROOT [50], which is the de-facto standard in HEP. Is used to store data, but also offers functionalities to plot and analyse data with build-in tools for HEP.

Usually an analysis starts by an idea, which is first tested on MC data. They need to resemble the actual data in all physical processes, from the generation of $\Upsilon(4S)$ events, over particle decays, to the interaction of the particles with matter. For the creation of $\Upsilon(4S)$ MC data the EvtGen [51] library is used, while various other packages are used for other processes like Pythia [52] for hadronisation. More information on the MC generators used in basf2 can be found in e.g. [53].

The detector geometry and interaction of particles with matter is taken care of by Geant4 [54]. Each particle is traversed through the detector, where their interactions are modeled, including radiation of photons. All information on the initial state of the particles from MC simulation, including their production vertex and their ancestors in case of decay products, are stored in the MCParticles DataStore.

The simulated detector signals are digitised and handled in the same way as real data, such that they can undergo the full reconstruction. In addition, they are compared to the real signals of the detectors and the simulation is tuned such that the simulated signals resemble the real once as accurately as possible. In an additional step, outside of basf2, the beam particles inside SuperKEKB are simulated. If this simulation results in particles leaving the beam pipe due to the background processes described in Section 2.6, they are transferred to Geant4 which then simulates the response of the Belle II detector. These background event information can be overlaid with the actual physics simulation to obtain more realistic MC data, which in the ideal case cannot be distinguished from real data. This, of course, is difficult to achieve, and the simulation of the detector and the backgrounds is steadily optimised. At a later step, recorded background data can be used to replace the simulated backgrounds.

Further software shipped with basf2 includes e.g. Jupyter Notebooks [55] which are mainly used for analysis [56].

CHAPTER 3

From detector measurements to particle trajectories

In this chapter the most concepts important for track finding in Belle II are introduced. First, a general overview of the purpose of track reconstruction is provided in Section 3.1, followed by the combination of track finding methods in Belle II in Section 3.2. Afterwards, the Hough Transformation (HT) is introduced in Section 3.3, since it is the main mathematical concept used for the developments in this thesis, followed by descriptions of the track finding algorithms currently employed in Belle II in Sections 3.4 to 3.6 and finally methods for extracting the track parameters are presented in Section 3.7.

3.1 Purpose of track reconstruction

As introduced in Chapter 2 the tracking sub-detectors provide spatial information that enable the reconstruction of the trajectories of charged particles traversing the detectors. To convert the single measurements into tracks, dedicated algorithms have been developed to identify and form the tracks from the measured hits, called *track finding*, or, in short, just *tracking*, and retrieve precise measurements of the initial kinematic track parameters, called *track fitting*.

The objective of the tracking algorithms is the identification of all tracks in an event, without constructing additional tracks from random combinations of hits or the reconstruction of actual background tracks. However, achieving both is hardly possible. First, the tracking detectors do not cover the full solid angle but only 91.1 %. In addition there are insensitive regions in the SVD and PXD, and over time readout electronics can fail or deteriorate, creating inefficient regions. Second, the track finding algorithms themselves can be inefficient, for instance when facing too many background hits in a specific region, or if a particle is deflected significantly during MS events, rendering it impossible to find the trajectory.

Some of the quantities to evaluate the track finding performance are the track finding efficiency, the hit efficiency, and the hit purity. They describe the efficiency of finding all tracks in an event, and to some extent the quality of these tracks based on how many correct hits are found for each track, and how many wrong hits are contained in a track, respectively. While the importance of a high track finding efficiency is self explanatory, high hit efficiency and purity influence the quality of the final track fit, as missing or wrong extra hits deteriorate the track fit result, potentially increasing the errors on the track parameters or yielding wrong track parameters.

Many analyses in Belle II make use of the clean environment and the precisely known initial state conditions, allowing for the measurement of the total invariant mass or the invariant mass of each of the two created B mesons. In case of inefficient track finding, final state tracks are missing, subsequently reducing the number of events available for a physics analysis, introducing a systematic error. In addition, the estimation of important analysis variables would be wrong, resulting in events to be discarded. Additional tracks on the other side can spoil an event by adding momentum, energy, and charge to an event, rendering it unusable for analysis, too. Finally, the track parameters need to be known precisely as the tracks are extrapolated to the IP and to the detectors surrounding the tracking volume. High precision PID in TOP and ARICH is only possible if the entry point of the charged tracks is known with high precision. Similarly the association of ECL clusters with tracks requires accurate predictions of the location a track enters the ECL. In case of the extrapolation to the IP the correct PXD hits need to be attached to the track first before the tracks are fitted again. Close to the IP the particles associated with the tracks are combined to heavier particles based on their properties, including mass hypotheses from PID. In order to combine two particles into a heavier particle their trajectories need to get close enough to each other in a single point to be combined, which is impossible if the track parameters of only one of them are wrong. Reconstructing a full decay chain requires both precise and accurate measurements of all decay vertices. A prominent example is the measurement of CP violation with Belle II. If the measurement errors of the decay vertices of the two B mesons are too large, it is not possible to tell the two vertices apart, making it impossible to measure the difference in lifetime and thus the CP violation in the process.

3.2 Track finding in Belle II

In HEP there are two distinct methods to find tracks: global and local methods. Global methods aim to use as many information as possible from all detector layers to define a track, while local methods try to connect information from neighbouring hits to find a tracklet. Each tracklet can then be combined with other tracklets from local methods into a global track. A tracklet is a set of spatially confined measurements like from a CDC SL (or from different SVD layers) that coarsely defines the trajectory of a particle. This section introduces both the global and local track finding algorithms used in the default Belle II tracking algorithms and in DATCON as well as the new tracking algorithm developed in this thesis.

For reference, the current tracking chain in Belle II is introduced first before breaking it up into its parts of CDC and SVD tracking and into the global and local parts. A schematic overview of the Belle II tracking chain is presented in Figure 3.1. Currently tracking starts with both a global and a local tracking approach in the CDC, namely the *CDC Legendre Track Finder* (global) and the *CDC Segment Automaton* (local). Information obtained from these algorithms are combined into CDC tracks that build the base for the CDC-to-SVD Combinatorial Kalman Filter (CKF). The CKF relies on a track seed and builds the track by locally checking additional hits, more details on the CKF are provided in Section 3.6. It attaches SVD hits to tracks that were already found in the CDC, extending them to the inner tracking and vertexing volume. All remaining SVD hits are used by the VXDTF2 to find tracks in the SVD alone, which afterwards are merged with CDC tracks that do not have any SVD hits attached to them by the CDC-to-SVD CKF. Afterwards the remaining SVD tracks that are not merged with an existing CDC track are extrapolated into the CDC employing the SVD-to-CDC CKF. All tracks found so far are combined into a single StoreArray and extrapolated towards the PXD by a

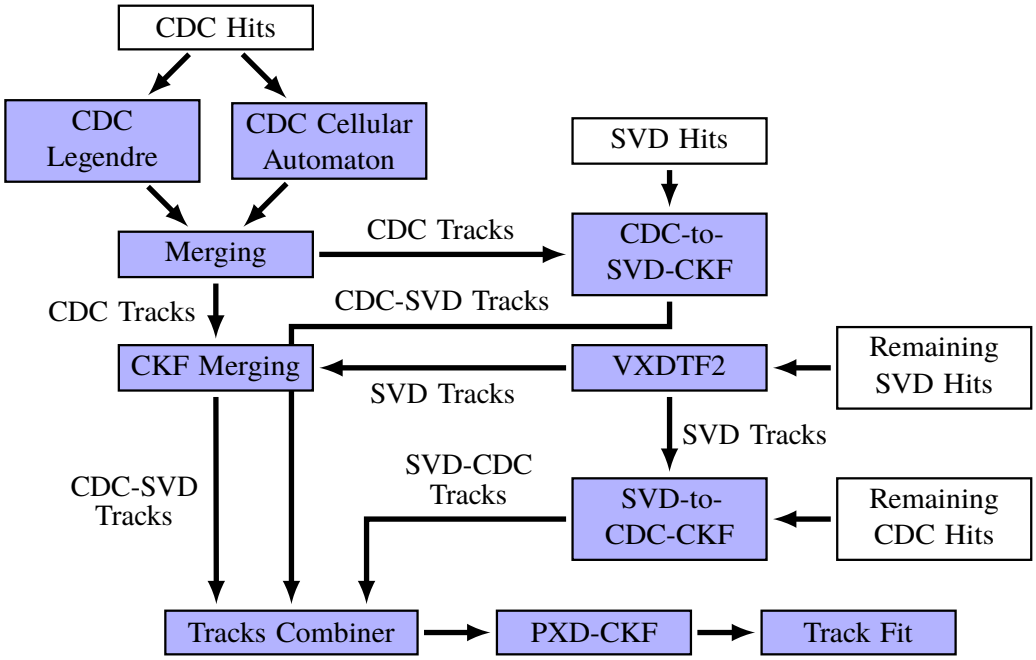


Figure 3.1: Track reconstruction flow in Belle II. A detailed guide through this scheme is given in the main text.

last CKF, after which a final track fit is performed.

3.3 Track finding using the Hough Transformation

Although track finding utilising the HT so far is not part of the default track finding chain in Belle II, it is described first, as it is the method employed in both DATCON and the newly developed SVD track finding described in Chapter 5. In Belle II the HT is so far used in the CDC trigger, and a regular CDC track finding algorithm based on it exists, too. However, it is not used by any of the default track finding algorithms. First, the general concept of the HT is introduced alongside some necessary mathematical concepts. Afterwards the usage of the HT with data of the Belle II SVD and the important concepts for this thesis are introduced.

3.3.1 Introduction of the Hough Transformation

The HT was first introduced and patented by P. V. C. Hough [57] in 1962 as a method of finding patterns (straight lines) in the images of bubble chambers [58]. In the initial HT a line is parametrised by its slope m and intercept b as

$$y(x) = m \cdot x + b \tag{3.1}$$

and the HT inverts this to

$$b(m) = -x \cdot m + y. \tag{3.2}$$

Using the HT, points in the image space are converted to lines in the parameter space or Hough Space (HS), and vice versa. This benefits track finding, as the task of finding a line in the image is

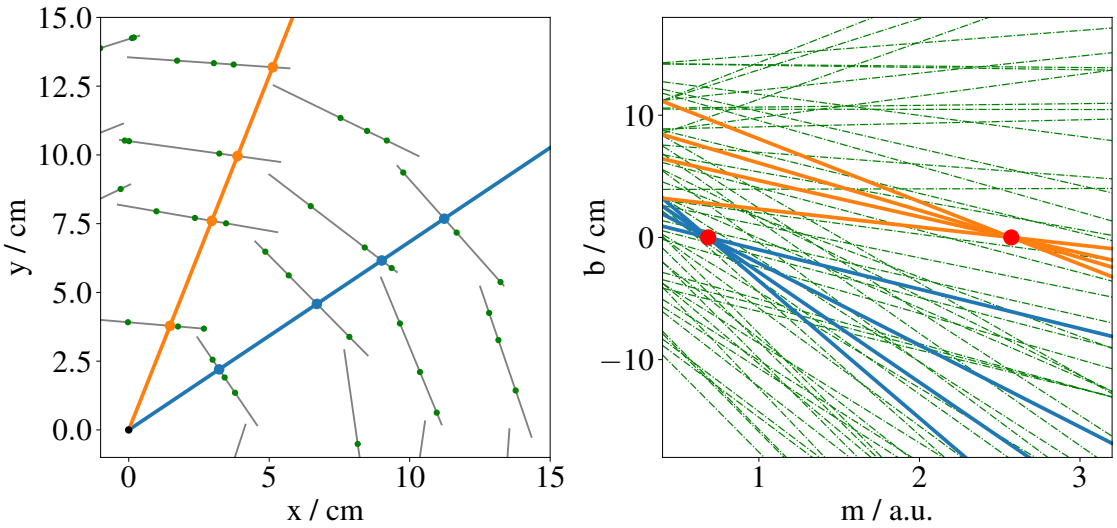


Figure 3.2: Example of the HT with hits on the SVD (example data). In the left half with orange and blue points mark track hits and green points mark background hits; two example tracks are indicated by the blue and orange lines. Applying the HT described in Equation (3.2), variables and parameters are exchanged. Background hits are transformed to the green dash-dotted lines in the right half, while the hits from the two example tracks are transformed to the blue and orange lines. For each of the two tracks (blue and orange) the lines corresponding to the hits intersect in one point, highlighted by the red dots.

reduced to finding a point of intersecting lines in the HS, which is illustrated in Figure 3.2. In the left half random hits are marked as green dots on the SVD (black lines), and two straight tracks are indicated by the blue and orange lines and points (as hits on the SVD). The Hough transformed lines are shown in the right half, with each line corresponding to one of the hits in the left part, and the blue and orange lines intersecting in one point each (highlighted by the red dots), marking the slope and intercept of both tracks.

However, with this parametrisation a slope m close to infinity requires an infinitely large parameter space. Utilising the Hesse normal form of a straight line [59], it can be described by

$$\rho(\theta) = x \cdot \cos\theta + y \cdot \sin\theta \tag{3.3}$$

in two dimensions with x, y as points on the line ρ as the distance of the line to the origin and θ as the angle between the x-axis and the connection line, as depicted in Figure 3.3. This limits the parameter space to $\theta \in [-\pi/2, \pi/2]$ with ρ potentially being infinite, which can be avoided by a proper choice of the coordinate system.[1] Using the Hesse form and applying the HT to the example points in Figure 3.2 results in Figure 3.4. Again, the background hits are indicated by the green dash-dotted lines, and the hits of the two straight tracks are sketched as blue and orange sinusoidals. For each of the two tracks the corresponding lines intersect in one point which indicates the angle θ and the distance ρ from the origin of the two straight tracks, marked by red dots. Since θ is the angle between the normal to the line and the x-axis, but not between the line itself and the x-axis, it is 90° off of the actual angles,

[1] The angle θ represents a generic angle in this theoretical description and is not linked to the track polar angle which also is denoted as λ.

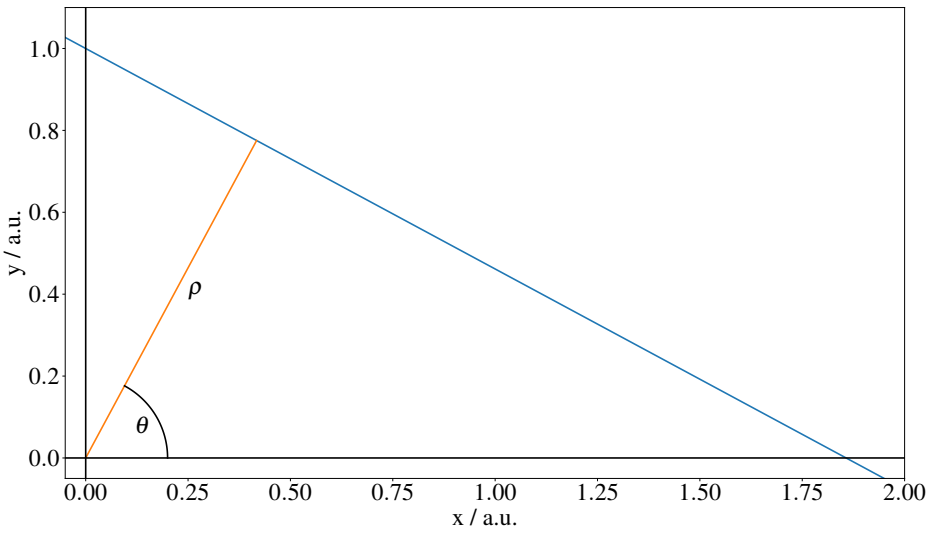


Figure 3.3: Hesse form of a straight line. Instead of representing the line as $y(x)$ as in Equation (3.1) it is represented as $\rho(\theta)$ as in Equation (3.3).

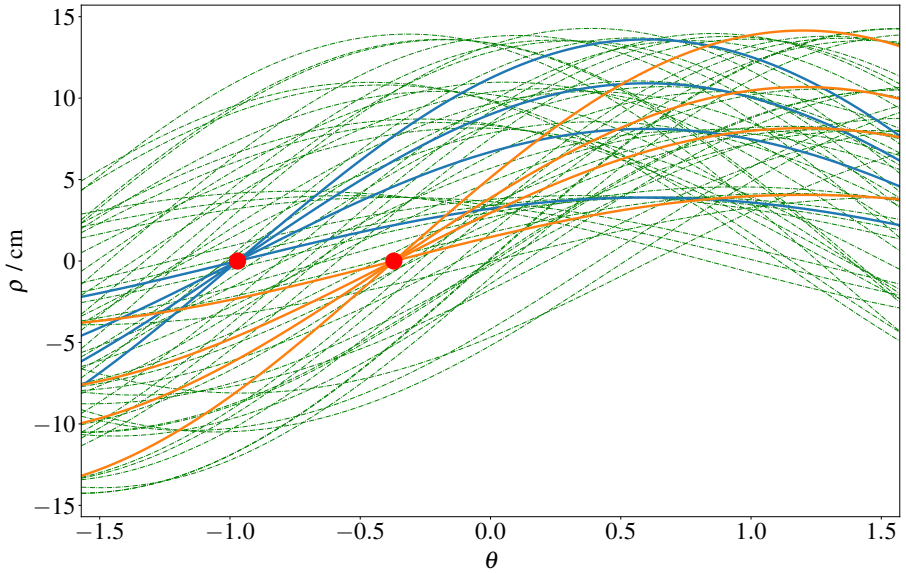


Figure 3.4: Representation of the Hesse form of the SVD hits in Figure 3.2. All green background hits are now represented by green dash-dotted lines, while the hits of the two tracks are shown as solid blue and orange lines. While for the background hits no intersection of four lines in one point is found, there are two distinct intersections for the blue and orange curves each, highlighted with red dots. These mark the parameters θ and ρ of the two tracks.

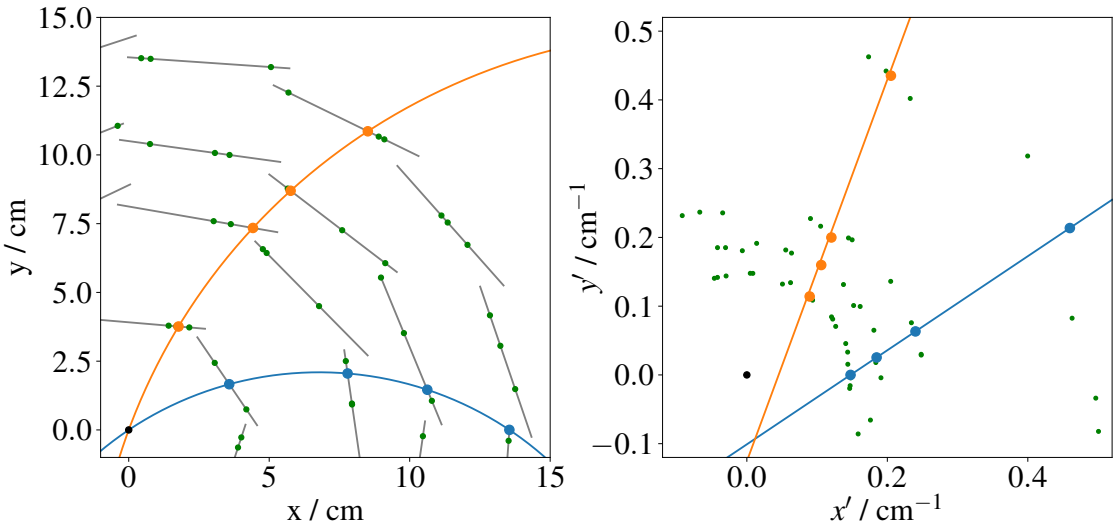


Figure 3.5: The conformal transformation using Equation (3.4) converts the two bent tracks in the left half into a straight line each in the right half. All green points marking background hits are transformed in the same way. Since the transformation involves a division by the squared radius of the point, the unit is an inverted length, and the values are smaller.

which has to be accounted for during track reconstruction.

So far the HT is applied to straight lines or straight tracks. However, the tracks in Belle II are bent by a magnetic field to measure momentum of the particles, approximating a circle through the origin in the x-y-plane in case they are created at the IP and neglecting energy loss and MS in the detector. Although a HT for circles exists, which aims for the estimation of the centre coordinates of the circle and its radius, a different method is employed in this thesis.

A certain set of angle-preserving transformations, called *conformal transformations*, is used to preserve the track's angles at the IP, while converting the circular lines into straight ones. An example for this is given in Figure 3.5. Both the orange and the blue track in the left half are converted into straight lines in the right half. In this case the conformal transformation is performed via

$$x', y' = \frac{x, y}{(x - x_c)^2 + (y - y_c)^2} \tag{3.4}$$

where x, y are the coordinates of the measurements, x_c, y_c are the coordinates of a point on the circle, for which usually the IP at the origin is is taken, and x', y' are the conformal mapped values of x and y. Now Equation (3.3) can be written as

$$\rho(\theta) = x' \cdot \cos\theta + y' \cdot \sin\theta \tag{3.5}$$

$$= \frac{2x}{x^2 + y^2} \cdot \cos\theta + \frac{2y}{x^2 + y^2} \cdot \sin\theta \tag{3.6}$$

with an additional factor of 2 allowing the normal distance ρ to represent the track's signed curvature $\rho = q/R$. Here q is the track's charge in multiples of unit charges, and R is its radius. Thus, when identifying the intersection of the conformal and Hough transformed hits in the $\rho - \theta$ space, the track's

curvature and its initial direction are known. With both values the transverse momentum p_T and the azimuthal angle φ are calculated as

$$\varphi = \theta - \frac{\pi}{2} \tag{3.7}$$

$$p_T/\text{GeV/c} = 0.3 \cdot B/\text{T} \cdot \rho^{-1}/\text{m}, \tag{3.8}$$

ensuring that φ remains within $[-\pi/2, \pi/2]$ by adding or subtracting π if necessary.

3.3.2 The Hough Transformation for track finding in the SVD

Since the HT is used for both DATCON and a newly developed SVD track finding, the concepts outlaid in this section are general and not tied to either implementation. After establishing the general concept of the HT and the mathematical foundations for finding ρ and φ, both values need to be extracted from the HS by finding the intersection point of sinusoidal curves corresponding to a track. Analytical solution of this task are unfeasible, as in case of track finding in the SVD potentially hundreds or thousands of hits and thus lines are present, and each line needs to be tested with all others for intersections. Another approach is to divide the HS into small sections and check for each sinusoidal line whether or not it passes through such a section. If so, a counter for each section a line passes through is increased by one, and finding tracks is reduced to finding local maxima in the HS. However, this approach is potentially very computing intensive. With just 128 sections in both horizontal and vertical directions, and 1 000 hits and thus lines in the HS, 16.384 million checks of sectors need to be performed. Instead insights about the SVD and the problem to be solved are exploited. Since (random) combinations of two intersecting lines are common, especially for lines originating from background hits, at least three lines must intersect in a given point. In addition, the SVD consists of four layers, and in general three two dimensional measurements are necessary to constrain a track as each track has five degrees of freedom, more details are provided in Section 3.7.1. Thus the requirement is that sinusoidal lines created by hits on at least three different SVD layers intersect for a valid track.

To reduce the computational effort, the HS is divided into two equal parts along both axes, resulting in four parts in total. Each part is then checked for sinusoidal lines stemming from hits on at least three different SVD layers passing it. Only if lines from at least three different SVD layers cross a section it is further processed by again dividing it in two equal halves in horizontal and vertical direction and repeating the check for each of these. The process is depicted in Figure 3.6 with the intersection of the orange lines as an example, and all other lines being ignored when further dividing a section. A section is not further divided in four equal parts if less than three lines from different SVD layers pass it, or if a maximum number if divisions is reached. In this case the section is considered a track candidate and marked for further processing. This approach is called *fast HT* and is equivalent to finding local maxima when checking all sections, as in each iteration only sections that fulfill the requirement of being passed by lines originating from three different layers are further processed. It is used widely in HEP in various experiments.

One major drawback of this method is that random combinations can occur when a section is passed by lines with both positive and negative slope, e. g. three lines from different layers, two of which with positive and one with negative slope. These lines cannot stem from hits from same track, not even low p_T curling tracks. To avoid random combinations of lines, the HS is extended from $\theta \in [-\pi/2, \pi/2]$

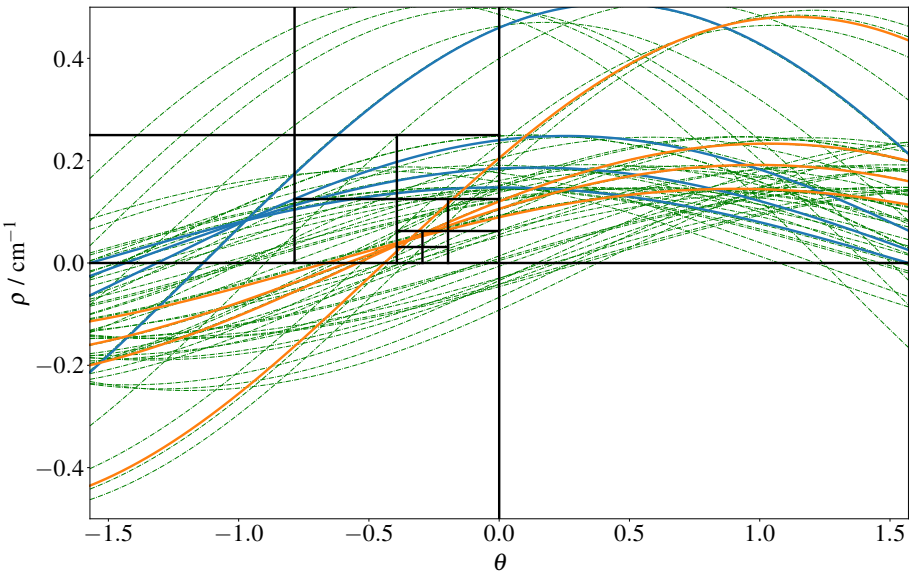


Figure 3.6: Educational example for the fast HT used with SVD data. In this case, the intersection of the orange lines is probed, the blue lines of the second track and the green lines from background hits are ignored when checking for the next division. With even more steps, the intersection can be found with better precision, at the cost of execution time.

to $\theta \in [-\pi, \pi]$, but only the part of each line with positive slope is considered when checking the passage of a section. A positive slope corresponds to the outgoing *arm* of a track, i. e. the part of a track traversing away from the IP. The term *arm* in this context refers to at maximum half a revolution in the detector. If a low p_T particle first travels outwards from the IP and then turns back towards the IP without leaving the tracking volume, it is considered to consist of two such arms. In case both of these arms of a track are found independently, they are assigned with opposite charges. While the approach described here avoids random combinations of lines with rising and falling slope to build a track, it poses the risk of finding tracks twice but with opposite charge. This may happen if the actual first outgoing arm and the first ingoing arm are both found independently, but they are not recognised as the same track, potentially creating a clone track. These need to be identified and either merged or removed from the track set in later steps during tracking.

3.4 Track finding in the SVD

While the HT is employed for a new SVD track finding algorithm developed in this thesis, Belle II already records data using the VXDTF2 for track finding in the SVD. Its concept of “Sectors on Sensors” was proposed by R. Frühwirth et al. in 2013 [60] and implemented into basf2 in its first version as VXDTF [61] and improved and refactored later to the VXDTF2. A second algorithm that attaches SVD hits to existing CDC tracks is the CDC-to-SVD CKF which was developed in [36] and is briefly described in Section 3.6. Because the VXDTF2 works with SVD hits only, an additional algorithm is necessary to combine and merge tracks found in SVD and CDC separately. Before the CKF was added to basf2, a simple cut-based method was used for this merging. Now a CKF dedicated

to this task is used.system

3.4.1 VXDTF2

During track finding each SVD sensor is divided into several parts called *sectors* in the tracking software in a first step. Only combinations of hits in sectors that share a friendship relation are considered for further analysis. These friendship relations are learned in a training process based on millions of MC events using the full simulation of the Belle II detector and are stored in the *SectorMap*, which based on a concept described in [60]. Additional filters are applied to combinations of two and three hits:

Two hit filters If two hits are in sectors sharing a relation, the hits are tested using geometrical properties like their 2D and 3D distance.

Three hit filters All two-hit combination sharing one hit are tested for compatibility to form a triplet. Again, geometrical properties are used in this step like the difference in φ and λ from the first to the second to the third hit.

As the SectorMap, the filters are trained on a large number of MC events. The decision of each filter not only depends on the actual values of the geometrical properties, but also of the location of the hits in the detector to apply different cuts in e.g. the forward and backward regions of the detector. Although it is possible to also create four hit combinations and apply a filter on them, three SVD hits are enough to define a track.[2] Thus, no four or even five hit filters is used.

Using a *cellular automaton*, all triplets are probed to combine them and build paths and later track candidates, starting at the outermost SVD layer. In a general cellular automaton, an action is taken for each *cell* based on its state. Here, a neighbour cell is another hit in the same triplet, or another triplet which contains shared hits. Usually longer paths are favoured, but it is possible to create a track candidate out of a path of only three hits that is a subset of a longer path. This allows for the creation of track candidates with a hit missing e.g. due to ineffective detector regions or MS or damaged readout electronics.

So far many track candidates are allowed to share several hits, and the number of track candidates can be significantly larger than the actual number of tracks in an event, since also track candidates consisting of a subset of hits in a path are accepted. Out of these track candidates the best ones need to be found. Two methods are used for this task. First, the track candidates are fitted with a simple track model, the different fits available for this task are introduced in Section 3.7.6, or evaluated with a multivariate method, yielding a quality indicator. Only the candidates with the highest quality indicator are tested for overlapping hits and the paths containing the most hits. The final set of tracks is build from the track candidates with the highest quality and the longest paths.

In principle the VXDTF2 can also include PXD hits, as it "learns" the detector geometry via the SectorMap. However, the VXDTF2 is only used with SVD hits because of the high occupancy in the PXD. Even after ROI selection and data reduction in the PXD, the number of hits is high and the hits inside the ROIs are close enough for the VXDTF2 to attach wrong hits to tracks too often to be acceptable. Thus, although technically possible, a different method of adding PXD information to tracks, which is found in the CKF explained in Section 3.6.

[2] Three hits on the SVD provide six degrees of freedom, while five degrees of freedom are enough to fully constrain a track. More details on this are provided below in Section 3.7.1.

3.5 Track finding in the CDC

As mentioned in Section 2.4, the CDC is the main tracking detector in the Belle II experiment, providing valuable information for the measurement of charged particle momenta. At the time of writing this thesis, the CDC track finding is the backbone of tracking in Belle II, as it is the first part in the tracking chain, as introduced in Section 3.2. Both a global and a local track finding method are used for track finding in the CDC to ensure that the tracks contain information from both axial and stereo SLs, as well as to find tracks of long lived particles that travel a long distance in the detector before decaying.

3.5.1 CDC Legendre algorithm

Similar to the HT in the SVD, a global track finding algorithm can be used in the CDC. As each hit is not as locally constrained as on the SVD, but carries some ambiguity due to the charges drifting towards the single wires, the algorithm employed is slightly different. While still neglecting interactions with the detector material like energy loss or MS in the gas or at the wires, and assuming the particles are on a perfect circular trajectory originating in the IP, Equation (3.6) can be rewritten as

$$\rho_{\pm}(\theta) = \frac{2 \cdot (x \cdot \cos\theta + y \cdot \sin\theta \pm d)}{x^2 + y^2 - d^2} \tag{3.9}$$

with x, y as the coordinate of the CDC wire that registered a hit, and d being the drift distance at each wire. Equation (3.9) describes the Legendre transformation, which was proposed for track finding in [62], and its implementation for Belle II is described in [63]. Due to the drift distance there is an ambiguity in the position estimation of the actual hit that needs to be considered, as the track can pass the wire at any point with distance d from it. Instead of one line in the HS, now there are two lines in the Legendre space, one for each of sign of d. Finding the intersection of lines in the Legendre space works the same as in the HS using the method of bisecting the Legendre space.

However, the Legendre transformation fails to find tracks that deviate too much from a circle in the x-y-plane, or that do not originate from the IP. Additionally, it only works with hits of the axial SLs of the CDC, but not with hits of the stereo SLs as their x-y-position changes along the z-axis. Either a minimisation is needed to include the stereo SL hits, or a different method combining already found axial tracks with stereo hit information. Further information on the CDC track finding can be found in in [64–66].

3.5.2 Cellular automaton in the CDC

As a local track finder for the CDC, a cellular automaton is employed. This concept was introduced in [67] and has previously been used in other HEP experiments [68, 69]. In a first step clusters of CDC hits within each SL are created. Weighted relations are build between neighbouring hits, allowing for gaps in the layers for instance in case of broken readout electronics or dead wires. Using these relations, *triplets* of hits are build, including an estimate of the track passing through each triplet, yielding a *facet*. Triplets are connected to each other step-by-step in case the next possible triplet to be attached shares exactly two hits with the previous triplet, and if selection criteria are passed, like geometrical cuts or a least squares fit with the full hit set. This process is repeated until no further hit in the same SL can be added. A cluster is the largest group of neighbouring hits in each SL.

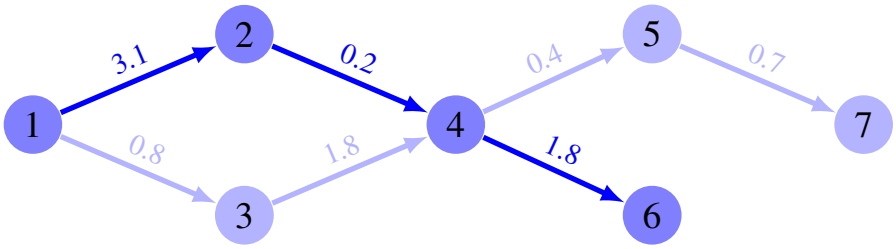


Figure 3.7: Example for a directed acyclic graph as used in track finding. The blue circles mark the nodes of the graph, and the arrows connecting them are the directed edges with weights. While the path 1–2/3–4–5–7 is the longest path, with 1–2–4–5–7 having the larger weight of both, the shorter path 1–2–4–6 has the highest weight in total. It therefore is highlighted as the result of the first iteration of path finding. Adapted from [70].

Afterwards the best set of hits in each cluster, called *path*, needs to be found. Using the weighted relations, a directed acyclic graph is build, with each triplet representing a node, and the weights of the relations between two triplets are represented by the edges. An example for such a graph is depicted in Figure 3.7. Starting with a randomly chosen hit, the path is built by recursively following the largest weights. The final path is the one with the highest sum of weights, even if this means that no hits of a layer are contained although being in the cluster before.

Finally, the best paths from the different SLs can be used to create new tracks, or to add them to existing tracks found with the Legendre transformation algorithm. After this the best tracks are selected and fake tracks, falsely assigned hits and background hits are removed.

3.6 Track finding using the CKF

While track fitting concepts including the functional principle of the Kalman Filter (KF) are described in Section 3.7 in more detail, the CKF as an algorithm suited for simultaneously finding and fitting tracks is described in this section. The concept itself was first introduced by R. Frühwirth in 1987 [71]. Although already being employed in various other HEP experiments, the usage of a CKF in Belle II was assumed to not be possible for long time due to the very different sub-detectors, and especially with the VXD only having few layers. Nonetheless, a CKF has been developed for Belle II and its development is described in [36]. It is used at various stages in the Belle II tracking, as introduced in Figure 3.1, and at the time of writing the To-PXD CKF is the only way to attach PXD hits to existing tracks, as the PXD itself is not part of track finding. Besides that, it plays a crucial role of combining information from the different sub-detectors, namely SVD and CDC, into a combined set of tracks.

The general concept of the CKF in Belle II is shown in Figure 3.8. For simplicity, the CDC-to-SVD CKF is used as example here. First, tracks found in the CDC serve as *track seeds* and are extrapolated to the SVD. For each track a pre-selection of hits to possibly attach is conducted in *hit filter 1*. Only for SVD hits passing this first selection, the costly extrapolation to the next layer is performed.[3] With information of the extrapolated position and each hit on the next layer, a second selection is performed in *hit filter 2*, but the hit on the next layer is not yet used as part of the track. All hits that pass this second selection are used in the *Kalman update* step, where the probed hit is used in the fit, followed

[3] In case of overlapping sensors, the next hit that is probed is not on another layer, but in the overlapping sensor closer to the IP. Thus the CKF is able to attach multiple hits on the same layer to a track in these regions.

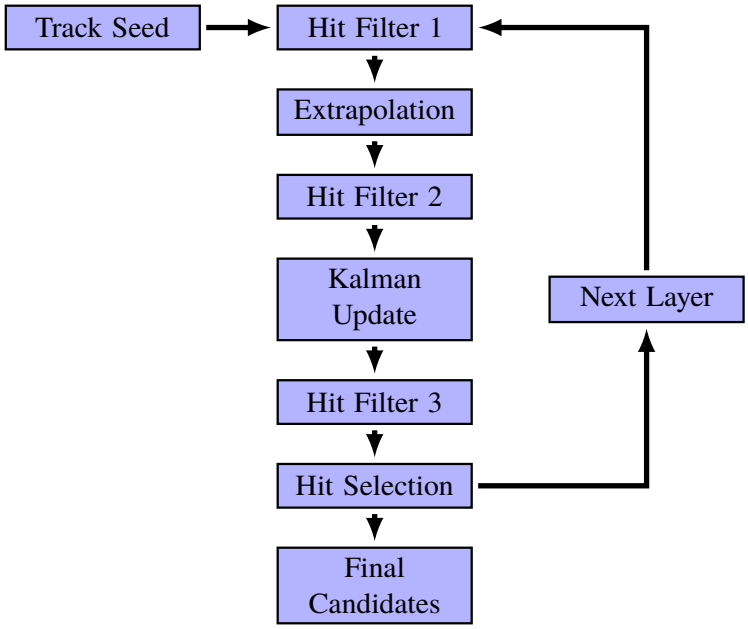


Figure 3.8: General concept of the CKF as employed in Belle II. Building up on a track seed, usually a measurement in one of the sub-detectors, hits in another tracking detector are probed one by one. Between each of the main KF steps (extrapolation and update), a filter is used to remove possible candidate based on all information available at the current state. Once a hit is selected, the process starts over from the first hit filter, until no more hits can be added. Last, a final track set is created and usually re-fitted before proceeding with track finding.

by a third filter, *hit filter 3*, and the selection of the n best hits.

If there are more layers closer to the IP, the procedure is repeated by creating intermediate tracks adding the aforementioned best n hits to the current track, and probing the hits on the next layer. Once the innermost layer is reached, or no more hits can be attached, the final track candidates are created. In addition, it is possible to revert the order of the CKF and traverse the detectors inside-out. For both the CKF towards the PXD and the SVD the three hit filters use Multi Variate Analysis (MVA) based methods to decide which hits to reject, utilising all information available at each step.[4]

For the SVD-to-CDC CKF the situation is slightly different. Here the three filters are included in the extrapolation, update, and hit selection stages, but are not separate entities. But the overall scheme is the similar to the CKFs to attach SVD and PXD hits. In contrast to the SVD and PXD CKF, hit rejection in the CDC is not based on MVA methods, but on simple selections on geometric properties or track extrapolation. And since the SVD-to-CDC CKF uses track seeds found in the SVD it starts at the innermost CDC layer and probes further hits inside-out.

Furthermore there are ideas to use a CKF based on ECL hit information, especially to find low p_T tracks, electrons in particular. However, this approach is not yet fully implemented and some issues have been identified by the author, but they have not been further investigated. This CKF is build on the idea that the energy deposited in the ECL and the position of the ECL cluster can be used for an initial guess of the particles transverse momentum p_T. Based on this guess the track seed is created

[4] The set of variables is the same in each hit filter, but information on χ^2 of the track fit or the residuals between the extrapolated position and the local hit coordinates are not the same or not available, depending on the filter stage.

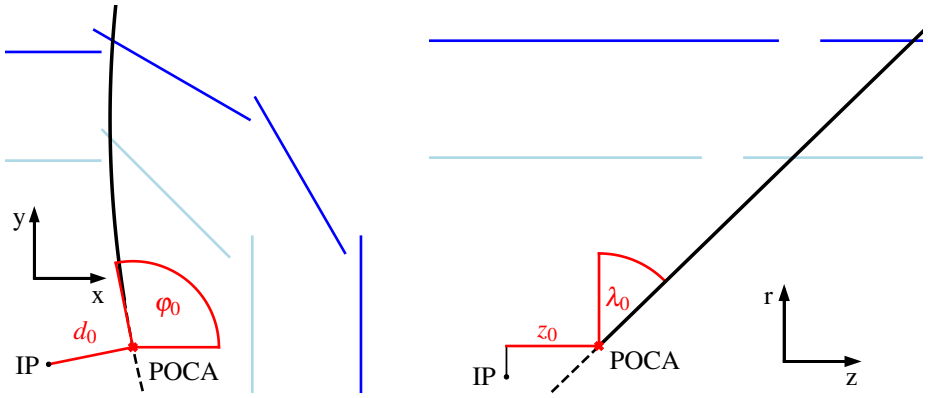


Figure 3.9: Definition of the track parameters in Belle II. All five track parameters are defined at the POCA. The left half shows the definitions of d_0 and φ_0 in the x-y-plane, and the right half shows the definitions of z_0 and λ in reference to the IP. ω_0 as fifth track parameter is the track curvature, which is defined as the inverse of the track radius $1/R$ in the x-y-plane. In general, a particle does not need to actually pass the POCA

and extrapolated towards the CDC by checking for CDC hits in close spatial proximity to the ECL cluster in the x-y-plane. However, there are some drawbacks of this approach. First, assuming the transverse momentum based on the cluster energy and the the radial position of the cluster can lead to bad track seed for low p_T tracks curling inwards after the maximum distance from the z-axis has been reached. Second, although the energy measurements in the ECL are very precise, a high energy muon can deposit a significant amount of energy, similar to what an electron with low p_T would deposit. Thus, the ECL-to-CDC CKF is still in development at the time of writing and not actively used for data taking or offline reconstruction.

3.7 Track parameters and fitting

Track fitting is the second essential part of track reconstruction. During the track fit, the track parameters need to be calculated as precisely and accurately as possible, with the smallest possible errors. Before introducing the most important track fitting algorithms used in basf2 and for this thesis, the track parameters that need to be calculated are introduced.

3.7.1 Track parameters

In Belle II a magnetic field is used to bend the tracks of charged particles in order to estimate their momentum. Without any material in the detector the trajectory of a charged particle in a magnetic field can be described by a helix. With material, the trajectory will deviate from a helix because of energy loss, MS, and further interactions, but a helix is still a good approximation. However, material effects need to be incorporated in the track fits to get the best possible estimate of the track parameters. A helix can be fully described by a starting point, the direction at the starting point, and the curvature, yielding five parameters in total – d_0, φ_0, ω, λ_0, z_0 –, which are depicted in Figure 3.9 (except for ω_0) and described in the following. In the left part the x-y-projection of the track at the Point of Closest Approach (POCA), while the right part shows the s-z-projection.

The POCA is defined as the closest distance of the track to the z-axis in the x-y-projection with the lowest possible value of z. This makes it well defined even for particles whose ancestors decay far off the IP, like K_S^0 or Λ^0 daughters, as the tracks are extrapolated back towards the z-axis to define their POCA. It is important to note that a particle does not need to pass through the POCA, as the POCA is defined for a general track and can be close to the IP even for secondary particles with a decay vertex far away from the IP. The definitions given here are used throughout the thesis.

$\boldsymbol{d_0}$ is the signed 2D distance of the POCA to the IP, or rather the z-axis, in the x-y-plane, and, by definition, the shortest distance. Its sign is defined by the sign of the angular momentum at this point.

$\boldsymbol{\varphi_0}$ is the azimuthal angle of the track, defined as the angle between the x-axis and a tangent to the track at the POCA. It ranges from $-\pi$ to π.

$\boldsymbol{\omega_0}$ is the signed curvature $1/R$ of the track in the x-y-plane with the track radius R. Its sign is the same as the sign of the charge of the particle.[5]

$\boldsymbol{z_0}$ is the z distance to the IP at the POCA.

$\boldsymbol{\lambda_0}$ is the *dip angle*. It is defined as the angle between the x-y-plane and the tangent of the helix at the POCA in the s-z-plane with s being the path length of the trajectory. It is defined in the range $-\pi/2$ (backward) to $\pi/2$ (forward).[6]

Since five track parameters need to be determined in the track fit, only tracks with at least five Number of Degrees of Freedom (NDF) can be fitted to fully constrain the parameters. The NDF of each individual measurement describes the amount of information that is provided by the measurement. While each pixel detector measurement provides two NDF, one for each direction on the sensor plane, SVD and CDC measurements only provide one NDF each, as each strip or wire yields a one dimensional measurement at the position of the track.

3.7.2 Track modelling

While the track parameters at the POCA – d_0, φ_0, ω, λ_0, z_0 – need to be found in the track fit, they are not directly used in the fit. Instead, the *state vector* $\mathbf{x}$ is used which is defined as

$$\mathbf{x} = (q/p, u', v', u, v)^T . \tag{3.10}$$

with q/p as the particle's charge divided by its momentum, $u' = \frac{\mathrm{d}u}{\mathrm{d}w}$, $v' = \frac{\mathrm{d}v}{\mathrm{d}w}$ describe the direction of the momentum vector at the measurement position as angles with the normal to the plane as reference, u, v as local measurements on a detector plane as introduced before, and $\mathbf{w} = \mathbf{u} \times \mathbf{v}$ pointing outwards from the IP in radial direction to define a right-handed coordinate system. In case of physical detector planes, like in the VXD, $\mathbf{x}$ is defined on the sensor planes. For other measurements, like in the CDC, the plane at the position of the measurement is constructed such that the track is perpendicular to the plane, i.e. $u' = v' = 0$. This definition significantly simplifies the calculation of covariance matrices.

[5] An alternative choice for the variable of the curvature is ρ, as introduced with the HT and the Legendre Transformation.

[6] Alternatively, $\theta_0 = \frac{\pi}{2} - \lambda_0$ is used often in tracking, or $\tan\lambda_0$ as the slope of the helix in the s-z-plane. θ_0 is defined between 0 and π.

Since the state vector is used for the track fit, the POCA parameters need to be converted into $\mathbf{x}_0$ first, or alternatively an initial guess of the state vector $\mathbf{x}_0$ at the IP is made. Then, the state vector is propagated to the measurements using the *system evolution* or *extrapolation* function f by

$$\mathbf{x} = f(\mathbf{x}_0), \tag{3.11}$$

or, for a specific evolution step

$$\mathbf{x}_i = f_{i-1}(\mathbf{x}_{i-1}) \tag{3.12}$$

or, in matrix notation

$$\mathbf{x}_i = \mathbf{F}_{i-1}\mathbf{x}_{i-1}. \tag{3.13}$$

For the sake of convenience, measurement errors, detector misalignment, and random noise are ignored at this point. More details including the aforementioned effects, and additionally non-linearities like MS and energy loss, can be found in e.g. [71, 72]. After expressing the state vector via the initial state and the evolution function, it has to be mapped into the measurement space, which for instance are the local coordinates on a detector plane. Thus, the five-vector $\mathbf{x}$ need to be reduced to a two-vector $\mathbf{m}^{\text{pred}}$ representing a prediction of the measurement. This is handled by the *measurement mapping function* denoted as h by

$$\mathbf{m}_i^{\text{pred}} = h_i(\mathbf{x}_i) = h_i(f_i(f_{i-1}(\cdots f_0(\mathbf{x}_0)))), \tag{3.14}$$

and in matrix notation as

$$\mathbf{m}_i^{\text{pred}} = \mathbf{H}_i\mathbf{x}_i = \mathbf{H}_i\mathbf{F}_i\mathbf{F}_{i-1}\cdots\mathbf{F}_0\mathbf{x}_0. \tag{3.15}$$

The function h_i / $\mathbf{H}_i$ is specific for each detector and measurement, and thus only needs to be applied for the current prediction i. Finally, the residuals r can be defined as the difference between the actual measurements m_i and the prediction:

$$\mathbf{r}_i = \mathbf{m}_i - \mathbf{m}_i^{\text{pred}}. \tag{3.16}$$

The task of track fitting now is to find the optimal estimate $\hat{\mathbf{x}}_0$ of the true initial state vector $\mathbf{x}_0$, from which the POCA track parameters are calculated.

3.7.3 χ^2 based track fitting

Very often an analytical model to describe data is known. In these cases the most common way to extract the free model parameters is by a so called χ^2 fit. It is based on the method of least squares, invented by German mathematician and natural scientist Carl Friedrich Gauss around the year 1800.

The base idea of this method is to minimise χ^2 which is defined as sum of squared residuals between the measured data and the values predicted by the model for the same independent variables by varying the model parameters

$$\chi^2 = (\mathbf{y} - f(\mathbf{x};\mathbf{a}))^T\ \mathbf{V}^{-1}\ (\mathbf{y} - f(\mathbf{x};\mathbf{a})) \tag{3.17}$$

with the measured (dependent) data $\mathbf{y} = (y_1, \ldots y_m)$, and $f(\mathbf{x};\mathbf{a})$ as the model for the data $\mathbf{y}$ with the independent data $\mathbf{x}$ and the k free parameters $\mathbf{a}$, e.g. $f(x;a) = a_0 + a_1x + a_2x^2$. Using the independent values $\mathbf{x}$ the parameters have to be optimised such that the model f best describes the data $\mathbf{y}$, which is achieved for the minimal value of χ^2. Last, $\mathbf{V}$ denotes the *covariance matrix* of the measurements.

Equation (3.17) can also be written in matrix form as

$$\chi^2 = (\mathbf{y} - \mathbf{F}(\mathbf{x})\mathbf{a})^T \, \mathbf{V}^{-1} \, (\mathbf{y} - \mathbf{F}(\mathbf{x})\mathbf{a}) \ . \tag{3.18}$$

Again, $\mathbf{y} = (y_1, \dots y_m)$ are the measured data points, but now $\mathbf{F}(\mathbf{x})$ is a matrix incorporating the functions modeling dependent data $\mathbf{y}$ from the independent data $\mathbf{x}$, which can be both a linear or a non-linear, and the parameters that must be optimised are in the k-vector $\mathbf{a}$.

To provide a purely educational example:

$$\mathbf{F}(\mathbf{x})\mathbf{a} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_0 + a_1 \cdot x_1 + a_2 \cdot x_1^2 \\ a_0 + a_1 \cdot x_2 + a_2 \cdot x_2^2 \\ a_0 + a_1 \cdot x_3 + a_2 \cdot x_3^2 \\ a_0 + a_1 \cdot x_4 + a_2 \cdot x_4^2 \end{pmatrix} \tag{3.19}$$

where the model matrix $\mathbf{F}(\mathbf{x})$ describes a second order polynomial with the independent data points (x_1, x_2, x_3, x_4), and $\mathbf{a}$ denotes the free parameters $\mathbf{a} = (a_0, a_1, a_2)^T$ that need to be found during the fit.

In the example in Equation (3.19) the model $\mathbf{F}(\mathbf{x})$ is purely analytical, in which case the minimising parameters can be calcluated directly as

$$\mathbf{a} = (\mathbf{F}^T \mathbf{F})^{-1} \mathbf{F}^T \mathbf{y}$$

in which case the solution is even unique. In case of an existing approximate solution for $\mathbf{a}$, a linearisation around this approximation can be made by evaluating the model at $\mathbf{a} + \delta\mathbf{a}$. If a linearisation is not possible, e.g. because the model is not differentiable, alternative methods like gradient descent need to be employed to minimise χ^2.

However, this is usually not the case for a track in a detector, even though a helix can be described analytically, allowing for only intermediate approximations of the optimal parameter vector $\mathbf{a}$. Due to inhomogeneities in the magnetic field, the particle trajectory will deviate from a helix, rendering a purely analytical solution of the equations of motion impossible. Additionally, MS and energy loss in the detector material cause a deviation from the ideal case of a helix, too.

Due to these non-linearities, no closed form for the solution of the equations of motion and they need to be solved numerically. Because of the potentially large deviations from the helix due to energy loss or MS, a linearisation will not provide a good approximation, too. In addition, the minimisation requires the inversion of the covariance matrix V, whose size increases linearly with the number of measurements, making it computationally expensive in case of correlations between single measurements. A typical track consists of 50 to 60 individual measurements, but as they are usually in two dimensions, the size of the covariance matrix is twice that of the number of measurements. For uncorrelated measurements however, it is just a diagonal matrix and the inversion is trivial by inverting each value in place. Last, the model $\mathbf{F}(\mathbf{x})$ needs to be evaluated for all measurements simultaneously, and in case the minimisation of χ^2 is performed iteratively, the evaluation needs to be performed multiple times until the process converges, resulting in an increased execution time.

While the χ^2 method is not the method of choice for the final track fit, it is still used for preliminary track fits to estimate the quality of the tracks fast, ignoring material effects. For instance it is used in both the SVD and CDC standalone track finding. The algorithms used by the SVD standalone algorithms are introduced in Section 3.7.6. The final fit uses all available information, and incorporates

material effects and noise, to get the best estimate of the track parameters. Since it needs to be more precise and uses more information, it also uses more computation time.

3.7.4 Kalman Filter track fitting

While a global χ^2 fit cannot be employed for the final track fit because it is difficult to describe local effects like MS, local approaches like the KF described in [73] are suitable for the task. It is a general fitting method used in a wide range of applications where data with noise and uncertainties need to be described by a known model, both inside and outside of physics. R. Frühwirth extended the concept of the KF to be used as a method for iterative and local track fits in [71]. The Kalman filter is the foundation of the CKF described in Section 3.6 which focuses on the application in the Belle II tracking.

Although the KF also aims to minimise the χ^2, it follows a different ansatz and does so by minimising the residuals of each measurement individually. As an iterative approach, only one measurement is evaluated at each time, followed by the next measurement, and so on. It uses the local track representation $\mathbf{x}$ described in Equation (3.10), and only uses the two local coordinates u and v. Because of this, the track propagation model $\mathbf{F}$ only needs to be evaluated once for each measurement, and the covariance matrix that needs to be inverted is of size 2×2, for which a direct analytical solution exists. To extract u and v from the state vector $\mathbf{x}$, the measurement extraction matrix takes the form of

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.20}$$

As a local method, each measurement k is evaluated with the information available at measurement $k-1$. Each iteration step $k-1 \to k$ comprises several steps, which are described in the following. With the local track representation, a good choice is to define the track state at the path distance s from the IP which for step k yields

$$\mathbf{x}(s_k) = \mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{3.21}$$

where, as before, $\mathbf{F}$ is the system evolution model and $\mathbf{w}_{k-1}$ is a random disturbance of the track between the points s_{k-1} and s_k, e.g. due to MS. This is called *extrapolation* step. The (predicted) measurement position m_k^{pred} is obtained by applying $\mathbf{H}$ to $\mathbf{x}$ as

$$m_k^{\text{pred}} = \mathbf{H}_k \mathbf{x}_k + \boldsymbol{\varepsilon}_k \tag{3.22}$$

with the measurement uncertainty $\boldsymbol{\varepsilon}_k$. All $\mathbf{w}_k$ and $\boldsymbol{\varepsilon}_k$ are assumed to be unbiased, independent, and of finite variance. Not only the measurements $\mathbf{m}_k$ have uncertainties represented by their covariance matrix $\mathbf{V}$, but also the state vector $\mathbf{x}_k$ has uncertainties which are given by its covariance matrix $\mathbf{C}_k$, and it is changing in each iteration step, too.

Before conducting the first Kalman step, a seed state $\mathbf{x}_{0,\text{init}}$ needs to be found as well as the corresponding covariance matrix $\mathbf{C}_{0,\text{init}}$. In case of track finding, this is usually a χ^2 fit as introduced in Section 3.7.3. Because the uncertainties of each parameter obtained by this fit, contained in $\mathbf{C}_{0,\text{init}}$, can already be very small, they could bias the result of the KF. Thus, to minimise their influence they are usually artificially increased by a factor between 500 and 1 000.

The individual steps that each iteration step comprises, as shown in Figure 3.10, are then:

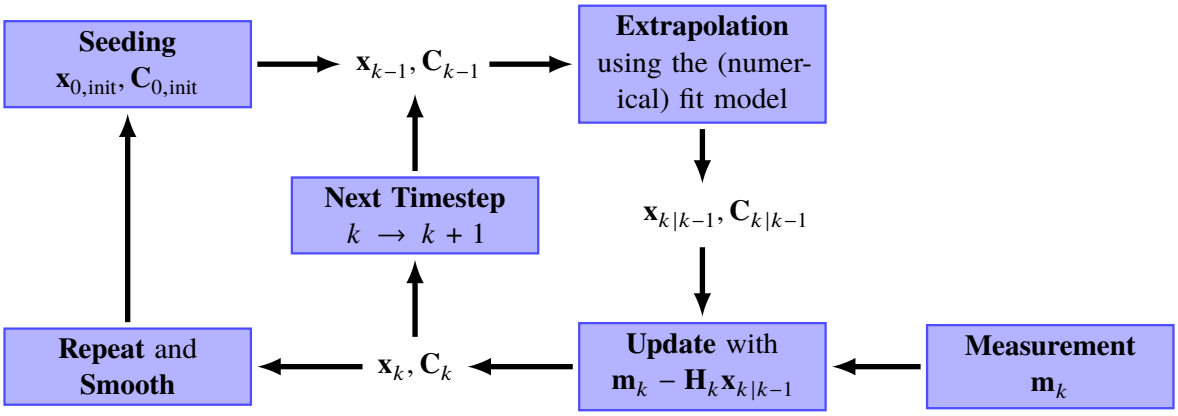


Figure 3.10: Schematic procedure of the Kalman filter. Figure after [74].

Extrapolation In the extrapolation step $k-1 \rightarrow k$ the state vector $\mathbf{x}_{k|k-1}$ and its covariance matrix $\mathbf{C}_{k|k-1}$ predicted based on all previous measurements up to and including $k-1$

$$\mathbf{x}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{x}_{k-1} \quad \text{and} \quad \mathbf{C}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{C}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}$$

with the covariance $\mathbf{Q}_{k-1}$ of the random disturbances $\mathbf{w}_{k-1}$ due to MS resulting in residuals $\mathbf{r}_{k|k-1}$ and their corresponding covariance matrix $\mathbf{R}_{k|k-1}$

$$\mathbf{r}_{k|k-1} = \mathbf{m}_k - \mathbf{H}_k\mathbf{x}_{k|k-1} \quad \text{and} \quad \mathbf{R}_{k|k-1} = \mathbf{V}_k + \mathbf{H}_k\mathbf{C}_{k|k-1}\mathbf{H}_k^T$$

Update In the update or filtering step, the state vector $\mathbf{x}_{k|k-1}$ and its covariance matrix are updated using the residual $\mathbf{r}_{k|k-1}$ between the state and the measured value $\mathbf{m}_k$. This is to decrease the difference between the predicted state and the measurement representing the target value

$$\mathbf{x}_k = \mathbf{x}_{k|k-1} + \mathbf{K}_k\mathbf{r}_{k|k-1} \quad \text{and} \quad \mathbf{C}_k = (\mathbb{1} - \mathbf{K}_k\mathbf{H}_k)\mathbf{C}_{k|k-1}$$

where the *Kalman gain matrix* $\mathbf{K}$ which represents both the uncertainties of the state and the measurement is defined as

$$\mathbf{K}_k = \mathbf{C}_{k|k-1}\mathbf{H}_k^T(\mathbf{V}_k + \mathbf{H}_k\mathbf{C}_{k|k-1}\mathbf{H}_k^T)^{-1} = \mathbf{C}_k\mathbf{H}_k^T(\mathbf{R}_{k|k-1})^{-1}.$$

This yields the updated residual and the corresponding covariance

$$\mathbf{r}_k = \mathbf{m}_k - \mathbf{H}_k\mathbf{x}_k = (\mathbb{1} - \mathbf{H}_k\mathbf{K}_k)\mathbf{r}_{k|k-1} \quad \text{and} \quad \mathbf{R}_k = (\mathbb{1} - \mathbf{H}_k\mathbf{K}_k)\mathbf{V}_k = \mathbf{V}_k - \mathbf{H}_k\mathbf{C}_k\mathbf{H}_k^T.$$

Finally the local χ^2 increment χ^2_+ and the updated total χ^2 value

$$\chi^2_+ = \mathbf{r}_k^T\mathbf{R}_k^{-1}\mathbf{r}_k \quad \text{and} \quad \chi^2_k = \chi^2_{k-1} + \chi^2_+$$

can be calculated. At this point, the state is a weighted average of all information from previous measurements via the state $\mathbf{x}_{k|k-1}$ and the current measurement $\mathbf{m}_k$.

Both the extrapolation and update steps are repeated until no more hits (= measurements) are available for attachment. Now the best estimate of the initial parameters is given by the final state vector, which includes all measurements. However, the result can still be improved by *smoothing*. For this, the full Kalman filter is executed again, with the final state obtained from the first execution serving as seed. As before, the uncertainties of the parameters represented by the covariance matrix are increased by a large factor. Because it is computationally more efficient, the second iteration of the full Kalman filter should start at the last measurement checked and then continue in reversed order. As all measurements are used again in this second iteration, the average of both iterations is the final result, and it also reduces the dependency of the estimated optimal track parameters on the initial seed.

While the track parameters are now determined with high precision, the track can still contain outliers among its set of hits, which deteriorate the quality of the final track parameters. Thus it would be beneficial for the final estimation of the track parameters to remove these outliers, a method for which is described in the following.

3.7.5 Deterministic Annealing Filter

The default final track fitting algorithm for Belle II is the Deterministic Annealing Filter (DAF) [75] implemented in the *GenFit2* library [76]. It is based on the idea of executing the KF multiple times while decreasing an artificial temperature to remove outliers or wrongly attached hits among the set of hits forming the track. These outliers can bias the track fit result and significantly deteriorate the fit quality.

After an initial fit based on all hits using the KF the residuals r for each hit in the track are calculated. In the next step, the weights of all hits with a residual larger than a maximum residual r_{max} are reduced, or the hits are even removed entirely from the fit. While each hit used in the KF has a weight based on the measurement uncertainties, these weights are the same regardless of whether or not the hit belongs to the track. The fit is repeated multiple times, in each step recalculating the residuals of each hit and reducing the temperature and with it r_{max}. With each iteration the track parameters are better defined, as the weight of the outliers and wrongly attached hits decreases.

3.7.6 Simple fits

Although more sophisticated track fits exist, as introduced in the previous sections, a simple track fit often yields a good estimation of the track parameters. All simple fits introduced in this section are χ^2 fits and used for both the VXDTF2 and the newly developed SVD Hough Tracking introduced in Chapter 5. Three different algorithms are used as simple methods to estimate the helix parameters, a circle fit, a triplet fit, and a helix fit. In addition to an estimate of the helix parameters they provide a *p-value* that can be used as a quality indicator. A p-value is a measure of how likely a fit result is, given the χ^2 of the fit and the NDF. Although the three algorithms are part of the newly developed SVD tracking in this thesis and thus play a crucial role for its performance, they are only introduced briefly and more information on them can be found in the corresponding references.

Circle fit

The first simple track fit is a *circle fit* following the description in [77]. At least four hits are required for the fit, as it is always possible to calculate a circle that passes perfectly through three points on a

plane. The fit calculates a circle center and the average distance of the hits from this center, which is the radius of the circle, and an approximation for the POCA and φ_0. Since all five helix parameters need to be calculated, λ_0 is approximated using the radius in the x-y-plane and the z-coordinate of the innermost hit, assuming $(0, 0, 0)$ as origin for the track, which is a valid assumption for many of the tracks seen in the Belle II detector. This method is used in a very similar way to perform intermediate fits of CDC tracks.

Triplet fit

A second method to estimate the track parameters using SVD information only is the *triplet fit* described in [78]. As the name indicates, triplets of hits are used for fitting, each subsequent triplet containing two hits of the previous triplet. For this fit method the hits need to be organised inside-out sorted by their radius, and the triplets are formed step-by-step from the first three hits to the last three hits. Although three hits in the track candidate are enough to perform this fit, its quality improves with more hits. Since triplets are used for the fits, material effects can be taken into account to some extent. Because the full three dimensional information of each hit is exploited, the fit yields a full 3D approximation of the track.

Helix fit

The last simple fit is a full 3D helix fit based on a Riemann sphere described in [79]. Like the triplet fit it makes full use of the 3D information of each hit. However, it does not approximate the particle's trajectory based on three hits over and over, but calculates the optimal set of helix parameters using all hits at the same time using a χ^2 based approach.

3.8 Tracking performance Figures of Merit

When writing any algorithm for reconstruction of particles it is essential to evaluate its performance before using it during data taking. For track reconstruction, high track finding efficiency and purity are important. It is possible to estimate the performance on recorded data, for which one method is introduced in Section 5.8, but it is easier using simulated data where all truth information available.

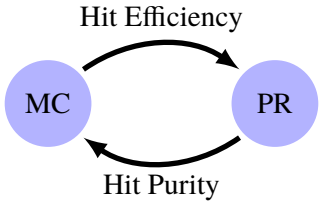
In each simulated event first MC tracks are created using the known simulated true hits. These are compared to the tracks found by the track finding algorithms, called Pattern Recognition (PR) tracks based on the hit content. As described in Section 3.7, a track needs to consist of enough hits to have at least five NDF for a valid track fit. Thus all MC and PR tracks with less than five NDF are not considered for track matching, as they cannot be fitted and thus be used in analysis. Due to this requirement, only traces from MC particles that are in the fiducial volume of Belle II are considered, and the maximum finding efficiency of 100 % is feasible.

When comparing the hit content of the MC and PR tracks the number of common hits n_c is counted, where the hit content of the MC track resembles the correct hits. Based on this number two tracking Figure Of Merit (FOM) can be calculated:

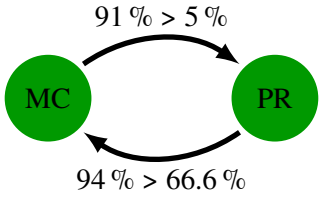
Hit Efficiency The hit efficiency is defined as the ratio of the number of common hits n_c with the total number of hits in the MC track n_{MC}: n_c/n_{MC}.

Hit Purity The hit purity is defined as the ratio of the number of common hits n_c with the total number of hits in the PR track n_{PR}: n_c/n_{PR}.

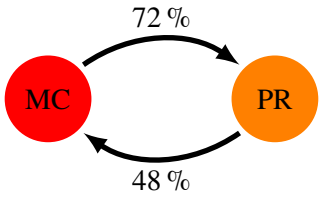
Ideally, both quantities are close or equal to 100 %. They cannot only be calculated on the full set of hits in a track, but also per sub-detector. Each track in the two sets is afterwards categorised into *found* or *missing* (MC tracks) and *matched*, *fake*, or *clone* (PR tracks), as follows (adapted from [70]).



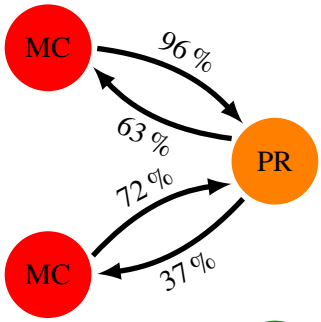
Example introducing the *hit efficiency* (upper arrow) and *hit purity* (lower arrow) in the relation between a MC track and a PR track. The notation with the two errors is used in the following to describe the relation between MC and PR tracks and the criteria to classify the tracks.



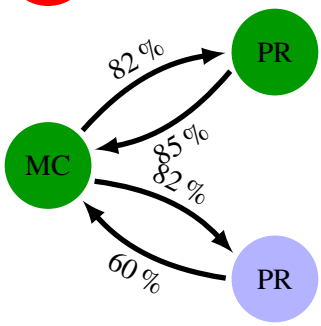
Each MC track that is related to at least one PR track with a hit efficiency of at least 5 % and a hit purity of at least 66.6 % is called found. Similarly, a PR track fulfilling these requirements is called matched.



It is possible for the PR track to contain less than 66.6 % hits from an MC track, but many additional hits from beam induced backgrounds. In this case its purity is too low, and it is called background or fake track. In contrast, if the MC track is not related to any PR track with high enough hit efficiency and purity, it is called missing.



In case a PR track is related to more than one MC track, it might contain a high number of hits from at least one of the MC tracks, e.g. 96 % hit efficiency to the first MC track. But since the PR track contains hits from several MC tracks, the hit purity with none of the related MC tracks is high enough to be a matched track, in which case it is a fake track, too. There are multiple reasons for this to happen, e.g. a decay in flight where one of the decay particles travels in nearly the same direction as the mother particle.



In contrast to the previous case, a MC particle can be related to multiple PR tracks, too. Only one of which can be the matched PR track. The other PR track, usually the one with lower purity, is considered a clone track. Especially low p_T curling tracks create clones, which travel through the tracking volume multiple times. The MC track is reconstructed as multiple PR tracks which are not merged into one final track.

Averaging over a larger set of tracks from many events allows for the calculation of three main track finding FOM:

Finding Efficiency The finding efficiency ε is the fraction of MC tracks marked as found over the

total number of MC tracks in the fiducial volume: $\varepsilon = n_{\text{found}}/n_{\text{MC}}$. As the detector acceptance is factored out, 100 % are theoretically feasible, but in practice never all tracks are found.

Fake Rate Dividing the number of PR tracks marked as fake by the total number of PR tracks yields the fake rate: $n_{\text{fake}}/n_{\text{PR}}$. As fake tracks often are from random combinations of hits, or actual tracks from beam induced backgrounds, but do not belong to the physics event of interest, the fake rate should be as low as possible to not deteriorate analyses by an impure track set.

Clone Rate Similar to the fake rate, the clone rate is estimated by dividing the number of PR tracks marked as clone by the total number of PR tracks: $n_{\text{clone}}/n_{\text{PR}}$. It should be as low as possible, too. Even though they are parts of actual tracks that are reconstructed but not added to the main reconstructed track for this particle, they add additional momentum and charge to an event, possibly decreasing the *missing momentum* variable used in semi-leptonic analyses, and disturbing the total charge distribution. The most common source for clone tracks are tracks found in different sub-detectors but not combined into one track, or curling tracks that are found in multiple parts and not merged into one track.

In addition to the single track FOM for the classification of each track, the hit efficiency and hit purity can be calculated globally on all tracks classified as matched or found, providing an insight to the average hit quality of all tracks. These values are used to judge the track finding performance in addition to the overall track finding efficiency, fake rate, and clone rate. Usually lower hit efficiencies and hit purities result in tracks of less quality, since the track fit can not use all information that potentially is available. Low hit purity can result in a deterioration of the track parameters, as often wrong hits that degrade the hit purity are further away from the actual particle trajectory. Thus, including them in the track fit worsens the quality of the track fit result, too.

At this point it needs to be clearly stated that the hit efficiency and purity are only calculated based on all matched tracks. This implies that both hit efficiency and purity can be 100 %, even if only a small fraction of tracks is actually found. All hits that are missed because the track they are on is not found do not contribute to the calculation. Thus, all tracking FOMs need to be considered in combination in order to assess the track finding quality.

CHAPTER 4

DATCON – FPGA-based Data Reduction for the PXD

Initially the plan for this work was solely to continue the optimisation of the DATCON system which was first developed and described in [2] and further developed and optimised in [3]. During the work described in this thesis, several limitations of the system were found which are described in this chapter. First, the FPGA implementation of DATCON is described. This is followed by the description of the according implementation in basf2, and by the presentation of the tracking and ROI finding results. Afterwards, a comparison of the ROI finding performance of DATCON and HLT as well as combining both to represent the actual data taking configuration is presented, and a summary and outlook including proposals for future improvements.

4.1 The FPGA implementation of DATCON

As introduced in Section 2.5, DATCON is one of two systems providing ROIs for the data reduction for the PXD, with the second being the HLT. In both cases tracks of charged particles are reconstructed from hits in the SVD (DATCON) and SVD and CDC (HLT), respectively. Each track found is extrapolated to the PXD layers and an ROI is calculated for each sensor the extrapolated track hits or where it is close enough to. For the calculation of the HLT ROI the full track finding chain is used online, and the extrapolation is based on a track fit using the DAF of GenFit2 (c.f. Section 3.7.5) and the full track model. In contrast, tracks found by DATCON are extrapolated using simple track models: a circle for the extrapolation in r-φ, and a straight line for the extrapolation in λ, both starting from the origin. This can cause large uncertainties because of the imprecise track models, which is explained in more detail later in this chapter.

The initial idea was to develop and optimise the algorithms for DATCON using basf2 and to port them to FPGA later. FPGAs consist of freely programmable logic gates. In contrast to a regular CPU, which is found in all modern day computers and phones, instructions are not only executed in series, but additionally all logic blocks work in parallel. Thus depending on the implementation many different instructions can be executed in just one clock cycle, which can offer large performance benefits compared to a regular computer, even at much lower clock speeds compared to modern CPUs. For instance, this allows for checking many HS sectors for whether one of the sinusoidal curves defined in Equation (3.6) passes through them in one clock cycle, e.g. a grid of 16 (horizontal) $\times$ 64 (vertical)

sectors at once. While the parallel execution is beneficial and helpful for a fast reconstruction of tracks, FPGAs have some drawbacks. The most important of which are:

1. In contrast to a regular CPU, which contains instructions for e.g. multiplication or addition of two values, and can perform this operation via its microcode, an FPGA consists only of a set of configurable logic blocks, incapable to perform even simple tasks on its own. A dedicated firmware is required, similar to the micro on a CPU, which has to be implemented specifically for the task at hand. This firmware defines the configuration of the logic blocks. While modern CPUs can have access to several GB of Random Access Memeory (RAM) to store both data and the instructions to be executed, FPGAs only have a few kB or MB to store the data. Since the number of logic blocks on an FPGA is finite, the firmware containing all the instructions cannot use more logic blocks than available, while a regular computer that executes all instructions in order can just proceed loading instructions from memory. If the firmware requires more logic blocks than available, either the complexity of the algorithm needs to be reduced, which can lead to loss of functionality, or a different FPGA model with more logic blocks is required. However, while different FPGA models are available, the exchange of the FPGA itself might require the development of a new carrier board that the FPGA integrated on, resulting in high development cost. Thus, the number of logic gates required for a specific firmware always need to be compared with the number of available logic gates, while a CPU does not have such a limitation.

2. FPGAs can only deal with integer values if no floating point capabilities are added via firmware, which would take some of the resources like storage and some of the logic gates. Thus, every floating point number either needs to be represented by an intermediate float-like data format with a limited amount of bits and dedicated logic, or by removing all decimal places. However, this is not an option if the absolute values of the floating point numbers are smaller than 1, as removing all the decimal places would result in zeros, like for sine and cosine. Another option is to multiply all values by some power of 2 or 10, and dropping the remaining decimal places afterwards. While the first one is computationally very inexpensive, as it can be realised by shifting bits, the latter one is easier to understand when reading the code. A value of 53.27 cm could be multiplied by 10 000 basically converting it to 532 700 µm. It's essential to carefully track the additional powers of 2 or 10, since the final value needs to be in the correct (initial) unit again to have a meaningful result.

3. Some algebraic instructions like a division, calculation of sine or cosine, or taking a square root can take multiple clock cycles, up to 36 for a division for instance. Thus reducing the number of these instructions as much as possible is important. Some of the resulting problems can be avoided by using Look-Up Table (LUT) in which values are stored that are known to be needed. For instance, instead of a division by 2, the value of 0.5 could be stored as 500 (multiplication by $1000 = 10^3$), while a value for a division by 20 (= 0.05) could be stored as 50 (still a multiplication by 1000). Of course this is not easily possible for the division of two arbitrary numbers a and b, but only for predefined values. In addition, also pre-calculated values for trigonometric functions can be stored in LUTsafter conversion to integer values by multiplication with powers of 2 or 10 if necessary, as their calculation is not really possible on FPGAs because of the missing floating point capabilities. Also values that are needed frequently can be stored in LUTsto avoid their repeated calculation.

For these reasons the values of sine and cosine that are needed for the calculation of the HT in Equation (3.6) are stored in LUTs, all multiplied by 1 000. The uncertainties resulting from using three digits only were decided to be acceptable by the developer of the FPGA version of DATCON, as other sources of uncertainties contribute much more to the final results.

The actual DATCON system consists of 14 FPGA boards in total. Twelve of these boards, called *concentrator boards*, are connected to the SVD FEE via optical links. The information they receive is

- The number of the readout board Fast Analog-to-Digitcal Converter (FADC) [80] that sent the data. Each FADC board is connected to up to 148 APV [81] chips which are directly mounted on the single sensors.
- The number of the single APV chip. Each APV chip is connected to 128 strips of one side of a SVD sensor.
- The number of each strip (0 to 127) connected to an APV that produced a valid signal after being hit by a traversing particle.

There is a unique mapping between the FADC boards and the concentrator boards. Since a precise hit reconstruction as performed in basf2 and as described in Section 2.5.1 is not feasible on the FPGAs available, a different approach is chosen. Each concentrator board first performs a data quality check on the data of single strips by checking whether the SNR of each strip signal is above a threshold (usually 5) by comparing the maximum of the six samples to the noise of the strip. In addition, the other samples are checked, too. The APV shapes the sample charges as

$$f(t) = A\frac{t}{\tau}e^{-\frac{t}{\tau}} \tag{4.1}$$

with a calibration constant A, (sample) time t, and the shaping time τ of 50 ns [39].

An exemplary sample distribution of a valid strip signal is shown in Figure 4.1. Samples are created every 31.44 ns by the APV. In DATCON the maximum sample is required to be the second, third of fourth of the six samples, and it needs to have a SNR of at least five, indicated by the horizontal black line in Figure 4.1 as an example. If these two requirements are not fulfilled, the digit is discarded. Next, the two samples neighbouring the maximum sample need to be above the noise threshold defined by half the ADU value of the maximum sample. In addition, the shape of the samples is checked, it needs to roughly follow the distribution described by Equation (4.1). Starting at the maximum sample, every sample is compared to its successor (predecessor) forward (backward) in time and needs to be larger than its successor (predecessor). The distribution of raw and filtered SVDDigits per event is shown in the left half of Figure 4.2. Only samples fulfilling all requirements are accepted and propagated to the clustering algorithm. Due to the noise filtering, the number of u-side (v-side) SVDDigits is reduced from 1763 ± 316.5 (1130 ± 180.2) to 409 ± 117.8 (291 ± 81.9). This searches for adjacent strips on the same side of the same sensor that pass the background rejection check. All neighbouring strips creating a valid signal are then combined into a cluster. As a next step, the centre of the cluster is estimated, which is defined to be

- The centre strip in case the cluster size is odd.
- The next larger integer compared to cluster size divided by two in case the cluster size is even.

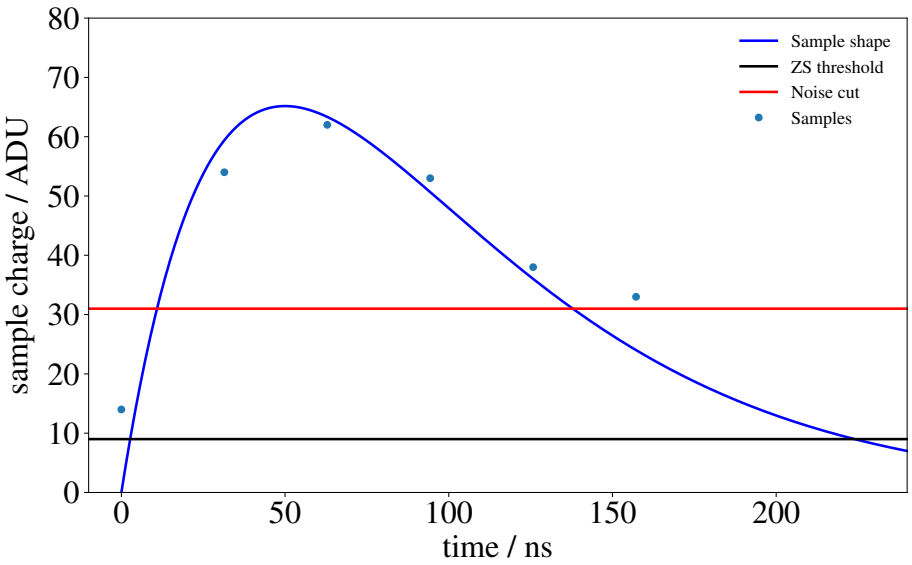


Figure 4.1: APV sample distribution over time for a hit on one strip in the SVD in Analog-Digital-Unit (ADU) (blue points). The blue curve defines the sample shape based on Equation (4.1). The horizontal black line indicates the zero-suppression threshold which is three times the noise of each strip for the SVD. Signals are only accepted on hardware level if the maximum sample is above this threshold. Last, the horizontal red line indicates the noise cut in DATCON which value is half of that of the maximum sample.

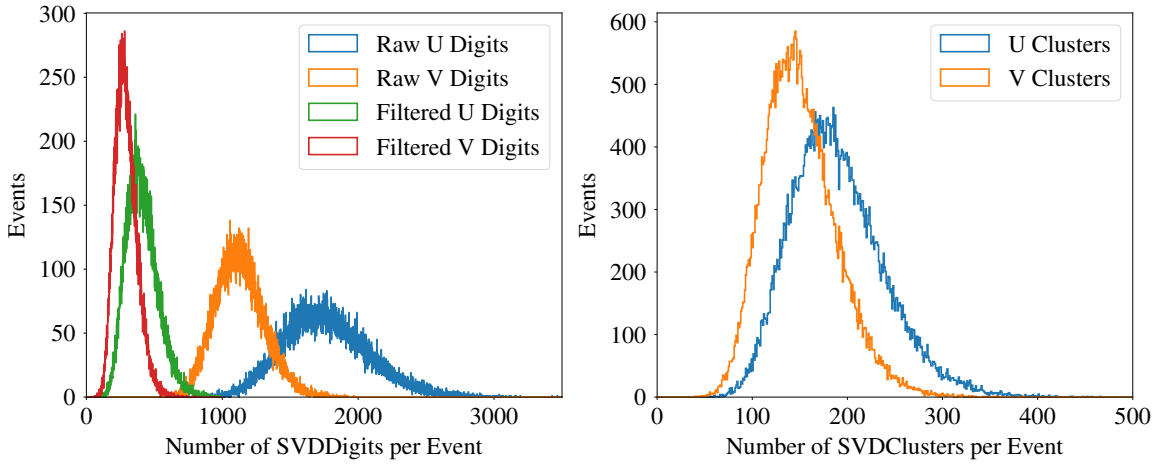


Figure 4.2: The left panel shows the number of raw SVDDigits for the u-side (blue) and v-side (orange), and noise filtered SVDDigits in green (u-side) and red (v-side). The noise filtering efficiently removes many noisy SVD strips, reducing the number of SVDDigits for clustering by a factor of around 4 to 5 depending on the side. In the right panel the distribution for the resulting u (blue) and v (orange) SVDClusters is shown. Around 189 ± 49.1 u clusters are created, as well as 150 ± 38.5 v clusters.

The distribution of SVDClusters per event is shown in the right half of Figure 4.2. Around 189 ± 49.1 u clusters are created, as well as 150 ± 38.5 v clusters. These are the clusters that enter the HT.

In contrast to the SVD reconstruction in basf2, the information of the single strip times or charges are not used during the creation of the SVD clusters. To limit the number of calculations on the FPGA, the corresponding x-y (z) coordinates for each strip on the u-side (v-side) are stored in LUTs, enabling to retrieve the coordinate directly from the seed strip number. Since for the r-φ HT the conformal mapped values of x and y are needed, these are saved in the corresponding LUT instead of the actual coordinates.

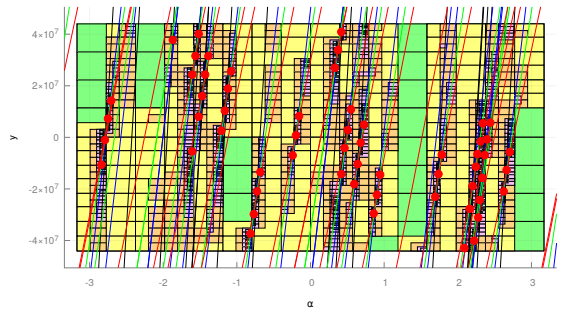
Subsequently the coordinates are propagated to the remaining two FPGA boards called *tracking boards*. These boards conduct the actual track[1] finding and the extrapolation to the PXD, and using more powerful FPGAs compared to the concentrator boards. One of the two tracking boards performs the HT on the u-side cluster coordinates to find the tracks in the r-φ-direction (i. e. in the x-y-plane), while the other one performs the HT on the v-side cluster coordinate to find the dip angle λ of the tracks, respectively. In the first step, all the sinusoidal curves from the HT are checked for the HS sectors they pass through. A HS sector passed by sinusoidal curves originating from hits on at least three different SVD layers is called *active*. While on the FPGA all sectors in both HSs are checked, with several sectors being checked simultaneously in one clock cycle, for sinusoidals passing, in the basf2 implementation a fast HT is used. This difference is be explained in more detail in Section 4.2. To reduce the amount of tracks from random combinations of hits, only sections of the sinusoidals with a positive slope are taken into account in this step. This provides the additional advantage that only the outgoing *arm* of each actual track is be found. Since the particles loose energy while traversing the detector, the second (ingoing) arm does not reach as close to the IP as the starting point for the first outgoing arm, which provides the best information on the actual track.

Afterwards the active sectors are clustered using a *depth first search* approach.[2] Starting in the lower left corner of the HS, only the top, right, and top-right neighbours are checked for being active or not. This approach is chosen because only the ascending sections of the sinusoidals are checked for passing a HS sector. For each active neighbour of the three neighbouring candidates currently under investigation, the checking procedure is repeated recursively with the corresponding neighbouring sectors. If no additional active neighbours can be found for an active sector, the search goes on with the last active sector for which so far not all active neighbours were checked and added to the cluster due to the recursive nature of the algorithm. A cluster is saved if it has a given minimum size and no more active sectors can be added, or if further sectors can be added, but it reaches a maximum threshold, in which case it is saved, too, and a new cluster is started. In total there are four user definable cluster size limits for each of the two HSs:

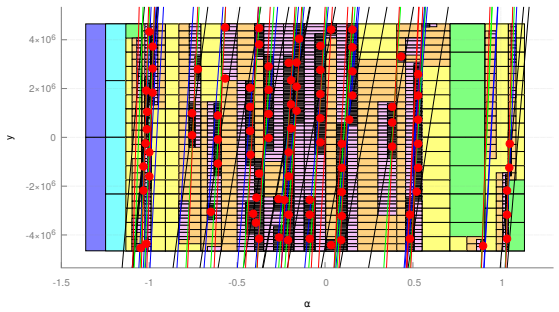
- *minimum cluster size*: only store a cluster if its size is at least this value, otherwise discard it
- *maximum cluster size*: store a cluster if its size is smaller or equal to this value, and start a new cluster from the next neighbouring active sector
- *maximum cluster size in x*: store a cluster if its size along the x-axis is equal to this value, and

[1] The term *track* is used very loosely in this context of the FPGA based DATCON, as the result of the HT are not actual tracks, but just combinations of a track radius r and corresponding azimuth angle φ for the r-φ HS, and dip angle λ for the λ HS obtained from the clusters in the corresponding HS.

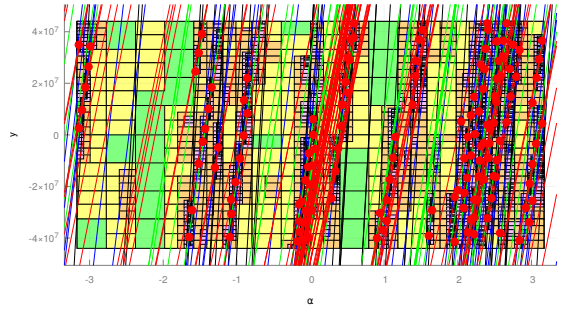
[2] The depth first search algoritm was first described by Charles Pierre Trémaux in the early 19th century as a strategy for solving mazes.



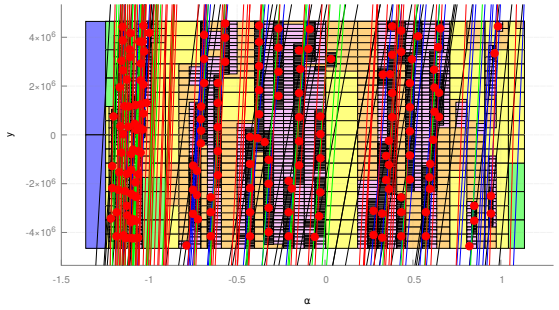
(a) Hough Space for the extraction of φ for a $\Upsilon(4S)$ event without beam backgrounds.



(b) Hough Space for the extraction of λ for a $\Upsilon(4S)$ event without beam backgrounds.



(c) Hough Space for the extraction of φ for a $\Upsilon(4S)$ event with beam backgrounds for nominal luminosity.



(d) Hough Space for the extraction of λ for a $\Upsilon(4S)$ event with beam backgrounds for nominal luminosity.

Figure 4.3: Examples for Hough Spaces for an $\Upsilon(4S)$ event with and without beam background for the extraction of φ (a) and λ (b) in case no beam backgrounds are present, and for the extraction of φ (c) and λ (d) in case beam backgrounds are present. The lines are the sinusoidal curves from the HT coloured according to the SVD layer of the hits for layer 3 (black), layer 4 (blue), layer 5 (green), and layer 6 (red). The different colours of the rectangles of the HS represent the iteration level in the fast HT, and the red dots mark the positions of found HS clusters. In this case, the fast HT is used, as it is computationally more efficient on a PC. Exaggerated views oIf each of the four figures can be found in Appendix A.1.

start a new cluster from the next neighbouring active sector

- *maximum cluster size in y*: store a cluster if its size along the y-axis is equal to this value, and start a new cluster from the next neighbouring active sector.

These limits are needed to avoid the creation of too many fake tracks (minimum cluster size) by random combinations of sinusoidal lines crossing a sector, and to retain a certain accuracy in the cluster creation, since a cluster that extends too far in either the x-axis or the y-axis yields inaccurate information about the cluster Center of Gravity (CoG), which is calculated for each cluster that is stored. An example for the sinusoidals in the HS as well as the found clusters and their CoGs is shown in Figure 4.3. In all four Figures 4.3(a) to 4.3(d) the active sectors in the final iteration of the fast HT are coloured in light blue, and often hidden under the red dot marking the CoG. In particular in Figure 4.3(b) and 4.3(d) for the λ HS it is visible that the clusters have a large extension along the x-axis, mostly caused by random combinations of sinusoidal lines from hits from an actual track with those of another track (Figure 4.3(b)), or with background hits (Figure 4.3(d)). If the real intersection

of sinusoidal lines belonging to a track is on the left edge of such a cluster, the angular coordinate wrongly be estimated with a value too large, which eventually would result in a bad extrapolation to the PXD. However, limiting the size of the clusters can result in a larger number of clusters, thus all four cluster size thresholds for each of the two HSs have to be optimised to minimise the number of HS clusters, while having a high ROI finding efficiency. For the best ROI finding performance, the four limits have to be optimised alongside the vertical size of HS, and the number of sectors in x and y direction, yielding seven parameters for each of the two HSs and 14 in total. The optimisation of the parameters is described in Section 4.3.1.

4.1.1 Extrapolation to the PXD

Once the two tracking boards are done with the track finding step, i. e. after clustering of the HS sectors, the track parameters defined by the CoG of the clusters of each HS are extrapolated to the two PXD layers as a circle in the r-φ-direction and as a straight line using λ. The different steps of the extrapolation are shown in Figure 4.4. From the r-φ HS the track radius r_t and the azimuth angle $\varphi_{t,0}$ are known. A track with these two parameters then hits a sensor plane at a radius of r_s at an angle $\varphi_{s,0}$ with the x-axis. The initial situation is depicted in Figure 4.4(a). To simplify the calculation, the track and the sensor are rotated by the angle of the sensor $\varphi_{s,0}$ such that the sensor is perpendicular to the x-axis, Figure 4.4(b). In this position, the y-coordinate of the intersection point between track and sensor can easily be calculated analytically as the intersection between a circle (representing the track) and a straight line (representing the sensor). The general solution to this problem is

$$y = y_c \pm \sqrt{r^2 - \left(x - x_c\right)^2} \tag{4.2}$$

with (x_c, y_c) being the coordinates of the centre of the circle and r as its radius. In this configuration the angular difference $\Delta\varphi = \varphi_{t,0} - \varphi_{s,0}$ is the only important angular variable. The (relative) track centre is at $\left(x_c,\, y_c\right) = \left(r_t \cdot \sin\Delta\varphi,\, r_t \cdot \sin\Delta\varphi\right)$, and the straight line is at $x = r_s$ Thus, the extrapolation can be performed via

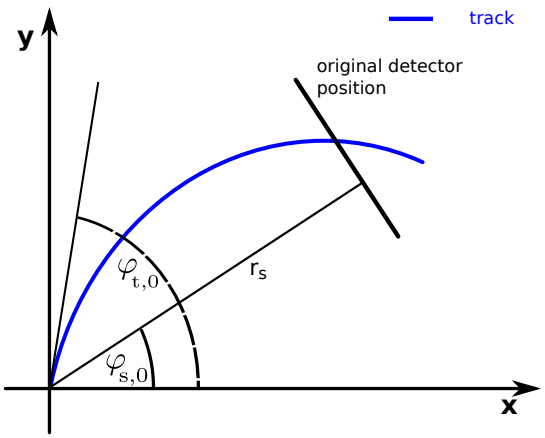
$$y_{1,2} = r_t \cdot \cos\Delta\varphi \pm \sqrt{r_t^2 - \left(r_s - r_t \cdot \sin\Delta\varphi\right)^2} \tag{4.3}$$

yielding two possible solutions for y. Both of these need to be checked for whether or not they actually are in the active area of the pixel sensor, taking into account the shift $\Delta y = 3.5\,\mathrm{mm}$ of the PXD sensors that is used to create the windmill structure. The condition for this is

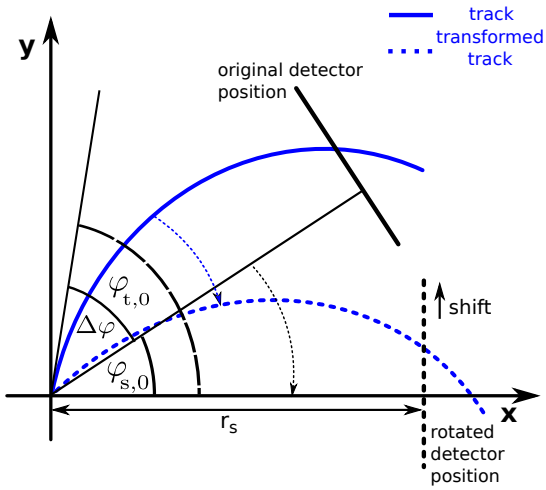
$$-\frac{w_s}{2} + \Delta y \;\leq y_{1,2} \;\leq +\frac{w_s}{2} + \Delta y \tag{4.4}$$

with w_s as the width of the sensor. If this is fulfilled the track intersects with the sensor. As only local coordinates on the sensor are relevant for the calculation of the ROIs, it's not necessary to rotate the system back to its original position. This procedure is repeated for each track with each of the 8 (12) layer 1 (2) ladders.

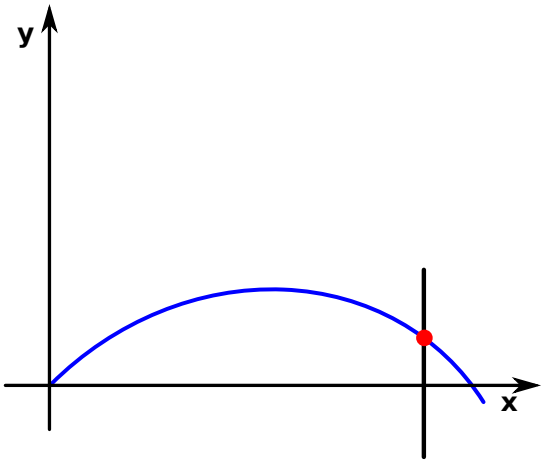
The extrapolation to obtain the global z coordinate of the intersection is simpler as the track is extrapolated as a straight line using $\tan\lambda$ as the slope. Now, the intersection of two straight lines needs to be calculated as sown in Figure 4.4(d). Ideally, for each y position that is accepted in the r-φ



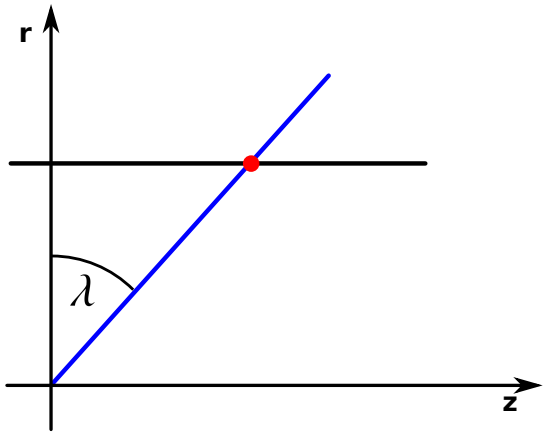
(a) Initial problem: Intersection calculation of a circle (= track, blue) with a straight line (= detector, black). In addition the two initial angles $\varphi_{s,0}$ and $\varphi_{t,0}$ for the sensor and track are shown, as well as the sensor radius r_s.



(b) Rotate the system by $\varphi_{s,0}$ so that the detector is perpendicular to the x-axis. The only angular variable of importance is the angular difference $\Delta\varphi = \varphi_{t,0} - \varphi_{s,0}$.



(c) Calculate the intersection (red dot) analytically as the intersection of a circle with a straight line. Since only local coordinates on the sensor are needed, no back-rotation is necessary.



(d) In the v-direction the extrapolated hit (red dot) is calculated from the intersection of two lines, with the black line parallel to the z-axis illustrating the detector and the blue line illustrating the track.

Figure 4.4: Concept of the track extrapolation to the PXD of tracks reconstructed from SVD hits. In the idealised case without energy losses this can be simplified to the analytical calculation of the intersection point of a circle with a straight line.

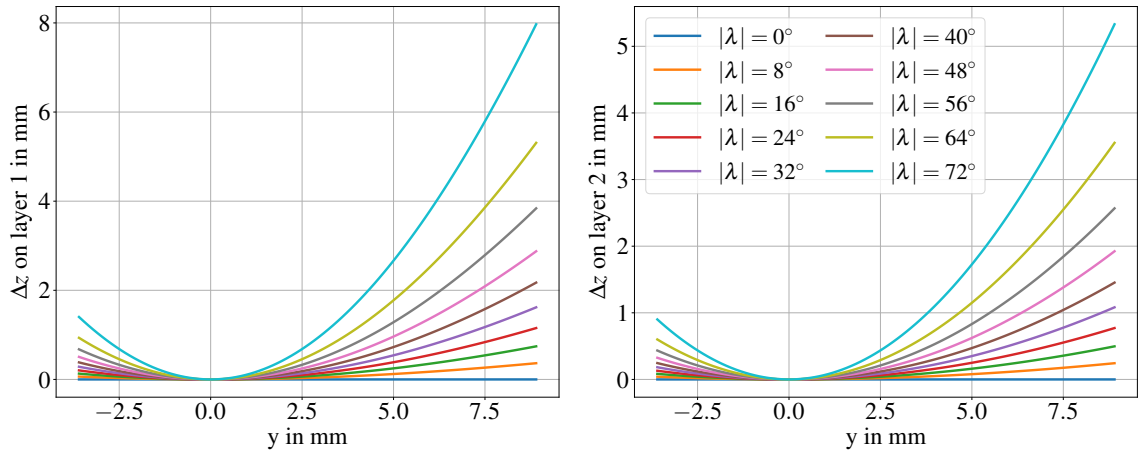


Figure 4.5: Systematic error Δz of the extrapolation in z neglecting the y coordinate of the extrapolated hit from the circle extrapolation in r-φ for different values of λ as function of the y coordinate on the sensor plane. The left (right) side shows the Δz on layer 1 (layer 2). The maximum errors correspond to 120 (60) pixels on layer 1 (layer 2).

extrapolation, the z coordinate is calculated as

$$z = \sqrt{x^2 + y^2} \cdot \tan\lambda = \sqrt{r_s^2 + y^2} \cdot \tan\lambda. \tag{4.5}$$

However, since the two FPGA tracking boards perform the track finding and the extrapolation independently, the values $y = y_{1,2}$ of the extrapolated hits in Equation (4.3) are not known during the z extrapolation and thus neglected, which simplifies Equation (4.5) to

$$z = r_s \cdot \tan\lambda. \tag{4.6}$$

In addition, since the extrapolation for both directions is performed independently, the z coordinate is valid for all sensors of each layer. However, neglecting the y coordinate in the extrapolation by using Equation (4.6) instead of using Equation (4.5) introduces an error

$$\Delta z = \left(\sqrt{r_s^2 - y^2} - r_s\right) \tan\lambda \tag{4.7}$$

in the extrapolation, which is shown in Figure 4.5. Due to the shift of the PXD sensors that is necessary to create the windmill structure, the lower bound of the active area of a PXD sensor in the rotated coordinate system (c.f. Figure 4.4(c)) is at $y = -3.6\,\text{mm}$, while the upper bound is at $y = 8.9\,\text{mm}$, which are the minimum and maximum values on the horizontal axis. In the worst case, the difference in pixels is about 120 for very forward and about 60 for very backward tracks on layer 1. The error is larger on layer 1 because the relative influence of $y = 8.9\,\text{mm}$ compared to the sensor radius of 14 mm is larger compared to the radius of 22 mm on layer 2.

In addition, the straight line extrapolation introduces a second error compared to an extrapolation using a helix, or a sinusoidal as the projection of a helix into the y-z-plane. This error depends on $\tan\lambda$ and p_T and the corresponding distributions are depicted in Figure 4.6. With 150 µm on layer 2,

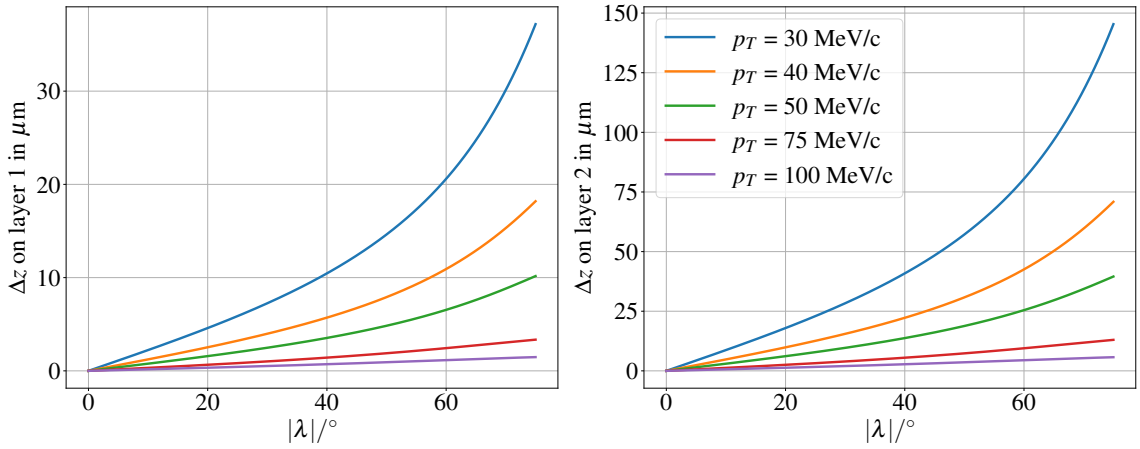


Figure 4.6: Systematic error Δz of the extrapolation in z introduced by using a straight line extrapolation compared to an extrapolation with a sine curve as function of $|\lambda|$ for different p_T values. Since the z component of the momentum vector increases with increasing λ by $p_z = p_\mathrm{T} \tan\lambda$, the track can be approximated as a straight line even for large values of λ because of the high total momentum, even for low p_T values. For a track with a p_T of 30 MeV/c the error is less than one pixel on layer 1, and less than two pixels on layer 2.

the extrapolation error is about two pixels, which is much smaller compared to the error introduced by not taking into account the extrapolated y position.

As the extrapolated y coordinate from Equation (4.3), the calculated z coordinate has to be confirmed to being contained inside the active sensor area, too, via

$$-\frac{l_s}{2} + \Delta z \;\leq z\; \leq +\frac{l_s}{2} + \Delta z \tag{4.8}$$

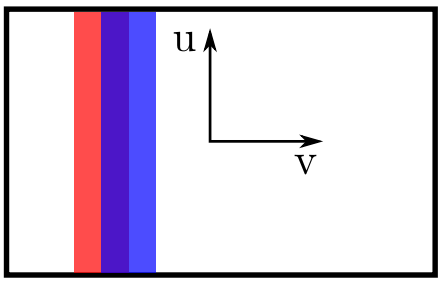
with l_s as the length of the PXD sensors, which is different between layer 1 and layer 2, and Δz as the shift of the sensor.[3] The extrapolated positions y and z have to be converted into pixel IDs id_{col} and id_{row} for the column and row pixel:

$$id_{col} = \frac{y - \Delta y}{u_{pitch}} \frac{w_s}{2} \tag{4.9}$$

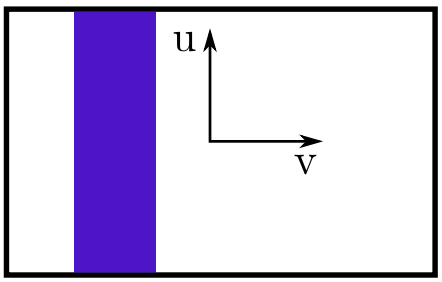
$$id_{row} = \frac{z - \Delta z}{v_{pitch}} \frac{l_s}{2} \tag{4.10}$$

where u_{pitch} denotes the pitch in u-direction, and v_{pitch} denotes the pitch in v-directionfor a given

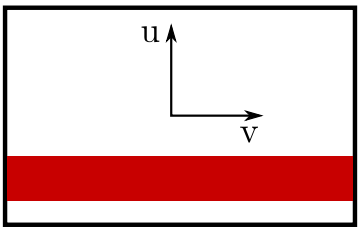
[3] There are four of these Δz shifts in total, one for each of the layer 1 forward and backward and the layer 2 forward and backward sensors.



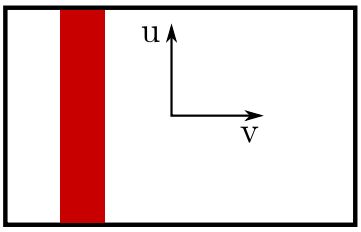
(a) Example of two overlapping ROIs from extrapolation in λ before...



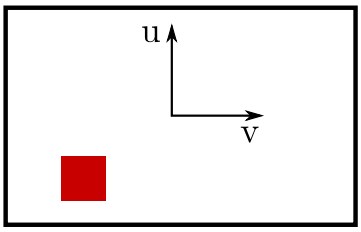
(b) ... and after being merged into one ROI to reduce the total number of ROIs.



(c) ROI in the u-direction from the extrapolation in r-φ covering the full length of a sensor.



(d) ROI in the v-direction from the extrapolation in λ covering the full width of a sensor.



(e) Combined ROI which is sent to ONSEN.

Figure 4.7: ROI creation on the FPGA DATCON system. The coordinate system u-v describes the local coordinates on the sensor. The local u-direction corresponds to the global r-φ-direction, and the local v-direction is parallel to the global z-axis.

sensor, respectively.[4] Finally, the ROIs are calculated as

$$\mathrm{ROI}_{col,1/2} = id_{col} \pm \frac{l_u^{\mathrm{ROI}}}{2} \tag{4.11}$$

$$\mathrm{ROI}_{row,1/2} = id_{row} \pm \frac{l_v^{\mathrm{ROI}}}{2} \tag{4.12}$$

with l_u^{ROI} (l_v^{ROI}) as the size of the ROI along the u-direction (v-direction). Since the cluster positions in the HSs are defined to be the centre of each sector, the possible values for $\rho = 1/r$, φ, and λ are known when constructing the HS. This can be taken advantage of to pre-calculate all the extrapolated positions on the PXD and also to pre-calculate the ROI positions and store them in LUTs, so that by finding the cluster positions in the HS the according ROIs can be looked up without costly calculations and with deterministic runtime on the FPGA, which is important to process the events in 30 µs on average at nominal luminosity.

However, these ROIs are not yet final: an ROI resulting from the extrapolation in r-φ-direction is a belt of a given number of pixels in the u-direction covering the full detector length on both sensors of the PXD ladder that was hit by the extrapolation, as shown in Figure 4.7(c). A ROI from the λ extrapolation creates a belt of a given number of pixels in the v-direction, as shown in Figure 4.7(d), but

[4] The conversion into id_{row} is not as straight forward as indicated in Equation (4.10), since there are different pixel pitches along the v-direction on the forward / backward and layer 1 / layer 2 sensors.

not only on one sensor, but on all sensors of each layer at the same z-coordinate. The two single-side ROIs are then merged into the final ROI, as shown in Figure 4.7(e). However, this ansatz has an obvious problem: the total number of ROIs can grow quickly due to the one-to-one combination of ROIs from both directions. Even if the track finding in both cases was perfect, meaning that both only found real existing tracks but no fakes, there would be on average eleven tracks per side, resulting in 44 preliminary ROIs in u-direction (two ROIs for each track on each ladder, for eleven tracks and on two layers), and 220 preliminary ROI in the v-direction (eight ROI for each track on PXD layer one, plus twelve ROI for each track on layer two). Combining the ROIs on each sensor, the total number of ROIs is at least 242: $11 \times 11 = 121$ ROIs on each of the two layers. Although this is a simplified case, it shows the combinatorial problem of this approach.

As the track finding using the HSs only is not perfect, many more tracks than actually are present an event are found due to combinatorics in each HS. In this case, the number of ROIs would increase very fast, and many of the single-side ROIs might overlap. In order to reduce the issue with combinatorics, overlapping ROIs on each sensor in u-direction or v-direction are merged to reduce the final number of ROIs, as exemplary shown in Figures 4.7(a) and 4.7(b) for two ROIs from extrapolation in λ. Only afterwards the merged single-side ROIs for both directions are combined. For each sensor with ROIs in both u-direction and v-direction the conjunction of the two is a final ROI, as depicted in Figure 4.7(e), that will be further processed. If, however, only an ROI from one of the two extrapolations is present, no final ROI is created. All final ROIs are then sent to the ONSEN which uses the ROIs from both DATCON and HLT to select the PXD hits inside the ROIs, and sends them to the storage system.

Due to background hits and random combinations of hits caused by particles originating from the $\Upsilon(4S)$ decay or secondary processes crossing the same HS sectors and activating them, a large number of irreducible fake tracks is found in both HSs, as shown in Figure 4.3. Reasonable values for the number of tracks actually found without any background are 25 in both the r-φ and the λ HS, as shown in Figures 4.3(a) and 4.3(b). This would yield 625 ROIs on each PXD layer without merging overlapping ROIs, and even with merging the total number of ROIs in each event can easily exceed 1 000 in total on both PXD layers. Creating over 1 000 ROIs on average would result in more than 25 ROIs on each of the 40 PXD sensors, which is difficult to impossible for the ONSEN system to cope with. It will also result in a low data reduction since most of the PXD sensors will be covered by ROI area, as is be shown in Section 4.3. With background, even with optimisation of the parameters of the HS search, the number of clusters / tracks obtained from each HS can easily exceed 200, as indicated by the red dots in Figures 4.3(c) and 4.3(d).

4.2 Implementation of the FPGA like version of DATCON in basf2

As mentioned before, the starting point for this development were the basf2 implementation of the DATCON algorithms described in [2, 3]. However, over time the development of the actual FPGA system, which is described in Section 4.1 and is fully described in [4], and the implementation in basf2 diverged. Thus the implementation in basf2 was adapted to better agree with the actual firmware developed for the FPGA, to compare both systems and to optimise the parameters using the basf2 version. This is because the throughput using a high level programming language for simulations is much larger than simulating the full FPGA firmware on a PC, which includes building a new version after minor changes, or even to simulate data with a PC and then send these data to an actual FPGA for evaluation.

Some shortcuts were taken in the basf2 implementation nonetheless. For instance, LUTs are only used in a few occasions, like for the boarders of the HS sectors, while in other cases the small performance penalty of calculating e.g. trigonometric values during simulation is accepted.Avoiding LUTs also means that all (floating point) values need to be converted into integers with the correct power of ten during processing.

Checking each of the HS sectors individually for all the hits from the HT would be very slow on a PC. With a HS consisting of 128×128 sectors and 1 000 hits this would mean 16.384 million checks for whether a sinusoidal curve passes through a sector, and most of the checks for each curve yielding a negative result. Thus the fast HT based on a *divide-and-conquer* approach is used, and only HS sectors that were passed by sinusoidals based on hits from at least three different SVD layers are further processed. Every active sector is divided in four equally sized parts by dividing the size in both the horizontal and vertical direction in two. These new subsectors are then each checked themselves for the sinusoidals to pass them, and the procedure is repeated until a maximum number of iterations is reached or until sinusoidals originating from hits from less than three SVD sensors pass a sector. This is illustrated for instance in Figure 4.3(a) by the different colours of the rectangles, each representing an iteration step. The active sectors obtained from both the fast divide-and-conquer approach and checking each of the predefined sectors are the same, but the employed method is much faster on a CPU.

4.2.1 Differences to the previous implementation

Although this work is based on [3], the algorithms presented in this chapter are implemented to be as close as possible to the actual FPGA implementations, several details in the implementation are different compared to the previous work. One key difference is that in the previous work additional merging steps were performed for the single track candidates obtained from each HS, e.g. by grouping track candidates with similar parameters. Afterwards the merged track candidates were combined to 3D hits based on their hit content, comparing the lists of hits contained in each r-φ and λ track candidate. Both steps are easy measures to reduce the number of tracks, but are not incorporated in the current version of DATCON. But it requires keeping track of which hit is contained in which track for both HSs. Because of this change, the y coordinate of the extrapolated hit in the rotated coordinate system is not known to and thus unavailable for the extrapolation in z. A detailed discussion of the bias of the z extrapolation is found in Section 4.1.1.

In addition, the track finding developed in the previous work was based on true hit positions on the SVD with each true hit providing x, y, and z coordinates and being bound to a SVD sensor. In the current FPGA implementation it is not foreseen to check the hit content of the found tracks, nor are 3D space points created in the first place. Furthermore, the previous implementation described in [3] contained a merging step for the extrapolated hits on the PXD such that for merged extrapolated hits only one ROI was created, which additionally reduced the number of ROIs. Since the previously described algorithm of merging ROIs is easier to implement on both FPGA as well as in C++, this change is considered beneficial compared to the algorithm of combining and merging extrapolated hits on each sensor which is described in [3], as this involves a moving mean as the average extrapolated hit position.

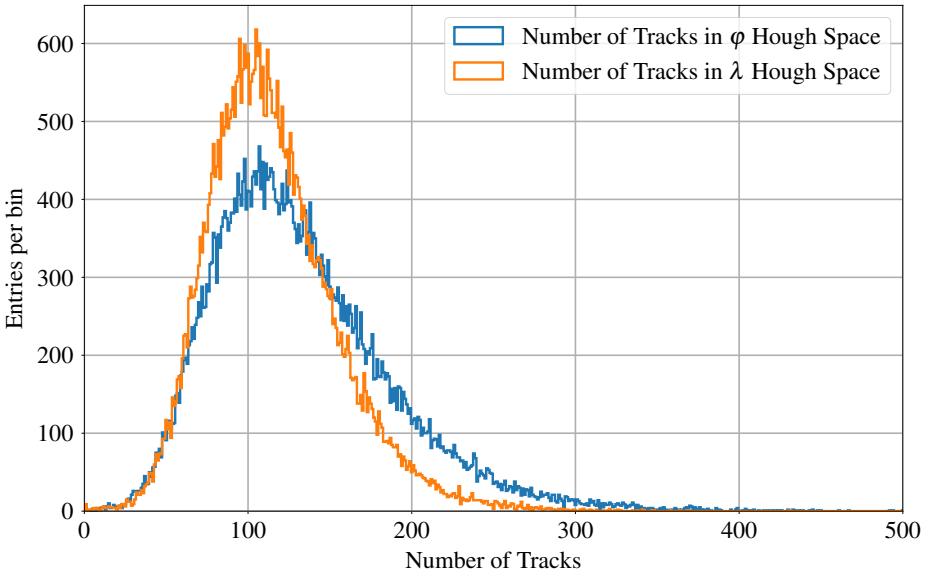


Figure 4.8: Number of tracks found in the r-φ HS (blue) and λ HS (orange) for $\Upsilon(4S)$ events with nominal background. On average 131 (114) tracks are found in r-φ (λ), with a long tail towards higher numbers. The tail is caused by events with many hits from beam backgrounds.

4.3 ROI finding performance with DATCON

As shown in Figure 4.3(c) and 4.3(d), around 100 clusters are found per HS for $\Upsilon(4S)$ events with nominal background conditions. Each cluster in this sense corresponds to one track. A clearer visualisation of the number of tracks per HS is presented in Figure 4.8, using 50 000 simulated $\Upsilon(4S)$ events with nominal background. While there are only few events with less than 50 tracks per HS, in many cases even more than 200 tracks are reconstructed in each of the two HSs. Just combining each track from the φ HS with each track from the λ HS would result in more than 10 000 ROI on average, thus the single-side ROI merging described in the Section 4.1 is indeed necessary.

Since hit information is not available to perform truth matching as described in Section 3.8, the only option to perform truth-matching with MC tracks is via comparison of the track parameters. DATCON provides only the track angles, and the curvature from the two HSs. With on average 100 to 150 tracks distributed more or less equally over 360° for r-φ, and 133° for λ, it is very likely that DATCON tracks with track angles close to the MC values exist. Figure 4.9 shows the residual of the reconstructed angles between the the MC tracks and the tracks found in DATCON without double-counting. The residual is defined as the smallest distance $\Delta\alpha = \alpha_{\mathrm{MC}} - \alpha_{\mathrm{HS}}$ with $\alpha = \varphi, \lambda$. In both cases the distributions are centered around 0, but with long tails, resulting in standard deviations of 2.0° and 1.5° for φ and λ. The residual distributions would be even narrower if more HS clusters were found by DATCON in case they were equally distributed over the corresponding angular range. Thus no tracking performance evaluation is performed, but only performance studies on the ROI finding efficiency, which is the more important performance metric for DATCON.

The ROI finding performance is best estimated with two FOM: the ROI finding efficiency and the Data Retention Fraction (DRF). In addition, the total number of ROIs is important, as it must be low enough for ONSEN to be able to cope with it. This number was chosen to be 31 by the developers

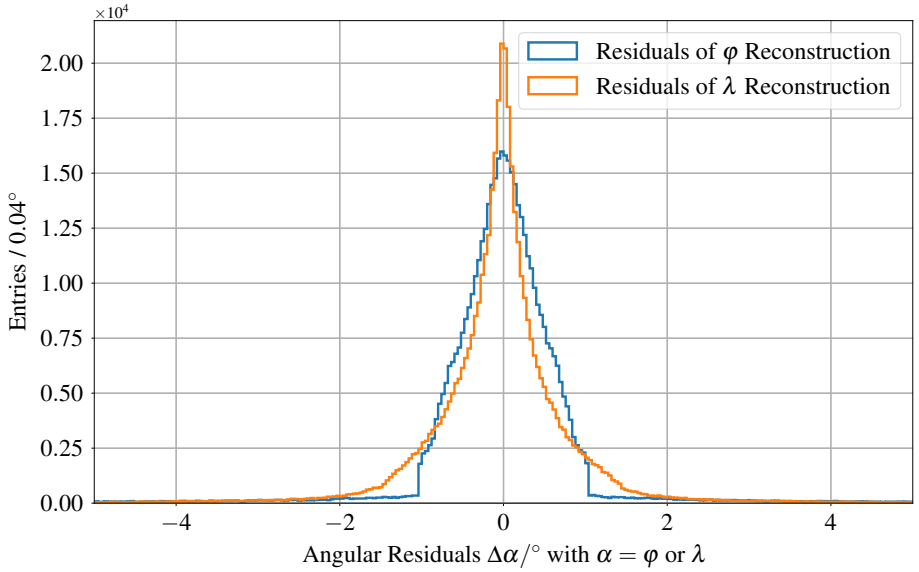


Figure 4.9: Angular residuals $\Delta\alpha = \alpha_{\mathrm{MC}} - \alpha_{\mathrm{HS}}$ with $\alpha = \varphi, \lambda$ for the angles corresponding to the positions of the clusters in each of the two HSs. The blue (orange) distribution shows the residuals in φ (λ). While the λ distribution has a narrower core, it has longer tails in both positive and negative direction compared to the φ distribution. The sharp edges visible in the φ residual distribution originate from the discrete HS sectors. The core of the φ residual distribution as a with of 0.44°, while the overall standard deviation is 2.0°. Similarly, the standard deviation of the λ residuals is 1.5°.

of ONSEN. While the ROI finding efficiency should be close to unity, the data reduction must on average be larger than 10, meaning that on average only about 10 % of PXD hits are saved by the ROI selection. The ROI finding efficiency and the DRF are defined as

$$\varepsilon_{\mathrm{ROI}} = \frac{\text{number of clusters of MC PXD hits inside of ROI}}{\text{total number of clusters of MC PXD hits}} \tag{4.13}$$

$$\mathrm{DRF} = \frac{\text{number of PXD hits inside all ROI}}{\text{total number of hits on the PXD}}. \tag{4.14}$$

The definition of the ROI finding efficiency given in Equation (4.13) is only valid for MC studies. While the DRF can easily be estimated on data, it is not as simple to estimate the ROI finding efficiency, not only for DATCON, as it is difficult to differentiate between

- A track that is not found and thus no ROI is created for this track,
- A poor estimation of the track parameters e.g. because of a poor track fit, such that the ROI position is off compared to the actual position of the track's traversal through the PXD
- Bad hit efficiency on the PXD, where an ROI is created, but the PXD does not register a hit, resulting in an ROI with no hit that is attached to a track.

All three cases can be at least partially disentangled with a sufficiently large data sample, but for single events it is not possible to differentiate them. In case no track is found in the first place, no ROI is created, thus even with a fully efficient detector no hits will be stored. If the track was found later in

the offline reconstruction, no PXD hit would be available to be attached to the track. However, if the track is also not found during offline reconstruction, all information of this track are lost, not only in the PXD, and, in fact, this effect can only be studied in MC, but not at all on data.

A poor track fit can be caused by wrongly attached hits in the SVD or the CDC, deteriorating the fit quality. Since DATCON does not perform a real track fit, nor does it use CDC hits, this is a general remark valid for other ROI finding algorithms, too, using a proper track fit and / or CDC hits. If the ROI resulting from a track extrapolation is too small to contain the correct PXD hit belonging to the track because of a bad track fit quality, the PXD hit is not stored and thus cannot be attached to the track even if the track finding and fitting improves with time. Similarly, if the extrapolated position on the PXD hit is far away from the correct hit, it might still be contained in the ROI, but other hits might better fit to the track during hit attachment with the To-PXD-CKF. In both cases described so far it is impossible to recognise a missing or wrong PXD hit, and thus no statement about the ROI finding efficiency is feasible.

Last, an inefficient pixel or region on the PXD can be found when often no PXD hit is close to a track that is extrapolated to the PXD in a certain region, even without any known physical cause. An inefficient region is also present if there is a known defect like damaged electronics. In this case ROIs covering the inefficient region are created, but PXD hits are never attached to tracks. Eventually this is seen as a reduced hit efficiency, but a large data set is required for this conclusion. However, usually damaged electronics usually are recognised more easily, and a reduced hit efficiency in a specific region rather indicates variations in the sensor and the silicon itself.

4.3.1 Parameter optimisation for DATCON

An optimisation of all 18 parameters conducting a grid search is not feasible due to the large number of possible combinations. Probing only three different values for each parameter, nearly 350 million simulation runs need to be conducted. As a simpler approach, for each parameter a minimum and maximum can be defined, between which values are randomly chosen for simulations. Using this random choice of parameters, $\mathcal{O}(10\,000)$ single simulation runs need to be performed to obtain a large enough and statistically meaningful dataset. However, this dataset is very likely to only provide a localisation of the optimum parameters, and the random search needs to be repeated a few times, limiting the individual parameters with each step.

The optimisation approach chosen for this thesis is different. Using the `scipy` Python library [82], a minimisation with `scipy.optimize.minimize` is performed. The library offers a wide variety of optimisation algorithms and functions. The user provides the function that needs to be optimised, and a set of starting values for the parameters the function is based on. In this case, the parameters are the DATCON parameters for the HSs and the ROI sizes, and the function is based on the ROI finding efficiency ε and the DRF, using a FOM. A set of starting values for the parameters is obtained from coarse manual optimisation.

Following the definition of the DRF in Equation (4.14) the largest value the DRF can take is unity, meaning no data reduction at all, or equally that all PXD hits are inside an ROI, and the smallest value is zero where no PXD hit at all is inside any ROI.[5] To achieve the target data reduction of at least a

[5] There is an alternative definition using the inverse of Equation (4.14). In this case the minimum value is one (no data reduction at all), and the maximum value would be infinity (no PXD hits inside any ROI). However, the definition in Equation (4.14) provides the benefit of a limited range for the DRF and is thus used in this thesis.

factor of 10 on average, the DRF following the definition in Equation (4.14) needs to be smaller than or equal to 0.1 on average.

While the ROI finding efficiency should be maximised, the DRF must be minimised and be smaller or equal to 0.1 on average, which is defined as the target value for the DRF. Two FOMs for the ROI finding performance and the combined FOM are defined as

$$\mathrm{FOM}_{\mathrm{ROI}\,\varepsilon} = \frac{1}{1 + \exp\left((0.9 - \varepsilon) \cdot 100\right)} \tag{4.15}$$

$$\mathrm{FOM}_{\mathrm{DRF}} = \exp\left(-\frac{1}{2}\frac{(\mathrm{DRF} - 0.1)^2}{(1/16)^2}\right) \tag{4.16}$$

$$\mathrm{FOM}_{\mathrm{total}} = -\mathrm{FOM}_{\mathrm{ROI}\,\varepsilon} \cdot \mathrm{FOM}_{\mathrm{DRF}}. \tag{4.17}$$

Equation (4.15) describes a broadened step-like function, resulting in $\mathrm{FOM}_{\mathrm{ROI}\,\varepsilon} = 0.5$ for an efficiency of $\varepsilon = 90\,\%$. This distribution is chosen as favours larger efficiency values because the function is strictly monotonically increasing. From previous experience, and with the combinatorial approach of ROI creation, it is known that DRF values of 0.1 or lower are difficult to achieve, and usually come at the cost of a reduced ROI finding efficiency ε. Thus a Gaussian distribution centered at the target value of 0.1 is chosen as the FOM for the DRF in Equation (4.16). It is narrowed by using $\sigma = 1/16$. This results in the combined FOM in Equation (4.17) which gives higher weight to higher finding efficiencies compared to an ever decreasing DRF.

The optimisation is performed simulating DATCON with 1 000 $\Upsilon(4S)$ events with nominal beam backgrounds. Using the current parameter set and the FOM defined in Equation (4.17), `scipy.optimize.minimize` then changes the parameters to minimise the FOM. Different optimisation algorithms were tested in this thesis, but many of which are not applicable for the current task. The main reason is that all parameters are integers, as DATCON on FPGA only can work with integer values, as described in Section 4.1. In contrast, the optimisers treat all values as floating point values, and change them in the decimal places only for optimisation, which barely has an influence on the values used for the DATCON simulation with basf2. In addition, many of the optimisers only change the parameters by a small amount, which does barely have an influence on the basf2 simulation, with no option of changing the parameter variation. While it is possible to define e.g. the vertical HS size as a floating point number and then converting it to an integer alongside a multiplication with some power of ten, non-integer cluster or ROI sizes are meaningless. To overcome this issue, the log2 of all values that are to be optimised are given to the algorithm, since the log2 of most integer values is a floating point, i.e. non-integer number. Thus the optimsation is performed on the log2 values of the parameters, while the DATCON simulation in basf2 uses the actual values. Additionally, the chosen optimiser *Nelder–Mead* [83] uses a comparably large parameter variation of 5 %.

The FOMs in Equations (4.15) and (4.16) are the result of series of optimisation attempts, many of which with similar FOMs for the individual variables that are to be optimised. Another FOM used for optimisation is the euclidian distance to the target values using the ROI finding efficiency ε and the DRF to span a 2D space. In this case the euclidian distance d is defined as

$$d = \sqrt{(1 - \varepsilon)^2 + (0.1 - \mathrm{DRF})^2}. \tag{4.18}$$

Figure 4.10 shows the results of nine different optimisation attempts for the distance d as function

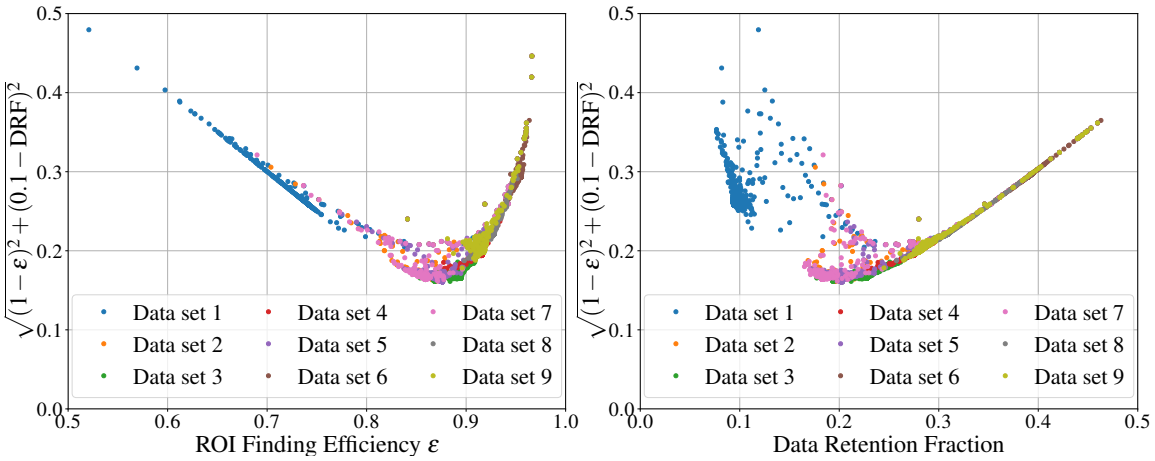


Figure 4.10: Smallest distance d to the desired working point of 100 % ROI finding efficiency and a DRF of 0.1 as defined in Equation (4.18) as function of the ROI finding efficiency (left) and DRF (right) for nine different optimisation runs using scipy.optimize.minimize. In both cases a lower bound is visible. Data set 1 indicates that a data reduction by a factor of 10 only is possible with a low efficiency, and oppositely data set 9 indicates that an efficiency of more than 90 % comes at the cost of a worse data reduction.

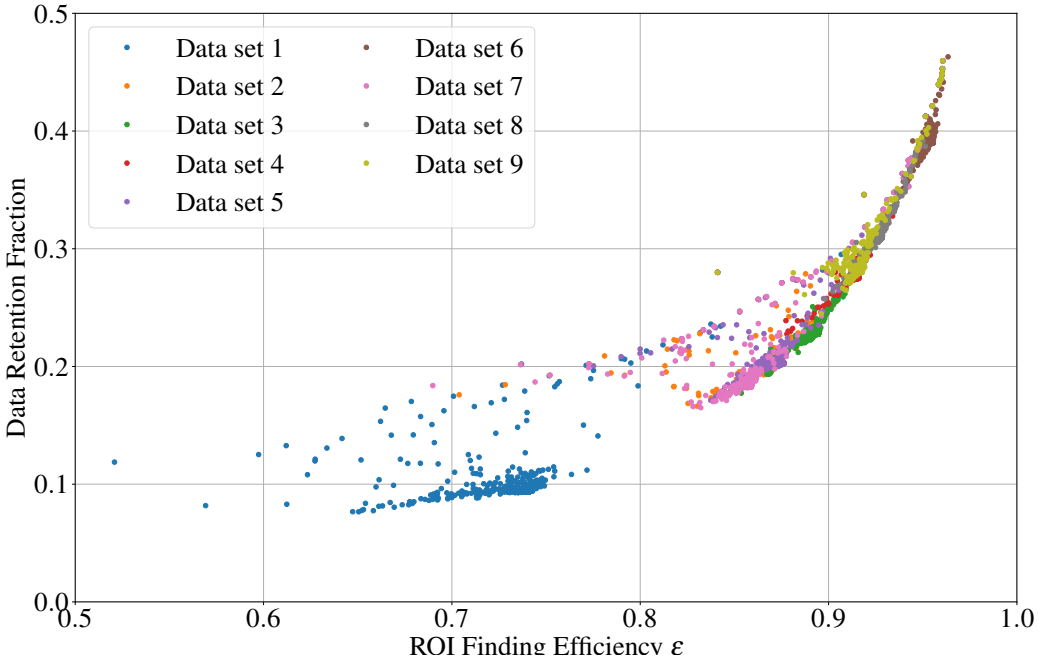


Figure 4.11: DRF as function of ROI finding efficiency for nine different optimisation runs. With the current implementation of DATCON it is not feasible to obtain an ROI finding efficiency of 90 % or more while reducing the amount of stored PXD data by a factor of 10.

of the ROI finding efficiency ε (left) and the DRF (right). Both FOM definitions from Equation (4.17) and Equation (4.18) are among the nine data sets. In both the left and the right part of Figure 4.10 a boundary is visible, indicating that with the current implementation of DATCON it is not possible to obtain both a high ROI finding efficiency ε and the desired data reduction by a factor of 10 on average simultaneously.

Similarly Figure 4.11 shows the DRF versus the ROI finding efficiency ε for each iteration of the optimisation in the nine data sets. Again a lower bound is visible. This indicates that a trade-off between ROI finding efficiency ε and data retention is to be made. Since an efficiency of less than 90 % is not acceptable, the chosen working point is the one with the lowest DRF at $\varepsilon > 90\,\%$, which is found to correspond to a data retention of 25 % or a data reduction factor of 5.27 on average.[6]

4.3.2 ROI finding evaluation

To evaluate the ROI finding performance, 50 000 simulated $\Upsilon(4S)$ events with nominal beam background are tested. Since the PXD cluster sizes vary with the z-position of the cluster on the sensor, the finding efficiency is evaluated on PXD clusters from MC particles being contained in any ROI, as noted in Equation (4.13). In contrast the DRF is calculated based on the number of single PXD digits (= pixels) inside and outside the ROIs, c.f. Equation (4.14).

In all figures showing the ROI finding performance as function of p_T, λ, or φ, a grey area indicates the distribution of the MC particle momentum parameters. In case the y-axis does not cover the full range of 0 to 1, the MC particle distributions are adapted such that the main aspects of the distribution, like higher rates in a specific region, are still fully visible. Additionally, dashed red lines in the λ figures indicate the angular acceptance region of the tracking detectors in λ from $-60°$ to $73°$, and λ values larger (smaller) than $0°$ are referred to as forward (backward), respectively. In order to set the results into context, Figure 4.12 shows the number of PXD digits per event with nominal beam backgrounds, i.e. the number of PXD pixels with a signal above threshold. The average number of PXD digits is 41623 ± 1216 per event.

Figure 4.13 shows the ROI finding efficiency as function of the transverse momentum p_T of the MC particles. While the average ROI finding efficiency is 90.1 %, it decreases significantly in the p_T region below 300 MeV/c which is the most abundant region of the phase space. The effect is more severe with lower values of transverse momentum, for example for particles with $p_\mathrm{T} < 100\,\mathrm{MeV}/c$ the ROI finding efficiency is less than 80 %. The step at around 250 MeV/c is caused by curling particles with λ being close to $0°$, as shown in the left half of Figure 4.14 and highlighted with a red circle. Tracks with p_T and λ in this region of the phase space only move very slowly along the z-axis, creating several PXD and SVD hits at different locations along z. Usually only the first two PXD clusters are contained in ROIs while missing all the others. Excluding this region from the ROI finding analysis in p_T, the efficiency distribution in the right half of Figure 4.14 is obtained. The step is significantly reduced when excluding MC particles with $p_\mathrm{T} < 250\,\mathrm{MeV}/c$ and $|\lambda| < 5°$ from the analysis as shown in orange. Figure 4.15 shows the dependence of the ROI finding efficiency of the two angles λ (left) and φ (right), again with the distribution of MC particle values in grey. Two red lines in the left part indicate the SVD acceptance region. Some structures are visible in the left part. First, the dip around $\lambda = 0°$ is caused by the same low momentum curlers that are found responsible for the step in the

[6] Since the data reduction factor in each event is the inverse of the of the DRF in each event, the mean value of the data reduction factor is not the inverse of the mean value of the DRF. The value of 5.27 for the data reduction factor is calculated by using the per-event data reduction factor.

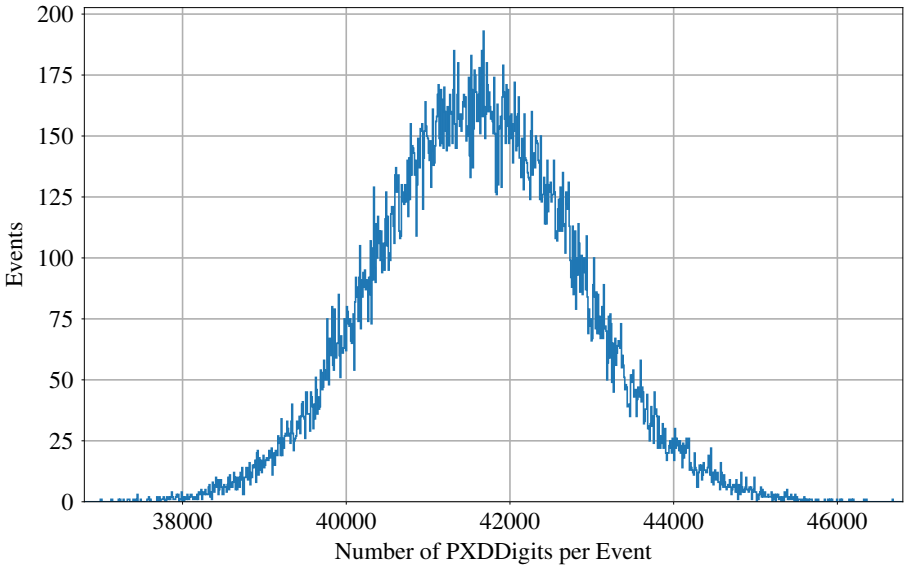


Figure 4.12: Number of PXDDigits per event. The average number of PXDDigits per event is 41623 ± 1216.

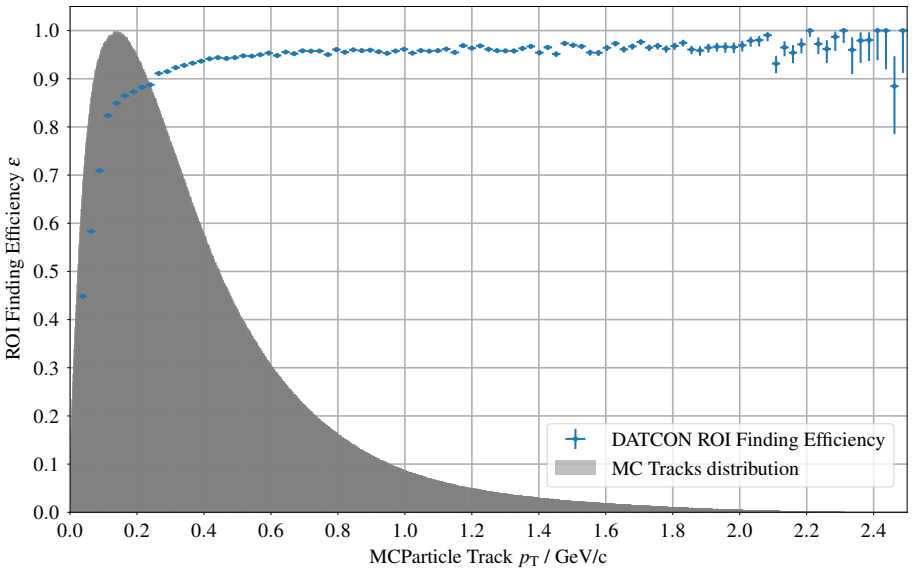


Figure 4.13: ROI finding efficiency ε as function of the transverse momentum p_T of the MC particle. The grey area marks the relative abundance of MC particle p_T values. On average the efficiency is 90.1 %, but especially in the p_T region below 300 MeV/c where the track density is the highest DATCON struggles to create ROIs containing the PXD hits of the $\Upsilon(4S)$ decay products. The cause of the step at 250 MeV/c in the efficiency distribution is discussed later in this section.

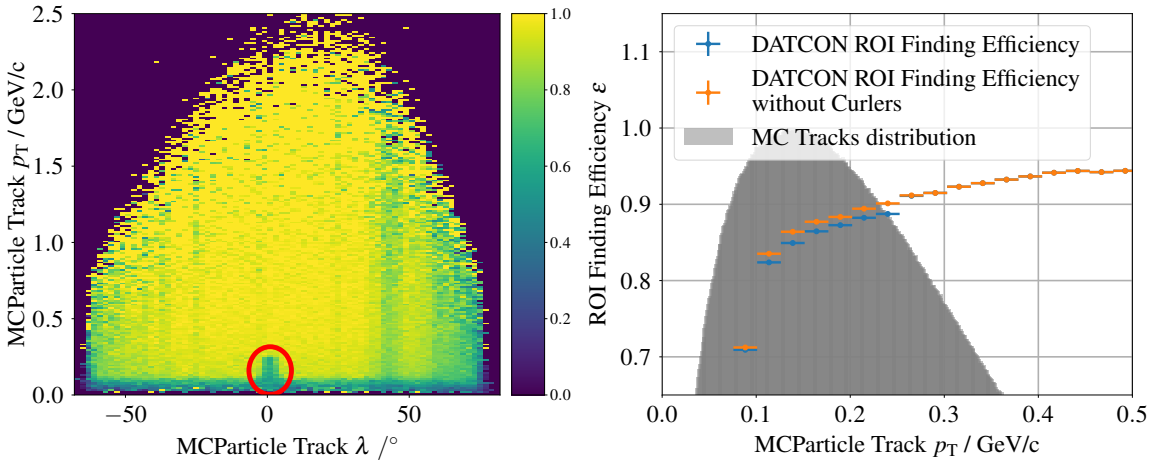


Figure 4.14: Left: ROI finding efficiency as function of λ and p_T. At low p_T values, an inefficient region is visible around $\lambda = 0°$, highlighted with the red circle. These particles are curlers that are nearly stationary in their z-position and perform multiple turns in the tracking volume since they are not able to enter the ECL. Right: Comparison of the ROI finding efficiency as function of p_T between the full sample (blue) and excluding MC particles with $p_T < 250\,\text{MeV}/c$ and $|\lambda| < 5°$. The step at $p_T = 250\,\text{MeV}/c$ is significantly reduced.

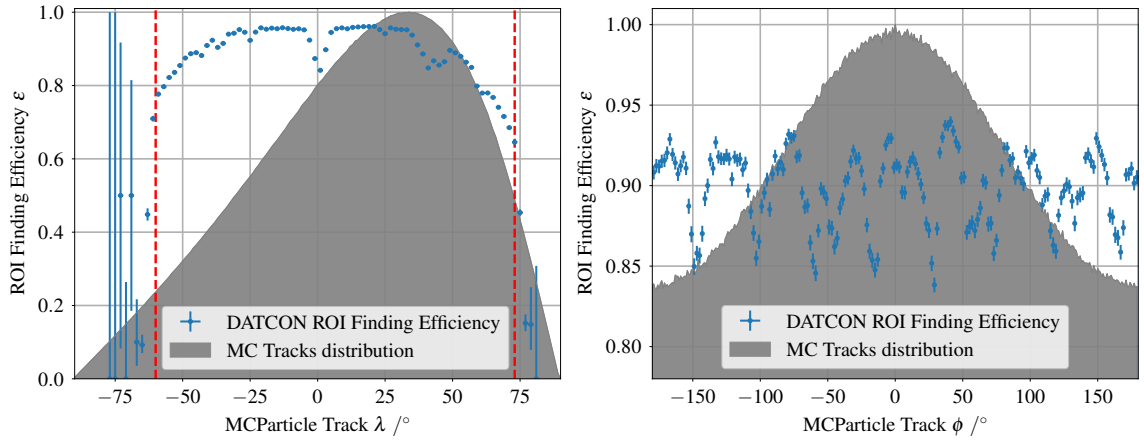


Figure 4.15: ROI finding efficiency ε as function of the dip angle λ (left) and azimuth angle φ (right) of the MC particle. The red lines in the left figure indicate the boarders of the SVD acceptance region, and the grey area marks the relative abundance of MC particle momentum values. While the ROI finding efficiency is nearly constant in both cases, it falls of in the forward region towards larger dip angles. The dip around $\lambda = 0°$ is caused by low p_T particles curling multiple times in the detector, where only ROI for the first outgoing arm are found. The structures in the right figure are caused by larger uncertainties of the extrapolation towards the upper and lower boundaries of the PXD sensors.

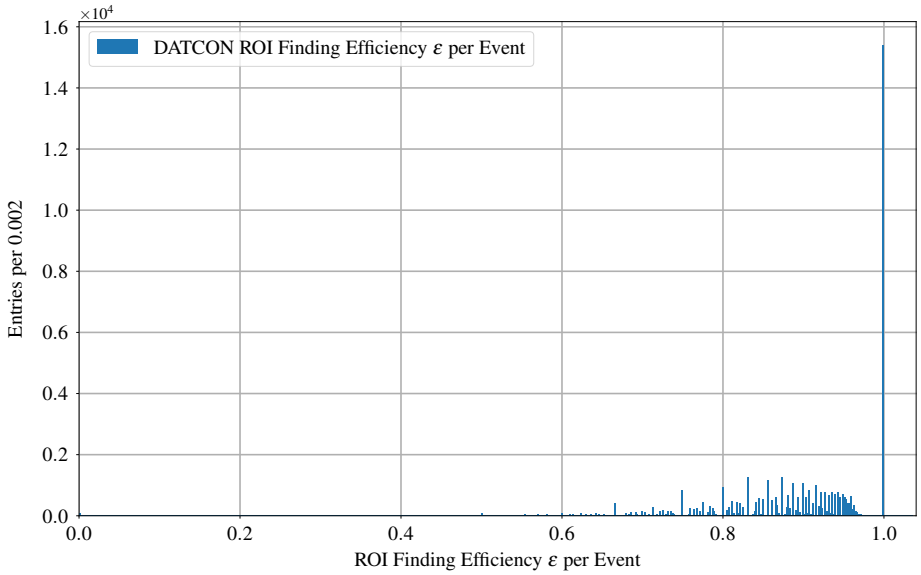


Figure 4.16: ROI finding efficiency ε in DATCON per event. All MC particle PXD clusters are contained inside ROIs in more than 30 % of the events.

p_T efficiency distribution. A second dip around $\lambda = 40°$ is caused by the insensitive region of the SVD in layer 3, where the two sensors of each ladder are glued together. It does not depend on p_T and manifests itself in a vertical band around $\lambda = 40°$ in the 2D efficiency plot in the left half of Figure 4.14. Although the insensitive region on the SVD is rather small, it is extended in the angular range because the SVD sensors are not cylindrical but planar. The further to the long sides of the ladder a particle is traversing the ladder, the larger the λ value of this hit, broadening the area where a layer 3 hit is missed. Since the requirement to find a track in the HS is to have hits from at least three layers, a hit on all layers 4, 5, and 6 needs to be present if the track passes the insensitive region on layer 3, thus reducing the chance of missing this track independent of its p_T.

In addition, the ROI finding efficiency decreases for forward and backward tracks with $|\lambda| > 45°$. This is caused by the extrapolation in λ using straight lines, although the tracks traversing on a helix having a sinusoidal projection in the x-z and y-z planes. Thus the error of the extrapolation gets larger with larger absolute values of the dip angle λ, reducing the ROI finding efficiency. Since the extrapolation in r-φ has larger uncertainties the closer the extrapolated hit is to the maximum / minimum u-coordinates on each sensor, the ROI finding efficiency drops in certain regions in φ. The eight distinct dips in the distribution in the right part of Figure 4.15 is dominated by missing clusters on the first PXD layer, corresponding to the regions where the sensors overlap. As a result of the optimisation, the ROI size in u-direction for layer 1 only is 49 pixels, while it is 68 pixels for layer 2. Thus, more PXD clusters are contained in ROI on layer 2. Further optimisation with e.g. a larger ROI size in u-direction on layer 1 could help to mitigate these drops.

Figure 4.16 shows the ROI finding efficiency per event. In more than 30 % of all events all MC particle PXD clusters are contained inside the ROIs, but in 13.4 % of all events less than 80 % of the PXD clusters are contained in ROIs. The discrete structure that is visible is caused by the small discrete number of PXD hits. If n PXD hits need to be contained in ROIs per event, the only options

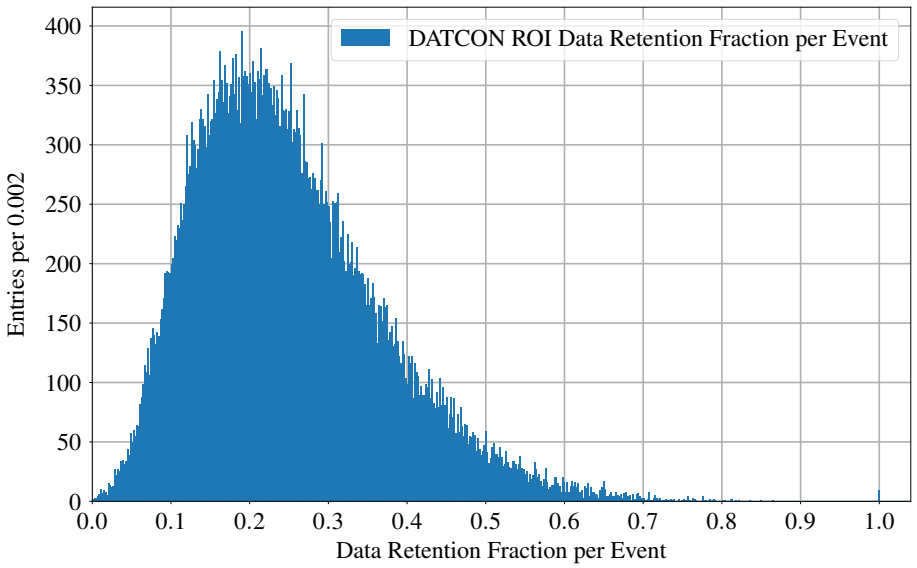


Figure 4.17: Distribution of the DRF per event. The target DRF of 10 % or less only is achieved in a small fraction of events.

for the ROI finding efficiency per event are $n/n = 1$, $n/(n-1)$, $n/(n-2)$, etc. As n is about 22 on average, since there are on average about 11 tracks per event in the acceptance region, the gap between the best possible ROI finding efficiency per event of 1 and the next lower value can be explained.

Figure 4.17 presents the DRF per event. With the current implementation of DATCON the average DRF is 0.25, equivalent in a data reduction by a factor of four. Thus, the achieved DRF with DATCON is much larger than the required value of 0.1.

The reason for this can be found in Figure 4.18 which shows the number of ROIs per event. On average about 163.76 ROIs are found per event, roughly half of which are on layer 1 and layer 2 each. This value is still to large for ONSEN, since at the time of writing, the requirement of ONSEN is that there are at maximum 128 ROI per event from DATCON and HLT combined. Thus, further optimisation for DATCON is required to reduce the number of ROI, and to improve both the efficiency as well as the data reduction.

Finally, in Figure 4.19 the combination of DRF and ROI finding efficiency for each event is shown. Only a small fraction of events fulfills the requirements of a high efficiency and low data retention, which is indicated by the red rectangle in the bottom right corner. In many events the efficiency is sufficiently high, but the fraction of PXD hits contained in the ROI is too large, resulting in a data retention fraction of 20 to 30 %.

4.3.3 Comparison of merged and unmerged single-side ROIs

To investigate the effect of the single-side ROI merging, the merging is switched off and the simulation is repeated on the same data set. Without merging on average 30 530 ROI are created per event, as shown by the orange distribution in Figure 4.20, represented by the x-axis at the top of the figure. This corresponds to a reduction of the average number of ROI per event by 99.5 %. For comparison, the distribution for the number of ROI per event with merging is shown in blue (bottom x-axis). While

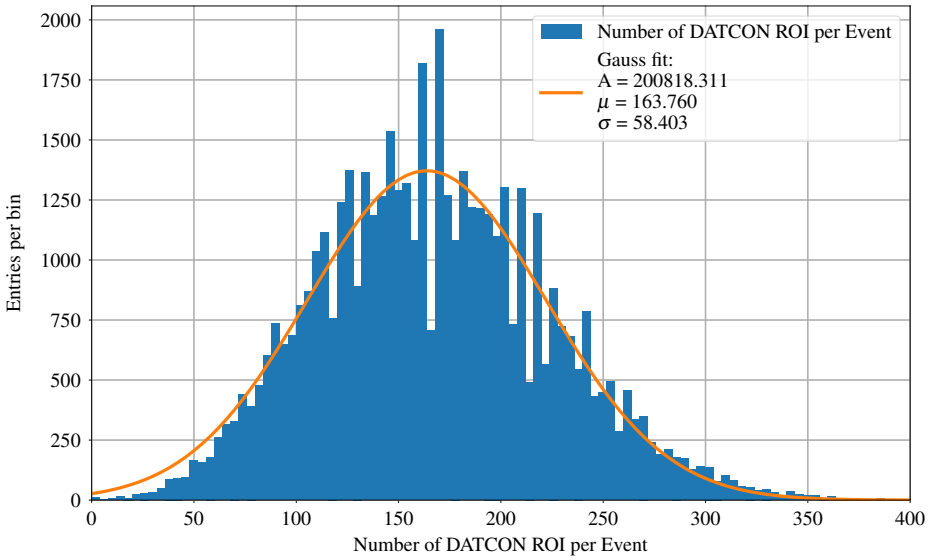


Figure 4.18: Number of ROIs per event for 50 000 $\Upsilon(4S)$ events with nominal beam backgrounds. On average 163.76 ROIs are calculated per event after single-side ROI merging.

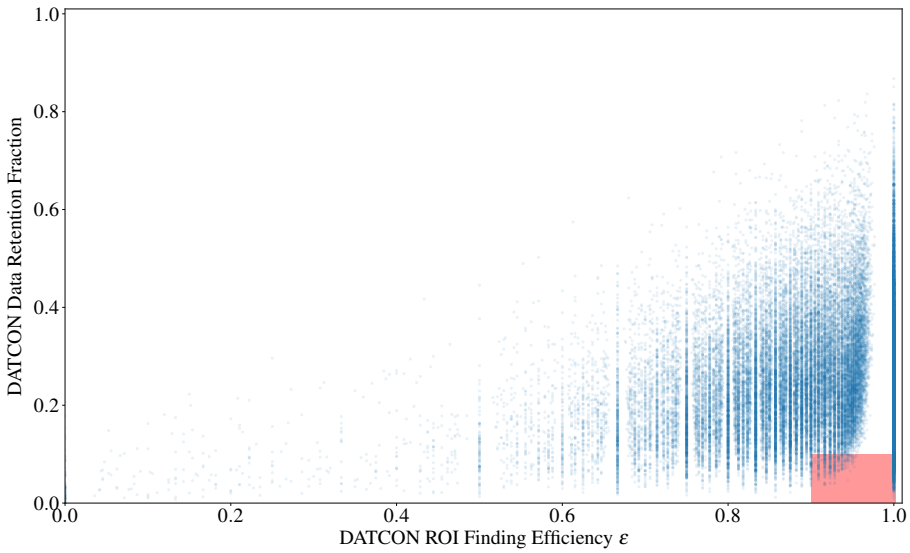


Figure 4.19: DRF vs. ROI finding efficiency ε per event. The target region with an efficiency of more than 90 % and a data retention of about 10 % is indicated by the red rectangle in the bottom right corner. Only a fraction of events fulfills the requirement of high efficiency and low data retention.

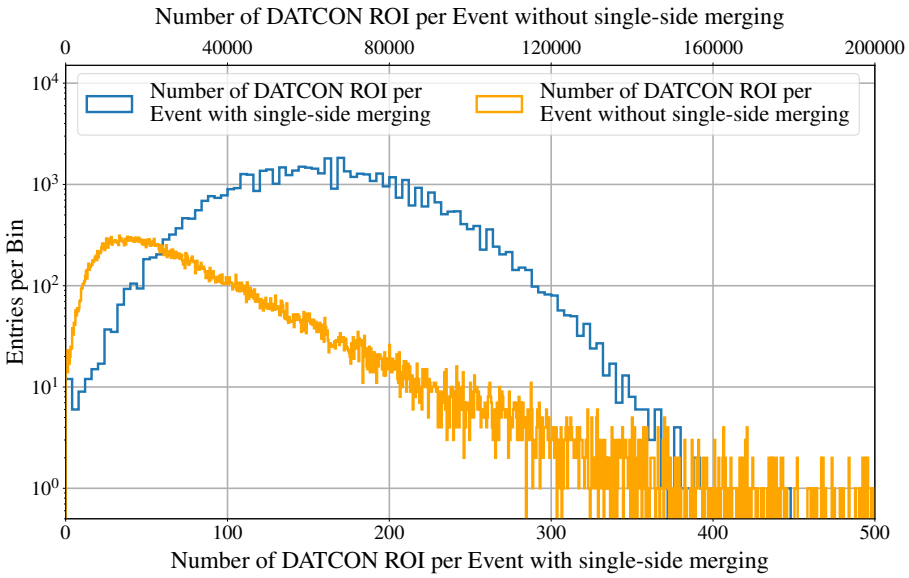


Figure 4.20: Comparison of the number of ROIs per event if the single side ROIs are merged (blue, bottom x-axis and in case no single-side ROI merging is applied (orange, top x-axis). Without single-side ROI merging, on average 30 530 ROI are created per event, and with merging 163.76 ROI are created per event (c.f. Figure 4.18).

the maximum of the distribution is located at around 15 000 ROI per event, there is a long tail towards higher numbers and even more than 200 000 ROI per event are possible. In addition, the size of each final ROI is different between the two simulations with and without merging. The corresponding distributions are shown in Figure 4.21 for the u-direction (v-direction) on the left (right), and again in blue with merging and in orange without merging. Without merging two distinct maxima are present in both figures, corresponding to the different ROI sizes in u and v for both PXD layers. In addition there are smaller values from ROIs close to the sensor boundaries where the size is limited by the sensor edge. With merging the ROIstend to be much larger, often ranging over the full width in u-direction, resulting in an average size of 118.30 pixels in u-direction and 97.02 pixels in v-direction.

4.4 Comparison with ROI finding on the HLT

Since both DATCON and the HLT are used for ROI finding online, their performance needs to be compared, and the benefits of using both systems need to be evaluated. Thus, in this section first a comparison of the ROI finding performance in terms of ROI finding efficiency and data retention is presented, followed by an evaluation of the combined performance.

Figures 4.22 to 4.23 show the ROI finding efficiency of DATCON and the HLT algorithms in comparison, as functions of p_T and λ. Overall 89.9 % of PXD clusters are contained in DATCON ROI, while the HLT finds 91.8 % of PXD clusters. While the HLT performance is better over most of the p_T range, DATCON is performing comparably well. For λ the situation is different. DATCON performs better in the central part, with both algorithms showing a reduced efficiency around $\lambda = 0°$ due to curling low p_T particles, c.f. Figure 4.14. However, DATCON falls short compared to the HLT in the very forward and backward regions where the HLT ROI finding is clearly superior due to its

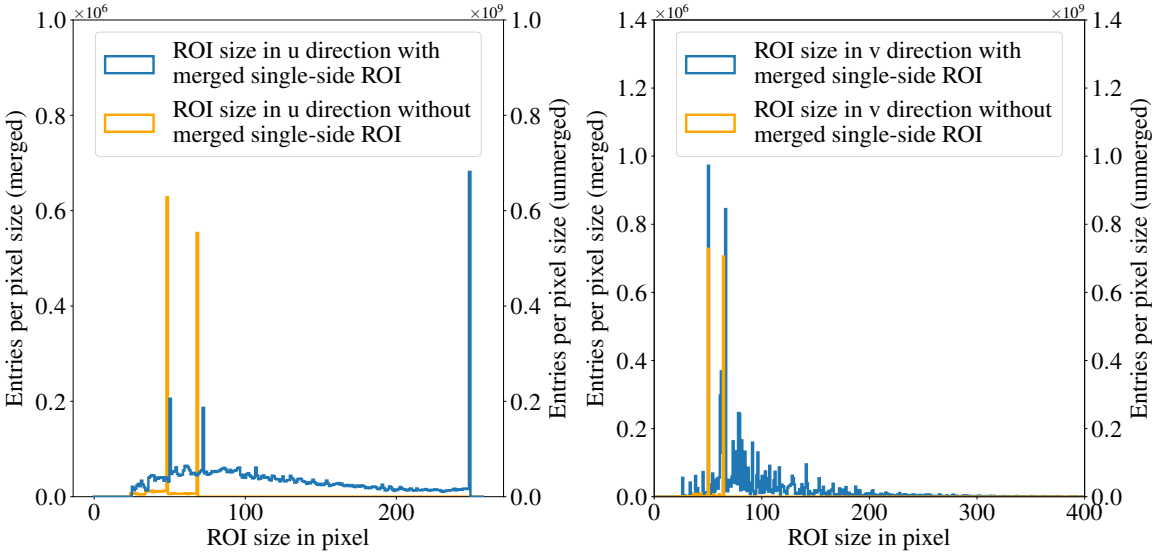


Figure 4.21: Comparison of the ROI sizes in u-direction (left) and v-direction (right). In both figures the left y-axis is for the case with, and the right y-axis for the case without single-side ROI merging. If the ROI are not merge, the ROI size is at maximum the size estimated from optimisation. With merging, a significant amount of ROI has a size in u which is the same as the sensor size, indicating that all pixels in u-direction are stored if they are also contained in an ROI in v-direction. In contrast, in v-direction there are few ROI larger than 200 pixels after merging.

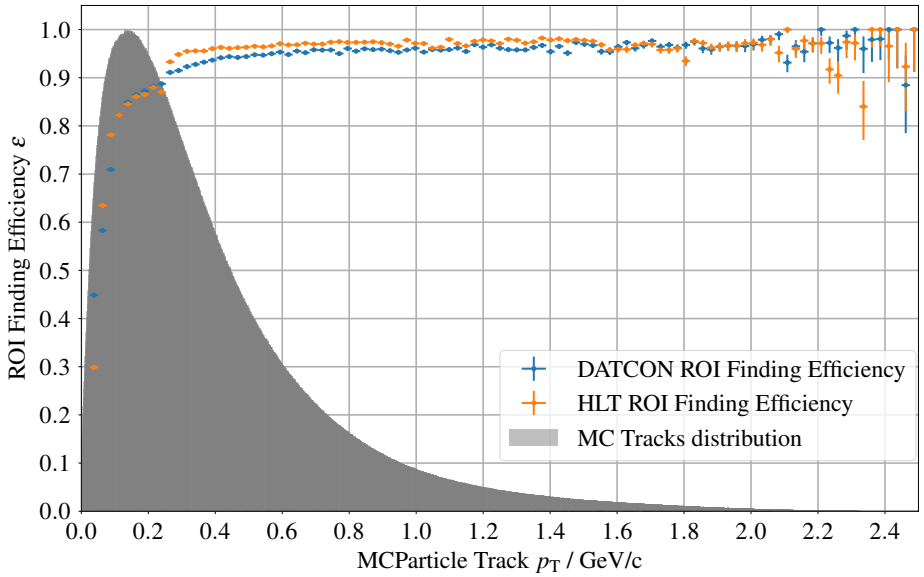


Figure 4.22: Comparison of the ROI finding efficiency as function of p_T of DATCON (blue) and HLT (orange). For most of the p_T range, the performance of the HLT ROI finding is superior to that of DATCON. However, although the efficiencies are comparably high, the DRF has to be considered in the comparison, as presented in Figure 4.24.

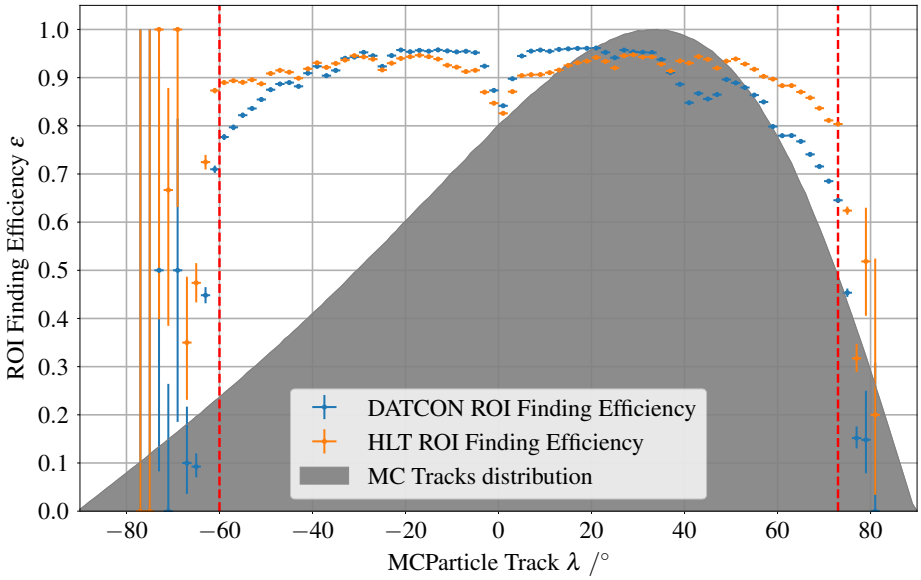


Figure 4.23: Comparison of the ROI finding efficiency as function of λ of DATCON (blue) and HLT (orange). The two vertical red lines indicate the acceptance region of the detector. While the performance of DATCON is slightly better in the central part, the HLT performs better in the forward and backward regions. In both cases a dip around $\lambda = 0°$ is visible caused by curling tracks that create multiple hits on the PXD of which only a fraction is contained in ROIs. Additionally, both distributions fall off towards high and low values of λ. This is partially caused by curling low p_T tracks where the PXD hits of the ingoing arm are not found, and by large extrapolation uncertainties of DATCON as introduced in Equation (4.7) and is depicted in Figure 4.5.

better track reconstruction and track modelling for ROI creation. The fall-off of the HLT ROI finding efficiency in the forward region ($\lambda > 60°$) is caused by low p_T curling particles where the part of the track returning to the beam axis is not reconstructed, and thus ROIs are missing for this. However, this efficiency cannot lead to a final conclusion alone, but only comparing the DRF at the same time.

Figure 4.24 shows the DRF of DATCON and HLT ROI finding. While the average DRF for DATCON is 0.25 again, it is clearly visible that the HLT ROI finding reduces the number of PXD hits significantly with an average DRF of less than 1 %.

4.5 Combined HLT and DATCON ROI finding performance

After comparing the individual performance, the combination of DATCON and HLT ROI finding needs to be evaluated. On average 98 % of all PXD clusters are contained in ROIs, as shown in Figures 4.25 to 4.26. Above transverse momenta of about 250 MeV/c the efficiency is close to 100 %. The degradation at lower p_T values is caused by curling particles around λ=0° and in the forward and backward regions in the angular acceptance. This coincides with reduced track finding efficiency in the HLT, as presented in Chapter 5. Thus, to recover PXD hits the track finding in the forward and backward directions needs to be improved.

As expected, the overall DRF, depicted in Figure 4.27, is dominated by the DRF from DATCON. It is slightly worse compared to only using DATCON as the HLT finds additional ROI which improves the finding efficiency but further degrades the data reduction performance.

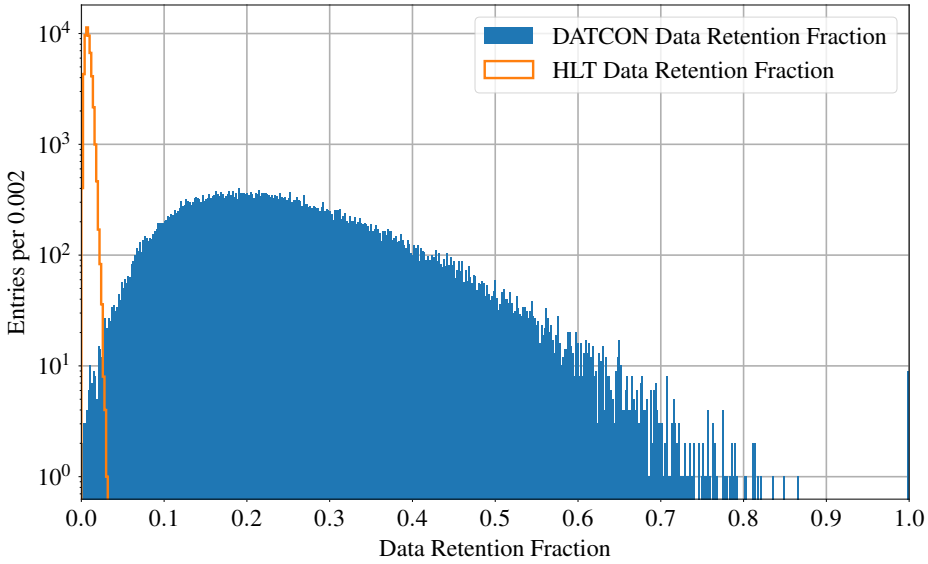


Figure 4.24: Data retention fraction per event for both DATCON and HLT ROI creation. While the average data retention fraction is 25 % for DATCON with a wide span, it is around 1 % for the HLT with a much narrower distribution of the individual values.

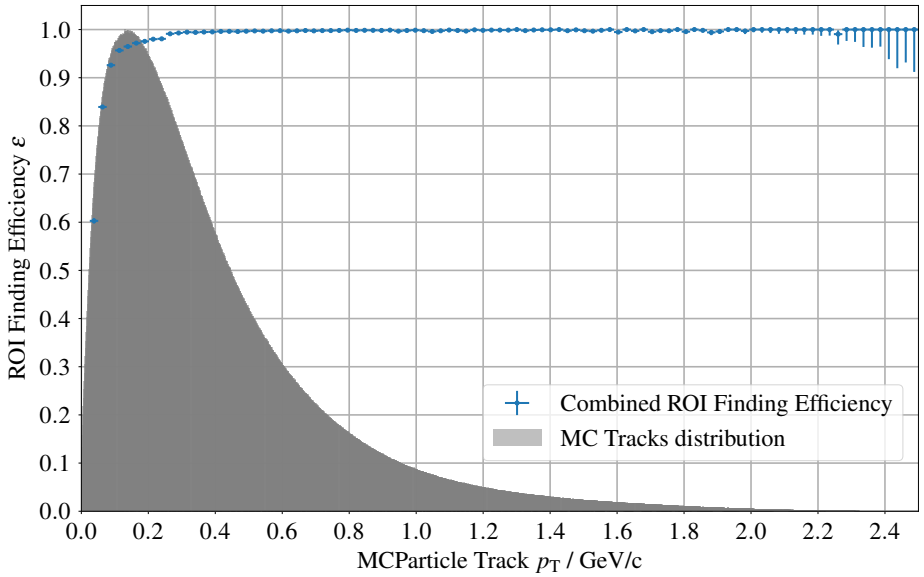


Figure 4.25: Combined ROI finding efficiency of DATCON and HLT as function of p_T, representing the setup used for data recording at KEK. The finding efficiency is close to 100 % over most of the p_T range, only falling off below 250 MeV/c, caused by curling particles.

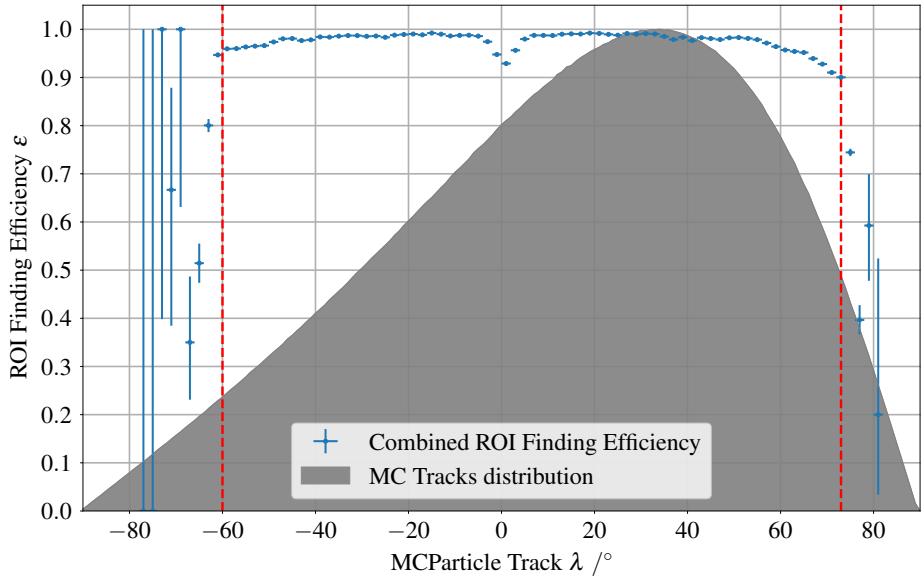


Figure 4.26: Combined ROI finding efficiency of DATCON and HLT as function of λ, representing the setup used for data recording at KEK with the red lines indicating the angular acceptance range of the tracking volume. The finding efficiency is above 90 % over the full λ range, with dips at 0° and in the forward and backward regions, caused by curling particles.

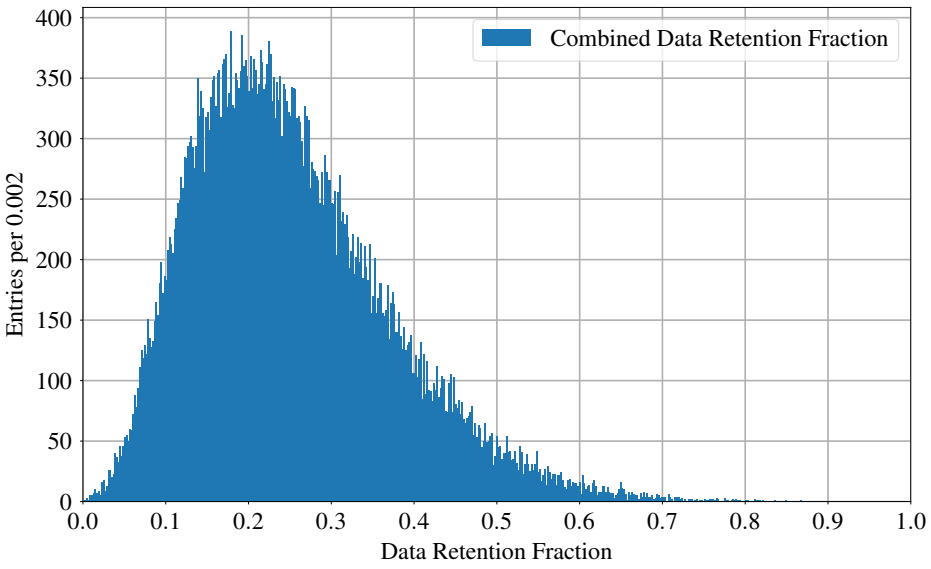


Figure 4.27: Combined DRF using DATCON and HLT ROI finding, representing the setup used for data recording at KEK. As expected, the combined DRF is very similar to that of using DATCON alone, as the HLT DRF only is about 1 %.

4.6 Summary and outlook for DATCON

Based on the simulation results, the current implementation of DATCON cannot suffice the ROI finding and data reduction requirements. Hence improvements are necessary, and some proposals for which are presented in this section. As the focus of this thesis is on simulation with basf2, likely not all of the proposals are possible to be implemented on FPGA. This may be because of general features of FPGAs, or because of the limited FPGA resources available for DATCON due to the aged hardware.

The most limiting feature of the current implementation is the combinatorics during ROI creation. Even when merging overlapping single-side ROI first before creating the final ROIs, more than 160 ROI are created per event on average, the average size of which is 118×96 pixels. But also ROIs covering complete PXD sensors are obtained, which is not ideal and needs to be avoided.

Compared to the ROI finding performance and DRF presented in [3] the ROI finding efficiency is higher by 2 % due to the chosen final working point in this study. In contrast the DRF is much worse. While the DRF in this work is 25 %, the average data reduction factor is 5.27 ± 2.76, which is less than half of the data reduction factor found in [3]. There are two main reasons for this. First, as mentioned before, in the previous work the track candidate lists from the two HSs were compared to create track objects with matching hits that are extrapolated to the PXD. This is no longer done in this work as it is not implemented on the FPGA. Combining the individual track lists based on the hit content not only reduced the problem of combinatorics, but it also enabled the usage of the y coordinate from the extrapolation in r-φ in the extrapolation in z, avoiding the extrapolation error described by Equation (4.7) and shown in Figure 4.5. But as the error of the extrapolation in z as function of the local y coordinate in the rotated coordinate system can be calculated in advance, this can in principle be corrected for during the combination of single-side ROI. However, this is difficult as the single-side ROIs are merged first. Afterwards it is not possible anymore to define the correction, as a different correction is necessary for each of the ROIs before merging. Thus, a reduction of both the combinatorics and using information from both HSs for extrapolation will improve the DRF and likely also the ROI finding efficiency. However, as shown in Section 4.3.1, a careful optimisation of all parameters needs to be performed.

Second, the understanding of the beam induced backgrounds improved over the last five years since the creation of [3], leading to a more accurate simulation of the backgrounds. With the beam induced backgrounds being underestimated in the simulations available during the work conducted for [3] the occupancy in both PXD and SVD was much lower. This lead to less sinusoidals and subsequently clusters in the two HSs and thus less track candidates even before combining the two track candidate lists, reducing the number of ROI and yielding a better data reduction.

However, there are several options for future improvements. First of all, creating actual 3D hits comparable to SpacePoints from information from both sides of the SVD can be considered, using additional timing information of the hits. While a fit of the function described in Equation (4.1) likely is not possible with a deterministic run time, a small neural network might be able to fulfill this task. More important is to use the 3D hit information in both HSs and compare which clusters consists of which hits. As already shown in [2, 3], using lists of hits for each track candidate originating from the two HSs before constructing the ROIs can significantly reduce the number of ROIs. A simplified check based on the sensor numbers instead of the actual hit content might be sufficient. In addition, using a pointer-like structure would be beneficial in general, as in this case only the pointers to the hit information need to be stored, but not the full hit information. In each reconstruction the hit information can be retrieved from memory via the pointers, instead of passing large chunks of data

between different parts of the FPGA firmware directly. Even an simple track fit might be possible in this case.

If SpacePoint-like 3D hits are available, another option is to only construct one HS instead of two. This approach is chosen for the newly developed SVD Hough Tracking, which is presented in Chapter 5. Combining all information from two HSs is not necessary in this approach, reducing the combinatorics problem to only checking which hits inside each HS cluster (= track candidate) can actually represent a track based on the hit coordinates. However, this requires additional selection on the hits in each HS cluster e.g. by using a neural network or another type of MVA.

Last, DATCON could be focused entirely on low p_T particles since the HLT is proven to find ROI for particles with high p_T reliably. In this scenario, the HS would not cover $-\rho$ to ρ, but only two horizontal bands in the φ HS enabling the reconstruction of tracks with e.g. $p_T < 225\,\mathrm{MeV}/c$, corresponding to $|\rho| > 0.02\,\mathrm{cm}^{-1}$. Combining this with the aforementioned improvements using 3D hit and timing information, DATCON can focus entirely on tracks (and thus ROI) that cannot or only hardly be found by the HLT tracking algorithms.

An additional emphasis can be made for / laid on *slow pions* from $D^{*\pm}$ decays, as they have a very low p_T. Because of their low energy and momentum, their energy loss in the SVD sensors is comparably large, even considering the statistical fluctuations of energy loss, thus using information on the deposited charge in the SVD it might be possible to reconstruct them reliably with hit energy measurements.

CHAPTER 5

Full Hough Transformation based tracking with SVD data

As shown in Chapter 4, the FPGA algorithms of DATCON are only able to provide a high ROI finding efficiency at the expense of the DRF. Several improvements to the algorithms used in DATCON are possible, but are not implemented on the FPGA so far due to time and resource limitations. Some of the improvements are briefly introduced in this chapter. This is followed by the description of a new track finding algorithm for Belle II based on the Hough transformation developed in this work, called *SVDHoughTracking*. To estimate the performance of the new tracking algorithm, it is compared to the VXDTF2 as the current SVD standalone tracking algorithm, and the full tracking chain, in terms of both track and ROI finding performance. Afterwards studies for ROI finding with different tracking algorithms are presented and the impact of adding PXD hits to tracks with the To-PXD-CKF is studied. This is followed by studies on track and ROI finding performance with increased beam backgrounds. Finally, a study on τ-pair events with data recorded by Belle II in 2021 is conducted.

5.1 Intermediate improvements of the DATCON basf2 implementation

Based on the work in [3], the algorithms for DATCON were further improved during the scope of this work. While the HT as the base remained unchanged, several new features were implemented and tested, many replacing previous algorithms. They are presented in the following.

Extension of the φ HS to cover 360° instead of 180° In the previous works on DATCON, both HSs were limited to a total range of 180°, and a HS cell could be *activated* when crossed by lines with both positive and negative slope. For the HS to find tracks in λ this does not pose any issue, as the total acceptance in λ only is 133° anyway. Requesting the slope of all sinusoidals crossing a sector to be the same, in this case to be positive, significantly decreased the number of tracks in the λ HS, as random combinations of curves with positive and negative slopes were avoided by this change. However, requesting all lines to have same slope, but limiting the φ HS to a range of 180° requires to keep track of the number of tracks with both a positive and negative slope, and the SVD layer the hit creating each line is on. This can be avoided by extending the φ HS to cover 360°, and only accepting sinusoidals with a positive slope when checking for the line to pass the sector. As described in Chapter 4, this approach only yields outgoing arms of the tracks (after removing random

combinations), but introduces the potential problem that a curling track could be found twice. Overall, this change significantly reduced the number of random combinations and also the number of clusters in the HS, and is used in both the current version of DATCON and the new SVD Hough Tracking algorithm.

Development of the clustering algorithm based on a depth first search This algorithm is already introduced and briefly described in Section 4.1, and is one of the few algorithms present in the DATCON implementation, the intermediate improvements, and the final SVD Hough Tracking. Previous to [3] and this work, a clustering algorithm for the HS existed [2], but was not used. Instead all active sectors were saved along with a list of hits which sinusoidal curves cross a given sector. Afterwards these lists were compared, and only if the hit lists of two active sectors were equal, which often was not the case, the lists were merged to reduce the number of track candidates. As the hits associated to each sector were kept track of, in the end these hit lists were also used to combine track candidates from the r-φ HS and the λ HS.[1] Since the clustering algorithm employing the depth first search algorithm is proved to be very reliable and efficient, and also easy to implement on FPGA, it was kept for both the basf2 and FPGA implementation of DATCON.

Implementation of a straight line fit to remove outliers Since a track resembles a straight line in both the r-z-plane and in the conformal space, a straight line fit can be performed to the data. Using this, fit outliers can be detected and removed by calculating the variance of the fit and removing all hits above a certain residual threshold in the next iteration. Repeating this process several times, each time lowering the threshold, efficiently removes most of the outliers, eliminating many of the track candidates with only random hits. But it also removes some signal hits from $\Upsilon(4S)$ decay products, especially in the r-z-plane as the track is only approximated as a straight line in this case, while the true trajectory is a sinusoidal in this projection. In many cases outliers are located on the same sensor, e.g. ghost hits on the SVD where several hits with the same x', y', but with different z values on the same sensor, c.f. Figure 2.7. In these cases wrong hits can be removed, and similarly for the fit of the z-r data where wrong hits in r-φ can be removed. As final results, more accurate information on φ, ρ, and λ are obtained compared to retrieving information just from the HS cluster CoG. Nonetheless, the value of the curvature ρ often still is rather inaccurate, as it the values retrieved from the conformal transformation assign an unproportional high weight to the hits on the innermost layers in the fit, while lowering the weight of the hits on the outermost layers.

ROI calculation in the Conformal Space Not only is it possible to fit the tracks in the conformal space, despite the uncertainties introduced by this, but it is also possible to extrapolate the tracks to the conformal mapped PXD sensors. The values obtained for ρ and φ can be used directly to extrapolate to the PXD. Although the conformal transformation of the VXD sensors yields flower-like structures in the conformal space, a straight line extrapolation to a sensor being represented as a straight line, too, instead of a curved line, is possible, reducing the computational cost of the extrapolation. While this approach provides good ROI without any background, adding background often deteriorates the track fit because actual track hits are removed but not background hits, eventually deteriorating the

[1] In previous works the HSs were either referred to as p/n for the different dopings of the SVD strips on the two different sides, u/v for the local coordinates, or r-φ/θ for the coordinates.

ROI finding efficiency.

Several of the developments described above showed good potential for improvements of DATCON, while at the same time it was not clear whether it would be possible to implement them on the FPGAs available for DATCON. Neither the ROI finding using the full tracking chain, as on the HLT, nor using the VXDTF2 alone for track finding and ROI creation yields 100 % efficiency, leading to inevitable loss of PXD data. While the main cause are inefficiencies in the track reconstruction, future improvements to the track finding algorithms could increase both the track and ROI finding efficiencies. However, all PXD hits that are not contained in a ROI are lost and thus cannot be used with newly found tracks from improved track finding, if they are not saved by other methods, which leads to a degradation of the vertex resolution of these tracks.

Thus, a new track finding algorithm for Belle II has been developed during the course of this work to potentially improve both the ROI finding and track finding performance. It is described in the following. Since the Hough transformation ansatz shows its potential with DATCON already, and is a well established method for track finding in particle physics in general, it is chosen as the basis for the new algorithm. Because this newly developed algorithm is meant to be used within basf2, but not to be ported to FPGA, there is no need to keep the limits of the FPGA. Its goals are

- track finding performance comparable to the VXDTF2 in terms of
  - track finding efficiency
  - fake and clone rate
  - hit efficiency and purity
- fast execution time
- low memory footprint
- tolerance to high background conditions

While a good track finding performance is an obvious goal for a track finding algorithm, the fast execution time and low memory footprint are of particular importance, too, as the new tracking algorithm might be used on the HLT in the future. As described in Section 2.5.3, the required nominal throughput rate for the HLT is 20 kHz, but the HLT is required to be able to cope with rates up to 30 kHz. This can only be achieved if all the necessary reconstruction algorithms are optimised for fast execution, but also for low RAM usage as all processes on the HLT share resources.

5.2 Implementation of the SVDHoughTracking

As outlined in Section 2.8, basf2 is based on single modules that execute specific code and interact with data from the DataStore. The design of the new SVDHoughTracking is based on *findlets*. Findlets are similar to modules in that each findlet fulfils a specific task. They can be combined in modules freely, such that a findlet can be used in several modules, or can be easily replaced by a different findlet within the same module. In addition, findlets can be nested with one findlet containing and executing several other findlets. This makes the code very flexible and easily expandable for future

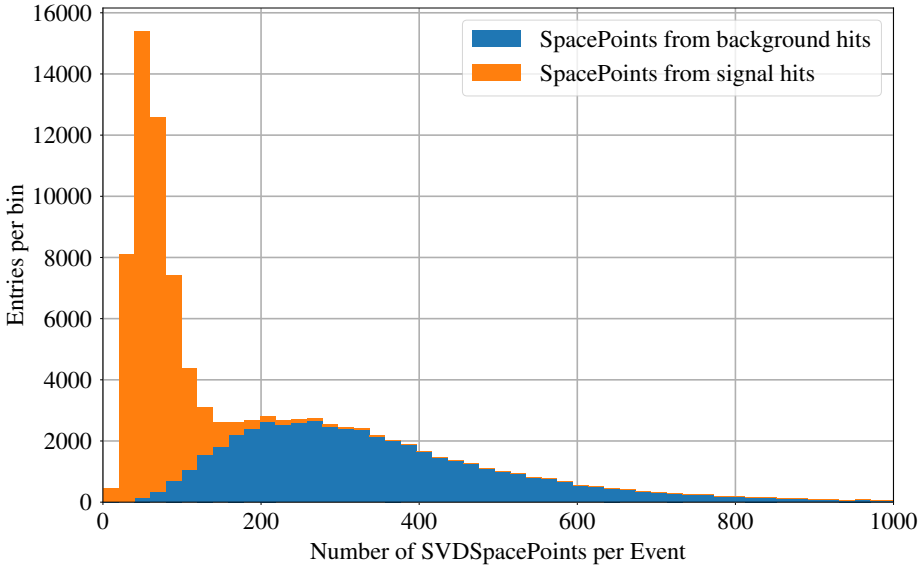


Figure 5.1: Number of SVD space points per $B\bar{B}$ event with beam induced background. On average 420 space points are created per event, while only about 60 space points are expected to originate from signal tracks. The remaining space points stem from hits from beam induced backgrounds.

developments. One findlet acts as the *FindletModule*, which contains all other findlets and is seen as a regular module by basf2.

The second important design pattern is a generic *filter* class. Filters are used to decide for a list of hits or tracks which hits or tracks (or track candidates) to retain and process further, and which of them to discard. Since they are a generic type, they can also easily be exchanged, for example via a different set of parameters in the steering file, to test and use different filters in different situations.

In contrast to the FPGA based DATCON, which uses a custom reconstruction of the SVD hit information, the SVDHoughTracking uses SVD space points based on the full SVD reconstruction. As described previously, each SVD space point provides 3D information about the hits and consists of two clusters, one for u-direction and one for v-direction each. The distribution of the number of SVD space points per $B\bar{B}$ event is shown in Figure 5.1. On average 420 space points are created per event, while only about 60 space points are expected to originate from the $B\bar{B}$ decays. All remaining space points stem from beam induced backgrounds and outnumber the space points produced by $\Upsilon(4S)$ daughters by a factor of almost ten. As the regular SVD reconstruction is used, each improvement to the reconstruction algorithms, for instance for background reduction or improvement of position resolution, will be directly available for the tracking with the new algorithm. To better cope with the expected high occupancy in the SVD, only one HS for finding track candidates is used, for which the r-φ HS is chosen. There are two reasons for this choice. First, using two HSs, the information of both HSs would need to be combined later on, potentially leading to combinatorial problems similar to those for the DATCON. This simplifies the identification of track candidates with a higher initial purity, reducing the effort needed to filter the correct hits for the final tracks from the initial track candidates. Additionally, all 3D information are available using space points, such that tracks can be fully reconstructed from only one HS and employing further processing of the track candidates.

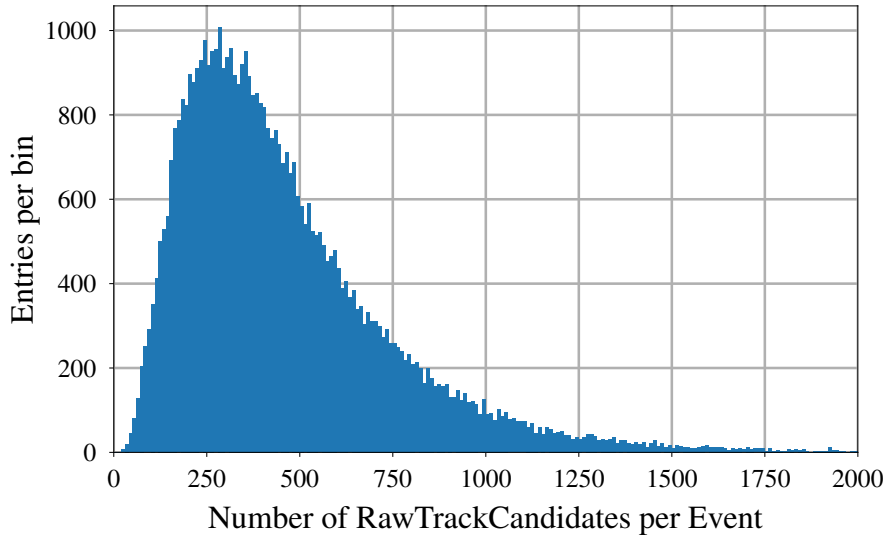


Figure 5.2: Number of RTCs per event obtained from the HS. Each cluster found in the HS yields one RTC that is further processed. On average 460 RTCs are found in each event, with the median at around 390 RTCs and the peak value at 300. Compared to on average eleven actual tracks in a $\Upsilon(4S)$ event, the number of RTCs is larger by a factor 40 on average.

Although on average there are more SVD space points than SVD clusters from the custom SVD reconstruction with DATCON, the execution time benefits from using additional information of the hits, and from evaluating only one HS instead of two.

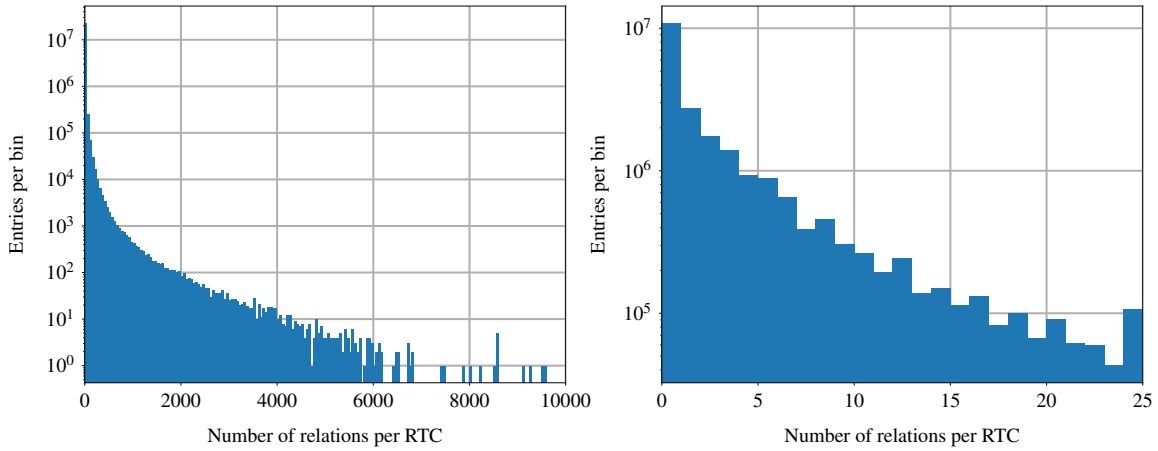
Evaluation of the Hough Space and creation of Raw Track Candidates After performing the conformal transformation on all hits, the sinusoidal curves are calculated and the HS is evaluated. First, the fast HT is used to identify active sectors which are crossed by sinusoidals from hits from at least three different SVD layers. These sectors are subsequently used to find HS clusters using the depth first search algorithm. On average about 460 clusters are present, each corresponding to one Raw Track Candidate (RTC), as shown in Figure 5.2, the median and peak values are 390 and 300, respectively. Each RTC contains a list of the hits that correspond to the sinusoidals passing the sectors contributing to it. While the maximum of the distribution is at 300 RTCs per event, there is a long tail towards higher values, and in some rare cases more than 1000 RTCs are created in one event. Only about 10 of the RTCs stem from a real physics track, often still containing wrong hits e.g. from beam induced backgrounds. All others either originate from tracks from beam induced backgrounds, or are just created by random combinations of SVD hits. In the following all purely random combinations of hits have to be removed from the set of RTCs, and for all RTCs representing actual tracks the wrong hits have to be removed from the list of hits.

Relation creation within the Raw Track Candidates In the next step relations between the hits of each RTC are created if

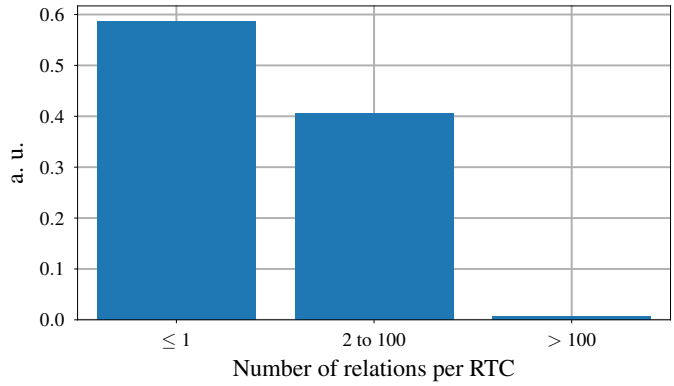
- The hits are on the same layer but on neighbouring ladders to account for the overlap regions in the SVD.

- The hits are on different layers and pass simple and loose cuts on the difference in λ and the difference in time between the u and v clusters of the two hits. It is possible to skip one layer.

It is necessary that one layer is allowed to be skipped because of the insensitive regions between the single SVD sensors, and because of possible inefficient or dead regions, for instance due to malfunctional readout electronics or damage to the silicon itself. The cut values in λ are different depending on the difference in layer number. They are necessary to identify and remove RTCs only consisting of random combinations of hits that potentially can form a circle in the x-y-plane, but are not compatible with a helix in 3D because they are randomly distributed in the r-z-plane. Additionally, they are used to remove wrong hits within correct RTCs, e.g. ghost hits that share the u-coordinate with the correct hit on the same sensor, c.f. Figure 2.7.



(a) Number of relations per RTC. In some cases nearly 10 000 relations are created for a single RTC, while in most cases the number is well below 100.



(b) Valid number of relations for the single RTCs. If no or only one relation can be created, no actual track can be constructed from a RTC. This is the case for nearly 60 % of all RTCs. In addition, for a small number of RTCs, too many (> 100) relations are created for 0.3 % of all RTCs. These RTCs are discarded to limit the execution time and RAM usage.

Figure 5.3: Number of relations per RTC.

The distribution of number of relations per RTC is shown in Figure 5.3(a). While the distribution peaks at zero, in some cases up to or even more than 10 000 relations are created for one RTC, as

shown in the left panel of Figure 5.3(a). Since a valid track requires the presence of at least three hits on different sensors, only RTCs for which at least two relations are created are further processed. A distribution for whether the number of relations is considered valid or not is shown in Figure 5.3(b). It is clearly visible that nearly 60 % of all RTCs only have zero or just one relation and thus cannot create a valid track. In addition, for 0.3 % of the RTCs more than 100 relations are created, in which case the processing of the corresponding RTC is aborted to avoid single outliers in both execution time and RAM usage. With otherwise same settings, but allowing for up to 10 000 relations per RTC instead of 100, the gain in track finding efficiency is marginal with 0.01 %, but the average execution time increases from 7.5 ms to 8.9 ms per event, and the standard deviation of the execution time per event increases by nearly a factor of 10 from 7.5 ms to 69.4 ms, while the RAM usage increases from 12 MB to 71 MB. The cause for single RTCs with a large number of relations is found to be a large number of background hits in close vicinity to each other, often in the very forward or backward regions of the SVD, in which only very rarely an actual track is hidden at the same time. These large piles of background hits can be caused by the beam particles being deflected slightly in a scattering process to hit the final focusing magnets. Thus removing these RTCs reduces the execution time and RAM consumption, while also decreasing the fake rate, but without having much impact on the tracking efficiency.

Tree Search and Hit Filtering For each of the remaining RTCs the relations are used to check if a (sub)set of hits in the RTC can form a track. Starting with the outermost hit(s), the next hits sharing a relation with the first one are combined into a *path* and a filter is applied. If the filter result is positive the next hit is attached to the path. The filters can differ depending on the current length of the path of hits. Different filters have been tested while writing this thesis, from simple geometrical filters using the angles and distances between the different hits, to fits to the hits with a circle [77], a triplet fit [78], or a helix fit [79]. The best results are achieved with a filter using the triplet fit. In case only two hits are checked, i.e. in the first step, a virtual IP is added, as the fit would not work otherwise. In all other cases the hits in the path are used without an additional IP constraint. However, due to the modularity of the findlets and the filters, it is easy to replace each filter, and e. g. use a virtual IP constraint also when the path is longer than two hits, or to replace the fit based filters by filters entirely based on geometrical constraints, or using MVA based filters.

In case of the fit based filters, the decision on whether or not to extend the current path with an additional hit is based on the *p-value* of the fit. If more than one hit is available at this point, all extended paths are sorted by their p-values, and only the n best candidates are further processed to add more hits, where n is a user definable quantity. This process of repeatedly adding a hit to a path based on the relations and applying the filters is called *tree search* as at each intermediate last point in the path the possible next hits create different branches within the RTC. It is ultimately stopped once the innermost layer is reached, or if the track candidate does not contain any further hits. Usually only one actual track is contained within one RTC, thus the best path is selected from all paths created for each RTC. For this, the paths are sorted by their length (= number of hits), and by their p-value if the length is the same, both in descending order.

After all RTCs are processed as described, a final set of track candidates is obtained. Since neither the HT approach nor the clustering are perfect, it is possible that at this point multiple track candidates with the same path and hit content are found. The final set of tracks is selected by comparing the

length of the path, the hit content, and the p-value. If tracks share hits, the longer track and the one with the higher p-value is saved, while the other ones are discarded. All tracks remaining after this final track selection are then stored in the basf2 data store.

As the SVDHoughTracking algorithm shares several features with DATCON, it has multiple parameters that need to be optimised to achieve the maximal track finding performance, too. The full set of parameters is:

- Seven HS parameters: vertical size of the HS, number of HS sectors in horizontal and vertical direction, as well as minimum and maximum (horizontal, vertical, total) cluster size.
- Three cut values on λ for the relation creation depending on the difference in layer number between two consecutive hits (0, 1, or 2). Additionally cuts on the difference in hit time are implemented that could be optimised, but these are not subject to optimisation procedure in this case and set to fixed values of 17.5 ns which are found by manual optimisation.
- The maximum number of relations per RTC.
- The maximum number of paths allowed for each path length, i.e. the number of hits in an RTC.
- For each of the hit filters the maximum number of hits to propagate to the next step, and the value for the cut on the p-value.

In total this results in 19 different parameters for optimisation, several of which being integer values that are difficult to optimise, as this is a *p-np complete* problem. For optimisation, a similar approach as for DATCON is chosen, with the track finding efficiency and fake rate following the definitions in Section 3.8 as FOMs to optimise for. All results shown in this section, e.g. on the number of RTCs or relations per RTC, are obtained using the final parameter set from optimisation.

5.3 Tracking performance studies on $\Upsilon(4S)$ MC events

For Belle II as a B-factory the reconstruction of final state particles from $\Upsilon(4S)$ decays has highest priority. Thus all reconstruction algorithms including the tracking are tested and validated with simulated $\Upsilon(4S)$ events with beam background. At the time of this work, only simulated beam backgrounds for the nominal luminosity conditions are available and used for all results presented in this chapter if not indicated otherwise. The new algorithm is compared to the full Belle II tracking chain (denoted as *default full tracking* in the figures) as described in Section 3.2, and the VXDTF2 in order to compare the performance using SVD hits only. Each simulation is conducted with 50 000 $\Upsilon(4S)$ events and the track data are analysed using the well tested tracking validation framework within in basf2 to ensure comparability.[2] As mentioned before, the focus of the performance comparison is on

- track finding efficiency, fake rate and clone rate
- SVD hit efficiency and hit purity.

[2] This framework is also used for the nightly validation of basf2.

The accuracy and precision of the estimation of the track parameters is not part of the studies, as it relies on a good track fit, and in general the parameters can be estimated more precisely if CDC hits (momentum estimate) and PXD hits (d_0, z_0) are part of the track. Thus fitting tracks from the full tracking chain would outperform the SVD-only track finders for any of the variables. Since PXD hits are only attached in a last step by the CKF, but not part of the track finding itself, all results on tracking performance were created without using PXD hits unless mentioned differently. All of the above quantities are evaluated as function of the transverse momentum p_T and the dip angle λ.These two track quantities are chosen as they are non-uniform, while e.g. the distribution of φ is nearly flat, with only a small non-uniformity caused by the initial particles' boost along the x-axis due to the collision angle of 41.5 mrad compared to the z-axis.

General information on the performance figures

As before, in all figures showing the tracking and ROI finding performance as function of p_T and λ, a grey area indicates the distribution of the MC particle momentum parameters. In case the y-axis does not cover the full range of 0 to 1, the MC particle distributions are adapted such that they are still fully visible. Additionally, vertical dashed red lines in the λ figures indicate the angular acceptance region of the tracking detectors in λ from $-60°$ to $73°$, and λ values larger (smaller) than $0°$ are referred to as forward (backward), respectively.

In many cases ratios are shown for efficiencies, purities, fake and clone rates. Since the SVDHoughTracking performance is the base line for evaluating performance improvements or losses, the bin contents of each of the distributions is divided by the bin content in the same bin of the SVDHoughTracking distribution. Thus, in efficiency and purity ratio plots a ratio smaller than unity indicates that the SVDHoughTracking performs better than the other algorithm it is compared to, while for a ratio larger than unity the SVDHoughTracking performs worse, as high efficiency and purity values are desirable. The opposite is true for fake and clone rate ratios as these values should be as low as possible; a higher ratio for a particular bin means that the SVDHoughTracking yields better values in this bin. Thus, in the ratio plot, the ratio of VXDTF2 and SVDHoughTracking is shown in orange, and the ratio of the full tracking and the SVDHoughTracking is shown in green, respectively.

Additional perfomance figures, for example as function of the impact parameters d_0 and z_0, are included in the appendix in Appendix A.2.

5.3.1 Track finding efficiency

Figure 5.4 shows the track finding efficiency as function of p_T and λ for three different track finding methods and evaluating the full track finding with SVD data alone.[3] The new tracking algorithm achieves more than 90 % efficiency above p_T values of 100 MeV/c and thus over most of the p_T range, with increasing efficiency with increasing p_T. Similarly, the efficiency is above 95 % in most of the acceptance region of the SVD in λ. However, a few steep dips are visible. These align with the insensitive regions between SVD sensors on the different layers. If a particle crosses such a region, in

[3] The MC track finding calculates the NDF of each MC track and assumes an MC track to be *findable* if the NDF is greater or equal to five. Thus, an MC track with e.g. 40 CDC hits and 1 SVD hit is counted as findable, like for instance a K_S^0 daughter, and if the CDC part is found, but the single SVD is missed, the SVD hit efficiency is reduced. Thus, the SVD standalone tracking needs to be compared to the full tracking based on tracks that can be fully defined by SVD only information.

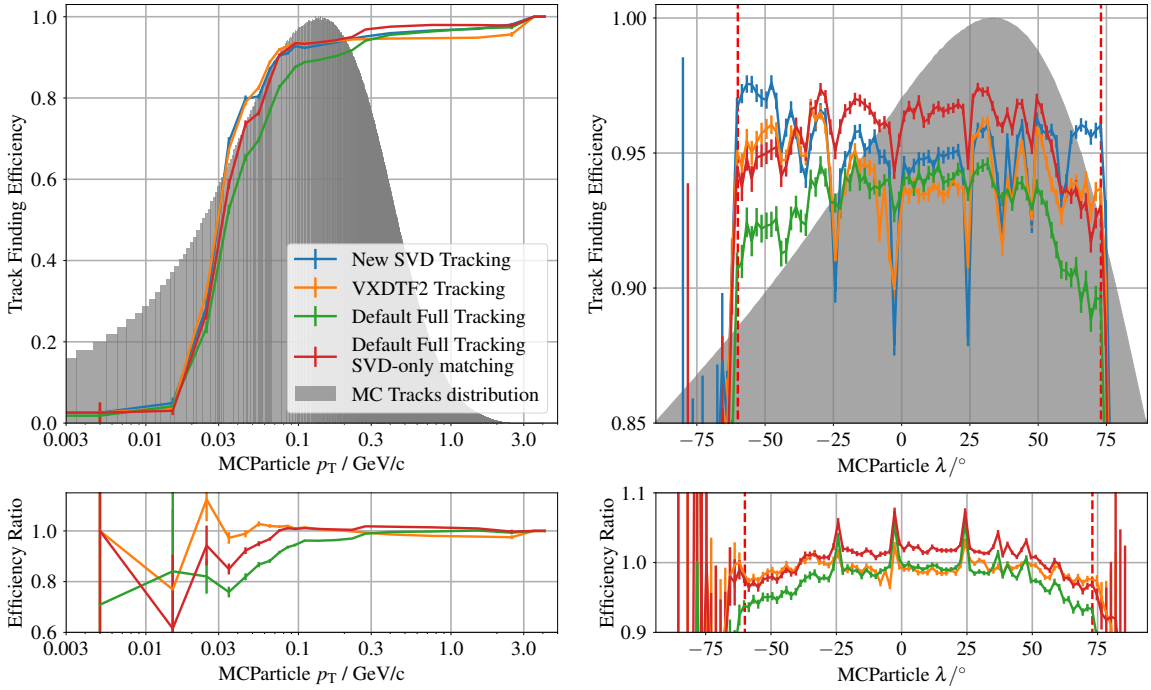


Figure 5.4: Track finding efficiency as function of p_T (left) and λ (right) for the new SVD tracking (blue), the VXDTF2 (orange), the full tracking chain (green), and the full tracking using only SVD information in MC track finding and truth matching (red) for comparison. The newly developed SVD tracking overall has the highest efficiency and is only outperformed by the VXDTF2 in the low p_T region between 50 and 200 MeV/c by about 1 %. The distinct dips in the efficiency in λ align with the insensitive regions between the single SVD sensors in each layer. The lower part shows the ratios of the new SVDHoughTracking and the other tracking methods, with the colours corresponding to the tracking algorithm the SVDHoughTracking is compared with. Only in the central region, the track finding efficiency of the full tracking, but only matching tracks based on SVD information, yields better results compared to the new SVDHoughTracking.

general only three hits are available to find the track. Thus, missing one of these hits results in missing the full track, as a track cannot be defined with only two SVD hits.

In comparison, the two SVD standalone methods are superior to the full tracking in the p_T region below 300 MeV/c where the particles do not always traverse the full CDC because they either curl back towards the VXD, or leave the CDC in the forward or backward end-plates if they have a sufficiently large longitudinal momentum component p_z. However, the efficiency of the full tracking using SVD information only for truth matching is highest above p_T values of 250 MeV/c, and in the central part of the λ spectrum. Of the two SVD algorithms, the new algorithm developed in this thesis has the overall higher finding efficiency of (94.73 ± 0.10) % compared to (93.89 ± 0.11) % of the VXDTF2, and the full tracking achieving (92.92 ± 0.11) % in efficiency. Only between 50 to 200 MeV/c the VXDTF2 performs slightly better by about 1 %. In this region all particles except for electrons usually are not yet MIPs and thus potentially suffer from increased energy loss and multiple scattering in the detector layers, causing the trajectories to deviate significantly from a circle, in which case they cannot be found in the HS anymore. Second, scattering in the r-z-plane can result in deviations in λ such that no relation is created between two hits on different sensors because of the cut on the difference in λ

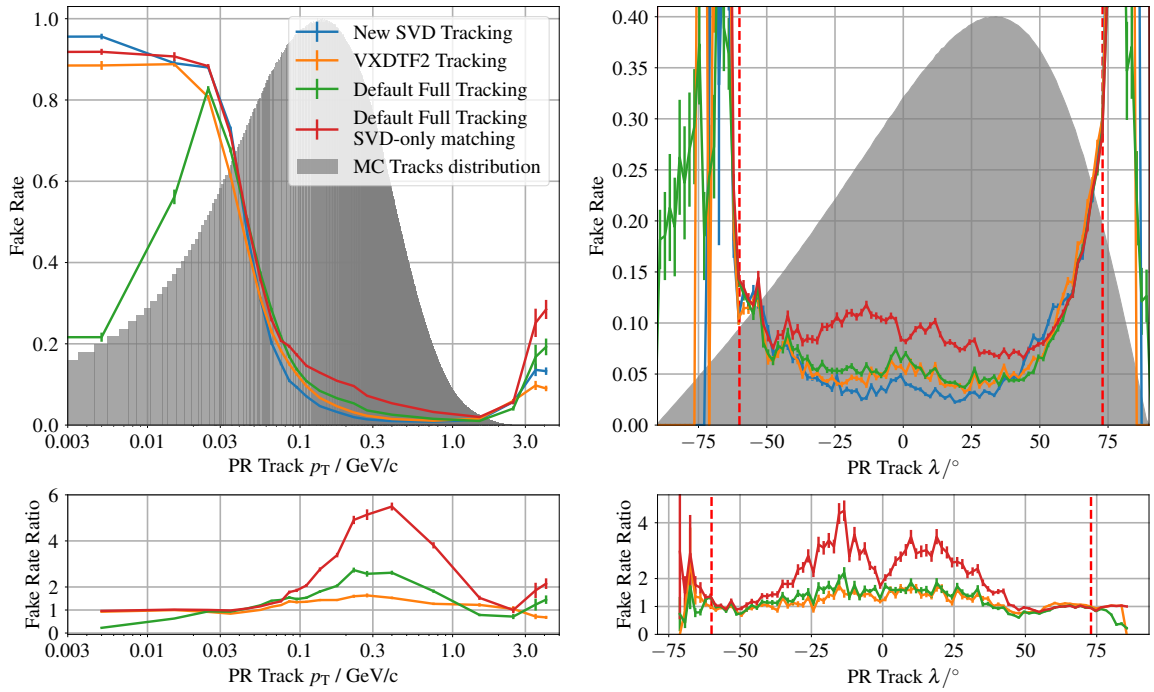


Figure 5.5: Fake rate as function of p_T (left) and λ (right) for the new SVD tracking (blue), the VXDTF2 (orange), and the full tracking chain (green), and the full tracking using only SVD information in MC track finding and truth matching (red) for comparison. Below 50 MeV/c the new algorithm has the highest fake rate, with similarly high fake rates in the VXDTF2 and the full tracking only using SVD information, which fully relies on the VXDTF2 in this p_T region. Above 100 MeV/c the new algorithm creates the lowest number of fake tracks. Between $\lambda = -40°$ and $\lambda = 40°$ the fake rate of the SVDHoughTracking algorithm is lower compared to the other algorithm. Most of the fake tracks are found in the very forward and backward regions in the detector, independent of the tracking algorithm used.

during relation creation. Both energy loss and scattering can be learned by the VXDTF2 sector maps during its training procedure, thus it is able to better cope with them.

As for the new track finding, there are similar distinct dips in the efficiency as function of λ for the VXDTF2, as for the same reason. This effect is less severe for the full tracking chain, as in most cases only low p_T particles need to be found by the VXDTF2 only, while for most particles CDC tracks are found, thus missing an SVD hit does not cause much harm. While all three algorithm have an efficiency of about 93 to 95 % in the central region around $\lambda = 0°$, the efficiency of the full tracking drops in the forward and backward regions, while the distribution of the two SVD algorithm is more uniform except for the aforementioned dips. Besides the dips at distinct values of λ, the new track finding performs best compared to the two other algorithms, with the largest performance advantages in the forward and backward directions.

5.3.2 Fake rate

Figure 5.5 shows the fake rate of the three different tracking methods as function of p_T and λ. While the fake rate of the new tracking algorithm is above 80 % below transverse momenta of 30 MeV/c,

it drops to below 10 % at around 100 MeV/c and keeps approaching 0 until it starts rising again at around 2 GeV/c. In the central region around $|\lambda| < 40°$, the fake rate of the new tracking is at a nearly constant level below 5 %, but it increases for larger $|\lambda|$ values, up to 30 % in the very forward region at $\lambda = 73°$. Since the fake rate increases for both low transverse momenta, and in the forward and backward region of the acceptance, it is likely that many of the tracks classified as fake are actual tracks from beam background processes, but not just random combinations of hits, e.g. from elastic Bhabha scattering, or from two-photon events. This hypothesis is supported by the way the new algorithm is implemented: when evaluating the RTCs and checking additional hits candidates to possibly be added to the track, the track candidates are fitted with the next hit in question, and only those with at least a certain p-value obtained from the fit are further processed, indicating that the fake tracks have a high p-value, which is much less likely for random combinations of hits. Overall the fake rate of the new algorithm is estimated to be (7.56 ± 0.12) %.

Compared to the other two tracking methods, the overall behaviour is similar: in all three cases the fake rate increases with lower p_T value of the PR track momentum estimate, and in both forward and backward direction. This also seconds the hypothesis that many of the found fake tracks are not random combinations of hits, but actual tracks from beam background, as it is unlikely that three very different algorithms find the same random combinations of hits to build a fake track. On average, the fake rate with the established algorithms is (7.81 ± 0.12) % and (7.64 ± 0.12) % for the VXDTF2 and the full tracking, respectively, which is slightly higher compared to the new algorithm, but all within the margin of error of one another.

The increased fake rate at low transverse momenta and in the forward and backward directions are a long standing problem already for the currently used algorithms. A different group in the Belle II collaboration is working to solve or at least reduce this issue, but the necessary tools are not yet available to be used for this work. These tools can then be used to reduce the fake rate of the new algorithm as well as the VXDTF2 and the full tracking chain. Thus, at this point the fake rate of the new tracking algorithm is not considered a problem on its own, as it is shared with the other two algorithms and mostly present at very low transverse momenta of $p_\mathrm{T} < 50\,\mathrm{MeV}/c$. In comparison the two established algorithms, the new SVD tracking reconstructs the lowest number of fake tracks above 60 MeV/c, while the full tracking reconstructs the highest number of fakes in this region. An explanation for this behaviour is that using CDC and SVD information increases the likelihood of adding too many wrong hits such that the purity criterion on tracks to be valid is not fulfilled any longer, in which case the corresponding track is classified as fake instead of found / matched.

5.3.3 Clone rate

Next to tracking efficiency and fake rate, the clone rate, shown in Figure 5.6, is the third important FOM to value the tracking performance. Again, the results of the new algorithm are shown in blue, while the results for the two established algorithms are shown in orange and green, respectively. Similar to the fake rate, the clone rate increases for low p_T values of the PR tracks for the new algorithm, as shown in the left part. Additionally, it peaks around $\lambda = 0°$, as shown in the right part of the figure. This is caused by curling tracks that are reconstructed multiple times, which is mostly the case for tracks around $\lambda = 0°$, as they are nearly stationary in their z-coordinate. In contrast, tracks with λ values significantly different from 0 travel along the z-axis, and are only reconstructed once even if they curl back towards the z-axis, as the second ingoing arm is not found. For transverse momenta above 300 MeV/c the clone rate is negligible, resulting in an average clone rate of (1.09 ± 0.05) %.

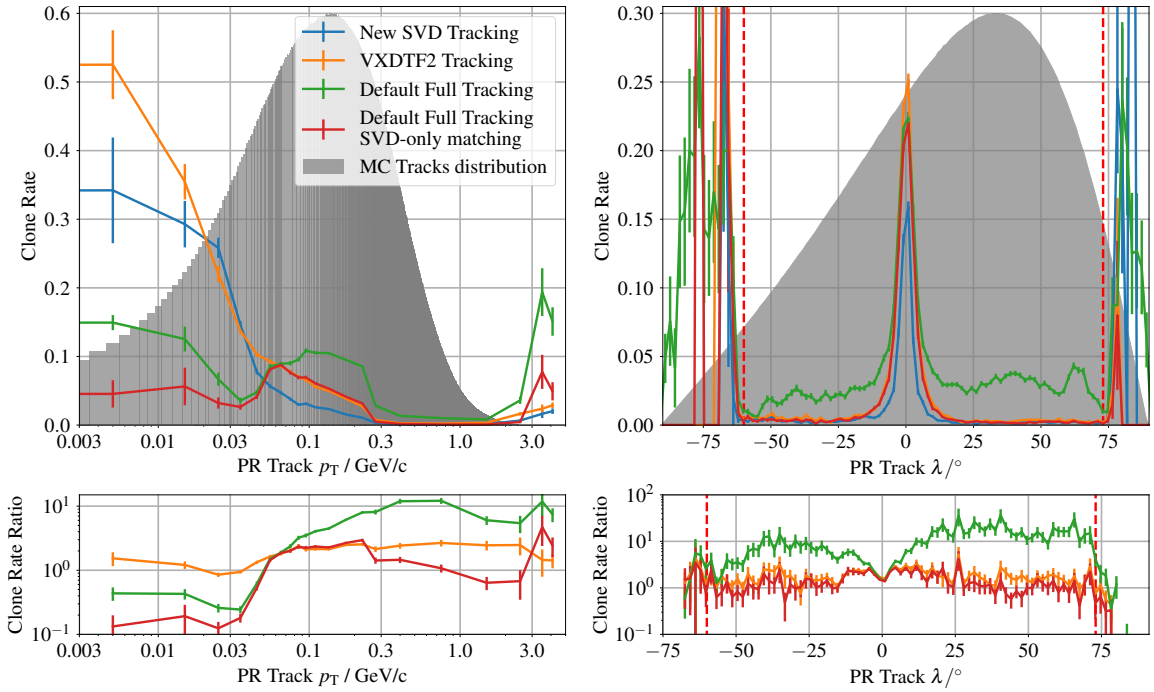


Figure 5.6: Clone rate as function of p_T (left) and λ (right) for the new SVD tracking (blue), the VXDTF2 (orange), and the full tracking chain (green), and the full tracking using only SVD information in MC track finding and truth matching (red) for comparison. All algorithms produce an increased number of clone tracks around $\lambda = 0°$, indicating that curling tracks are found multiple times. This is supported by the fact that both SVD standalone algorithms find more clone tracks in the very low p_T region where tracks tend to curl in the SVD only. In the full tracking configuration the clone rate is higher compared to the two SVD algorithms because CDC and SVD tracks can be found separately and not merged together.

Similar to the fake rate, the clone rate is increased for the VXDTF2, too, at low p_T values of the PR tracks, and around $\lambda = 0°$, for the same reasons as for the new tracking algorithm. Overall, a clone rate of $(2.24 \pm 0.07)\,\%$ is achieved by the VXDTF2. The additional clone rate is caused by finding low p_T tracks multiple times based on the sector map information, but not being able to combine the individual tracks. This can be due to missing information in between where the track passes the CDC. In contrast, for the full tracking the clone rate is only mildly increased for tracks with very low transverse momentum, which are barely reconstructable, but instead there is an increase between 50 and 250 MeV/c, as well as a nearly constant but higher level over most of the λ range compared to the SVD standalone algorithms, resulting in an average clone rate of $(4.34 \pm 0.09)\,\%$. Curling tracks in the CDC contribute significantly to the clone rate if several arms of a track are found in the CDC but not merged into one track. Often the first outgoing arm is reconstructed correctly, containing SVD and CDC information. Further arms are only reconstructed partially, while still containing enough true hits and a high purity to be considered a valid track such that they contribute to the clone but not the fake rate.

As discussed before, both clone and fake tracks are additional tracks in an event that need to be avoided, removed from the track set if possible, or ignored by the analyst, as they may spoil the analysis by adding additional tracks, momentum, or charge to an event. Similar to the efforts to reduce the fake

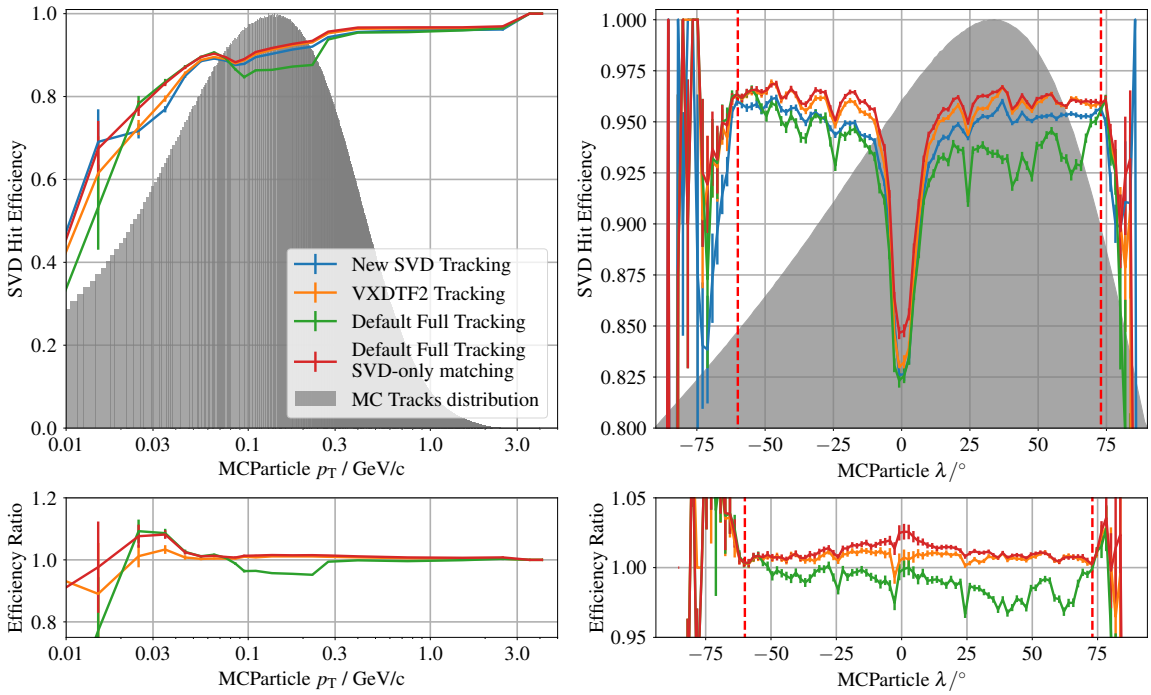


Figure 5.7: SVD hit efficiency of matched tracks for all three tracking methods as function of p_T (left) and λ (right). The SVDHoughTracking algorithm achieves an efficiency of more than 90 % over the majority of the p_T range, and is on a near constant level as function of λ, with a dip around $\lambda = 0°$ due to curling tracks.

rate, there is ongoing work to reduce the number of clone tracks by attributing them to the correct track. This, however, has to be done carefully, as usually the first outgoing arm provides the most valuable information for the track fit, while further arms often degrade the track fit result.

5.3.4 Hit efficiency and purity

The distribution of the SVD hit efficiency of matched tracks is shown in Figure 5.7 as function of p_T and λ. The new algorithm not only finds tracks with a very high efficiency, but it also does find the correct SVD hits efficiently, which is represent by the average hit efficiency of (93.92 ± 0.11) % and hit purity of (99.17 ± 0.04) %. This indicates that the tree-search algorithm and hit filtering based on simple track fits and cuts on the p-value of the fitted tracks are sufficient to discriminate correct hits from wrong hits. Additional improvements might be possible by employing MVA based filtering methods during hit filtering or final track selection. While on a near constant value of 95 % over most of the λ range, the efficiency dips between $-10° < \lambda < 10°$ due to low p_T curling tracks. Compared to the VXDTF2 the hit efficiency is slightly lower by about 0.7 % as the VXDTF2 achieves (94.60 ± 0.10) % efficiency, while the hit purity is slightly increased by about 0.4 % ((98.80 ± 0.05) % for the VXDTF2). The slight reduction of the hit efficiency compared to the default SVD tracking algorithm is mostly found in the low p_T regime, and caused by the VXDTF2 being able to learn about curling tracks and incorporating them in the sector maps. However, this seems to go on cost of the hit purity, indicating that the hit filters in the VXDTF2 can be optimised. With a hit efficiency of (92.70 ± 0.12) % and a hit purity of (99.21 ± 0.04) % the full tracking achieves similar values to the

Table 5.1: Summary of the five most important FOMs for the three investigated tracking methods for nominal backgrounds (from simulation). The last column contains results using the full track finding for track reconstruction, but only SVD information with the MC tracks and in MC matching to have a direct comparison of tracks that can be found in SVD alone.

Values in percent	New SVD track finding	VXDTF2	Full track finding	Full track finding SVD only matching
Finding efficiency	94.73 ± 0.10	93.89 ± 0.11	92.92 ± 0.11	95.72 ± 0.09
Fake Rate	7.56 ± 0.12	7.81 ± 0.12	7.64 ± 0.12	10.62 ± 0.14
Clone Rate	1.09 ± 0.05	2.24 ± 0.07	4.34 ± 0.09	1.94 ± 0.06
SVD Hit Efficiency	93.90 ± 0.11	94.60 ± 0.10	92.70 ± 0.12	94.99 ± 0.10
SVD Hit Purity	99.18 ± 0.04	98.80 ± 0.05	99.21 ± 0.04	99.28 ± 0.04

two SVD standalone algorithms. However, the hit efficiency is nearly 2 % lower compared to the standalone VXDTF2, even though the VXDTF2 contributes part of the full tracking, which is mostly caused by particles with transverse momenta between 100 to 250 MeV/c, while it recovers above this range, as shown in Figure 5.7. Combined with the higher clone rate of the full tracking this leads to the assumption that a significant number of tracks are found in both the CDC and SVD standalone algorithm, but not properly merged. Because of the higher number of hits the CDC track is assigned the *matched* flag, while the SVD track is assigned the *clone* flag, which explains both the lower hit efficiency and higher clone rate.

Summary of the standalone track finding performance

Compared to the well established VXDTF2 as SVD standalone track finding algorithm, and the default full tracking which the VXDTF2 is part of, the newly developed SVDHoughTracking performs better in terms of four of the five FOMs (track finding efficiency, fake rate, clone rate, and SVD hit purity), only the SVD hit efficiency is slightly worse. A summary of the most important FOMs for tracking in Belle II, and with the SVD in particular, is provided in Table 5.1. New tools to decrease the fake rate are in development at the time of writing, and all three track finding methods will benefit from those, especially in the very low p_{T} regime where barely any actual physics tracks are present. Nonetheless, careful optimisation is necessary to not reject any tracks from B meson decays, like slow pions.

Using a slightly different set of parameters for the SVDHoughTracking, even average tracking efficiencies of above 95 % are possible, at the cost of in increased fake rate of about 9 % instead of (7.56 ± 0.12) %. Thus, additional optimisation of the parameters can likely retain the increased finding efficiency, while achieving a similarly low fake rate.

5.3.5 Execution time evaluation

Since the full Belle II track reconstruction is executed on the HLT, which requires a fast execution time to process all incoming events in a timely manner, the track finding algorithms need to perform the track finding as fast as possible. Although the SVD standalone track finding in the current data flow only follows the CDC-To-SVD-CKF, the execution times of the VXDTF2 and the new SVDHoughTracking are compared running standalone, as this marks the worst case scenario. The execution time per event for both algorithms is shown in Figure 5.8. As the VXDTF2 consists of 10

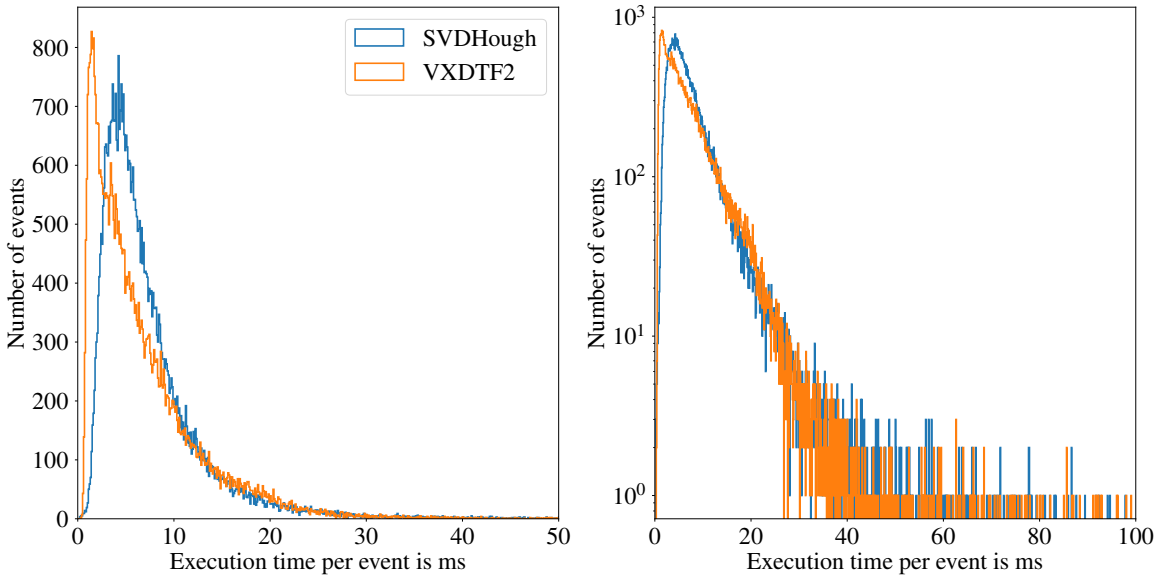


Figure 5.8: Execution time per event for the SVDHoughTracking and the VXDTF2 running standalone with linear axes (left) and a logarithmic y-axis (right). While the maximum of the execution time distribution for the VXDTF2 is at 1.5 ms compared to 4.7 ms, it is has a longer tail for the VXDTF2, with individual events taking up to 1.4 s per event.

individual modules, their execution times per event are added up to retrieve a combined value. On average, the SVDHoughTracking takes about (7.5 ± 6.3) ms per event to execute the track finding, while the VXDTF2 takes (7.0 ± 11.9) ms per event. The the most probable values being 4.7 ms for the SVDHoughTracking and 1.5 ms for the VXDTF2, while the median values are 5.8 ms and 5.0 ms, respectively.

At this point it has to be noted that the VXDTF2 aborts the execution of an event if in one of its parts a certain number of combinations or track candidates is found, which happens in about 1 of 10000 cases with nominal background, resulting in no tracks from the VXDTF2 in these events. If the values are close to but below the abort thresholds, the VXDTF2 execution time can be rather long, in this study up to 1.4 s for one event are observed. In contrast, the SVDHoughTracking does not abort the track finding for an event entirely, but only for individual RTCs, as shown in Figure 5.3, which happens for 0.3 % of all RTCs.

These results show that the SVDHoughTracking is not only a compatible track finding algorithm in terms of its track finding performance, but also with respect to the execution time, as it is similar to the VXDTF2. Using the SVD standalone track finding algorithms as part of the full track finding, the execution times are significantly lower. Many of the SVD hits belonging to tracks are already attached to the tracks by the CDC-To-SVD-CKF, which takes about ten times as long with around 70 ms per event. However, this is still a short time in comparison to the CDC standalone algorithms, which have a combined execution time that is another order of magnitude larger at 600 to 700 ms per event on average. Thus, the small increase in average execution time of the SVDHoughTracking in comparison to the VXDTF2 is fully acceptable, especially considering the increased track finding efficiency and reduced fake and clone rates.

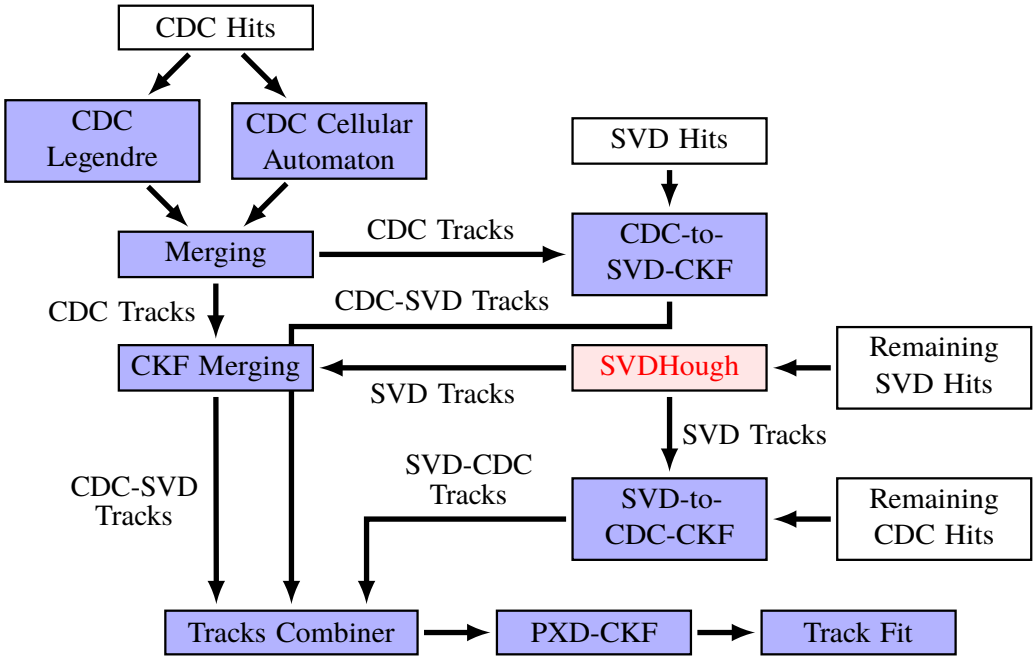


Figure 5.9: Updated track reconstruction flow in Belle II with the SVDHoughTracking replacing the VXDTF2 (indicated in red), with the SVDHoughTracking only using so far unassigned SVD space points. The rest of the track reconstruction chain remains unchanged; the original data flow is depicted in Figure 3.1.

5.4 SVDHoughTracking as VXDTF2 replacement in the full tracking

Now that the capabilities and the potential of the new SVDHoughTracking are proven, it is tested as a drop-in replacement for the VXDTF2 in the full tracking chain. This only introduces a small change in the full tracking chain, which is shown in Figure 5.9. Due to the modularity of basf2 it is very easy to conduct this change, as only a modification of the Python scripts defining the tracking chain is necessary.

In Figure 5.4 it is visible that the SVDHoughTracking achieves the highest track finding efficiency for λ values of approximately $|\lambda| > 50°$ in both forward and backward direction, while being slightly superior in the more central part of the detector with $|\lambda| < 50°$. On average, the track finding efficiency is increased from $(92.92 \pm 0.11)\,\%$ using the VXDTF2 in the full tracking to $(93.36 \pm 0.11)\,\%$ with the SVDHoughTracking as replacement. Thus, if replacing the VXDTF2 by the SVDHoughTracking in the full tracking chain yields improvements in the overall tracking performance, they are expected to materialise in the regions of the phase space where it performs better compared to the VXDTF2 standalone. While in the standalone case the SVDHoughTracking used the full set of SVD space points, it is now limited to those not already used by CDC-To-SVD-CKF, which is known to attach SVD hit information to existing CDC tracks with both high hit efficiency and purity, as shown in [36].

As expected, replacing the VXDTF2 with the SVDHoughTracking increases the track finding efficiency for $\lambda < -40°$ and $\lambda > 60°$, while providing similar performance for the region inbetween, as shown in the right half of Figure 5.10. However, below p_T values of $60\,\mathrm{MeV}/c$ the full tracking including the VXDTF2 provides a higher track finding efficiency of up to $40\,\%$ around transverse momenta of $p_T = 30\,\mathrm{MeV}/c$. In contrast, for $p_T > 60\,\mathrm{MeV}/c$ the SVDHoughTracking achieves a

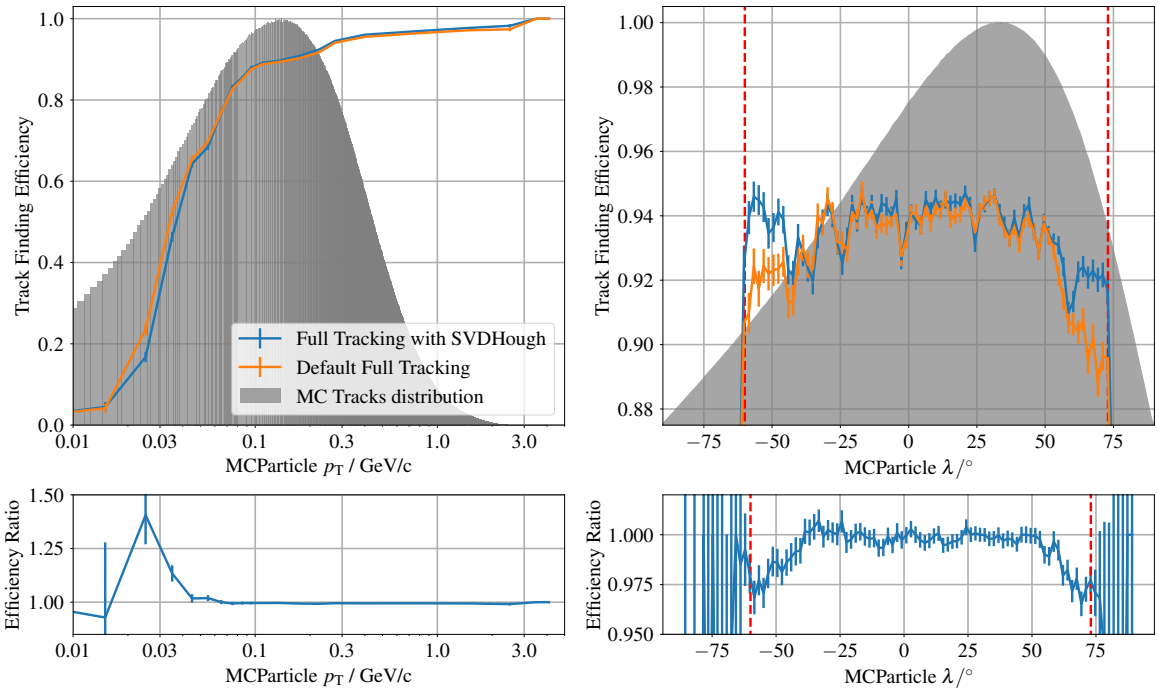
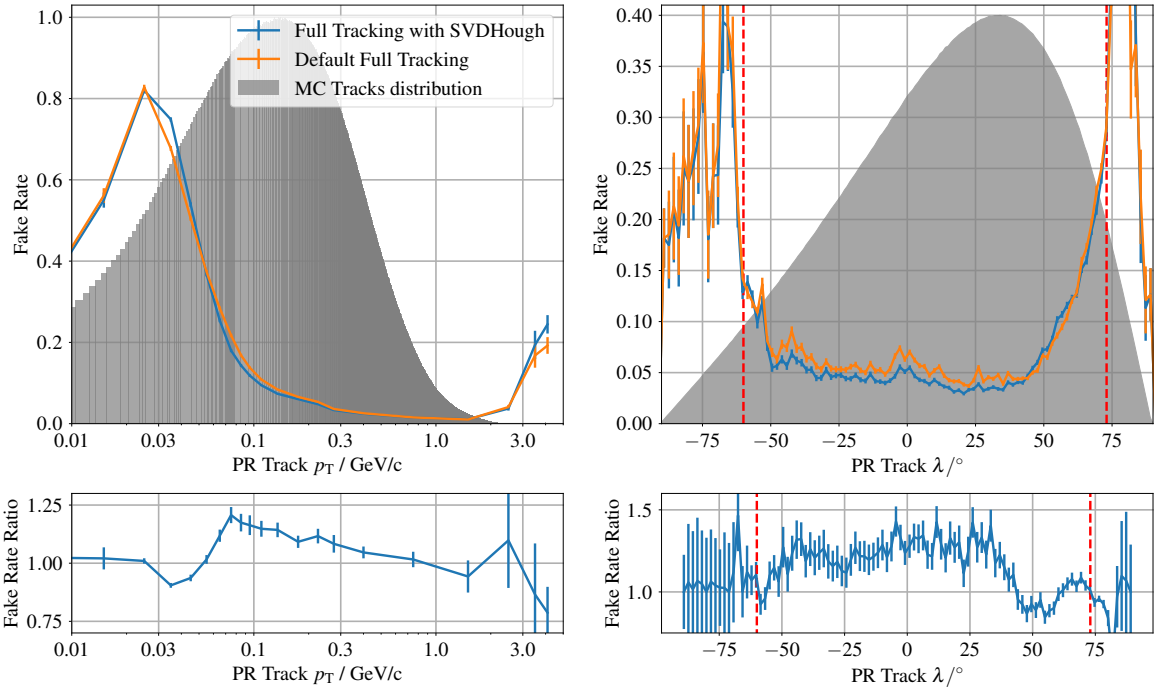


Figure 5.10: Track finding efficiencies using the default full tracking chain (orange) and the SVDHoughTracking as a drop-in replacement for the VXDTF2 (blue). Below p_T values of 60 MeV/c the default full tracking provides a higher finding efficiency of up to 40 % around p_T = 30 MeV/c. Beyond that, the full tracking using the SVDHoughTracking algorithm as a replacement performs slightly better. The largest improvements are visible in the very forward and backward direction, i.e. for large values of $|\lambda|$.

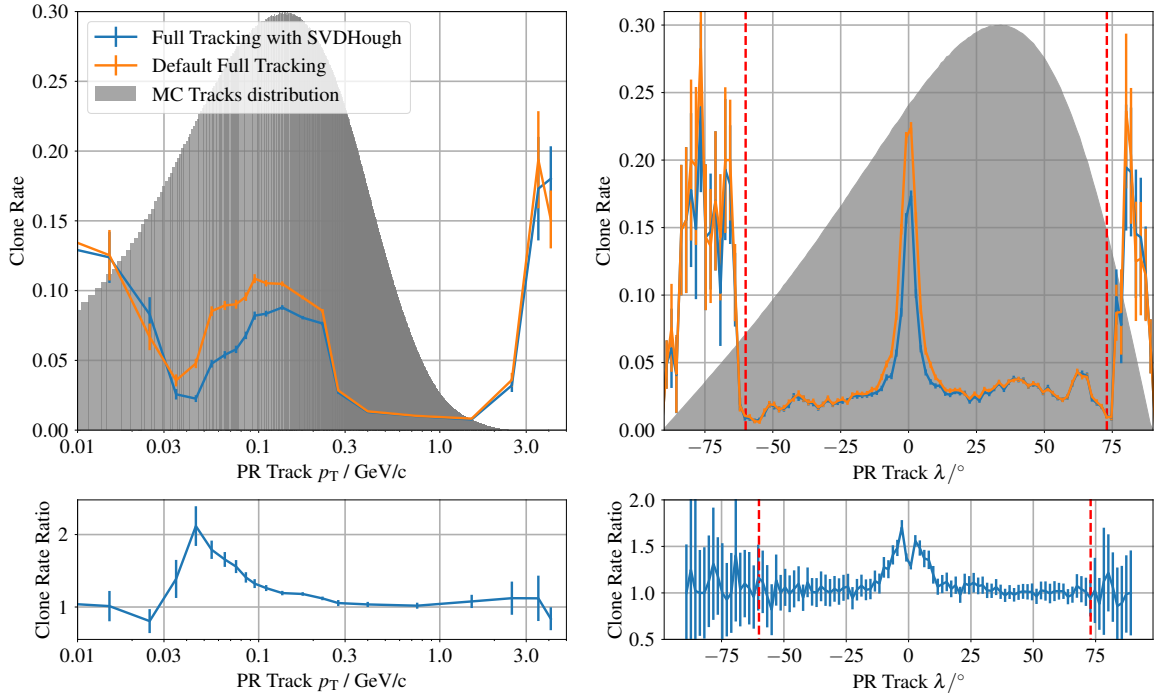
higher track finding efficiency.

For comparison, the fake and clone rate distributions as functions of p_T and λ are shown in Figure 5.11. Both rates are decreased overall using the SVDHoughTracking, with the largest improvements in the fake rate between 50 MeV/c and 800 MeV/c in p_T, and between −50° and 40° in λ, respectively, for the fake rate, and in the central region in λ and transverse momenta below 300 MeV/c for the clone rate. Overall, the fake rate decreases from (7.64 ± 0.12) % using the VXDTF2 in the full tracking to (7.15 ± 0.12) % using SVDHoughTracking, while the clone rate decreases from (4.34 ± 0.09) % to (3.61 ± 0.08) %. All values are obtained using the full MC track matching. As before when using the SVDHoughTracking and the VXDTF2 standalone, the SVD hit efficiency is slightly lower with the SVDHoughTracking at (92.32 ± 0.12) % compared to (92.70 ± 0.12) %, while the SVD hit purity is similarly high at more than 99 %. A summary of all average values, including those using only SVD information for MC track finding and truth matching, is provided in Table 5.2.

Similar to using the SVDHoughTracking alone, the SVD hit efficiency is slightly decreases compared to using the VXDTF2, while the SVD hit purity is slightly increased, c.f. Section 5.3.4. However, although the values and trends are consistently different, they are only about two standard deviations apart. Overall, the studies have proven that it might be beneficial to replace the VXDTF2 with the SVDHoughTracking in the full tracking chain, or combine their results to profit from the advantages of both algorithms. For instance, no indication on the impact on the reconstruction of specific processes like decays of D^* or K^0_S can be derived from the results presented thus far.



(a) Fake rate comparison of the default full tracking chain using the VXDTF2 or the SVDHoughTracking.



(b) Clone rate comparison of the default full tracking chain using the VXDTF2 or the SVDHoughTracking.

Figure 5.11: Fake rate and clone rate comparison replacing the VXDTF2 with the SVDHoughTracking in the full tracking chain. Using the new SVDHoughTracking tracking in the full tracking chain, both a lower fake rate and a lower clone rate are achieved.

Table 5.2: Summary of the most important FOMs for track finding with the SVD for replacing the VXDTF2 with the SVDHoughTracking. The columns two and three contain the FOMs for using both SVD and CDC information in MC tracks and for MC truth matching, while the data in columns four and five are based on the same track finding but only using SVD information in MC tracking and MC truth matching.

Values in percent	Full Tracking SVDHough	Default Full Tracking	Full Tracking SVDHough	Default Full Tracking
	CDC + SVD matching		SVD only matching	
Finding efficiency	93.36 ± 0.11	92.92 ± 0.11	96.21 ± 0.09	95.72 ± 0.09
Fake Rate	7.15 ± 0.12	7.64 ± 0.12	10.29 ± 0.14	10.62 ± 0.14
Clone Rate	3.61 ± 0.08	4.34 ± 0.09	1.08 ± 0.05	1.94 ± 0.06
SVD Hit Efficiency	92.32 ± 0.12	92.70 ± 0.12	94.54 ± 0.10	94.99 ± 0.10
SVD Hit Purity	99.29 ± 0.04	99.21 ± 0.04	99.38 ± 0.04	99.28 ± 0.04

5.4.1 Slow pion and K_S^0 case studies

The impact of replacing the VXDTF2 by the SVDHoughTracking is checked with two specific processes. For low momenta, the decay $D^{*+} \to D^0\pi^+$ is investigated, and for large distances of the decay vertex to the IP the decay of K_S^0 is checked, as the loss of one of the two K_S^0 daughters results in the loss of the K_S^0 itself.

Slow pion reconstruction

The D^{*+} and D^0 mesons have a very similar mass, with a mass difference $\Delta m = m_{D^{*+}} - m_{D^0} = 146\,\mathrm{MeV}/c^2$ which is only slightly higher than the mass of the pion. This results in very low momenta for the pions produced in the decay, with transverse momenta of only up to 250 MeV/c, and the peak in the p_T distribution at only 50 MeV/c, making them hard to find. Using 10000 $\Upsilon(4S)$ events with nominal background, the finding efficiency for the low momentum pions is probed. In each event one of the two B mesons always decays via $\overline{B} \to D^{*+}l^-\bar{\nu}_l$ with $l = e, \mu$. The D^{*+} then decays as described above, and the second B meson decays generically and its decay particles are ignored in this study.

The efficiency for the reconstruction of slow pions is shown in Figure 5.12 for the two SVD standalone tracking algorithms SVDHoughTracking (blue) and VXDTF2 (orange), the default track finding (red), and the full track finding with the SVDHoughTracking replacing the VXDTF2 in the full tracking chain (green). In contrast to studies presented before, where fully generic $B\overline{B}$ decays are used, the efficiency using the SVDHoughTracking is worse than using the VXDTF2 both standalone and as part of the full track finding as shown in the ratio plots in orange and red, respectively. Comparing the two SVD standalone algorithms, the with the VXDTF2 (92.64 ± 0.26) % of all slow pions are found, while with the SVDHoughTracking only (90.11 ± 0.30) % are reconstructed, which is a reduction of 2.5 %. Using the full track finding the difference is smaller, but still the default achieves a higher efficiency of (89.00 ± 0.00) % compared to (88.00 ± 0.00) %. The reduced efficiency is mainly caused by two effects. Firstly, the pions suffer from increased multiple scattering, as the scattering angle is proportional to the inverse of the momentum, c.f. Equation (2.2). Secondly, they experience an increased energy loss in the material since the pions are not MIPs anymore, but on the steeply rising part of the Bethe-Bloch-Formula in Equation (2.1) with low $\beta\gamma$ values, as shown in Figure 2.10. If the scattering angle is too large, the relation creation in the SVDHoughTracking can fail because of

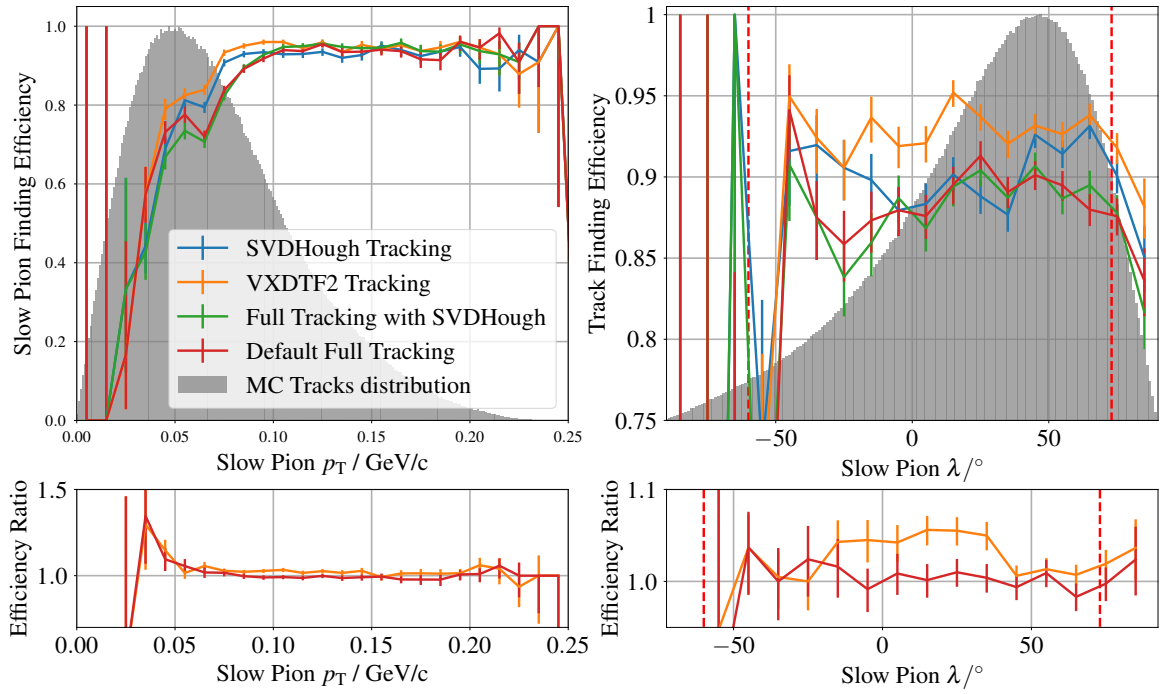


Figure 5.12: Track finding efficiencies for pions from $D^{*+} \to D^0\pi^+$ decays for the two SVD standalone tracking algorithms SVDHoughTracking (blue) and VXDTF2 (orange), the default track finding (red), and the full track finding with the SVDHoughTracking replacing the VXDTF2 (green). Using the SVDHoughTracking yields a slightly lower finding efficiency in both cases, which is mostly due to increased multiple scattering and energy loss at such low momenta, which the VXDTF2 can better cope with.

Table 5.3: Summary of the slow pion finding efficiency.

Values in percent	SVDHough Tracking	VXDTF2	Default full track finding	Full track finding with SVDHoughTracking
Finding efficiency	90.11 ± 0.30	92.64 ± 0.26	88.58 ± 0.32	88.37 ± 0.32

the selection on λ and φ. In addition, if the track deviates too much from a helix, the p-value of the track fit used in each step when adding a new hit to the track might be too low to pass the minimum required p-value to consider the track as good. Both issues can be addressed by further tuning the selection parameters of the SVDHoughTracking algorithm.

K_S^0 reconstruction

For simplicity, the reconstruction efficiency of K_S^0 mesons is only probed with 30000 particle gun events, where 5 K_S^0 mesons are generated per event and evenly distributed in the dip angle λ between $-60°$ and $70°$, as well as in momentum up to 1 GeV/c. For the evaluation of the reconstruction performance the default validation tools are used to ensure comparability with previous K_S^0 reconstruction studies. As no beam backgrounds are overlaid, this marks the best case scenario for the reconstruction of K_S^0 and their daughter particles, which are charged or neutral pions in most cases. However, as neutral

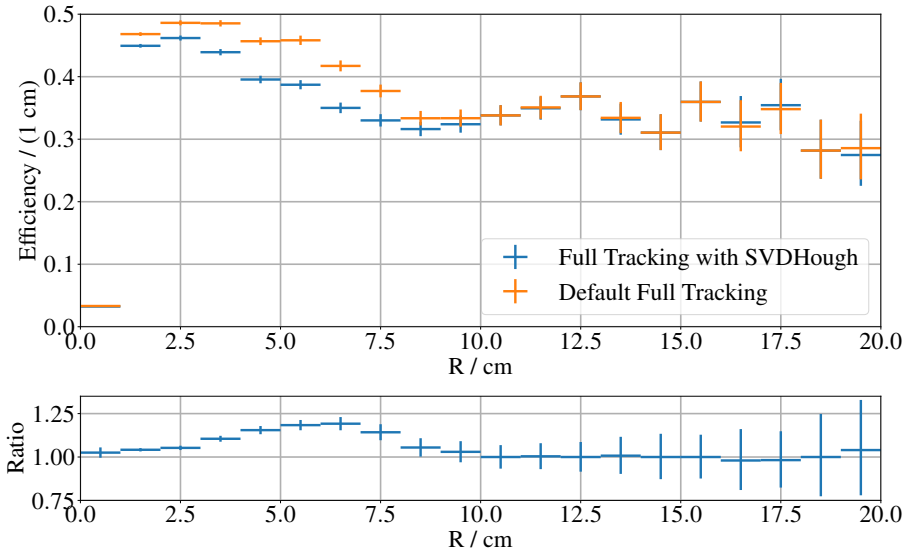


Figure 5.13: K_S^0 reconstruction efficiency as function of decay radius using the full tracking chain with SVDHoughTracking (blue) and VXDTF2 (orange, default). Below decay radii of 10.4 cm the usage of the VXDTF2 in the full tracking chain is beneficial for the reconstruction of K_S^0, at larger radii the reconstruction efficiency is only determined by the CDC tracking. The reduced efficiency using SVDHoughTracking is a result of two independent IP constraints during the track reconstruction.

pions are not visible in the tracking volume, only the decay in charged pions is of interest for this study. The main difficulty for the reconstruction is that both charged pions must be reconstructed, and a common vertex must be calculated. K_S^0 can only be successfully reconstructed if both of the pions are reconstructed and a common vertex is found.

The successful reconstruction of K_S^0 daughters by the SVDHoughTracking can be prevented by two features of the track reconstruction. First, during the conformal transformation the IP is used as reference point, and is thus assumed to be part of the track. However, tracks which do not pass the IP, or at least very close to it, could still be found in the HS. But a second IP requirement is used when probing two-hit combinations for whether or not they can be part of a track. Since a track fit is not possible with just two space points, the IP is used as a third virtual point on the track. The selection of possible two-hit combinations based on the p-value obtained from the track fit with the virtual IP can thus discard actual good combinations.

This is visible in Figure 5.13, which shows the reconstruction efficiency for K_S^0 as function of the decay radius for the full tracking chain using the SVDHoughTracking (blue) and the default tracking chain with the VXDTF2 (orange). For decay radii of up to 10.4 cm the pions produce at least three hits on the SVD and can thus in principle be found by the SVD standalone tracking. As the corresponding SVD hit and sector combinations are seen by the VXDTF2 during the training of the sector maps, the VXDTF2 is able to find both pion tracks more often than the SVDHoughTracking, enabling a higher K_S^0 reconstruction efficiency. This is independent of the K_S^0 momentum or its dip angle λ, as shown in Figure 5.14. Above decay radii of about 10.4 cm, at which the third SVD layer is located in the barrel region, the two pion tracks can only be reconstructed by the CDC standalone track finding, and SVD hits can solely be added to the tracks by the CDC-To-SVD-CKF, thus the efficiency ratio

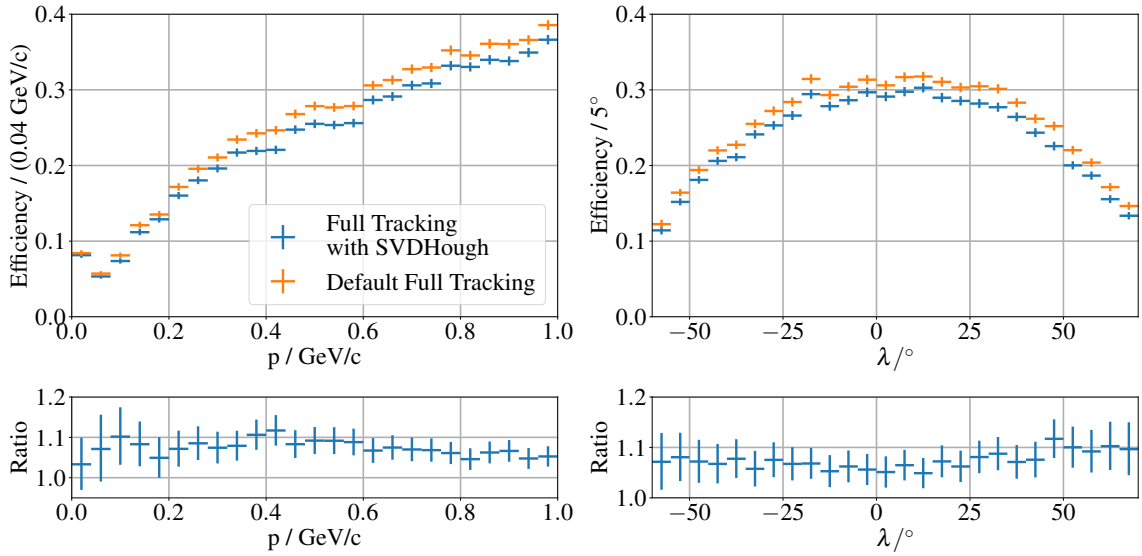


Figure 5.14: K_S^0 reconstruction efficiency as function of K_S^0 momentum (left) and dip angle λ (right) for the full tracking using SVDHoughTracking (blue) and VXDTF2 (orange, default). On average, using the VXDTF2 in the full tracking chain provides a 5 to 10 % higher K_S^0 reconstruction efficiency over the probed momentum and dip angle range.

is 1. Since the difference in reconstruction efficiency below radii of about 10 cm can be explained by the VXDTF2 finding the pion tracks, another track finding algorithm has the largest impact on the K_S^0 reconstruction efficiency outside of the third SVD layer, but also in general. For decay radii above 1 cm, i.e. outside of the beam pipe, the CDC standalone tracking has a large impact on the K_S^0 reconstruction, and thus needs to be improved to recover more K_S^0. Otherwise the K_S^0 reconstruction efficiency will remain limited to 50 % even in the ideal case of all K_S^0 stemming from the origin and without any beam backgrounds. The actual K_S^0 reconstruction efficiency in $B\overline{B}$ events is likely even lower because of the higher overall track density, and hits from beam backgrounds. Thus, before replacing the VXDTF2 in the full tracking as default SVD standalone track finding algorithms, further and more detailed studies are necessary to understand and recover the K_S^0 finding efficiency.

5.5 ROI finding performance studies on $\Upsilon(4S)$ MC events

The main objective of DATCON is to find ROIs effectively to help reduce the amount of PXD data to be stored. Since the development of the new SVDHoughTracking is based on the same concept, ROI creation is performed with the new track reconstruction algorithm presented in this chapter, too. Two methods of ROI calculation with the new track finding are compared to each other, to the ROI finding performance using the VXDTF2 only (as used in MC simulation) and the full tracking chain (as used on data). The first method is directly integrated into the newly developed tracking algorithm and can be enabled or disabled by a parameter in the steering file and the results obtained from this method are presented first. The second method, which is also used with the VXDTF2 and the full tracking, first performs a fit of the full track using the DAF and subsequently extrapolates the track onto the

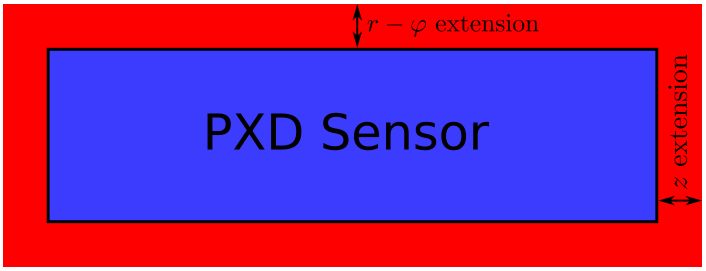


Figure 5.15: For the extrapolation to the PXD sensors, the extrapolated positions can also be in a range around the sensor (red). In these cases the ROIs still can extend into the active region to save PXD hits. The extension ranges for r-φ and z can be defined separately from each other, and they can take different values for the two PXD layers.

PXD sensor planes using the full helix track model with Genfit. It is therefore the default ROI finding algorithm.

5.5.1 ROI finding performance using a simple extrapolation

All tracks found are re-fitted with a triplet fit with an additional point at the IP to further constrain the fit, and because the fit yields the track parameters at the innermost point of the track, which would be the innermost SVD hit without the refit. From the momentum vector φ and λ are extracted, as well as the radius which is calculated as

$$r/\mathrm{cm} = \frac{p_T \,/\, \mathrm{GeV}/c}{0.00299792458 \cdot B/\,\mathrm{T}}$$

where r is the resulting track radius, p_T is the transverse momentum, and B is the magnetic field along the z-axis. Afterwards the extrapolation is performed in a similar way as for DATCON as sketched in Figure 4.4 and using the calculations in Equations (4.3) to (4.12). In addition to the calculations in these equations some corrections are included. First, the IP position is included as the starting point of the extrapolation, and its coordinates are rotated as

$$\begin{aligned} IP_{X,\mathrm{rot}} &= IP_X \cdot \cos\varphi_\mathrm{s} + IP_Y \cdot \sin\varphi_\mathrm{s} \\ IP_{Y,\mathrm{rot}} &= -IP_X \cdot \sin\varphi_\mathrm{s} + IP_Y \cdot \cos\varphi_\mathrm{s} \end{aligned}$$

with $IP_{X/Y}$ as the IP position in the lab frame, φ_s as the azimuthal angle of the sensor onto which the track is extrapolated, and $IP_{X/Y,\mathrm{rot}}$ as the rotated IP coordinates in the new reference frame.

Since neither track finding nor the track fit are perfect, the extrapolated positions on the PXD are not always within the active area of the PXD sensor they should have hit. In order to compensate for this effect, extrapolated positions contained in an extended acceptance can create valid ROIs. A sketch of this region is depicted in Figure 5.15 with the active area of the sensor coloured in blue, the extended region coloured in red. For each extrapolated hit contained within the active sensor area and the extended region a ROI is created. If it covers at least part of the active region, it is stored, and discarded if none of the four corner points are within the active area. Because the track radius usually is much larger than the sensor radius, a small error of the φ estimate has a larger influence on the extrapolation than a small error on the radius estimate. Thus, the extension range in r-φ is defined by

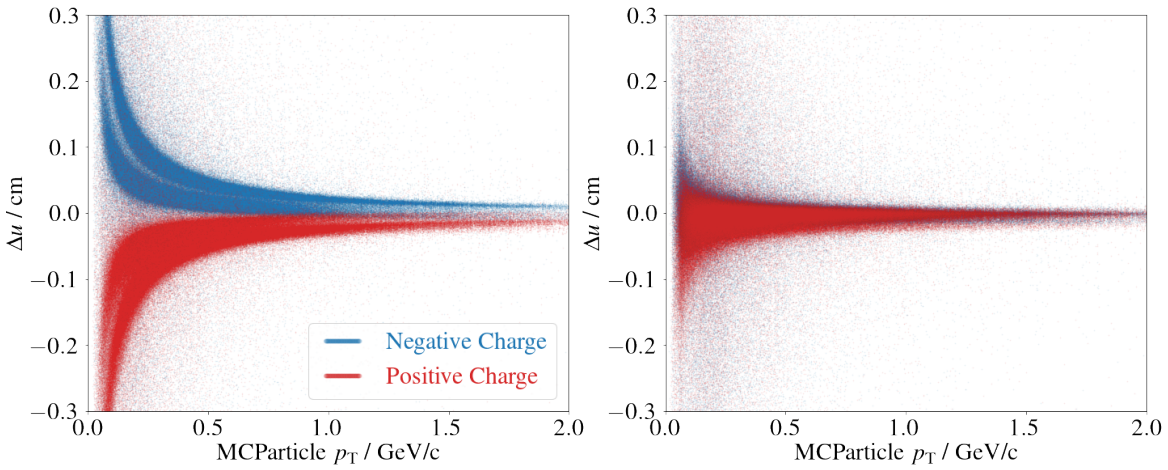


Figure 5.16: Residuals of the extrapolated tracks in u-direction calculated as $\Delta u = u_{\text{true}} - u_{\text{extrapol}}$ vs the transverse momentum p_{T} before (left) and after correction (right). Before adding a correction term, there is a dependence of the residual of the charge and the transverse momentum, as well as the layer on which the track is extrapolated to, visible as two distinct bands for both charges. A small asymmetry between the different charges remains after the correction.

$\Delta\varphi$ which is then converted into a length by multiplying it with the layer radius $\Delta(r\text{-}\varphi) = r_{sensor} \cdot \Delta\varphi$, resulting in different r-φ extension ranges for the two PXD layers. In contrast, the extension range Δz in z-direction is a fixed value and the same for both layers. The size of the ROIs are obtained by an optimisation similar to the one conducted for DATCON with fixed size ROIs in terms of number of pixels for each of the two PXD layers.

Using the extrapolation described above, a p_{T} and charge dependent bias of the extrapolated hits is observed in the residuals in u-direction with the residual being defined as $\Delta u = u_{\text{true}} - u_{\text{extrapol}}$, as shown in the left half of Figure 5.16. Since the residuals depend on the charge and layer, and increase with lower transverse momenta, they can be corrected. As the distribution appears to roughly follow $1/p_T$ behaviour, the correction term is calculated as

$$\text{correction term} = \frac{\text{correction factor} \cdot \text{track charge}}{p_T \,/\, \text{GeV}/c}$$

where the *correction factor* depends on the layer that is extrapolated to.

The result of the correction, after optimisation of the correction factor, is shown in the right half of Figure 5.16. Both the charge and p_{T} dependent bias nearly vanish, only a small asymmetry between the charges remains. Further reduction of the residuals could be achieved by using a p_{T} scaling that is not just reciprocal, but follows a $1/p_T^a$ behaviour with $a > 1$.

Nonetheless, the applied correction significantly improves the residuals, as shown in the left half of Figure 5.17 as a direct comparison of the u-residuals before and after the correction as a 1D projection. Before the correction the residual distribution is rather broad with a double-peak structure (orange), it is much narrower and with only one peak after the correction (blue). The residuals for the extrapolation in λ yielding the v-coordinate are shown the right half of Figure 5.17. The distribution is much broader than the u residual distribution. This is caused by the rather imprecise straight line extrapolation instead of using a helix or sine extrapolation, which would yield a more precise result, but is much

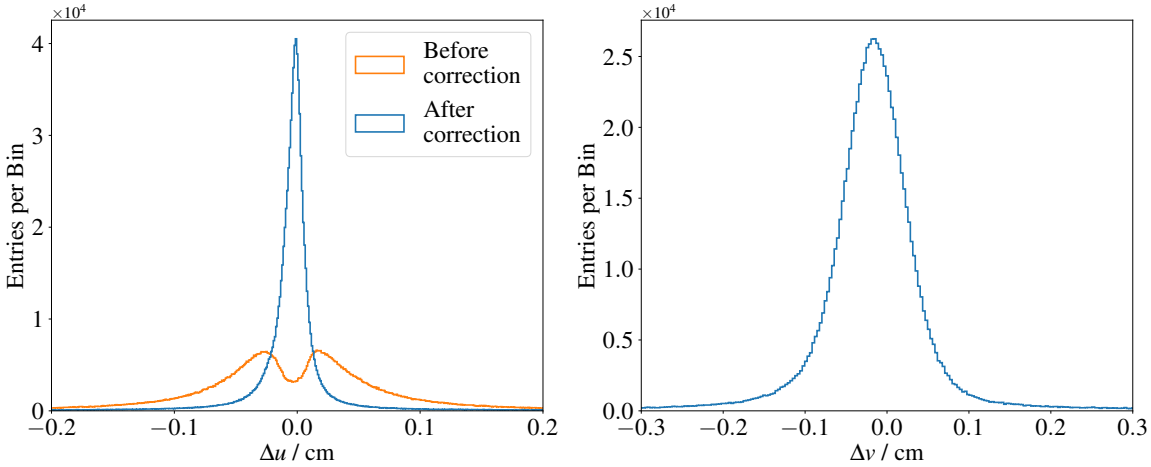


Figure 5.17: Residuals of the extrapolation in the u-direction (left) before and after correction, and in v-direction (right). The residual distribution in u-direction are much narrower compared to v-direction, which is caused by the straight line extrapolation in λ, compared to the track actually being represented as a sine. This is difficult to account for, but not impossible, and since the ROI finding efficiency is still high, no correction is performed.

more difficult to perform quickly in a similar amount of time. A residual of 1 mm corresponds to 20 pixels in u-direction, and between 12 and 18 pixels in v-direction depending on the layer and on position on the sensor.

ROI finding efficiency using a simple extrapolation

Figure 5.18 shows the ROI finding efficiency for the new SVD track finding with the simple PXD extrapolation (blue), VXDTF2 (orange), and the default tracking (green), both using the default ROI finding, as function of the transverse momentum p_T (left) and the dip angle λ (right).[4] While with ROI created by all three methods the efficiency is similar for tracks with very low transverse momentum (below 50 MeV/c, the new track finding with a simple extrapolation achieves the lowest ROI finding efficiency in the p_T region with the highest track density (50 to 300 MeV/c). For transverse momenta with values above 300 MeV/c all methods achieve more than 90 % finding efficiency with the default tracking performing best. The reduced efficiency in the p_T region below 300 MeV/c is a result of curling tracks, for which only parts of the tracks are reconstructed, fitted, and extrapolated to the PXD correctly. For all additional arms of the curlers, no extrapolation to the PXD is performed, and thus the corresponding hits are not contained in any ROI. This is seconded by the efficiency distribution as function of λ, where the efficiency shows the steepest drop around $\lambda = 0°$ for the new track finding with simple extrapolation. However, the new SVD tracking combined with the simple extrapolation yields the highest efficiency in the very forward and backward regions, where the efficiency of the full tracking is the same as the efficiency of the VXDTF2 due to lack of CDC hits, and thus tracks being reconstructed in the SVD only. This behaviour is caused by the increased track finding efficiency of the new algorithm in the forward and backward region, as shown in the right half of Figure 5.4. As the ROI finding efficiency for both VXDTF2 and the full tracking is reduced in the low p_T region, and similar for both algorithms in the forward and backward region, the ROI finding with the full tracking

[4] A description of the default PXD extrapolation and ROI creation is given in the next section.

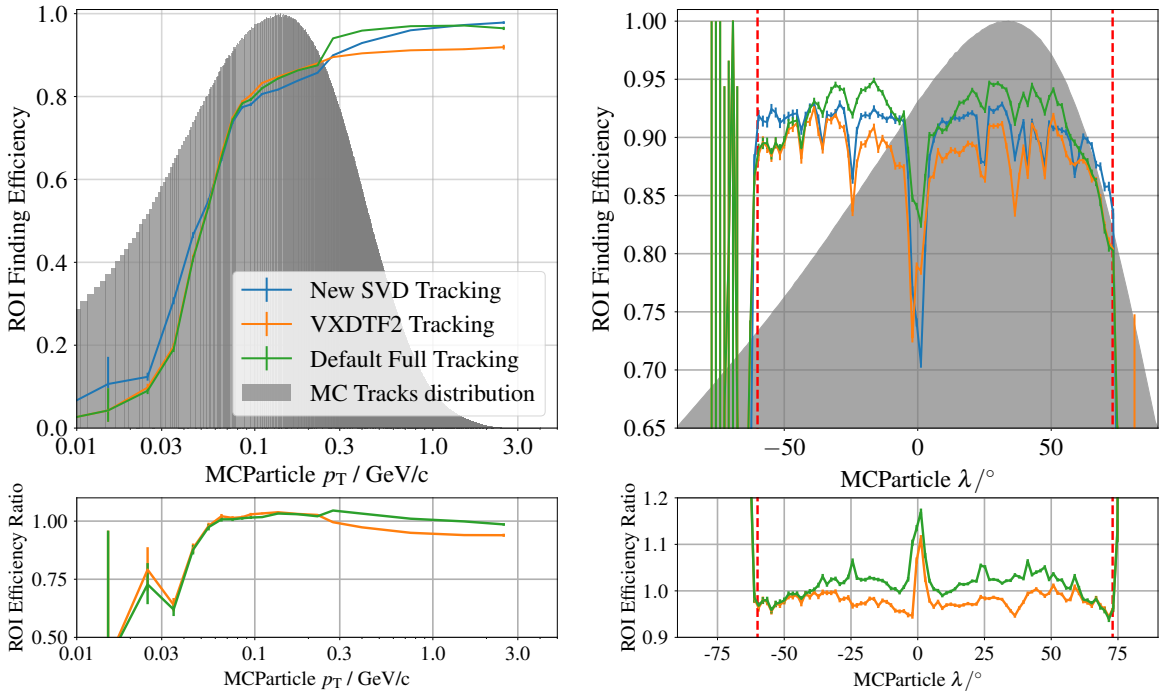


Figure 5.18: ROI finding efficiency using the simple extrapolation method (blue) as function of p_T (left) and λ (right) compared to ROIs created with VXDTF2 and full tracking. The ROI finding efficiency of the new tracking with the simple extrapolation compares similarly well as the other two methods, with a slightly worse efficiency in some regions of the p_T and λ spectra, and slightly better efficiency compared to at least one of the two other methods in other regions of the spectra. A more detailed discussion is given in the main part.

fully relies on the VXDTF2 in these regions. This reduction will complicate vertexing and possibly result in a low reconstruction efficiency for e.g. $D^{*+} - > D^0\pi^+$ decays, which is the dominant decay mode of D^{*+} mesons with 67.7 %. Because of the very low transverse momentum of these pions, a mis-reconstruction can subsequently spoil the reconstruction of the full B meson decay.

In addition, several other features are visible in the ROI finding efficiency distribution as function of λ in the right half of Figure 5.18. Most of the dips align with the angles of the insensitive SVD regions, which is expected from the decreased track finding efficiency in these cases. The steep drop for the new SVD tracking combined with the simple extrapolation around $\lambda = 0°$ however has two reasons. First, the insensitive region of SVD layer 3 is at $\lambda = -2°$, resulting in missing information for the simple χ^2 track fits used as a base for the extrapolation. Second, curling tracks have a large influence in this region, as they are nearly stationary in their z-coordinate and potentially create many PXD hits in close proximity.

Data Retention Fraction with the simple extrapolation

Next to the efficiency, the data retention is the second important FOM for ROI finding. A comparison of the DRF between the three used methods is shown in Figure 5.19 (right) together with the distribution of the number of ROI per event (left). While there are on average about 25 ROI per event from both VXDTF2 and the full tracking, the simple extrapolation used creates 32 ROI per event on average,

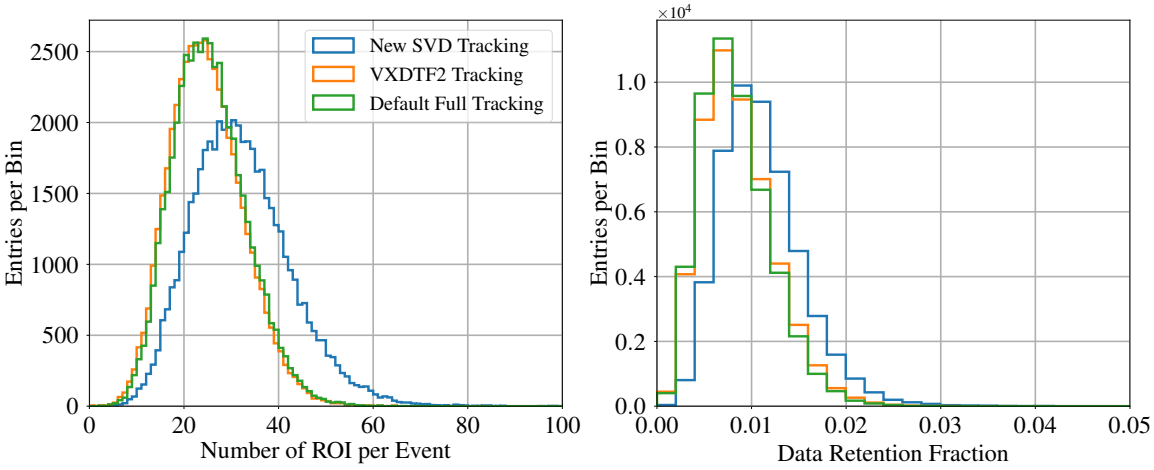


Figure 5.19: Number of ROI (left) and DRF (right) per event for the simple ROI finding. The averages of both quantities are increased compared to the two other ROI finding methods, where a larger number of ROI causes a higher DRF. The cause for this shift is the broad acceptance window around each PXD sensor which is depicted in Figure 5.15. Since the average DRF of 1 % is well below the target value of 10 %, this increase is acceptable.

Table 5.4: Summary of the ROI finding performance for the new SVD tracking with simple extrapolation, and VXDTF2 and full tracking with the default ROI creation method.

	Average ROI finding efficiency	Average Number of ROI per event	Average DRF per event
New SVD tracking	90.17 ± 0.13 %	32.06 ± 10.38	1.11 ± 0.44 %
VXDTF2 tracking	88.31 ± 0.14 %	24.90 ± 7.78	0.88 ± 0.39 %
Full tracking	91.78 ± 0.12 %	25.44 ± 7.77	0.84 ± 0.37 %

despite the fact that the number of reconstructed tracks is similar. The increased number of ROI is caused by a combination of slightly larger ROI on average with the new method, and because the extension region around the active PXD sensor region is chosen to be rather large. However, even creating ROI with two systems, e.g. with the default tracking and the new tracking, less than 60 ROI are created on average, which ONSEN can easily cope with. A similar effect is visible in the distribution of the DRF. As expected from the larger number of ROI from the new algorithm, the DRF is slightly higher with (1.11 ± 0.44) % compared to around 0.9 % achieved by the two established ROI calculation methods. Nonetheless, as the requirement on the DRF is to be less than 10 % on average, the achieved data reduction fully meets the requirement. The most important results of the ROI finding are summarised in Table 5.4.

5.5.2 ROI finding performance with the default Genfit extrapolation

While the simple extrapolation method proves to be fast and reliable yielding a high ROI finding efficiency over most of the phase space available for tracks in Belle II, it competes with the *PXDROIFinderModule* used in basf2. The PXDROIFinderModule requires a fitted track, which are usually fitted with GenFit, and uses the track model of GenFit itself and a Runge-Kutta-Nyström

algorithm [84] to extrapolate the tracks to the PXD sensors. Both track fit and extrapolation take a significant amount of time compared to both SVD only track finding algorithms with 20 to 25 ms (track fit) and 3 ms (extrapolation and ROI creation) per event, respectively. However, because the extrapolation is more precise, this method provides more accurate results in terms of residuals of the extrapolated hits and thus a higher ROI finding efficiency even with smaller ROI compared to the simple extrapolation approach described in the previous section.

After extrapolating to each PXD sensor plane, the track parameter uncertainties are used to define the size of the ROI in this plane as

$$u_{size} = n_{\sigma,u} \cdot \sqrt{\sigma^2_{\text{stat,u}} + \sigma^2_{\text{syst,u}}} \tag{5.1}$$

$$v_{size} = n_{\sigma,v} \cdot \sqrt{\sigma^2_{\text{stat,v}} + \sigma^2_{\text{syst,v}}} \tag{5.2}$$

yielding individual ROI sizes for each track instead of fixed sizes as used with the simple extrapolation. Here $n_{\sigma,u/v}$ denotes a constant factor the uncertainty is multiplied with, $\sigma_{\text{stat,u/v}}$ is the extrapolation uncertainty on the plane, and $\sigma_{\text{syst,u/v}}$ denotes the systematic uncertainty of the process. Both $n_{\sigma,u/v}$ and $\sigma_{\text{syst,u/v}}$ are user-definable variables of the PXDROIFinderModule, and in case of no extrapolation uncertainty the minimum ROI size is given by $n_{\sigma,u/v} \cdot \sigma_{\text{syst,u/v}}$. In addition, in order to avoid ROIs from becoming too large, a maximum size is defined for both directions. The result of the calculations in Equation (5.1) and Equation (5.2) is a ROI size in length but not number of pixels. Half of the size in each direction is added and subtracted to the extrapolated position on the detector plane and the resulting coordinates are then compared to the active sensor area, similar to the method employed with the simple extrapolation and depicted in Figure 5.15. As for the simple extrapolation, the values of Δz and $\Delta\varphi$ (to calculate the frame width in r-φ-direction as $\Delta r - \varphi = r_{sensor} \cdot \Delta\varphi$) are used to define the extended frame.

ROI finding efficiency

Using the default ROI creation with the tracks found by the SVDHoughTracking, the average efficiency improves by about 1.3 % to (91.16 ± 0.13) %. Its performance, along that of ROI creation based on tracks found by the VXDTF2 and the full tracking, respectively, is presented in Figure 5.20. The most significant increase in the efficiency is the p_{T} regime between 100 MeV/c and 500 MeV/c where many tracks are present, and in the very forward and backward regions, where up to 5 % more hits are contained inside a ROI at a given value of λ compared to the simple extrapolation. In addition, the tip around $\lambda = 0°$ is not as deep as before with the simple extrapolation, while further dips are visible at the λ values where the single SVD layers have insensitive regions. Between $\lambda = -30°$ and $\lambda = 60°$ the SVDHoughTracking combined with the default extrapolation is nearly on par with the ROI finding based on the full tracking, which achieves an efficiency of (91.78 ± 0.12) %. Although achieving a similarly high track finding efficiency as the SVDHoughTracking, the ROI finding efficiency based on tracks found by the VXDTF2 is significantly lower by more than 3 % at (88.31 ± 0.14) %. While the reasons for the difference in ROI finding efficiency, and the lower effiiciency using the VXDTF2 in particular, are not understood, they are also not topic of this work and thus not further investigated. One highly speculative hypothesis is the slightly worse SVD hit purity of the VXDTF2 tracking, causing a degradation of the track fit used for extrapolation. A second reason could be the slightly increased fake rate of the new SVDHoughTracking compared to the VXDTF2, where PXD hits of

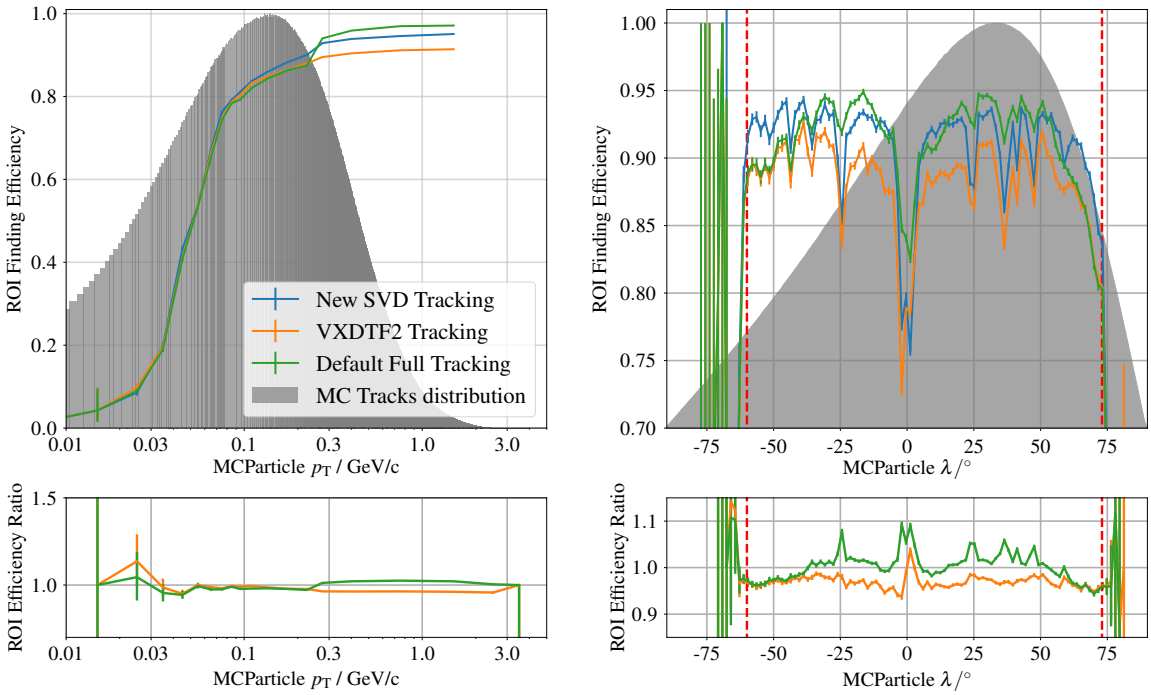


Figure 5.20: ROI finding efficiency using GenFit2 for extrapolation to the PXD. While the distributions for the VXDTF2 (orange) and full tracking (green) are the same as before, the ROI finding efficiency with the new SVD tracking (blue) is significantly improved, especially in the low p_T region up to 500 MeV/c. This is also represent by the λ distribution, where the dip at $\lambda = 0°$ is not as deep as before, and in the forward and backward regions in λ. In fact, the algorithm developed in this work provides the best ROI finding performance in the very forward and backward regions of the detector acceptance, while using the same extrapolation and ROI finding as the two well established algorithms.

physics tracks are contained in ROI created based on fake tracks, e.g. for tracks that curl back towards the PXD. Further investigations on this are needed to determine the cause of the different ROI finding performance between the two SVD track finding algorithms despite yielding a similarly high track finding efficiency.

Data Retention Fraction

Using the default PXD extrapolation and ROI calculation, the distribution of the number of ROI and the DRF per event are reduced compared to using the simple extrapolation, and comparable to the two established ROI calculation methods. As the SVDHoughTracking yields a higher track finding efficiency but also fake rate compared to the VXDTF2, the number of ROI per event created based on SVDHoughTracking tracks is marginally higher compared to the VXDTF2(25.48 compared to 24.90). With an average DRF of (0.90 ± 0.41) for the SVDHoughTracking, the target of 10 % is undershot by a factor of 10. The lowest DRF is achieved by the ROI finding based on the full tracking chain. Since the reconstructed tracks in this case contain CDC information, the estimate of the track parameters obtained from the track fit is much better compared to the two SVD methods, resulting in smaller ROI and thus a lower DRF despite higher ROI finding efficiency and a comparable number of ROI per

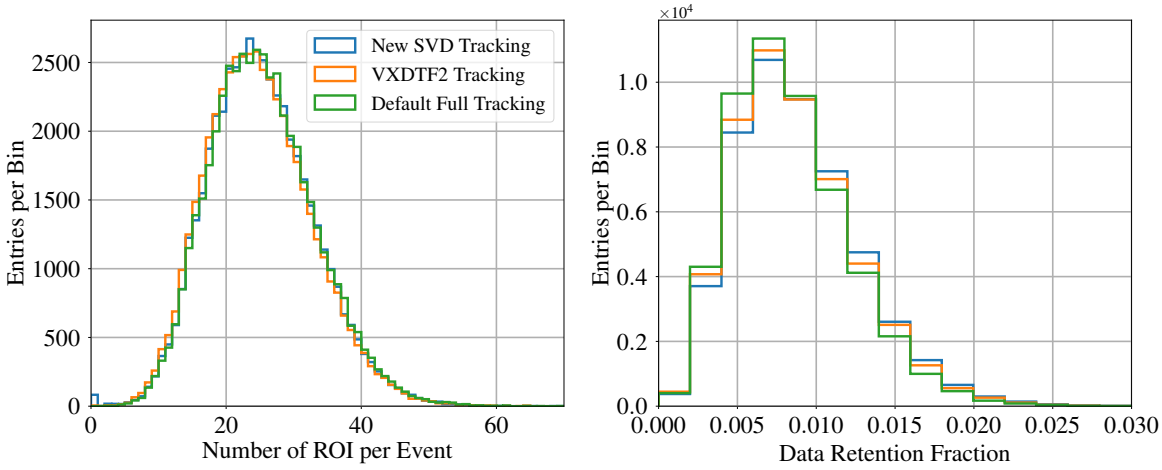


Figure 5.21: Number of ROI (left) and DRF (right) per event using SVDHoughTracking for ROI finding, compared to ROIfinding using tracks from the VXDTF2 and the full track finding. In contrast to the usage of the simple extrapolation, the number of ROI per event and the DRF are very similar for all three methods investigated. With the VXDTF2 a slightly lower number of ROI is created, which is represented by a lower ROI finding efficiency. The DRF using the full tracking is lower compared to the two SVD standalone tracking methods because the additional CDC hits improve the uncertainty of the track parameters which is used for the calculation of the ROI size.

Table 5.5: Summary of the ROI finding performance for the new SVD tracking, VXDTF2 and full tracking, all using the default ROI creation method.

	Average ROI finding efficiency	Average Number of ROI per event	Average DRF per event
New SVD tracking	91.16 ± 0.13 %	25.20 ± 7.79	0.89 ± 0.40 %
VXDTF2 tracking	88.31 ± 0.14 %	24.90 ± 7.78	0.88 ± 0.39 %
Full tracking	91.78 ± 0.12 %	25.44 ± 7.77	0.84 ± 0.37 %

event.

The results of the ROI finding performance of the new SVDHoughTracking and the two existing methods are summarised in Table 5.5.

Comparison of ROI finding and tracking

Since the ROI finding efficiency is reduced compared to the track finding efficiency, it is worth to compare them directly, as shown in Figure 5.22. In the low transverse momentum region ($p_T < 100\,\mathrm{MeV}/c$) the ROI finding efficiency is significantly lower than the track finding efficiency. The reason for this are curling tracks creating multiple PXD hits at different locations, only a fraction of which are contained in a ROI. This is also the cause of the large dip around $\lambda = 0°$.

For p_T values above 250 MeV/c, the ratio of ROI finding and track finding efficiency, asymptotically approaches unity for the new SVDHoughTracking and the full tracking, as shown in the lower part, while for the VXDTF2 the plateau in the ratio is at around 95 %. The conclusion from this comparison is that the ROI finding works very efficiently for all tracks found with the SVDHoughTracking and the

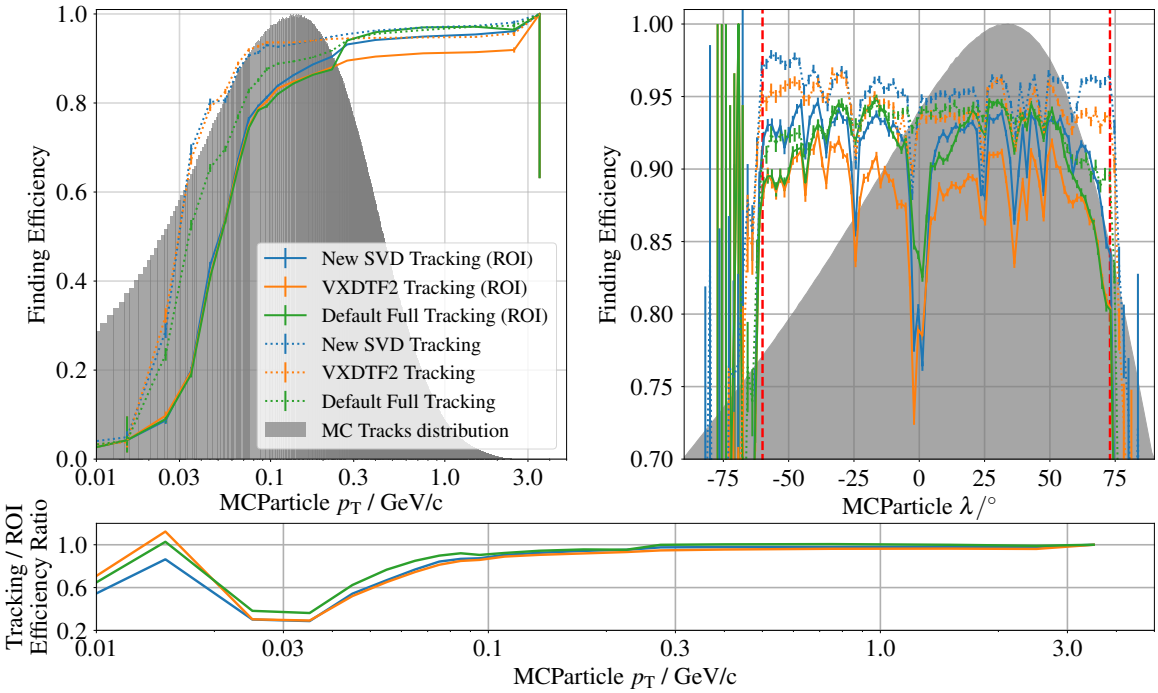


Figure 5.22: Direct comparison of the ROI finding efficiency (solid) and track finding efficiency (dashed) for the three algorithms as function of p_T (top left), λ (top right), and the ratio of the ROI finding over the track finding efficiency (bottom) as function of p_T. In the low p_T region of $p_T < 250\,\mathrm{MeV}/c$ the track finding efficiency is up to three times higher than the ROI finding efficiency. However, tracks in this region of the p_T spectrum curl and create PXD hits in multiple locations, only a part of these hits is contained in a ROI. Similarly, the track finding efficiency is higher for all three algorithms over the full λ range, where the drop ROI finding efficiency is highest around $\lambda = 0°$ where low p_T particles pass the PXD multiple times.

full tracking above transverse momenta of about 250 MeV/c where particles do not curl but leave the CDC, as the ratio approaches unity. If a track is found correctly in this regime, also the corresponding PXD hits are contained in a ROI, which, however, does not automatically result in them being attached to the tracks in the following steps. While curling tracks are usually found because the hit efficiency and purity requirements are met, only the first outgoing arm provides accurate information for ROI finding. Additional PXD hits on secondary or even tertiary arms are not contained in a ROI anymore and thus lost. The efficiency and purity of PXD hits that are attached to tracks either by the To-PXD-CKF following the single tracking algorithms, or by the full tracking after ROI finding, is investigated in Section 5.6.

5.5.3 Combination of ROI finding methods

Using the two existing track finding algorithms as well as the SVDHoughTracking as base for creating ROI on the PXD, at least 8 % of PXD hits are not contained in any ROI while achieving a data reduction by more than a factor of 100. As the three algorithms yield different track finding efficiencies in different regions of the tracking phase space, the ROIs found often are similar or even identical, but in many cases the algorithms find different ROIs because of the different track sets used. Thus, the

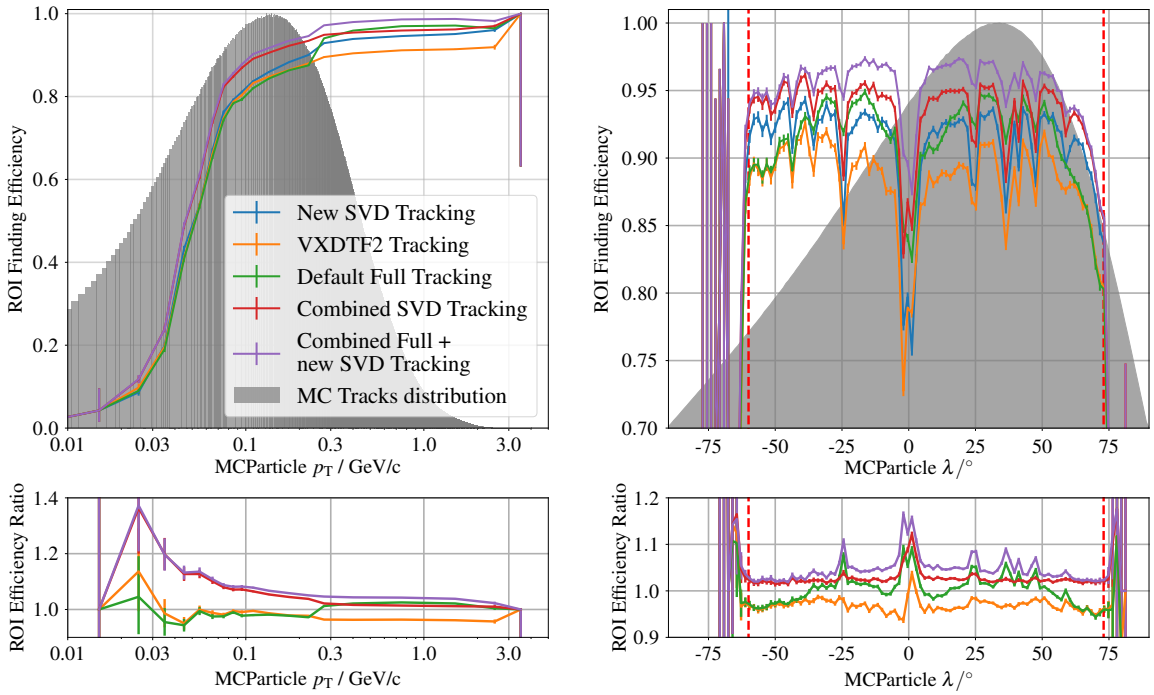


Figure 5.23: Comparison of the three ROI finding methods presented so far, and adding the SVDHoughTracking as a second ROI finding method after the VXDTF2 (Combined SVD Tracking) and the full tracking chain (Combined Full + new SVD Tracking). The ROI finding efficiency is increased by up to 5 % in some regions of the phase space.

ROI obtained from the individual algorithms can be combined to improve the ROI finding efficiency.

The results of adding the SVDHoughTracking as a second track and ROI finding method using the full set of SVD hits after the VXDTF2 and the full tracking is shown in Figure 5.23. Creating ROIs based on the track sets found with both VXDTF2 and SVDHoughTracking yields in a higher ROI finding efficiency of (93.42 ± 0.11) %, and is shown as red line in the figure. While the efficiency for $p_T > 300\,\mathrm{MeV}/c$ is similar to before, it is increased by up to 5 % in the p_T region below $300\,\mathrm{MeV}/c$ where the highest track density per p_T interval is seen. Similarly, the efficiency is increased over the full λ range compared to the two the ROI created with the tracks from the two SVD standalone algorithms, with the same distinct dips in the spectrum as before due to insensitive regions between the SVD sensors and due to curling tracks. Since most of the ROI found by the two algorithms are very similar, the DRF is only increased slightly to (1.01 ± 0.48) %, while the average number of ROI per event nearly doubles.

The improvement of using the new SVDHoughTracking after the full track finding is even larger, as shown by the purple lines in Figure 5.23. On average, (95.44 ± 0.09) % of all PXD clusters are contained in at least one ROI. For transverse momenta above $300\,\mathrm{MeV}/c$ the efficiency is constantly higher than 97 %. As before, the DRF increases slightly to (1.04 ± 0.48) % and the average number of ROIs doubles as expected. Both cases demonstrate the potential benefit of using a second ROI finding algorithm during data taking and MC generation. A summary comparing the combined ROI finding is presented in Table 5.6.

Table 5.6: Summary of the ROI finding performance using the new SVD tracking, the VXDTF2, the default full tracking, and combining VXDTF2 or full tracking ROIs with ROI finding with the new SVD Hough tracking.

	Average ROI finding efficiency	Average Number of ROI per event	Average DRF per event
New SVD tracking	91.16 ± 0.13 %	25.20 ± 7.79	0.89 ± 0.40 %
VXDTF2 tracking	88.31 ± 0.14 %	24.90 ± 7.78	0.88 ± 0.39 %
Full tracking	91.78 ± 0.12 %	25.44 ± 7.77	0.84 ± 0.37 %
Combined SVD tracking	93.44 ± 0.11 %	49.59 ± 15.24	1.01 ± 0.47 %
Combined Full + new SVD tracking	95.44 ± 0.09 %	49.91 ± 15.26	1.04 ± 0.48 %

Table 5.7: Summary of the ROI finding performance using the full tracking with the VXDTF2 as SVD standalone algorithm (default), or replacing it with the SVDHoughTracking.

	Average ROI finding efficiency	Average Number of ROI per event	Average DRF per event
Full tracking with SVDHoughTracking	92.96 ± 0.11 %	25.56 ± 7.79	0.85 ± 0.38 %
Default full tracking	91.78 ± 0.12 %	25.44 ± 7.77	0.84 ± 0.37 %

The creation of ROI with both tracks from both SVD standalone algorithms is of special interest, as during MC production, ROI currently are only created using tracks reconstructed by the VXDTF2. However, for $p_\mathrm{T} > 300\,\mathrm{MeV}/c$ the ROI finding efficiency using tracks from the full tracking is much higher compared to the usage of tracks of the VXDTF2. In these cases it is thus very likely that no PXD hit can be attached to the tracks during MC reconstruction, degrading the impact parameter resolution. Adding a second ROI finding method and restoring only a fraction of the otherwise missed PXD hits can help, as more tracks with precise vertex information will be available. This also holds for ROI creation on data, as PXD hits not contained in a ROI are inevitably lost and cannot be restored. Future improvements to the track reconstruction, for example potentially increasing the track finding efficiency in the low p_T regime, might not benefit from the precise PXD measurements if the tracks were not found during data taking and thus no ROI was created. Thus, despite the large potential of using two different ROI finding methods, the impact on PXD track finding needs to be studied in more detail to give quantitative results, which is presented the next section.

5.5.4 ROI finding with the SVDHoughTracking as VXDTF2 replacement

Because the usage of the SVDHoughTracking instead of the VXDTF2 in the full tracking proves to be advantageous, the same tracking setup is also used for ROI finding. The result is shown in Figure 5.24. On average, a ROI finding efficiency of (92.96 ± 0.11) % is achieved using the SVDHoughTracking, compared to (91.78 ± 0.12) % with the default tracking. Again, the largest improvements are visible in the very forward and backward regions in λ, but also over nearly the full p_T range. In the very forward and backward regions in λ improvements of up to 5 % are possible in comparison to ROIs based on the default tracking. All values including those for the number of ROI and the DRF per event are summarised in Table 5.7.

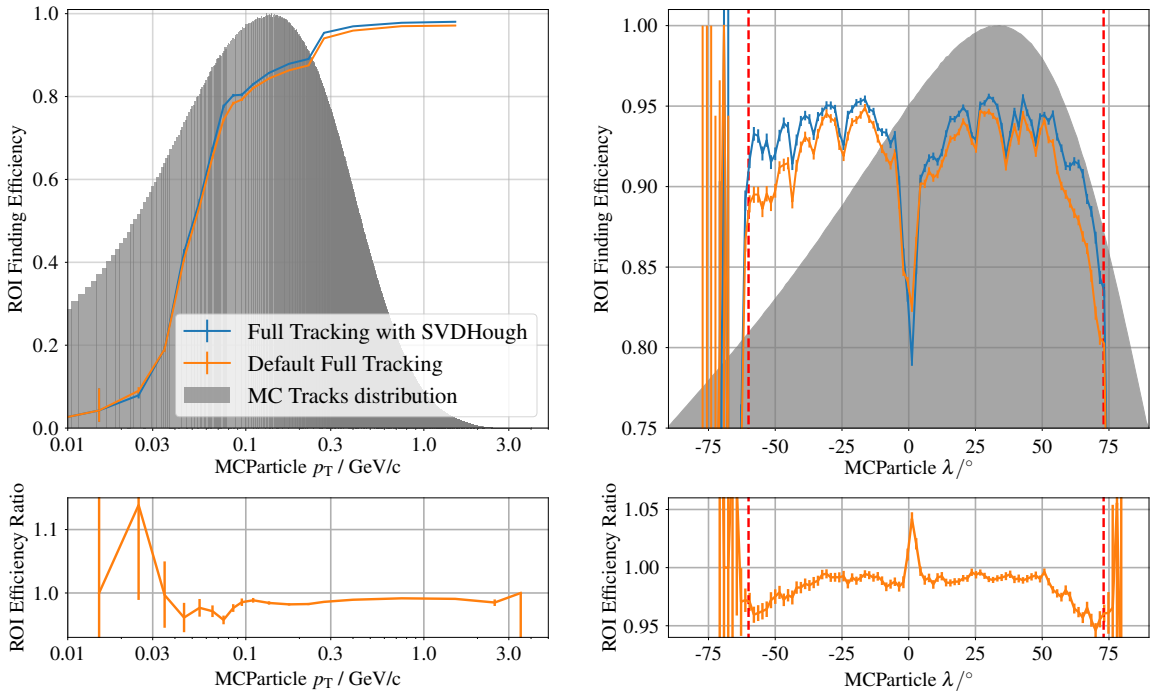


Figure 5.24: ROI finding efficiencies using the default full tracking chain (orange) and the full tracking with the SVDHoughTracking as a drop-in replacement for the VXDTF2 (blue). Using the SVDHoughTracking in the full tracking instead of the VXDTF2 yields a higher ROI finding efficiency over nearly the full p_T range and nearly full λ range. The usage of the VXDTF2 is only advantageous below $p_T = 30\,\mathrm{MeV}/c$ and around $\lambda = 0°$, with an average improvement of 1.2 % using the SVDHoughTracking.

5.6 Adding PXD hits to the tracks

As described in Section 3.2, information from the PXD is not used in the track finding step, but are added to previously found tracks using the To-PXD-CKF. The creation of ROI is essential for the CKF. During the relation creation step of the CKF, the number of relations is too high to be considered sensible without data reduction in the PXD. However, there is an interplay between track finding for the CKF, and the track finding for ROI creation: Tracks that are not found during ROI creation will result in loss of PXD hit information – but PXD hits that are contained in ROI will not be used if the corresponding track is not found during the actual tracking, but only during tracking for ROI creation. Thus, in this section the track finding performance with PXD hits, and the efficiency of adding PXD hits to the tracks is evaluated. This is evaluated with two different scenarios.

- Using a track finding algorithm for ROI creation and afterwards the same tracks to add PXD information:
  1. SVDHoughTracking → ROI finding → To-PXD-CKF
  2. VXDTF2 → ROI finding → To-PXD-CKF
  3. Full tracking → ROI finding → To-PXD-CKF (default for data taking)[5]

[5] For data taking, the ROI are created with the full tracking online, but the full tracking is executed again offline. The

- Using the SVD standalone track finding algorithms for ROI creation and the full tracking with all available information afterwards:
  4. SVDHoughTracking → ROI finding → full tracking
  5. VXDTF2 → ROI finding → full tracking (current default for MC data creation)
  6. VXDTF2 + SVDHoughTracking → ROI finding for each → full tracking

While the first three cases allow for the direct comparison of the three tracking algorithms for ROI creation and the subsequent attachment of PXD hits by the CKF, it is possible to estimate the benefits of using the new SVDHoughTracking for ROI finding during MC production with the latter three cases, either alone (case 4), or in addition to the VXDTF2 (case 6), which both needs to be compared to the current default (case 5).

As described in Section 5.5.3, the ROI finding efficiency is significantly increased when using tracks obtained from both SVD track finding algorithms to create ROI. Thus, the naive assumption is that the PXD hit efficiency increases in case 6 compared to case 5, or stays the same at least. The reason is that the PXD hits contained in the VXDTF2 ROIare still available as before, but additional ROI are obtained from the SVDHoughTracking. Additionally, the ROI finding efficiency of the full tracking is higher compared to the VXDTF2, which indicates that using only the VXDTF2 results in loss of PXD hits for the full track finding. This hypothesis is checked in the following.

Since PXD hits are only attached to existing tracks in a very last step using the To-PXD-CKF, the tracking FOMs are dominated by the CDC and SVD track finding. Only for a small fraction of tracks with few hits the matching status changes from *matched* to *fake* or vice versa after adding PXD hits, as this changes the overall hit purity value of the tracks, which has the largest influence on tracks only found in the SVD. Thus, the following sections focus more on the PXD hit efficiency and purity, as the other tracking FOMs (track finding efficiency, fake rate, and clone rate) only change slightly when adding PXD hits.

5.6.1 Adding PXD hits to the tracks used for ROI creation

As a first step, the same tracks used for ROI creation are used for adding PXD hits with the To-PXD-CKF afterwards, for the three track finding algorithms SVDHoughTracking, VXDTF2, and the full tracking (cases 1 to 3). As usually each found track results in ROIs on both PXD layers, the expectation is that the PXD hit efficiency is similarly high in all three cases. However, it must be remembered here that the hit efficiency and hit purity values are only based on successfully matched PR tracks.

First, if an algorithm finds only 5 % of all tracks correctly, but finds all, and only all, hits of those tracks it can achieve 100 % hit efficiency and purity for each of the subdetectors. Yet it still miss the vast majority of all hits, as 95 % of the tracks and their corresponding hits are not found.

Second, only the detectors used in track finding are also used for MC track finding and MC truth matching. Changing which detectors are used for truth matching can have significant impact on e.g. the track finding efficiency, as is presented in the summary in Table 5.1. In that case the track finding efficiency increases by about 2.6 % when using only SVD information instead of SVD and

reason is that only hit information are stored, but not the tracks created on the HLT. Due to improved offline calibration, the tracks used in the offline tracking can be different, or can have different track parameters. The tracking flow used here resembles the case where offline tracks and online tracks are the same, thus the tracks can directly be used as the input for the To-PXD-CKF after PXD data reduction.

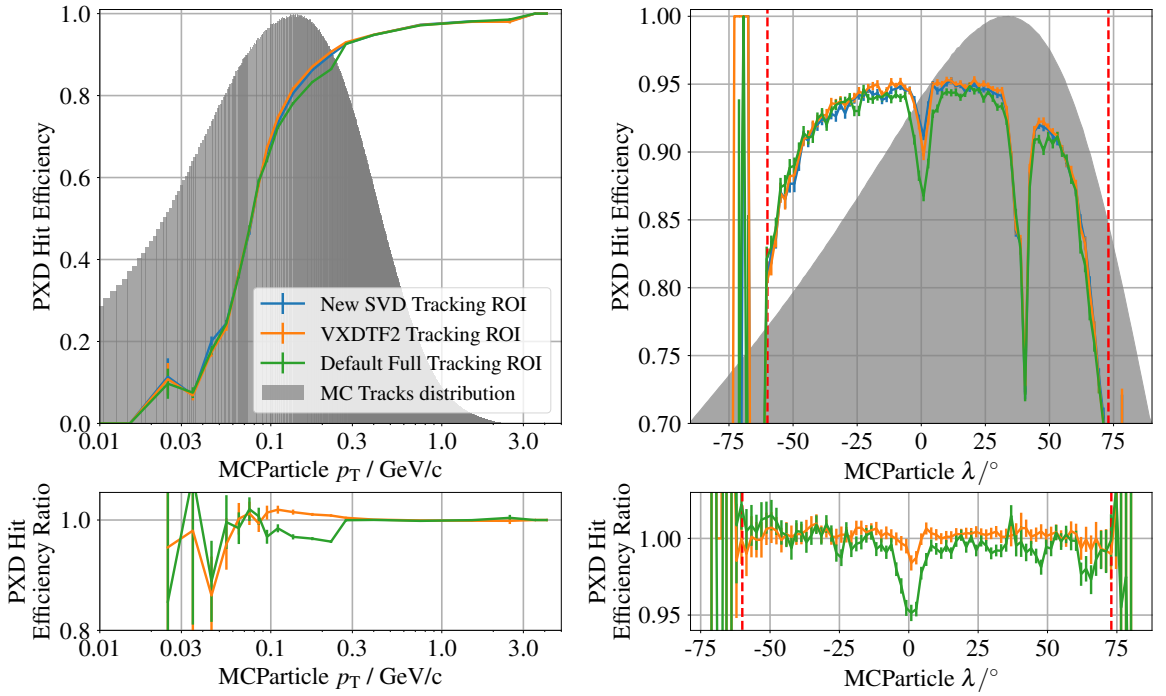


Figure 5.25: PXD hit efficiency when using the tracks uses for ROI creation are also used to add PXD hits to with the To-PXD-CKF. Using tracks found by the SVDHoughTracking for ROI finding to add PXD hits results in a PXD hit efficiency of (90.42 ± 0.13) %, similar to using the VXDTF2 for both tasks, while the overall track finding efficiency using the SVDHoughTracking is higher by about 1.3 %. Using the full tracking for ROI creation and PXD hit attachment yields the highest track finding efficiency, but a lower PXD hit efficiency.

CDC information for tracks found by the full track finding chain, indicating that often tracks were not merged correctly, or that too many wrong CDC hits resulted in a hit purity too low to fulfill the hit purity requirement for a matched track. As PXD hits are now added to the tracks, they also have to be utilised in MC truth matching. This will impact the track finding efficiency of all track finding algorithms, as tracks with two PXD hits and one SVD hit provide 6 degrees of freedom and can thus be fitted and contribute to the valid MC tracks. Thus, these tracks are now part of the truth sample, but cannot be found by any of the currently existing track finding algorithms, and subsequently the track finding efficiency will decrease because the number of found and matched tracks nearly remains the same, but the number of MC tracks in the denominator increases.

5.6.2 Full track finding after ROI selection with different SVD tracking algorithms

As during MC creation the ROI creation and selection is simulated by using only tracks found by the VXDTF2 instead of the full tracking to reduce the execution time of MC data production, the potential benefit of using a different set SVD tracking based ROI for adding PXD hits to tracks during the full tracking is investigated, corresponding to the cases 4 to 6. The PXD hit efficiency achieved with the three methods as function of p_T and λ is depicted in Figure 5.26. While all three distributions are similar for the two kinematic variables, combining ROI created with tracks reconstructed by both SVD track finding algorithms independently does not yield a higher PXD hit efficiency, contradicting the

Table 5.8: Summary of the tracking FOMs in case the tracks from SVDHoughTracking, VXDTF2, or full tracking, are used for ROI finding first, and then extrapolated to the PXD to attach PXD hits employing the To-PXD-CKF.

Values in percent	SVDHough (1) + ROI finding + To-PXD-CKF	VXDTF2 (2) + ROI finding + To-PXD-CKF	full tracking (3) + ROI finding + To-PXD-CKF
Track Finding Efficiency	92.71 ± 0.12	91.39 ± 0.13	93.32 ± 0.11
Fake Rate	6.61 ± 0.11	7.20 ± 0.12	8.11 ± 0.12
Clone Rate	0.72 ± 0.04	1.74 ± 0.06	3.85 ± 0.09
SVD Hit Efficiency	94.28 ± 0.10	95.09 ± 0.10	93.15 ± 0.11
SVD Hit Purity	99.34 ± 0.04	99.11 ± 0.42	99.28 ± 0.04
PXD Hit Efficiency	90.42 ± 0.13	90.57 ± 0.13	89.88 ± 0.14
PXD Hit Purity	94.84 ± 0.10	94.60 ± 0.10	95.09 ± 0.10

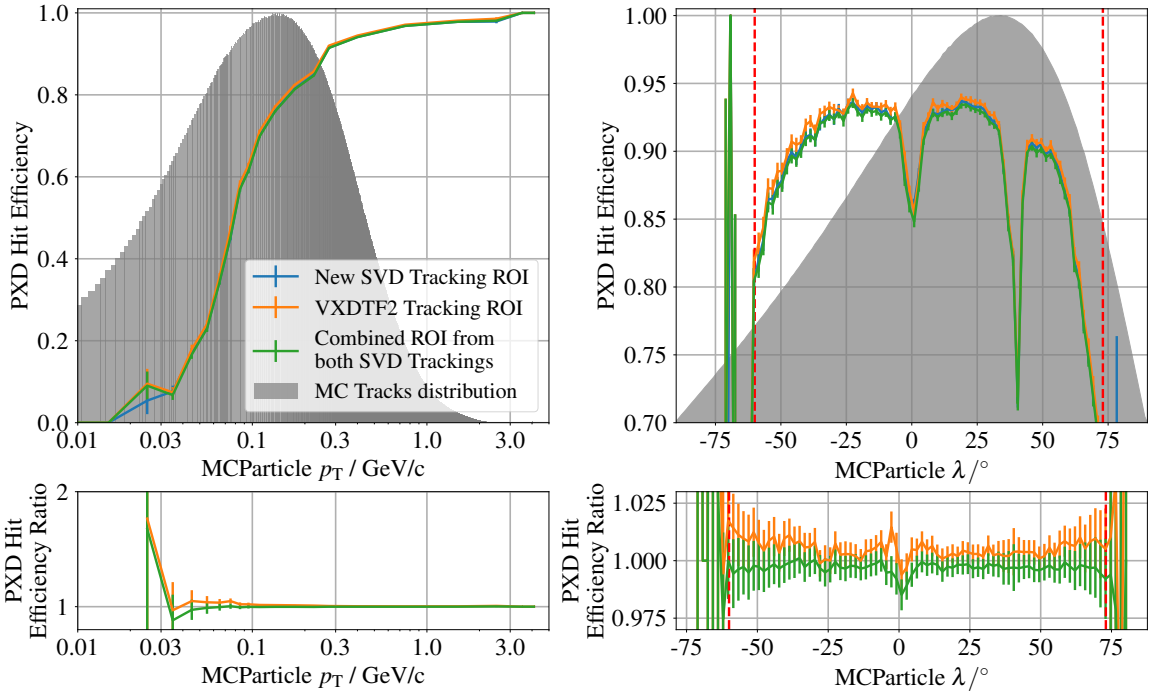


Figure 5.26: PXD hit efficiency when using SVDHoughTracking (blue), VXDTF2 (orange) for ROI creation, as well as combining both (green), followed by the default full track finding. In contrast to the expectation, using tracks from both SVD based track finding algorithms to create ROI yields a lower PXD hit efficiency, although the ROI finding efficiency is increased.

Table 5.9: Tracking FOMs in case the ROI finding is only based on the SVD standalone track finding algorithms, and the default full track finding is conducted afterwards with all SVD hits, and all PXD hits contained in any ROI created by the SVD track finding algorithms.

	SVDHough (4)	VXDTF2 (5)	SVDHough + VXDTF2 (6)
Values in percent		+ ROI finding + full tracking	
Track Finding Efficiency	92.50 ± 0.12	92.34 ± 0.12	92.31 ± 0.12
Fake Rate	7.83 ± 0.12	8.05 ± 0.12	8.06 ± 0.12
Clone Rate	3.90 ± 0.09	3.86 ± 0.09	3.86 ± 0.09
SVD Hit Efficiency	93.10 ± 0.11	93.14 ± 0.11	93.14 ± 0.11
SVD Hit Purity	99.25 ± 0.04	99.27 ± 0.04	99.28 ± 0.04
PXD Hit Efficiency	88.89 ± 0.14	89.24 ± 0.14	88.62 ± 0.14
PXD Hit Purity	95.70 ± 0.09	95.35 ± 0.09	95.36 ± 0.09

initial assumption. As the ROI based on VXDTF2 tracks are the same in cases 5 and 6, it is expected that at least the same PXD hit efficiency is achieved, but not a lower value.

Several possible explanations exist. First, the To-PXD-CKF was trained with PXD hits contained in ROI based on VXDTF2 tracks. This could potentially introduce a bias in the CKF MVA based filters. Second, it is possible that PXD hits in the additional ROI are attached to tracks which would not have had any PXD hits attached to them otherwise, likely found in the SVD only, or in the CDC without any SVD hits. Attaching wrong PXD hits to these tracks could lower the overall hit purity below the required threshold of 2/3, causing them to be assigned a fake track instead of being a matched track. This hypothesis is seconded by the fact that the overall track finding efficiency is slightly decreased in case 6 compared to case 5, as shown in the summary in Table 5.9, although the difference in efficiency is still in the margin of error. At this point, no final conclusion on the seemingly reduced PXD hit efficiency using two independent sets of SVD tracks for ROI creation can be made, but further investigations are necessary in the future.

5.7 Resilience against higher beam backgrounds

In the previous sections the track and ROI finding capabilities of the new SVDHoughTracking algorithm, as well as comparisons to the algorithms that are currently used for these tasks are discussed. Since the overall rate of beam induced backgrounds in the Belle II detector, and the VXD in particular, is not well defined, due to the lack of synchrotron radiation and injection backgrounds in the simulated background samples, it is important to test the tracking and ROI finding algorithms against higher background rates. Thus, both track finding and PXD ROI finding are applied to simulated data with twice the expected beam induced backgrounds. All results presented in this section are thus based on 20000 simulated $\Upsilon(4S)$ events with twice the expected beam backgrounds.

5.7.1 Track finding with increased beam background rates

As before, the track finding efficiency is evaluated first. The track finding efficiency as function of p_{T} and λ with twice the expected amount of beam backgrounds is depicted in Figure 5.27. Over nearly the full p_{T} and λ range the SVDHoughTracking yields the highest track finding efficiency, exceeding the

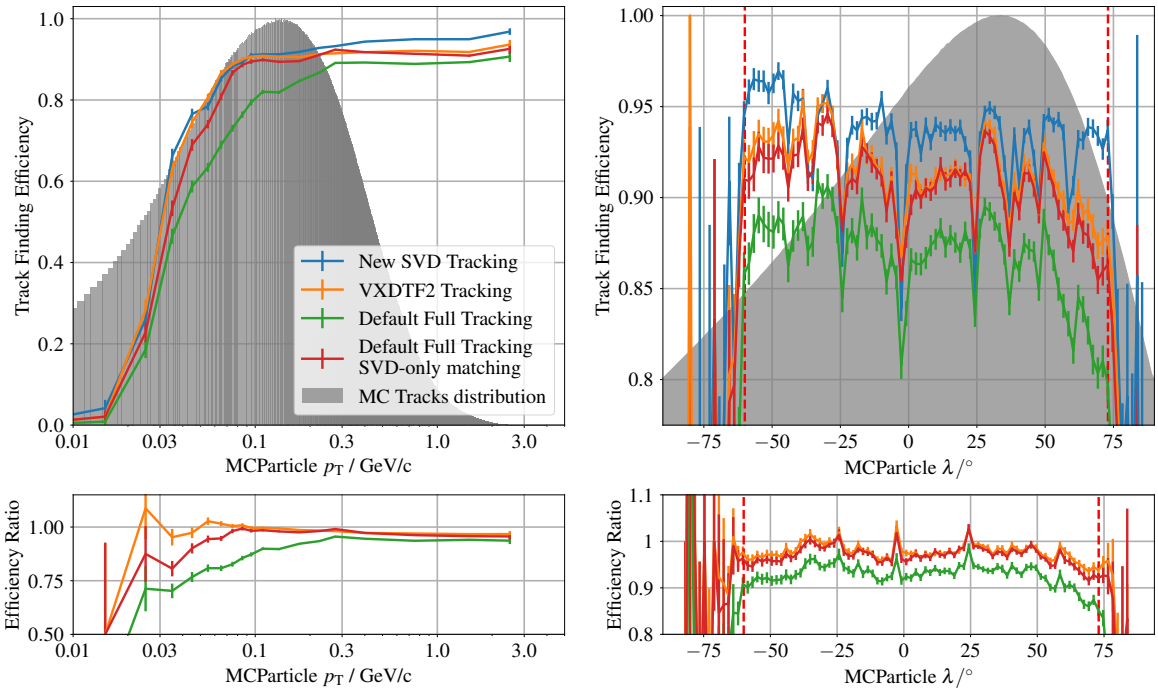


Figure 5.27: Track finding efficiency as function of p_T (left) and λ (right) for twice the expected background rates. The newly developed SVD tracking overall has the highest efficiency in general and in most of the p_T and λ spectra. As before, the distinct dips in the efficiency in λ align with the insensitive regions between the single SVD sensors in each layer.

full tracking (considering SVD and CDC hits) by up to 40 %, and up to 20 % only taking SVD hits into account. The efficiency is higher in the low p_T region, and over nearly the entire λ range. However, at (93.01 ± 0.18) % average track finding efficiency about 2 % fewer tracks are found compared to the nominal background case with the SVDHoughTracking. As expected, all features of the distributions remain the same, like the dips in the λ efficiency distribution.

In comparison, the SVDHoughTracking yields the highest track finding efficiency above 100 MeV/c. Both established tracking algorithms suffer significantly more from the harsher background conditions, with the VXDTF2 only achieving (91.11 ± 0.20) % compared to (93.89 ± 0.11) % at nominal background conditions, while the efficiency of the full tracking is reduced to (86.31 ± 0.24) % (nominal background: (92.92 ± 0.11) %) and thus loosing 6 % more tracks. All tracking performance results are summarised in Table 5.10.

Similar to the nominal background conditions, the new SVDHoughTracking obtains the highest fake rate below transverse momenta of 50 MeV/c, but achieves the lowest fake rate between 80 MeV/c and 800 MeV/c where the majority of tracks are. Also, the fake rate of the SVDHoughTracking is the lowest of all algorithms for $|\lambda| < 30°$, but increases drastically for larger values of $|\lambda|$ to about 60 % in the most forward part of the acceptance region. Around $\lambda = -50°$ and $\lambda = 50°$ the fake rate of the SVDHoughTracking is significantly higher compared to the other algorithms. However, the overall fake rate is the highest compared to the VXDTF2, and the full tracking evaluating all hits or SVD hits only. Due to the appearingly distinctive features of the tracks (low p_T and forward / backward) it

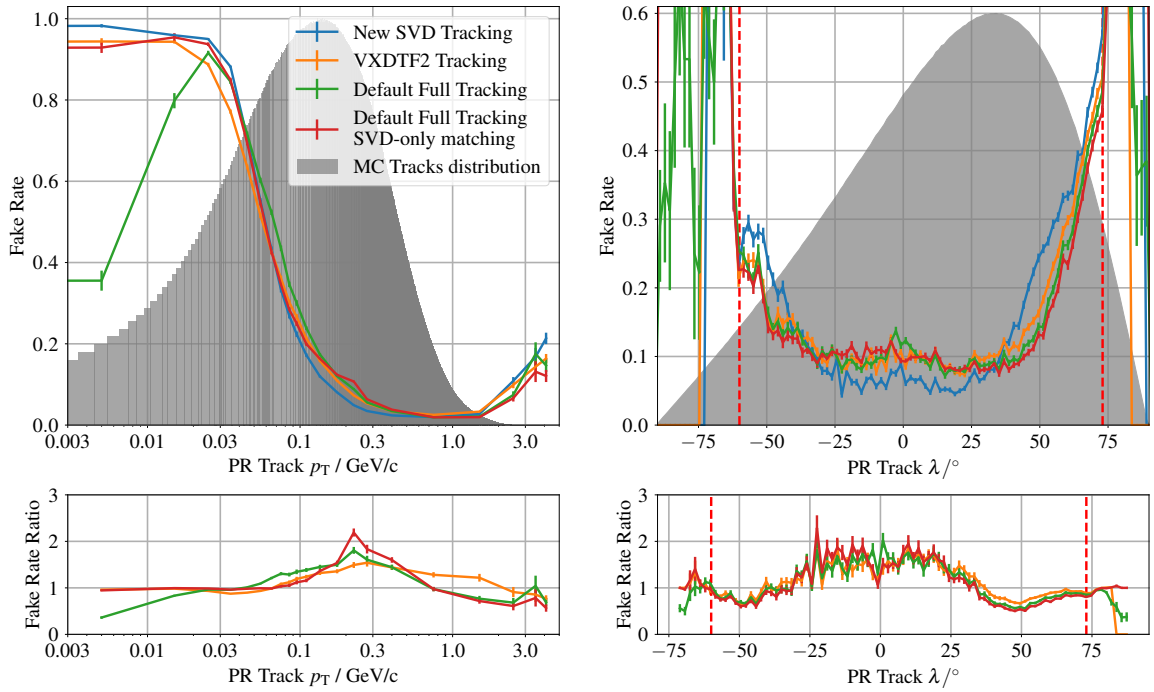


Figure 5.28: Fake rate as function of p_T (left) and λ (right). Below 50 MeV/c the new algorithm has the highest fake rate, approaching 100 % of fake rate below 30 MeV/c, while it finds the smallest number of fake tracks between 80 MeV/c and 800 MeV/c where most of the tracks of $\Upsilon(4S)$ decays are expected. As before, most of the fake tracks are found in the very forward and backward regions in the detector, and the SVDHoughTracking creates nearly twice the amount of fakes around $\lambda = 50°$.

seems likely that the fake tracks from background can be marked as such in the future with tools that need to be developed. Nonetheless, the higher track finding efficiency is of more value, especially considering that most of the fake tracks that the SVDHoughTracking finds are in the very low p_T region and would thus not be used by many physics analyses.

Since the clone rate and SVD hit efficiency and purity are similar to the tracking with nominal conditions, and similar between the algorithms, they are not depicted here but in **??** and summarised in Table 5.10.

Similar to the nominal background simulation, the SVDHoughTracking is used to replace the VXDTF2 in the full simulation with twice the nominal background rates. In this case the gain in track finding efficiency is even larger, as shown in Figure 5.29 and in the summary in Table 5.11. Overall an increase in track finding efficiency of about 2.50 % is achieved with the SVDHoughTracking in the full tracking chain compared to using the VXDTF2, both comparing the full set of CDC and SVD hits, or performing SVD-only MC matching. However, the fake rate is increased by about 0.55 %, while the clone rate is decreased by 0.7 %. As discussed previously, most of the tracks classified as fake are actual tracks from beam induced backgrounds. But the increase of about 10 % compared to the nominal background conditions indicates that additional random combinations of hits are accepted as a track. Some of the tracks classified as fake contain some hits from physics tracks, and some hits from background such that the final hit purity of these tracks is too low to consider the track as correctly found.

Table 5.10: Summary of the five most important FOMs for the three investigated tracking methods for twice the nominal beam backgrounds (from simulation). The last column contains results using the full track finding for track reconstruction, but only SVD information with the MC tracks and in MC matching to have a direct comparison of tracks that can be found in SVD alone.

Values in percent	New SVD track finding	VXDTF2	Full track finding	Full track finding SVD only matching
Finding efficiency	93.01 ± 0.18	91.11 ± 0.20	86.31 ± 0.24	90.35 ± 0.21
Fake Rate	19.16 ± 0.28	17.02 ± 0.27	15.74 ± 0.26	15.14 ± 0.25
Clone Rate	1.06 ± 0.07	2.11 ± 0.10	3.38 ± 0.13	1.73 ± 0.09
SVD Hit Efficiency	91.96 ± 0.19	92.77 ± 0.18	91.84 ± 0.19	93.10 ± 0.18
SVD Hit Purity	98.68 ± 0.08	98.08 ± 0.10	98.58 ± 0.08	98.47 ± 0.09

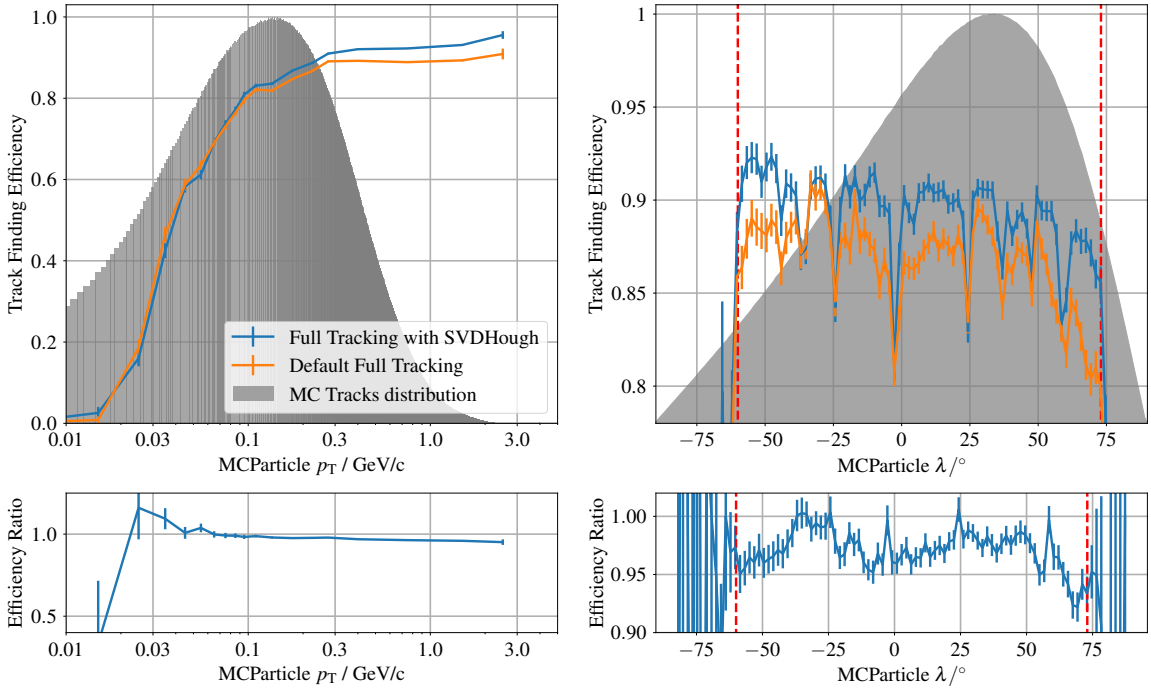


Figure 5.29: Track finding efficiencies using the default full tracking chain (orange) and the SVDHoughTracking as a drop-in replacement for the VXDTF2 (blue) for twice the nominal beam backgrounds. Below p_T values of 60 MeV/c the default full tracking provides a higher finding efficiency of up to 40 % around p_T = 30 MeV/c. Beyond that, the full tracking using the SVDHoughTracking algorithm as a replacement performs slightly better. The largest improvements are visible in the very forward and backward direction, i.e. for large values of $|\lambda|$.

Table 5.11: Summary of the most important FOMs for track finding with the SVD for replacing the VXDTF2 with the SVDHoughTracking for twice the nominal beam background rates. The columns two and three contain the FOMs for using both SVD and CDC information in MC tracks and for MC truth matching, while the data in columns four and five are based on the same track finding but only using SVD information in MC tracking and MC truth matching.

Values in percent	Full Tracking SVDHough	Default Full Tracking	Full Tracking SVDHough	Default Full Tracking
	CDC + SVD matching		SVD only matching	
Finding efficiency	88.80 ± 0.11	86.31 ± 0.11	92.75 ± 0.09	90.36 ± 0.09
Fake Rate	16.29 ± 0.12	15.74 ± 0.12	16.05 ± 0.14	15.14 ± 0.14
Clone Rate	2.69 ± 0.08	3.39 ± 0.09	0.88 ± 0.06	1.73 ± 0.06
SVD Hit Efficiency	91.02 ± 0.12	91.83 ± 0.12	92.20 ± 0.10	93.10 ± 0.10
SVD Hit Purity	98.88 ± 0.04	98.58 ± 0.04	98.83 ± 0.04	98.47 ± 0.04

5.7.2 ROI finding with increased beam background rates

After demonstrating that track finding works under harsher background conditions, efficient ROI finding needs to be confirmed. As the combination of ROIs created by tracks from two different tracking algorithms proves to be beneficial for the nominal background conditions, it is directly included here. This is shown in Figure 5.30. Compared to the VXDTF2 and the full tracking as base for ROI finding, the ROI finding efficiency is highest when using tracks reconstructed by the SVDHoughTracking, consistently over the full p_T and λrange, respectively. This is direct consequence of the higher track finding efficiency of the SVDHoughTracking algorithm.

Similar to the nominal background conditions, adding the SVDHoughTracking as a second ROI finding algorithm after the VXDTF2 or the full track finding improves the ROI finding efficiency. As before, this indicates that the two algorithms in most occasions find the same tracks, also considering the similar track finding efficiency, but that often one algorithm finds a track that is not found by the other. Although the average track finding efficiency decreases, the average number of ROI is similar as in the case of nominal beam backgrounds, and so is the DRF, as shown in Figure 5.31. However, since the number of background hits is increased by a factor of two, also twice as many PXD hits are contained in ROIs. An overview of the ROI finding performance with increased backgrounds is provided in Table 5.12.

In addition, the ROI finding with the SVDHoughTracking replacing the VXDTF2 is also tested, the result is shown in Figure 5.32. Using SVDHoughTracking in the full tracking for ROI finding instead of the VXDTF2 improves the ROI finding efficiency by about 4.5 % from (83.67 ± 0.26) % to (88.21 ± 0.23) %. Improved efficiency is visible over both the full p_T and the full λ ranges, where up to 10 % improvement are achieved in the very forward direction around $\lambda = 73°$. The average number of ROI per event increases slightly from 24.92 ± 7.81 to 25.15 ± 8.12, while the DRF is increased by about 0.07 % from (0.84 ± 0.38) % to (0.91 ± 0.41) %, which, however, still yields a higher data reduction factor than required.

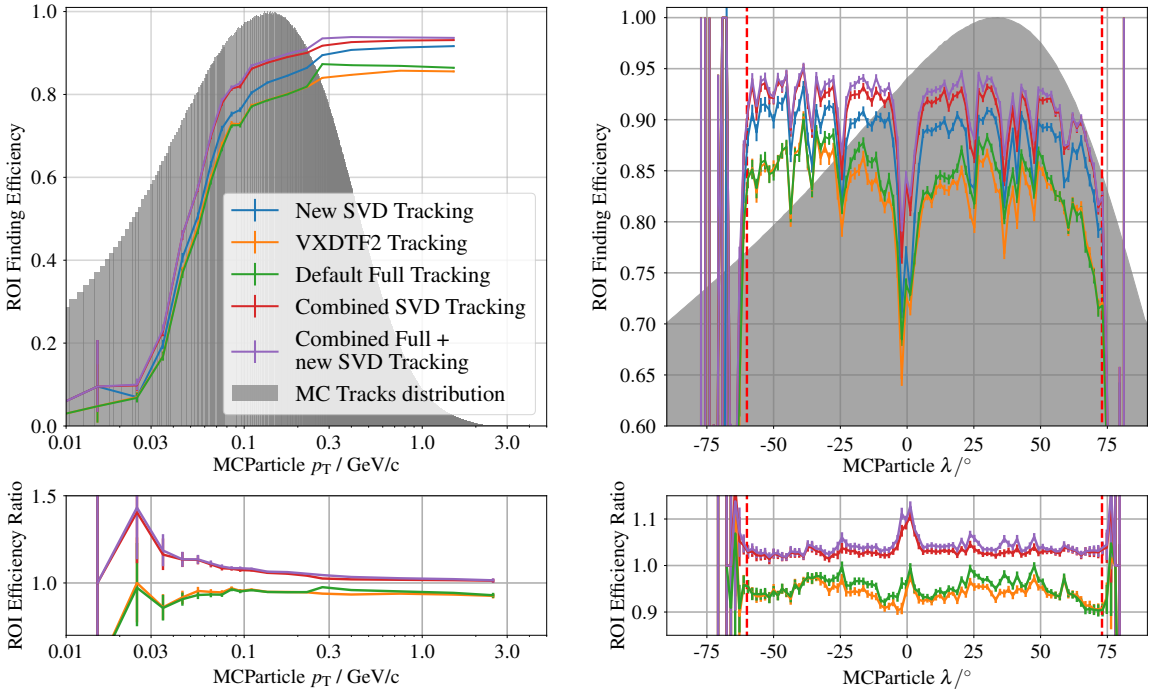


Figure 5.30: ROI finding efficiency with twice the nominal amount of beam background. Compared to the nominal background case, the efficiency is reduced by between 4 % and 8 %, with the new SVD tracking providing the highest efficiency compared to the VXDTF2 and the full tracking chain. Again, it is beneficial to add ROI finding with SVDHoughTracking tracks as a second step after the VXDTF2 and the full tracking, in which case on average more than 90 % efficiency are possible.

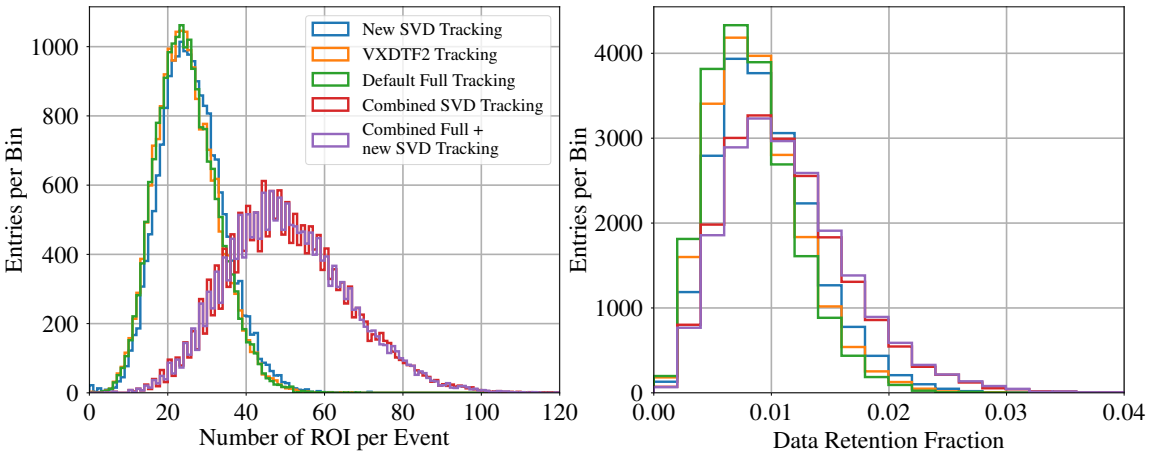


Figure 5.31: Number of ROI (left) and DRF (right) for twice the nominal expected background rates. In addition to the ROI finding with the SVDHoughTracking (blue), the VXDTF2 (orange), and the full tracking (green), also the distributions for combining both SVD standalone track findings (red), as well as the SVDHoughTracking with the full tracking (purple) are shown. Both the number of ROI per event, as well as the DRF are comparable to the ROI finding under nominal beam background conditions.

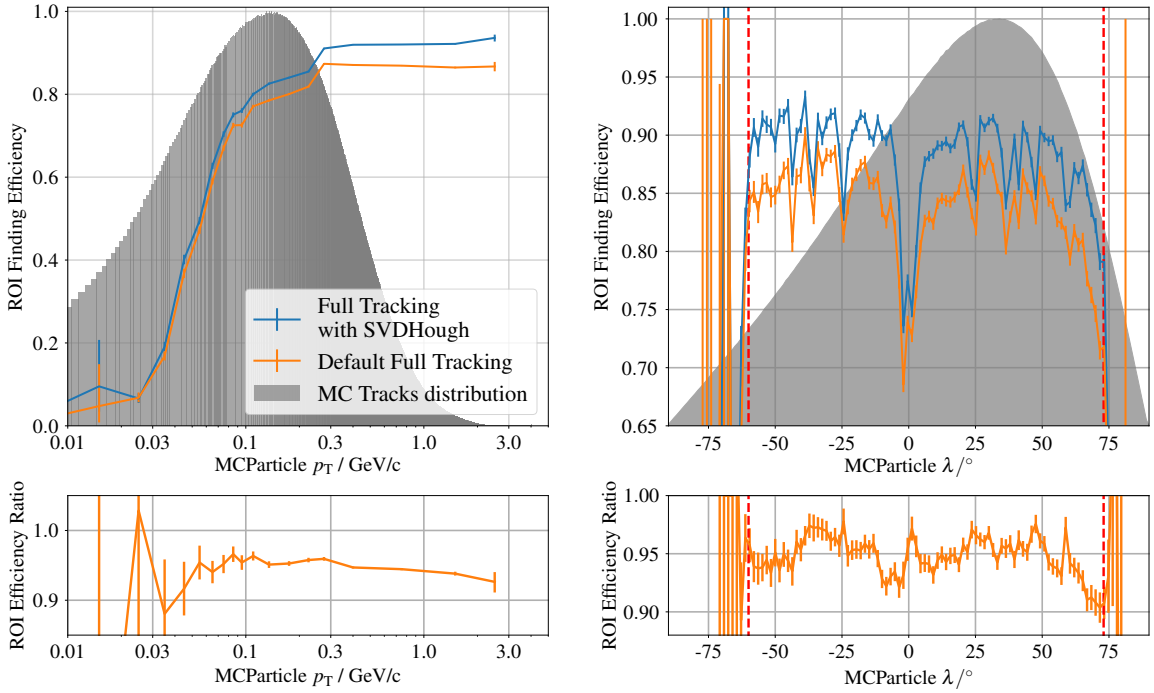


Figure 5.32: ROI finding efficiency with twice the nominal amount of beam background, comparing VXDTF2 (orange) and SVDHoughTracking (blue) in the full tracking. Using SVDHoughTracking in the full tracking for ROI finding instead of the VXDTF2 improves the ROI finding efficiency by about 4.5 % from (83.67 ± 0.26) % to (88.21 ± 0.23) %. At the same time the number of ROI per event increases slightly, as well as the DRF, which is still below 1 % on average.

Table 5.12: Summary of the ROI finding performance for twice the expected beam induced background. The shown values are for the standalone SVD track finding methods, the default full tracking, the full tracking with the SVDHoughTracking as replacement for the VXDTF2, and combining two ROI finding methods (both SVD standalone methods, and the default full tracking followed by the SVDHoughTracking).

	Average ROI finding efficiency	Average Number of ROI per event	Average DRF per event
New SVD tracking	87.77 ± 0.23 %	26.22 ± 8.18	0.95 ± 0.43 %
VXDTF2 tracking	82.51 ± 0.27 %	24.90 ± 7.85	0.88 ± 0.40 %
Default full tracking	83.67 ± 0.26 %	24.92 ± 7.81	0.84 ± 0.38 %
Full tracking with SVDHoughTracking	88.21 ± 0.23 %	25.15 ± 8.12	0.91 ± 0.41 %
Combined SVD tracking	90.37 ± 0.21 %	50.62 ± 15.49	1.13 ± 0.53 %
Combined Full + new SVD tracking	91.29 ± 0.20 %	50.64 ± 15.45	1.15 ± 0.54 %

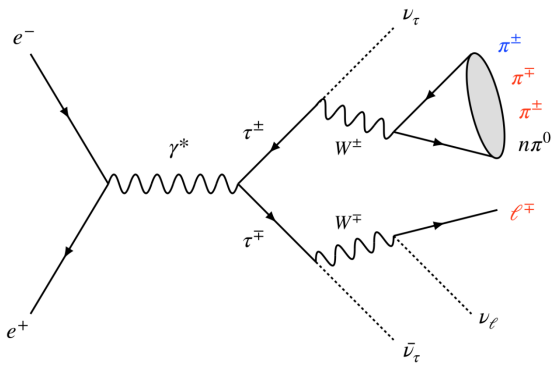


Figure 5.33: Feynman diagram for the 1-prong-3-prong $e^+e^- \to \tau^+\tau^-$ process. The three tag particles (l^+, π^-, π^+) are highlighted in red, and the probe pion is highlighted in blue, respectively (from [85]).

5.8 Tracking performance studies on τ-pair events

To evaluate the track finding efficiency and the fake rate on data, a process with a clear signature is needed, as no truth information for comparison is available. $\Upsilon(4S)$ decays are not suitable for this purpose because the decay chains often incorporate neutrinos which cannot be found, and it might be unclear whether a particle was just not found, or outside of the acceptance region of the Belle II detector. In addition, the variation in the number of tracks per event is rather large, rendering it difficult to define the number of missed tracks. A clean signature can found in the decay of the two τ leptons originating from the process $e^+e^- \to \tau^+\tau^-$. Although the decay of a τ lepton produces at least one neutrino, the overall signature is different from that of a $\Upsilon(4S)$ decay, and the number of tracks created by each τ lepton is limited. This study was conducted for the full Belle II tracking in [85] where further information can be found. Key features of the study are summarised in the following.

At the energies accessible with the SuperKEKB accelerator the production cross section for the production of a τ-pair is nearly as large as the one for the production of a B meson pair from a $\Upsilon(4S)$ decay, yielding a large number of events to study with approximately 1 million events per fb. In addition, the momentum of the decay particles covers a large range, making this process suitable to study the tracking performance in a large momentum range from about 200 MeV/c up to 3.5 GeV. The study uses processes where one of the two τ leptons decays hadronically into three charged pions via $\tau \to 3\pi^\pm\nu_\tau + n\pi^0$.[6] The second τ lepton decays leptonically via $\tau \to l\nu_l\nu_\tau$ with $l = e, \mu$. This signature is called *3-prong-1-prong* and the method is called *tag-and-probe*. While each event contains four neutrinos, the four tracks provide the necessary clean signature. With this, two channels are defined from the 1-prong τ decay:

- *electron channel*: the 1-prong track is an electron with $\tau \to e\nu_e\nu_\tau$
- *muon channel*: the 1-prong track is a muon with $\tau \to \mu\nu_\mu\nu_\tau$

Due to mis-identification there might be a small contribution stemming from the hadronic 1-prong decay $\tau \to \pi^\pm\nu_\tau + n\pi^0$ with the charged pion being identified as an electron (1.89%) or a muon (6.36%).

[6] For simplicity the τ and the ν_τ are noted in a generic way, avoiding an assignment of being a particle or an anti particle.

Two of the charged pions as well as the light lepton are called the *tag* particles, while the remaining charged pion is the *probe* particle. In some occasions during the analysis, a difference is made between the two tag pions having same (opposite) charge, indicated by the abbreviation SS (OS). The process is depicted in Figure 5.33. In this scenario the track finding efficiency ϵ_{track} can be defined as

$$\varepsilon_{track} \cdot A \;=\; \frac{N_4}{N_3 + N_4} \tag{5.3}$$

where A is an acceptance factor to incorporate the spatial acceptance of the Belle II detector for the probe track, N_4 is the number of events in which all four tracks are found, with zero total charge of the four tracks. N_3 is the number of events where the probe track is not found, i.e. only the three tag tracks are found. For simplicity, the track samples will be referred to as 4-track (N_4) and 3-track (N_3) samples, respectively.

5.8.1 Event and track selection

As the τ-pair events have a specific topology, they can be pre-selected by requiring a subset of the L1 trigger channels to fire which are:

High energy threshold trigger. The total energy deposited in the ECL needs to be above 1 GeV, while the event is not classified as an ECL Bhabha event.

Low multiplicity three cluster trigger. At maximum three neutral ECL clusters are allowed, at least one of which with a cluster energy of more than 300 MeV/c, while the event is not classified as an ECL Bhabha event.

Only one of the two ECL triggers is required to give a positive signal for an event. A study of their efficiency with respect to different CDC track triggers is provided in [85].

After selecting the events based on the L1 trigger decision, the single tracks need to be selected, and background tracks and events need to be removed. The τ-pair events are required to contain two good pion tracks as well as one good lepton track. They are selected based on the transverse momenta of the tag tracks, as well as their distance from the IP at their POCA, as shown in Table 5.13. Overall, the charge of the three tag tracks needs to be ± 1. While the probe pion only needs to fulfil a rather loose selection (c.f. column 2 of Table 5.13), the three tag tracks have to satisfy more stringent selections. Among these, the selection criteria for the two lepton modes are orthogonal to each other, and they are orthogonal to the tag pion tracks, too.

Different track multiplicities are required in the different samples, as summarised in Table 5.14. Due to the selection criteria, the tag pion track set is a subset of the probe pion track set.

5.8.2 Background suppression

After the event and track selection, a non-negligible contribution of background tracks is left, e.g. from continuum events, radiative di-lepton processes ($e^+e^- \to l^+l^-\gamma,\ l = e, \mu$), and two-photon background creating two muons ($e^+e^- \to e^+e^-\mu^+\mu^-$) are contained in the sample of selected events. They need to be removed or at least suppressed as much as possible. Thus, additional selections are performed. All selections and vetos were carefully optimised in [85] on MC and compared to data. For this reason the same selections are used in this work and directly applied to data. Additional information and comparison figures can be found in the reference.

Table 5.13: Track selection criteria. The first three rows mark the selections defining the *good tracks* used for this analysis (from [85]).

	Probe pion track	Tag pion track	Tag electron track	Tag muon track
p_T / GeV/c	-	> 200	> 200	> 200
$\lvert z_0 \rvert$ / cm	< 3	< 3	< 3	< 3
$\lvert d_0 \rvert$ / cm	< 1	< 1	< 1	< 1
$\frac{E_{\text{cluster}}}{p}$	< 0.8	< 0.6	(0.8, 1.2)	< 0.6
E_{cluster}	-	> 0	-	> 0
muonID	-	< 0.9	-	> 0.9

Table 5.14: Track multiplicity criteria. Notice that the $N^{\text{tag}}_{\text{pion}}$ list is by definition contained in the looser list $N^{\text{probe}}_{\text{pion}}$. So for the 3-track samples where no additional pion probe is needed, the candidate multiplicity for the two lists is required to be exactly the same (from [85]).

	$N^{\text{probe}}_{\text{pion}}$	$N^{\text{tag}}_{\text{pion}}$	$N^{\text{tag}}_{\text{electron}}$	$N^{\text{tag}}_{\text{muon}}$
electron channel, 3-track sample	2	2	1	0
electron channel, 4-track sample	3	≥ 2	1	0
muon channel, 3-track sample	2	2	0	1
muon channel, 4-track sample	3	≥ 2	0	1

Angular isolation requirement. The τ leptons are in different hemispheres in the CMS frame, leading to an angular separation of their daughters. Thus, in the CMS frame the angle between the 1-prong track and each of the 3-prong tracks has to satisfy $\cos\theta^* < -0.5$.

Vertex fit quality. The two probe pions need to originate from a common vertex. Using *Rave* [86] to fit the vertex, the fit needs to yield a p-value of more than 0.01 (Prob($\chi^2 > 0.01$)).

Neutral particle multiplicity. At most one π^0 and two additional good photons are allowed. The single neutral particle criteria are:

- π^0 reconstruction: $E_\gamma > 100\,\text{MeV}/c$, -0.8660 < $\cos\theta_\gamma$ < 0.9563, 115 < $m_{\gamma\gamma}$ < 152 MeV/c^2, clusterNHits > 1.5
- good γ reconstruction: $E_\gamma > 200\,\text{MeV}/c$, -0.8660 < $\cos\theta_\gamma$ < 0.9563, clusterNHits > 1.5, not a π^0 photon

1-prong momentum selection. In order to suppress continuum and $e^+e^- \to e^+e^-\mu^+\mu^-$ background, the 1-prong track is required to have a momentum of at least 20 % of the beam energy. Requiring the 1-prong momentum to be below 80 % of the beam energy reduces radiative di-lepton backgrounds.

2-prong angular separation. The opening angle between the two tag pions is required to be greater than 0.05 (0.2) rad for the muon (electron) channel, respectively.

Mass selections. The invariant mass of all tag tracks (M_{tag}) must be below 8.5 GeV, and the invariant mass of the two tag pions ($M_{\pi\pi}$) is required to be smaller than the τ lepton mass.

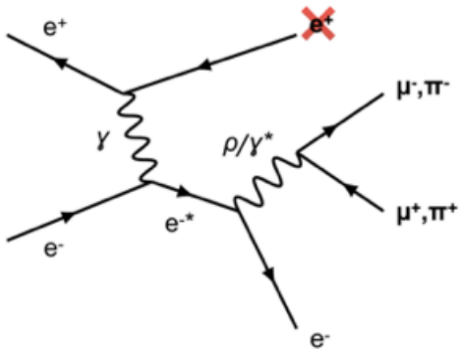


Figure 5.34: Feynman diagram for the $ee\gamma^*$ background where one of the two electrons is missed, as indicated by the red cross on the positron, and a μ-pair or π-pair is created from the virtual photon (from [85]).

***ee*γ^* veto.** After all previous selections, a significant contribution of what appears to be $ee\gamma^*$ is left in the 3-track sample of the electron channel with both tag pions having opposite charge (electron OS-channel). The process is depicted in Figure 5.34. A $ee\gamma^*$ enriched sample in data is used to define a veto against it. This sample contains events with 1-prong tracks with transverse momenta $p_T \in (2, 3.7)$GeV/c and opening angles between the 1-prong and each of the 2-prong tracks $\in (120°, 150°)$. Using these information, the $ee\gamma^*$ veto is defined to discard events in the electron OS-channel if the missing mass squared (M^2_{miss}) is below 20 GeV^2/c^4, and the dip angle λ is above 50° or below −45°.

In contrast to the study performed by the Belle II tracking group, the track finding efficiency study is not performed individually for the two leptons, and for the same sign and opposite sign channels of the tag pion pair, but only on the combined sample. Also, no data-MC correction factor is calculated for the three different track finding methods due to the lack of MC data.

5.8.3 Track finding efficiency with τ-pair events

Figure 5.35 shows the track finding efficiency as function of p_T and λ using the di-τ tag-and-probe method for the new SVDHoughTracking (blue), the well established VXDTF2 (orange), and the default full track finding (green) using $\int \mathscr{L}\, dt = 16.267\,\text{fb}^{-1}$ of data recorded in 2021. With an efficiency of (92.32 ± 0.13) % on average, the SVDHoughTracking yields the second best performance with only the full tracking achieving a higher efficiency of (92.92 ± 0.12) % on average. Compared to $\Upsilon(4S)$ MC studies, the VXDTF2 performs worse. As the tag-and-probe method requires $p_T > 200\,\text{MeV}/c$, the low transverse momentum region where both the SVDHoughTracking and the VXDTF2 achieve a much higher track finding efficiency compared to the full track finding on $\Upsilon(4S)$ MC (c.f. Figure 5.4) is not included in this study. Using $\Upsilon(4S)$ events the full track finding achieves the highest track finding efficiency above transverse momenta of 300 MeV/c, too. In addition, in the data used in this study the rate of beam induced backgrounds is much lower than in the $\Upsilon(4S)$ studies with nominal backgrounds presented in Section 5.3. Comparing the performance of the full tracking with nominal backgrounds and twice the nominal backgrounds it is obvious that its performance is better with lower background rates.

However, a seemingly reduced efficiency in a particular bin does not necessarily indicate an actually reduced efficiency in that bin. This is, because the ratio N4/(N3+N4) is computed for each bin of the

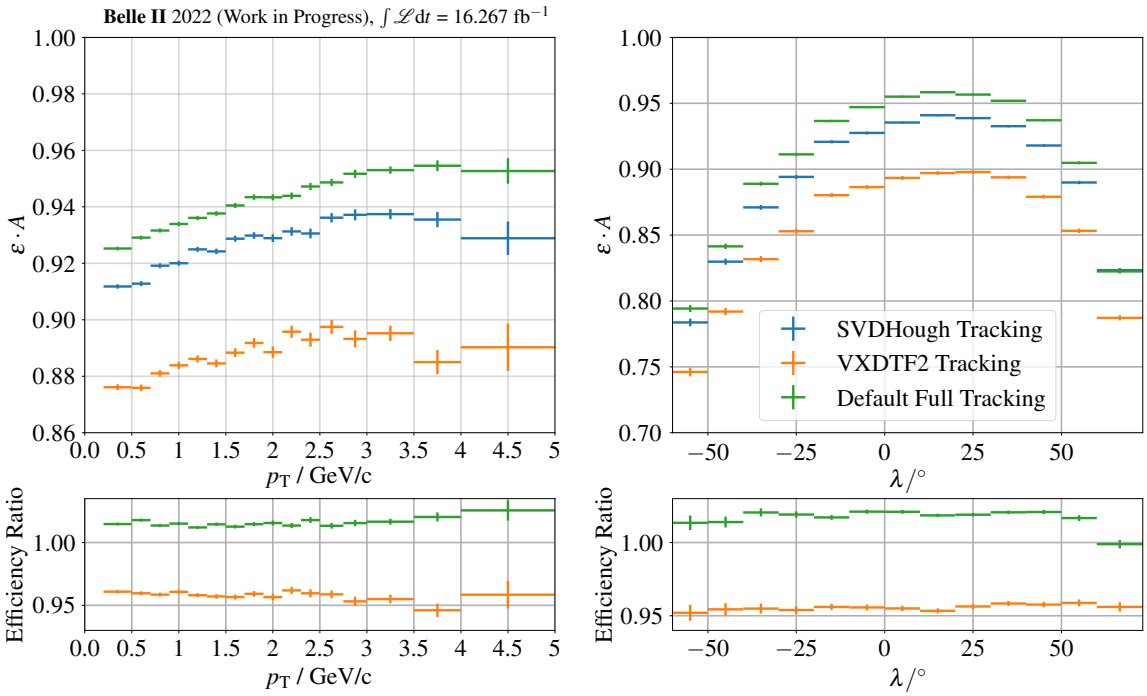


Figure 5.35: Track finding efficiency estimate with di-τ events for the new SVDHoughTracking (blue), the default SVD track finder VXDTF2 (orange), and the default full track finding (green). The data set consists of $\int \mathscr{L}\, dt = 16.267\,\mathrm{fb}^{-1}$ recorded in 2021. On average the SVDHoughTracking finds $(92.32 \pm 0.13)\,\%$ of all tracks, which is a significantly higher efficiency compared to the VXDTF2 at $(88.57 \pm 0.15)\,\%$. The efficiency is consistently above 90 % as function of p_T of the 1-prong particle. All three track finding algorithms work most efficiently in the central-forward region in λ, but with significant inefficiencies in the very forward and backward regions.

1-prong particle p_T and λ. A lower ratio indicates that N3 has a higher relative value in comparison to N4, but it does not mean that the missing probe track actually would be in this bin. Nonetheless, the total ratio is a valid estimate for the overall tracking efficiency.

5.8.4 Fake rate with τ-pair events

Similar to the track finding efficiency, the fake rate can be estimated with τ-pair events. In this case, instead of measuring

$$\varepsilon \cdot A = \frac{N_4}{N_4 + N_3}$$

as before, the fake rate is calculated with events containing four and five tracks, and defined as

$$r_{\text{fake}} = \frac{N_5}{N_4 + N_5}. \tag{5.4}$$

While the track selection criteria are the same as in Table 5.13, the track multiplicity selection is different and shown in Table 5.15. In contrast to the efficiency study, only the electron channel is considered for a higher sample purity.

Table 5.15: Track multiplicity criteria for the fake rate study (from [85]).

	$N_{\text{pion}}^{\text{probe}}$	$N_{\text{pion}}^{\text{tag}}$	$N_{\text{electron}}^{\text{tag}}$
electron channel, 4-track sample	3	3	1
electron channel, 5-track sample	4	≥ 3	1

Similar to the efficiency study, a track and event selection is conducted:

Hemispherical separation. The thrust axis of the event is used to define two opposite hemispheres. All 3-prong tracks are required to be in one, and the 1-prong track is required to be in the other hemisphere. This condition is similar to the angular separation selection of the track finding efficiency study.

Invariant mass selection. Three different requirements are made here:

- $|m_{\pi\pi}^{\text{OS}} - m_{\rho}| < 100\,\text{MeV}/c^2$ where $m_{\rho} = 775.26\,\text{MeV}/c^2$ [32] is the mass of the ρ meson, and $m_{\pi\pi}^{\text{OS}}$ is the invariant mass of the two probe pions with opposite charge. As the 3-prong particles are required to have a total charge which is opposite to the lepton (= electron or positron) charge, the OS-channel appears twice. Requiring the two tracks to have an invariant mass similar to that of an intermediate ρ meson originating from a τ lepton decay significantly reduces the background from continuum.
- $300\,\text{MeV}/c^2 < m_{\pi\pi}^{\text{SS}} < m_{\tau}$, where $m_{\tau} = 1\,776.86\,\text{MeV}/c^2$ [32] is the mass of the τ lepton, and $m_{\pi\pi}^{\text{SS}}$ is the invariant mass of the two tag pions with same charge sign.
- The invariant mass of the three tag pions $M_{\pi\pi\pi}$ must not be larger than $1.3\,\text{GeV}/c^2$.

Transverse momenta selection. The overall transverse momentum p_{T} of the 3-prong particles is required to be larger than $3\,\text{GeV}/c$, similarly $p_{\text{T}} > 1\,\text{GeV}/c$ is required for the 1-prong track.

Particle identification. Since only the electron channel is used for the fake rate estimate, the 1-prong track must fulfil an election identification probability of at least 0.9. In addition, the single particle of the 3-prong sample, which charge is opposite to the other two, is required to have a Kaon identification probability of less than 0.6.

Neutral multiplicity. Neither π^0 nor good gammas are allowed in the final selection.

Since no distinction between fake and clone tracks can be made easily without MC truth information, clone tracks are considered additional tracks, and thus count as fake tracks in this study.

The fake rate as function of p_{T} and λ is shown in Figure 5.36. Due to the tight selection the amount of events with 5 tracks is very low, limiting the statistical significance of the results. However, as with $\Upsilon(4S)$ MC with nominal backgrounds, the SVDHoughTracking achieves the overall lowest fake rate with only 0.3 %. Similar features to the $\Upsilon(4S)$ study are visible, for instance the decreasing fake rate with increasing p_{T}. Despite the nearly flat distribution of the fake rate in λ, no clear indication for an increase in fake rate in the very forward or backward region is visible due to the lack of statistical significance and the low number of entries in the first and last bins of the λdistribution.

A summary of the track finding efficiency and fake rate for the di-τ tag-and-probe study is given in Table 5.16.

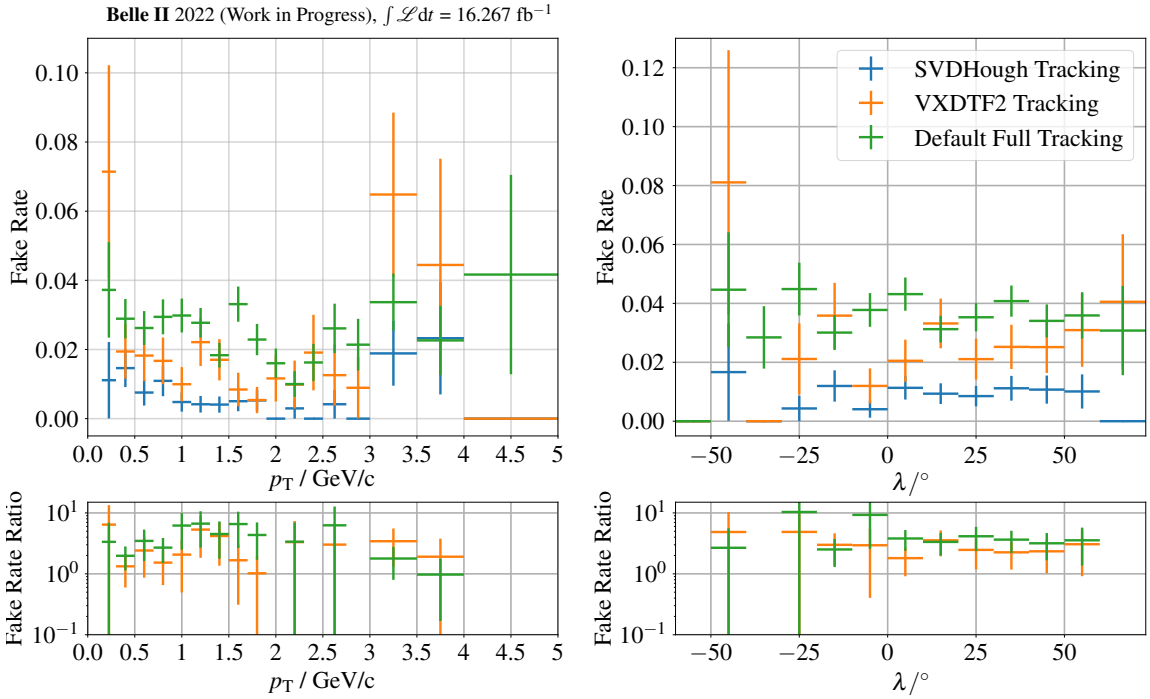


Figure 5.36: Fake rate estimate with di-τ events for the new SVDHoughTracking (blue), the default SVD track finder VXDTF2 (orange), and the default full track finding (green). The shape of the distributions and the relative abundance is similar to what is observed with $\Upsilon(4S)$ MC with nominal backgrounds with the SVDHoughTracking finding the smallest number of fake tracks. The data set consists of $\int \mathscr{L}\, dt = 16.267\,\text{fb}^{-1}$ recorded in 2021.

Table 5.16: Summary of the average track finding efficiencies and fake rates of the new SVDHoughTracking, the VXDTF2, and the default full track finding.

Values in percent	SVDHoughTracking	VXDTF2	Full track finding
Finding efficiency	92.32 ± 0.13	88.57 ± 0.15	92.92 ± 0.12
Fake Rate	0.68 ± 0.10	1.91 ± 0.21	2.79 ± 0.14

5.9 Summary for the SVDHoughTracking

The new SVDHoughTracking algorithm is a valuable addition to the set of Belle II track finding algorithms. It provides higher track finding efficiencies than the VXDTF2, the current default SVD standalone track finding, both standalone, as well as as part of the full track finding chain, while achieving lower fake rates at nominal luminosity and background rates. In addition, ROI finding is improved with the SVDHoughTracking, too, both on MC where only SVD information are used to find tracks for ROI creation, as well as in the full tracking. Furthermore, the SVDHoughTracking proves to be more robust against increased background rates, achieving higher track and ROI finding efficiencies, which ensures high finding efficiency also with increased beam backgrounds. The per event execution time of the SVDHoughTracking is similar to the VXDTF2 with (7.515 ± 6.324) ms compared to (7.022 ± 11.895) ms. Thus, it can be used as a replacement for the VXDTF2 in the full tracking chain. However, further in-detail studies are necessary, for instance with K_S^0 mesons, to confirm that the track finding efficiency is not worse in specific cases, despite the overall higher track finding efficiency.

CHAPTER 6

Conclusion and Outlook

The Belle II experiment has successfully been recording collision data since March 2018. Since then, first results demonstrated the capabilities of the detector and the reconstruction software, already providing world-leading results. To accomplish this, high track finding efficiency and precise vertex information is essential, among other challenges in particle reconstruction and background rejection. For precise vertexing, information from the PXD are crucial.

Since the integration time of the PXD sensors is too long to use PXD hits in the online reconstruction, and an overwhelming amount of background hits is collected during this time, data reduction on the PXD is required. To achieve the data reduction, Regions of Interest (ROI) on the PXD are calculated using information from the surrounding tracking detectors, and only PXD hits inside the ROIs are stored. The DATCON is one of two systems to provide these ROIs. Its foundations were laid seven years ago in [2], and it has been further developed and improved in this work. The replication of the FPGA-based DATCON described in [4] in the Belle II software framework basf2 has been developed in the scope of this thesis. In its current implementation about 90 % of PXD clusters originating from decay products of B mesons are contained in the ROIs, while rejecting about 75 % of mostly background hits, corresponding to a Data Retention Fraction (DRF) of 25 % at nominal luminosity. Although these performance figures do not fulfil the requirements, they show the potential of the system. Further improvements are necessary to achieve the target values of a DRF of 10 % with an increased ROI finding efficiency.

ROIs from random combinations of SVD hits are found to be the biggest challenge for an improved DRF. Further rejection of background SVD hits, as well as a reduction of random combinations, for instance by using track fits for the track candidates found in the Hough Space, is required. The software developed in this thesis can be extended to study the effects of possible improvements before implementing them on FPGA.

Based on the developments for DATCON, the SVDHoughTracking algorithm has been developed. It is an SVD standalone track finding algorithm which achieves a higher track finding efficiency as, as well as fake and clone rates comparable to, the currently default SVD standalone track finding algorithm VXDTF2. Since the SVD track finding algorithms are not used standalone on data, but are embedded into the full track reconstruction, the track finding performance of the full system needs to be evaluated. With the SVDHoughTracking as a replacement for the VXDTF2 in the full track reconstruction chain, the track finding efficiency is increased by 0.5 %, as shown in the left half of Figure 6.1, while achieving both a lower fake and clone rate. Its capabilities have been proven

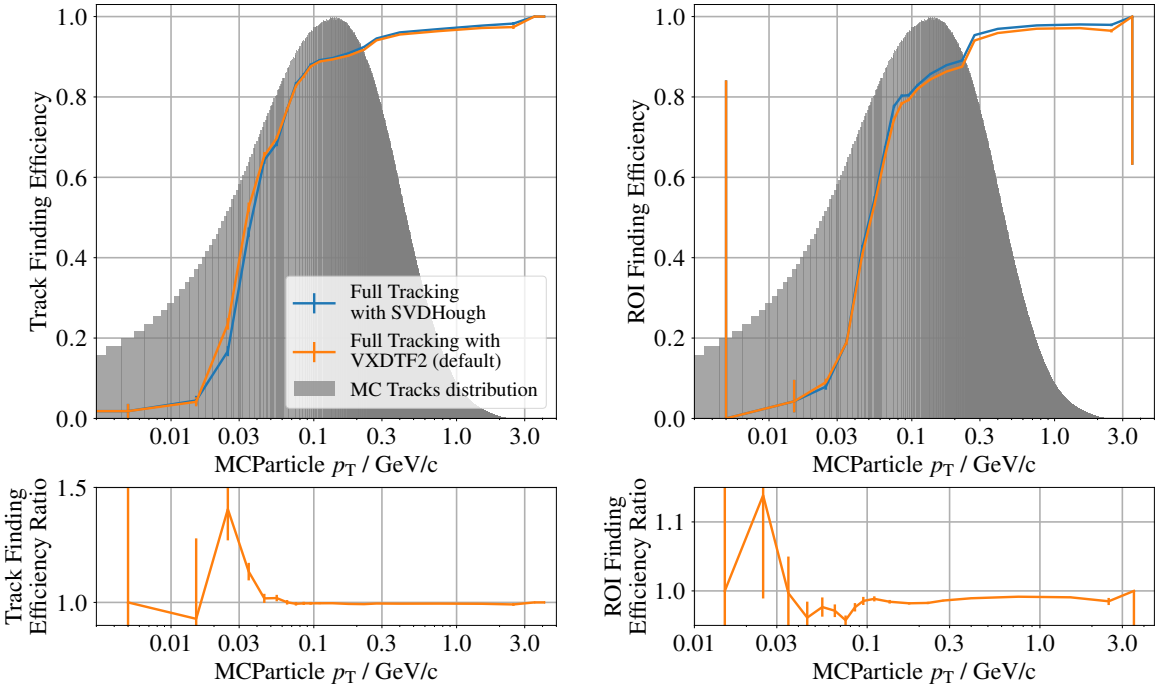


Figure 6.1: Summary of the track and ROI finding efficiency comparing the VXDHoughTracking and the VXDTF2 in the full tracking.

with data from di-τ decays, with a track finding efficiency that is higher by 3.75 % compared to the VXDTF2 when used standalone.

The improved track finding efficiency also results in an increased ROI finding efficiency compared to the VXDTF2, both standalone or as part of the full tracking chain, as shown in the right half Figure 6.1. This makes it a reasonable replacement option for the VXDTF2. However, the new SVDHoughTracking suffers from a reduced finding efficiency of slow pions from $D^{*\pm}$ decays, as well as for the decay products of long-lived neutral particles like K_S^0 or Λ^0 due to the assumption of tracks being perfect circles passing the interaction point. These limitations prevent the SVDHoughTracking from being a direct replacement for the VXDTF2. Nonetheless, a combination of both SVD standalone track finding algorithms is beneficial to the Belle II track reconstruction as well as PXD data reduction. The fast execution time of the new tracking algorithm does not impose an additional burden for the fast reconstruction on the HLT.

Two important parts of the Belle II reconstruction have been developed or improved. While the PXD data reduction currently implemented on FPGAs and employed at KEK does not fulfil all requirements, it shows a good potential for improvements. The newly developed SVD tracking algorithm can play a crucial role for the track reconstruction in the future, allowing Belle II to achieve its goals in providing world-leading results in the field of particle physics.

APPENDIX A

Supplemental Figures

A.1 DATCON

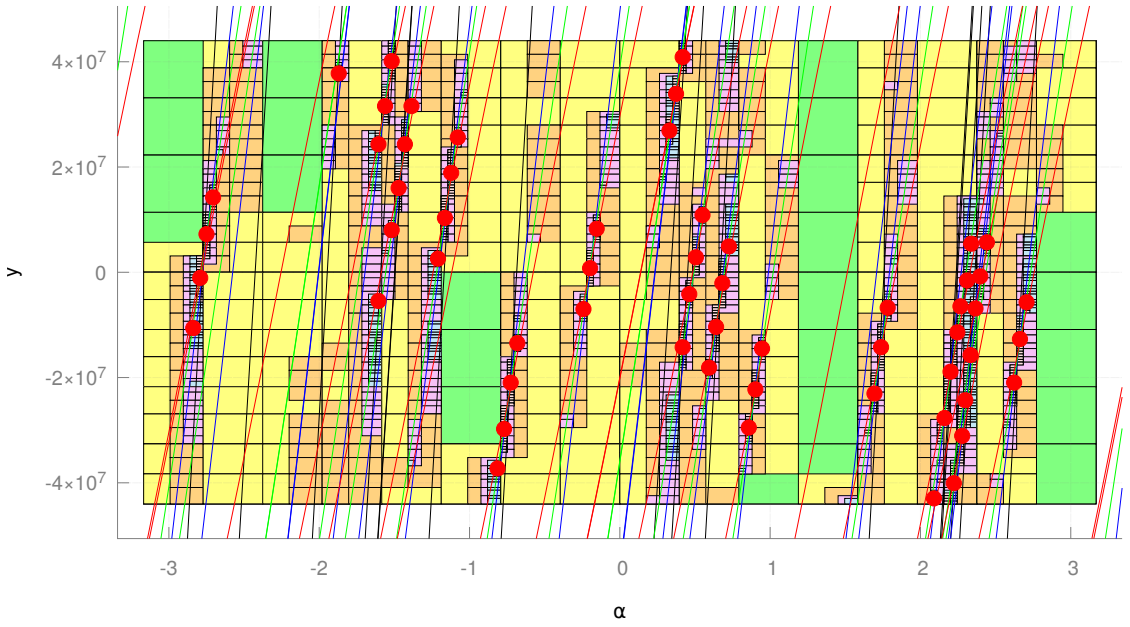


Figure A.1: Hough Space for the extraction of φ for a $\Upsilon(4S)$ event without beam backgrounds.

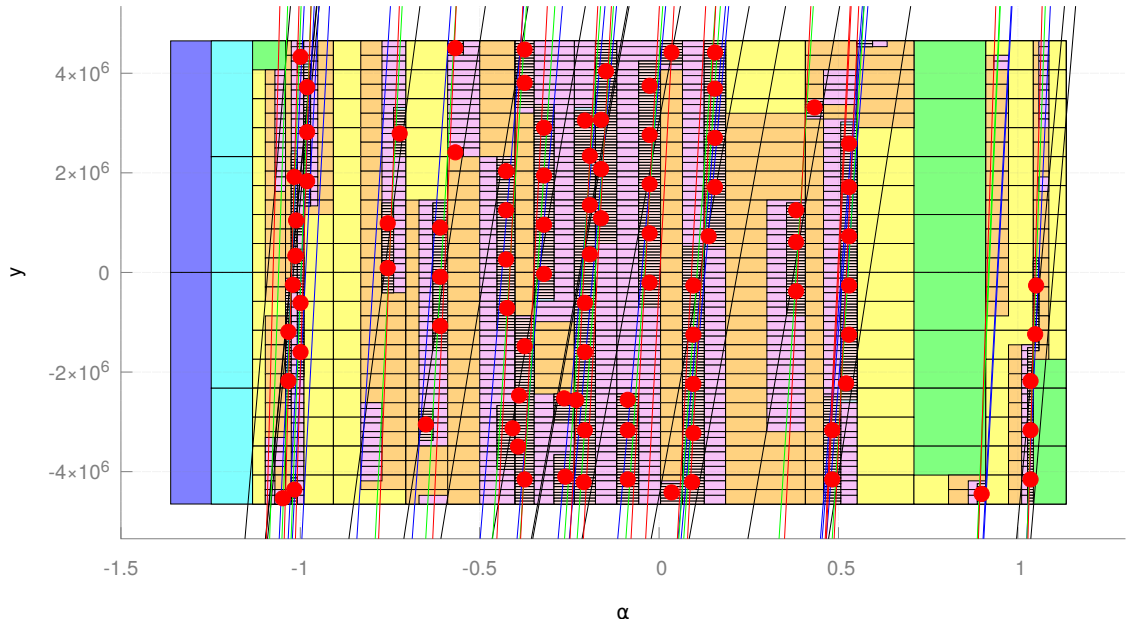


Figure A.2: Hough Space for the extraction of λ for a $\Upsilon(4S)$ event without beam backgrounds.

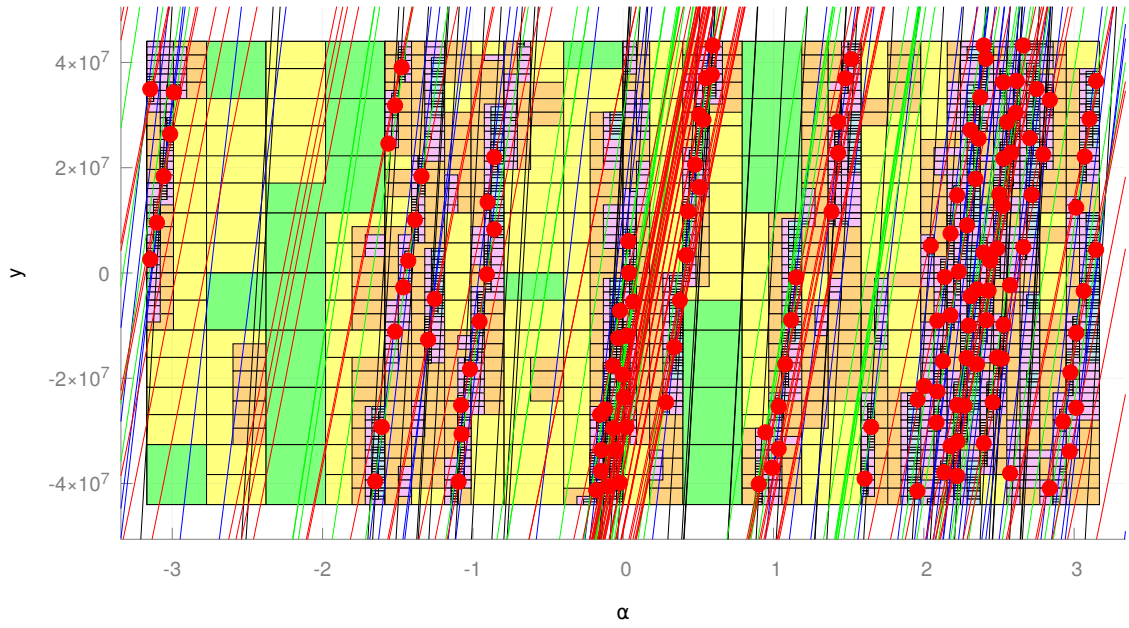


Figure A.3: Hough Space for the extraction of φ for a $\Upsilon(4S)$ event with beam backgrounds for nominal luminosity.

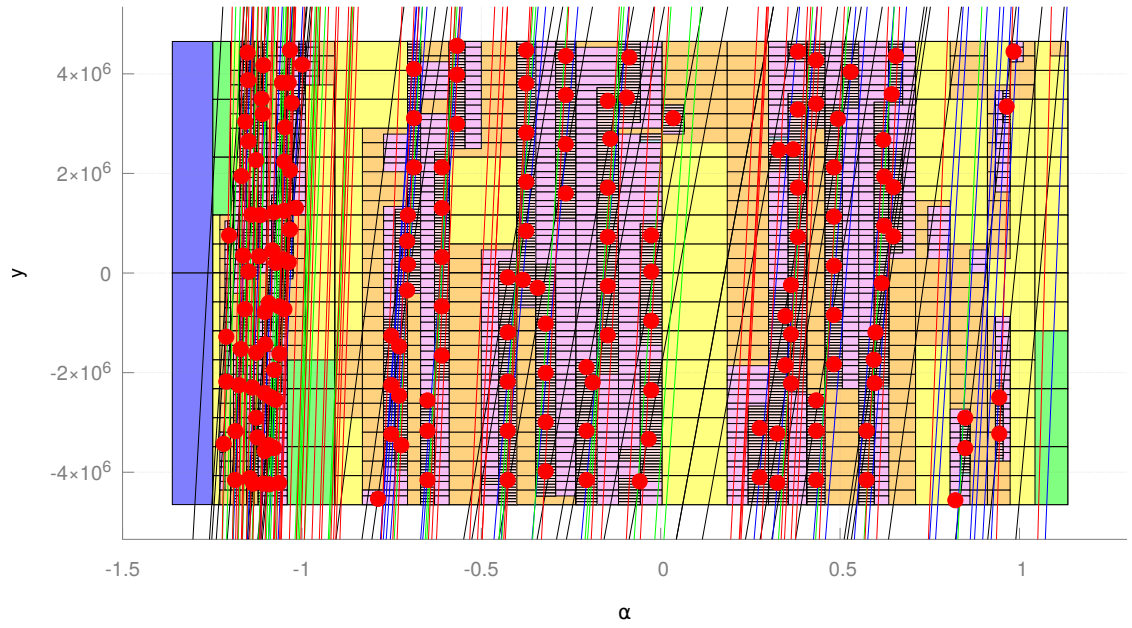


Figure A.4: Hough Space for the extraction of λ for a $\Upsilon(4S)$ event with beam backgrounds for nominal luminosity.

A.2 SVDHoughTracking

A.2.1 Standalone tracking performance comparison

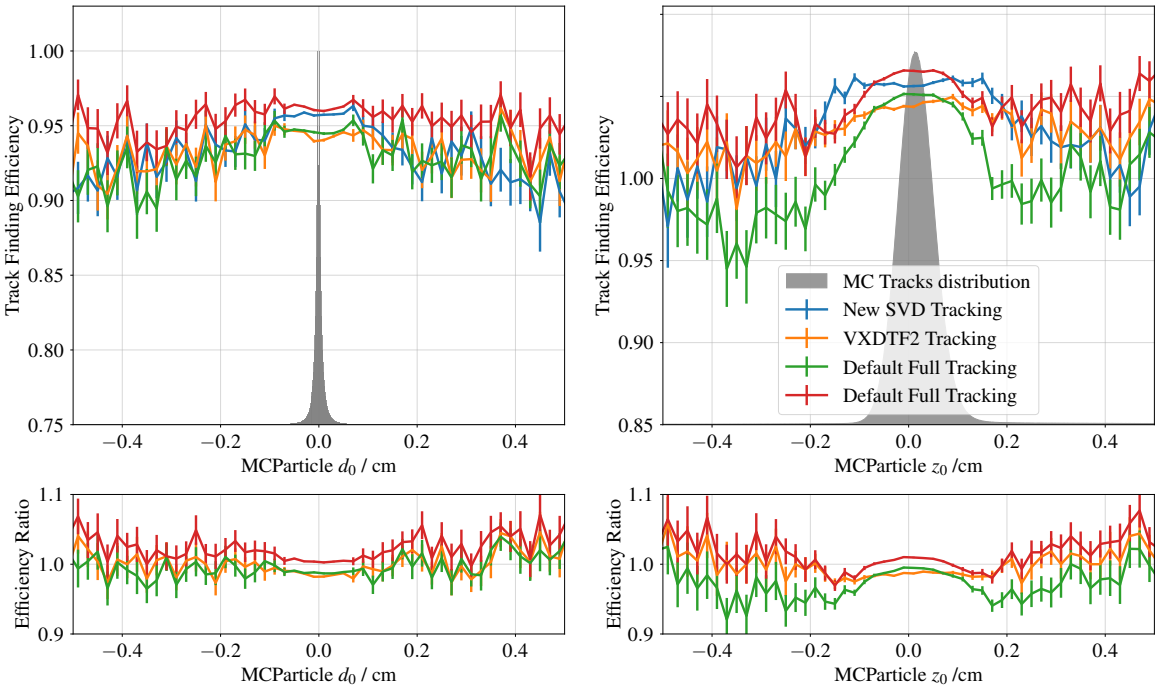


Figure A.5: Track finding efficiency as function of the impact parameters d_0 and $z_0$for the standalone SVDHoughTracking in comparison to the VXDTF2 and the full tracking chain.

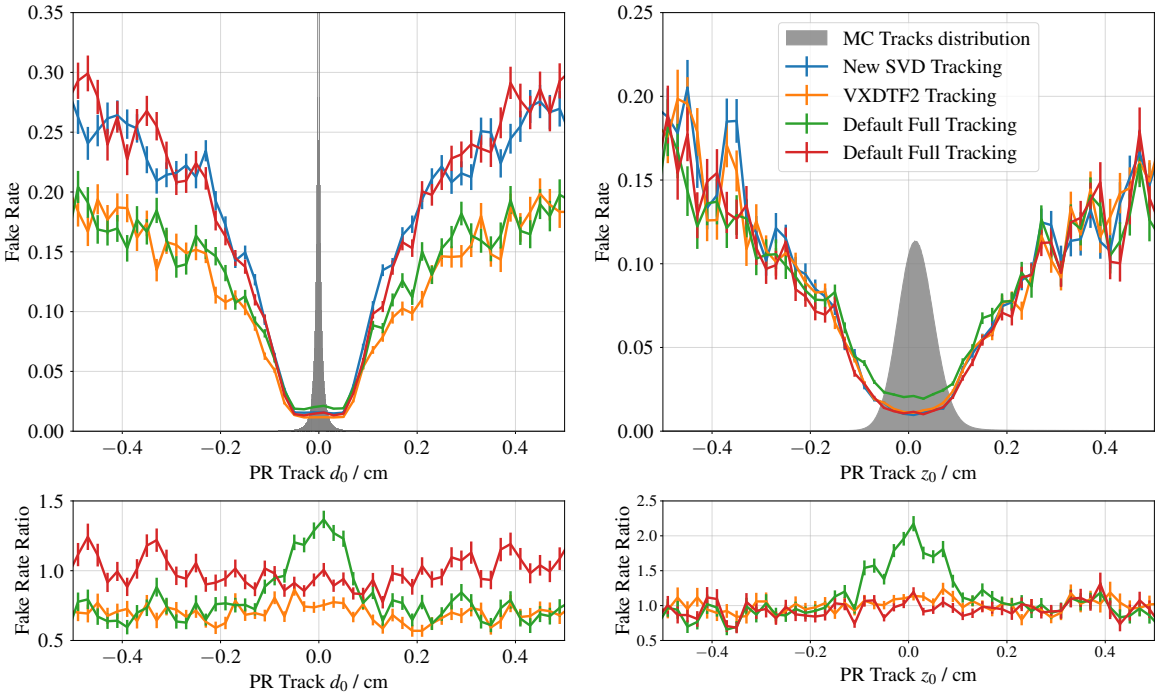


Figure A.6: Fake rate as function of the impact parameters d_0 and $z_0$for the standalone SVDHoughTracking in comparison to the VXDTF2 and the full tracking chain.

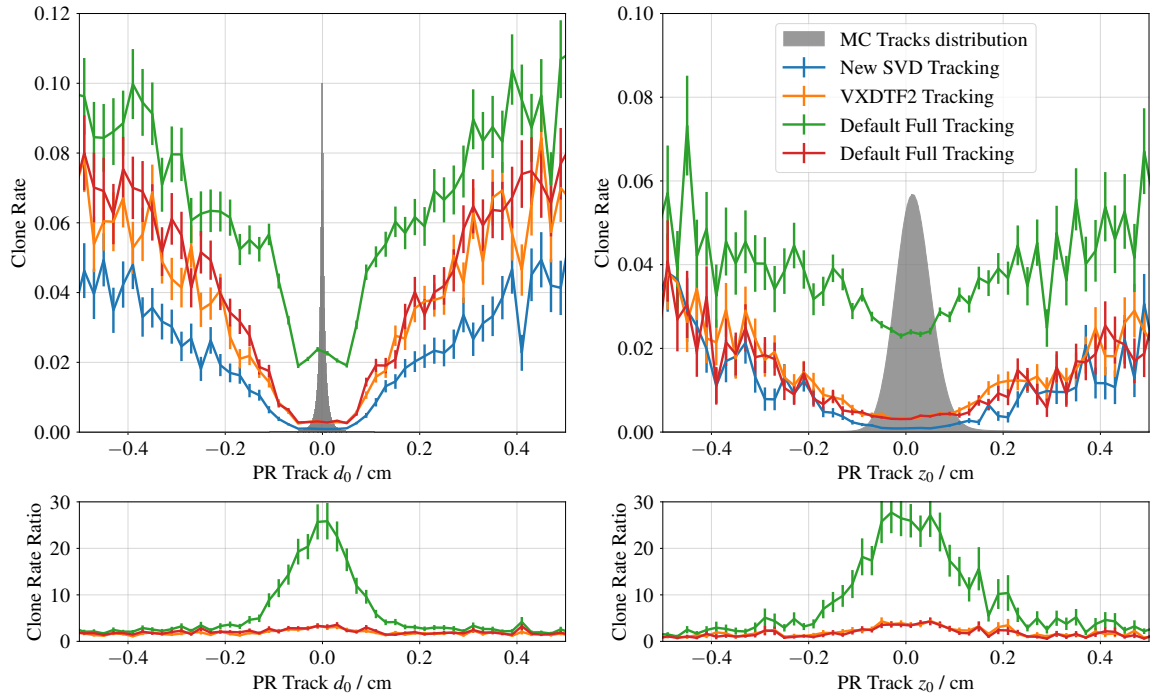


Figure A.7: Clone rate as function of the impact parameters d_0 and $z_0$for the standalone SVDHoughTracking in comparison to the VXDTF2 and the full tracking chain.

A.2.2 Full tracking performance comparison

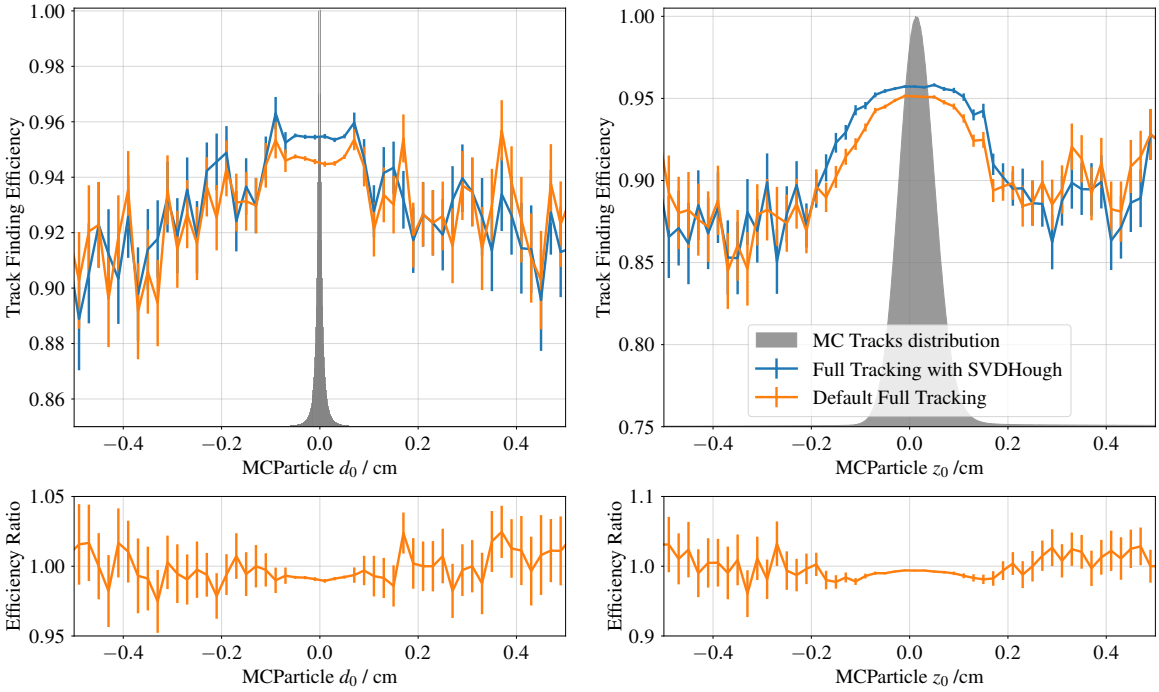


Figure A.8: Track finding efficiency as function of the impact parameters d_0 and z_0 when using the SVDHoughTracking instead of the VXDTF2 in the full tracking.

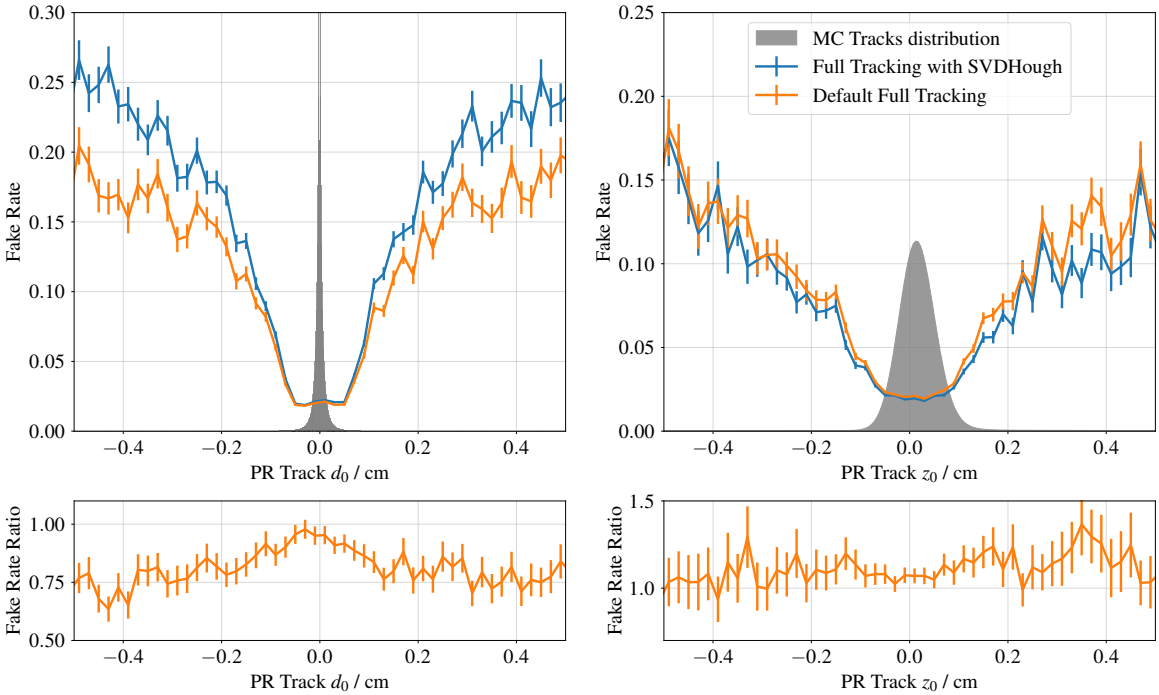


Figure A.9: Fake rate as function of the impact parameters d_0 and z_0 when using the SVDHoughTracking instead of the VXDTF2 in the full tracking.

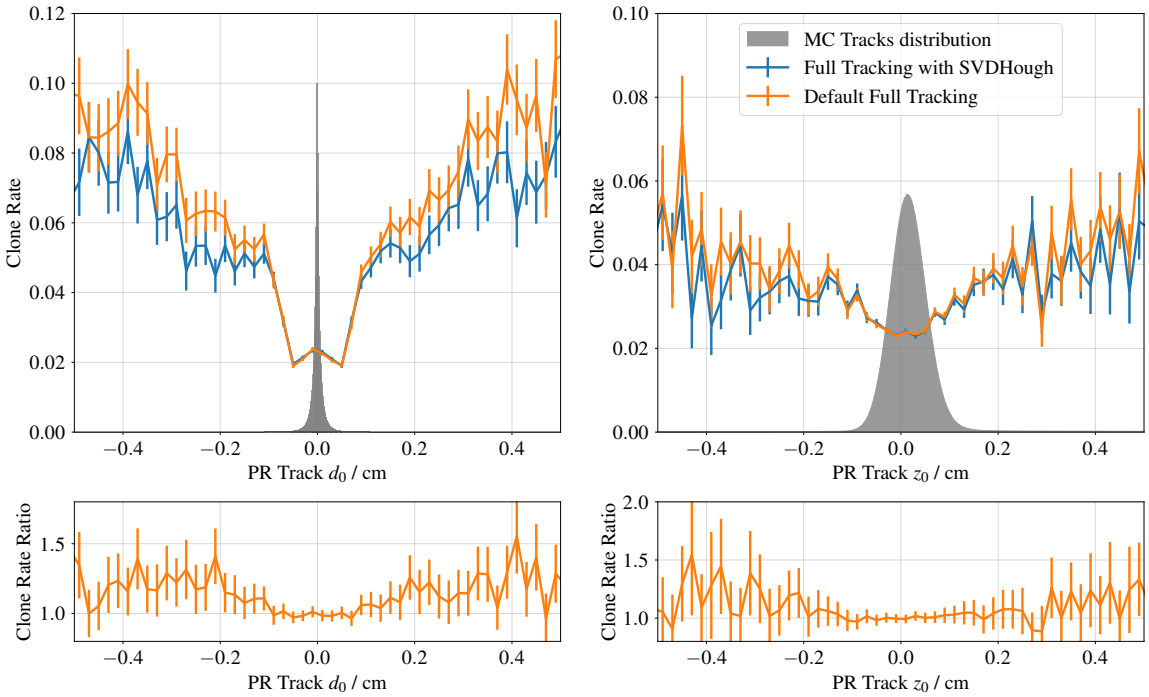


Figure A.10: Clone rate as function of the impact parameters d_0 and z_0 when using the SVDHoughTracking instead of the VXDTF2 in the full tracking.

A.2.3 ROI finding performance comparison

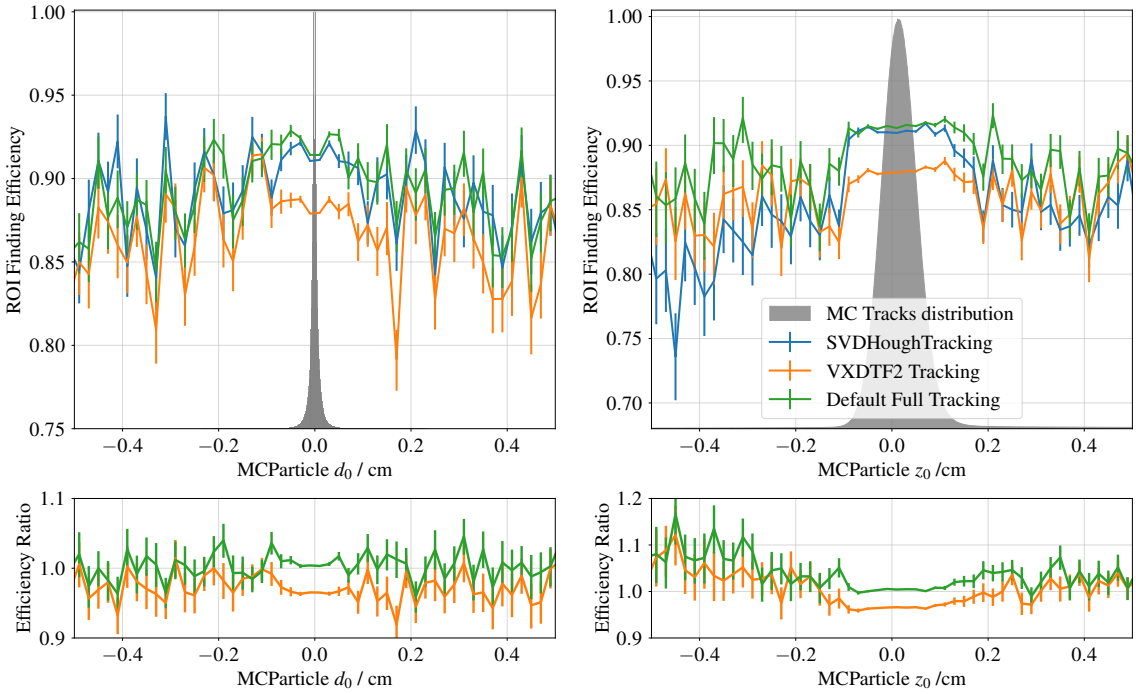


Figure A.11: ROI finding efficiency as function of d_0 and z_0 using the SVDHoughTracking standalone in comparison to the VXDTF2 and the full tracking chain.

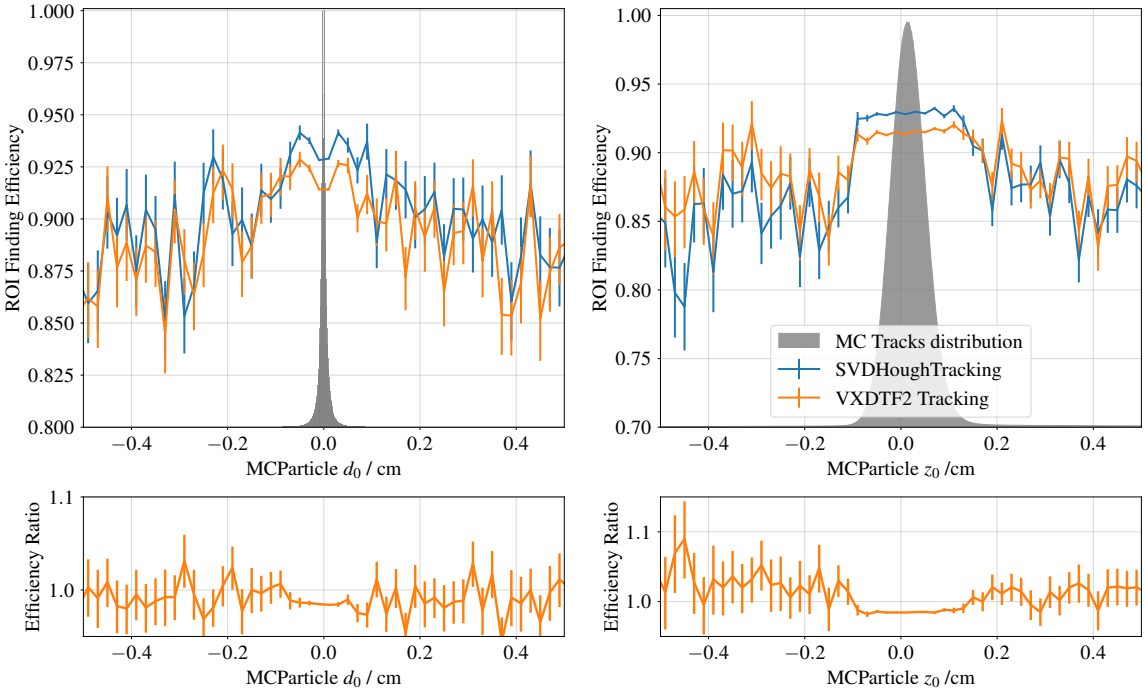


Figure A.12: ROI finding efficiency as function of d_0 and z_0 using the SVDHoughTracking as a replacement for the VXDTF2 in the full tracking chain.

APPENDIX B

Acronyms

ADC Analog-to-Digital Converter

ADU Analog-Digital-Unit

ALICE A Large Ion Collider Experiment

APV Analogue Pipeline Voltage

ARICH Aerogel Ring Imaging Cherenkov

ASIC Application Specific Circuit

ATLAS A Toroidal LHC ApparatuS

basf2 Belle Analysis Software Framework 2

BBF Bethe-Bloch Formula

BEAST Beam Exorcism for A STable experiment

BSM Beyond Standard Model

CDC Central Drift Chamber

CERN European Organization for Nuclear Research

CKF Combinatorial Kalman Filter

CKM Cabibbo-Kobayashi-Maskawa

CMOS Complementary Metal Oxide Semiconductor

CMS Center of Mass System

CoG Center of Gravity

CP Charge-Parity

CPU Central Processing Unit

DAF Deterministic Annealing Filter

DAQ Data Acquisition

DATCON Data Acquisition Tracking and Concentrator Online Node

DCD Drain Current Digitizer

DE Dark Energy

DESY Deutsches Elektronen SYnchroton

DEPFET DEpleted P-channel Field Effect Transistor

DHP Data Handling Processor

DM Dark Matter

DRF Data Retention Fraction

DSSD Double-Sided Silicon Strip Detector

EB1 Event Builder 1

EB2 Event Builder 2

ECL Electromagnetic Calorimeter

FADC Fast Analog-to-Digitcal Converter

FAIR Facility for Antiproton and Ion Research

FEE Front End Electronics

FHT Fast Hough Transformation algorithm

FOM Figure Of Merit

FPGA Field Programmable Gate Arrays

GEM Gas Elelectron Multiplicator

HEP High Energy Particle Physics

HER High Energy Ring

HLT High Level Trigger

HS Hough Space

HT Hough Transformation

ILC International Linear Collider

ILD International Linear Detector

IP Interaction Point

IR Interaction Region

KEK kō-enerugī kasokuki kenkyū kikō (High Energy Accelerator Research Organization)

KF Kalman Filter

KLM K-Long-Muon detector

LER Low Energy Ring

LFV Lepton Flavour Violation

LHC Large Hadron Collider

LHCb Large Hadron Collider beauty

LINAC Linear Accelerator

LUT Look-Up Table

MC Monte Carlo

MICROMEGA Micromesh Gaseaous Structures

MIP Minimum Ionizing Particle

MOSFET Metal Oxide Field Effect Transistor

MS Multiple Scattering

MVA Multi Variate Analysis

MWPC Multi Wire Proportional Chamber

NDF Number of Degrees of Freedom

ONSEN ONline SElector Node

PANDA AntiProton ANnihilations at DArmstadt

PC Personal Computer

POCA Point of Closest Approach

PID Particle Identification

PR Pattern Recognition

PS Parameter Sweep

PXD Pixel Detector

QCD Quantum Chromo Dynamics

QED Quantum Electro Dynamics

RAM Random Access Memeory

RBB Radiative Bhabha

RF Radio Frequency

RTC Raw Track Candidate

RO Read Out

ROI Region of Interest

SL Super Layer

SLAC Stanford Linear Accelerator Center

SM Standard Model

SNR Signal-to-Noise-Ratio

SOI Silicon on Insulator

SUSY SUper SYmmetry

SVD Silicon strip Vertex Detector

TC Track Candidate

TDC Time Digital Converter

TOP Time Of Propagation

TPC Time Projection Chamber

VXD Vertex Detector

VXDTF2 Vertex Detector Track Finder 2

VTX Upgraded Vertex Detector

XML Extended Markup Language

Bibliography

[1] T. Abe et al., "Belle II Technical Design Report", (2010), arXiv: 1011.0352, URL: http://arxiv.org/abs/1011.0352 (cit. on p. 1).

[2] M. Schnell, "Development of an FPGA-based Data Reduction System for the Belle II DEPFET Pixel Detector", Universität Bonn, 2015 (cit. on pp. 1, 15, 47, 58, 76, 80, 133).

[3] C. Wessel, "Optimisation of the Online Data Reduction Algorithms of the Belle II Experiment using DATCON", Universität Bonn, 2016 (cit. on pp. 1, 15, 47, 58, 59, 76, 79, 80).

[4] B. Deschamps, "Development and Deployment of DATCON, an FPGA-based Data Reduction System for the Belle II Pixel Detector", Universität Bonn, 2022 (cit. on pp. 1, 58, 133).

[5] M. Planck, "Ueber das Gesetz der Energieverteilung im Normalenspektrum", (1900) (cit. on p. 3).

[6] E. Schrödinger, "Quantisierung als Eigenwertproblem", (1926) (cit. on p. 3).

[7] E. Schrödinger, "An Undulatory Theory of the Mechanics of Atoms and Molecules", Phys. Rev. **28** (6 1926) 1049, URL: https://link.aps.org/doi/10.1103/PhysRev.28.1049 (cit. on p. 3).

[8] P. A. M. Dirac, "The Quantum Theory of the Emission and Absorption of Radiation", Dordrecht: Springer Netherlands, 1988 157, ISBN: 978-94-009-3051-3, URL: https://doi.org/10.1007/978-94-009-3051-3_9 (cit. on p. 3).

[9] E. Fermi, "Quantum Theory of Radiation", Rev. Mod. Phys. **4** (1 1932) 87, URL: https://link.aps.org/doi/10.1103/RevModPhys.4.87 (cit. on p. 3).

[10] F. Bloch and A. Nordsieck, "Note on the Radiation Field of the Electron", Phys. Rev. **52** (2 1937) 54, URL: https://link.aps.org/doi/10.1103/PhysRev.52.54 (cit. on p. 3).

[11] J. R. Oppenheimer, "Note on the Theory of the Interaction of Field and Matter", Phys. Rev. **35** (5 1930) 461, URL: https://link.aps.org/doi/10.1103/PhysRev.35.461 (cit. on p. 3).

[12] S. Weinberg, "A Model of Leptons", Phys. Rev. Lett. **19** (21 1967) 1264, URL: https://link.aps.org/doi/10.1103/PhysRevLett.19.1264 (cit. on p. 3).

[13] S. L. Glashow, "The renormalizability of vector meson interactions", Nuclear Physics **10** (1959) 107, ISSN: 0029-5582, URL: https://www.sciencedirect.com/science/article/pii/0029558259901968 (cit. on p. 3).

[14] A. Salam and J. C. Ward, "Weak and electromagnetic interactions", Il Nuovo Cimento (1955-1965) **11** (1959) 568 (cit. on p. 3).

[15] H. Fritzsch, M. Gell-Mann and H. Leutwyler, "Advantages of the color octet gluon picture", Physics Letters B **47** (1973) 365, ISSN: 0370-2693, URL: https://www.sciencedirect.com/science/article/pii/0370269373906254 (cit. on p. 3).

[16] D. J. Gross and F. Wilczek, "Ultraviolet Behavior of Non-Abelian Gauge Theories", Phys. Rev. Lett. **30** (26 1973) 1343, URL: https://link.aps.org/doi/10.1103/PhysRevLett.30.1343 (cit. on p. 3).

[17] H. D. Politzer, "Reliable Perturbative Results for Strong Interactions?", Phys. Rev. Lett. **30** (26 1973) 1346, URL: https://link.aps.org/doi/10.1103/PhysRevLett.30.1346 (cit. on p. 3).

[18] P. W. Higgs, "Broken Symmetries and the Masses of Gauge Bosons", Phys. Rev. Lett. **13** (16 1964) 508, URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.508 (cit. on p. 3).

[19] F. Englert and R. Brout, "Broken Symmetry and the Mass of Gauge Vector Mesons", Phys. Rev. Lett. **13** (9 1964) 321, URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.321 (cit. on p. 3).

[20] B. Pontecorvo, "Mesonium and Antimesonium", Soviet Journal of Experimental and Theoretical Physics **6** (1958) 429 (cit. on p. 3).

[21] B. Pontecorvo, "Neutrino Experiments and the Problem of Conservation of Leptonic Charge", Soviet Journal of Experimental and Theoretical Physics **26** (1968) 984 (cit. on p. 3).

[22] Y. Fukuda et al., "Evidence for Oscillation of Atmospheric Neutrinos", Phys. Rev. Lett. **81** (8 1998) 1562, URL: https://link.aps.org/doi/10.1103/PhysRevLett.81.1562 (cit. on p. 3).

[23] Q. R. Ahmad et al., "Measurement of the Rate of $\nu_e + d \rightarrow p + p + e^-$ Interactions Produced by ${}^8B$ Solar Neutrinos at the Sudbury Neutrino Observatory", Phys. Rev. Lett. **87** (7 2001) 071301, URL: https://link.aps.org/doi/10.1103/PhysRevLett.87.071301 (cit. on p. 3).

[24] M. H. Ahn et al., "Measurement of neutrino oscillation by the K2K experiment", Phys. Rev. D **74** (7 2006) 072003, URL: https://link.aps.org/doi/10.1103/PhysRevD.74.072003 (cit. on p. 3).

[25] Y. Abe et al., "Indication of Reactor $\overline{\nu}_e$ Disappearance in the Double Chooz Experiment", Phys. Rev. Lett. **108** (13 2012) 131801, URL: https://link.aps.org/doi/10.1103/PhysRevLett.108.131801 (cit. on p. 3).

[26] C. S. Wu, E. Ambler, R. W. Hayward, D. D. Hoppes and R. P. Hudson, "Experimental Test of Parity Conservation in Beta Decay", Phys. Rev. **105** (4 1957) 1413, URL: https://link.aps.org/doi/10.1103/PhysRev.105.1413 (cit. on p. 3).

[27] J. H. Christenson, J. W. Cronin, V. L. Fitch and R. Turlay, "Evidence for the 2π Decay of the K_2^0 Meson", Phys. Rev. Lett. **13** (4 1964) 138, URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.138 (cit. on p. 3).

[28] K. Abe et al., "Observation of Large *CP* Violation in the Neutral *B* Meson System", Phys. Rev. Lett. **87** (9 2001) 091802, URL: https://link.aps.org/doi/10.1103/PhysRevLett.87.091802 (cit. on p. 3).

[29] B. Aubert et al., "Measurement of *CP*-Violating Asymmetries in B^0 Decays to *CP* Eigenstates", Phys. Rev. Lett. **86** (12 2001) 2515, URL: https://link.aps.org/doi/10.1103/PhysRevLett.86.2515 (cit. on p. 3).

[30] V. C. Rubin, J. Ford W. K. and N. Thonnard, "Rotational properties of 21 SC galaxies with a large range of luminosities and radii, from NGC 4605 (R=4kpc) to UGC 2885 (R=122kpc).", The Astrophysical Journal **238** (1980) 471 (cit. on p. 3).

[31] K. G. Begeman, A. H. Broeils and R. H. Sanders, "Extended rotation curves of spiral galaxies: dark haloes and modified dynamics", Monthly Notices of the Royal Astronomical Society **249** (1991) 523, ISSN: 0035-8711, eprint: https://academic.oup.com/mnras/article-pdf/249/3/523/18160929/mnras249-0523.pdf, URL: https://doi.org/10.1093/mnras/249.3.523 (cit. on p. 3).

[32] P. Zyla et al., "Review of Particle Physics", PTEP **2020** (2020) 083C01 (cit. on pp. 3–6, 8, 20, 129).

[33] P. Franzini and J. Lee-Franzini, "Upsilon resonances", Annu. Rev. Nucl. Sci.; (United States) **33:1** (1983), URL: https://www.osti.gov/biblio/6584253 (cit. on p. 5).

[34] URL: https://hep.syr.edu/quark-flavor-physics/previous-experiments-and-projects/physics-of-cleo/,%20last%20checked%20on%20April%2023rd%202022,%2017:57%20CEST (cit. on p. 5).

[35] E. Kou et al., "The Belle II Physics Book", Progress of Theoretical and Experimental Physics **2019** (2019), URL: https://doi.org/10.1093%2Fptep%2Fptz106 (cit. on p. 5).

[36] N. Braun, "Combinatorial Kalman Filter and High Level Trigger Reconstruction for the Belle II Experiment", Karlsruhe Institute of Technology, 2018 (cit. on pp. 6, 12, 32, 35, 95).

[37] SuperB Collaboration et al., "SuperB Technical Design Report", (2013), arXiv: 1306.5655, URL: http://arxiv.org/abs/1306.5655 (cit. on p. 7).

[38] URL: https://www.belle2.org/e21595/e21770/infoboxContent25431/belle2.pdf,%20last%20checked%20on%20April%2023rd%202022,%2018:10%20CEST (cit. on p. 8).

[39] K. Adamczyk et al., "The Design, Construction, Operation and Performance of the Belle II Silicon Vertex Detector", (2022), arXiv: 2201.09824 [physics.ins-det] (cit. on pp. 11–13, 49).

[40] T. Kuhr, C. Pulvermacher, M. Ritter, T. Hauth and N. Braun, "The Belle II Core Software", (2018), arXiv: 1809.04299, URL: http://arxiv.org/abs/1809.04299 (cit. on pp. 12, 22).

[41] F. Müller, "Characterization and optimization of the prototype DEPFET modules for the Belle II Pixel Vertex Detector", 2017,
URL: http://nbn-resolving.de/urn:nbn:de:bvb:19-210714 (cit. on p. 11).

[42] J. Kemmer and G. Lutz, "New Detector Concepts",
Nuclear Instruments and Methods in Physics Research **253** (1987) 365,
URL: https://doi.org/10.1016/0168-9002%2887%2990518-3 (cit. on p. 14).

[43] C. Wessel, "Studies of the Track Reconstruction Algorithm in the Vertex Detector of the Belle II Experiment", Universität Bonn, 2014 (cit. on p. 15).

[44] H. J. Strutt, "LVIII. On the scattering of light by small particles", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **41** (1871) 447,
eprint: https://doi.org/10.1080/14786447108640507,
URL: https://doi.org/10.1080/14786447108640507 (cit. on p. 19).

[45] A. H. Compton, "A Quantum Theory of the Scattering of X-rays by Light Elements",
Phys. Rev. **21** (5 1923) 483,
URL: https://link.aps.org/doi/10.1103/PhysRev.21.483 (cit. on p. 19).

[46] A. Moll, "The Software Framework of the Belle II Experiment",
Journal of Physics: Conference Series **331** (2011) 032024, ISSN: 1742-6596, URL:
http://iopscience.iop.org/article/10.1088/1742-6596/331/3/032024/pdf
(cit. on p. 22).

[47] Standard C++ Foundation, The Standard : Standard C++,
URL: https://isocpp.org/std/the-standard (visited on 23/04/2018) (cit. on p. 22).

[48] Python Software Foundation, Welcome to Python.org,
URL: https://www.python.org/ (visited on 23/04/2018) (cit. on p. 22).

[49] B. Dawes, D. Abrahams and R. Rivera, Boost C++ Libraries, 2007,
URL: https://www.boost.org/ (visited on 23/04/2018) (cit. on p. 22).

[50] R. Brun and F. Rademakers, "ROOT - An object oriented data analysis framework",
Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **389** (1997) 81, ISSN: 01689002,
URL: http://linkinghub.elsevier.com/retrieve/pii/S016890029700048X
(cit. on p. 23).

[51] D. J. Lange, "The EvtGen particle decay simulation package",
Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **462** (2001) 152, ISSN: 01689002,
URL: https://ac.els-cdn.com/S0168900201000894/1-s2.0-S0168900201000894-main.pdf?_tid=10489e10-9189-4f5f-acf7-87cedf64b28c&acdnat=1524480961_905eab864acd43a4f358afa21cf979c2
(cit. on p. 23).

[52] T. Sjöstrand, S. Mrenna and P. Skands, "A Brief Introduction to PYTHIA 8.1", (2007),
arXiv: 0710.3820, URL: https://arxiv.org/pdf/0710.3820.pdf (cit. on p. 23).

[53] P. Urquijo and T. Ferber, "Overview of the Belle II Physics Generators", (2016), URL: https://docs.belle2.org/record/282/files/BELLE2-NOTE-PH-2015-006-v2.pdf (cit. on p. 23).

[54] S. Agostinelli et al., "Geant4—a simulation toolkit", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506** (2003) 250, ISSN: 01689002, URL: https://ac.els-cdn.com/S0168900203013688/1-s2.0-S0168900203013688-main.pdf?_tid=2ab2887a-7a2b-4813-9966-25cc44845930&acdnat=1524482129_7273334d05ad6f2c66d093ae3c67b132 (cit. on p. 23).

[55] Project Jupyter, Jupyter Homepage, 2018, URL: http://jupyter.org/ (visited on 16/11/2018) (cit. on p. 23).

[56] N. Braun, T. Hauth, C. Pulvermacher and M. Ritter, "An Interactive and Comprehensive Working Environment for High-Energy Physics Software with Python and Jupyt r Notebooks", Journal of Physics: Conference Series **898** (2017) 072020, ISSN: 1742-6588, URL: http://stacks.iop.org/1742-6596/898/i=7/a=072020?key=crossref.eff9ad3575e53bda6c70f5408e03c2cf (cit. on p. 23).

[57] P. V. C. Hough, "Method and means for recognizing complex patterns", (1962), Patent, URL: http://www.google.com/patents/US3069654 (cit. on p. 27).

[58] D. A. Glaser, "Some Effects of Ionizing Radiation on the Formation of Bubbles in Liquids", Phys. Rev. **87** (4 1952) 665, URL: https://link.aps.org/doi/10.1103/PhysRev.87.665 (cit. on p. 27).

[59] O. Hesse, Vorlesungen aus der analytischen Geometrie der geraden Linie, des Punktes und des Kreises in der Ebene, 1865, URL: https://archive.org/details/bub_gb_at6qA3g2YDwC/page/n24/mode/1up?q=Normalform (cit. on p. 28).

[60] R. Frühwirth, R. Glattauer, J. Lettenbichler, W. Mitaroff and M. Nadler, "Track finding in silicon trackers with a small number of layers", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **732** (2013) 95, Vienna Conference on Instrumentation 2013, ISSN: 0168-9002, URL: https://www.sciencedirect.com/science/article/pii/S0168900213008607 (cit. on pp. 32, 33).

[61] J. Lettenbichler, "Real-time Pattern Recognition in the Central Tracking Detector of the Belle II Experiment", PhD thesis: Technische Universität Wien, 2016, URL: http://www.ub.tuwien.ac.at (cit. on p. 32).

[62] T. Alexopoulos, M. Bachtis, E. Gazis and G. Tsipolitis, "Implementation of the Legendre Transform for track segment reconstruction in drift tube chambers", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **592** (2008) 456, ISSN: 01689002, URL: https://www.sciencedirect.com/science/article/pii/S0168900208005780 (cit. on p. 34).

[63] V. Trusov, "Development of Pattern Recognition Algorithms for the Central Drift Chamber of the Belle II Detector", PhD thesis: KIT, 2016, URL: https://ekp-invenio.physik.uni-karlsruhe.de/record/48882/files/EKP-2017-00009.pdf (cit. on p. 34).

[64] O. Frost, "A Local Tracking Algorithm for the Central Drift Chamber of Belle II", MA thesis: KIT, 2013, URL: http://ekp-invenio.physik.uni-karlsruhe.de/record/48172/files/iekp-ka2013-6.pdf (cit. on p. 34).

[65] N. Braun, "Momentum Estimation of Slow Pions and Improvements on the Track Finding in the Central Drift Chamber for the Belle II Experiment", MA thesis: KIT, 2015, URL: https://ekp-invenio.physik.uni-karlsruhe.de/record/48740/files/EKP-2015-00052.pdf (cit. on p. 34).

[66] M. Eliachevitch, "Tracking Performance Studies on Cosmic Rays and a Track Quality Estimation for the Central Drift Chamber of the Belle II-Experiment", MA thesis: KIT, 2018, URL: https://ekp-invenio.physik.uni-karlsruhe.de/record/49043 (cit. on p. 34).

[67] T. Toffoli and N. Margolus, Cellular automata machines : a new environment for modeling, MIT Press, 1987, ISBN: 0262200600, URL: https://dl.acm.org/citation.cfm?id=22919 (cit. on p. 34).

[68] I. Abt, D. Emeliyanov, I. Kisel and S. Masciocchi, "CATS: a cellular automaton for tracking in silicon for the HERA-B vertex detector", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **489** (2002) 389, ISSN: 01689002, URL: http://linkinghub.elsevier.com/retrieve/pii/S0168900202007908 (cit. on p. 34).

[69] V. Kovalenko, "Cellular automaton and elastic net for event reconstruction in the NEMO-2 experiment", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **389** (1997) 169, ISSN: 01689002, URL: http://linkinghub.elsevier.com/retrieve/pii/S0168900297000806 (cit. on p. 34).

[70] N. Braun, "Momentum Estimation of Slow Pions and Improvements on the Track Finding in the Central Drift Chamber for the Belle II Experiment", KIT, 2015, URL: https://ekp-invenio.physik.uni-karlsruhe.de/record/48740/files/EKP-2015-00052.pdf (cit. on pp. 35, 45).

[71] R. Frühwirth, "Application of Kalman filtering to track and vertex fitting", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **262** (1987) 444, ISSN: 01689002, URL: http://linkinghub.elsevier.com/retrieve/pii/0168900287908874 (cit. on pp. 35, 39, 41).

[72] A. Salzburger, "Track Simulation and Reconstruction in the ATLAS Experiment", Leopold-Franzens-Universität Innsbruck, 2008, URL: http://physik.uibk.ac.at/hephy/theses/diss_as.pdf (cit. on p. 39).

[73] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", Journal of Basic Engineering **82** (1960) 11, ISSN: 00219223, arXiv: NIHMS150003, URL: http://fluidsengineering.asmedigitalcollection.asme.org/article.aspx?articleid=1430402 (cit. on p. 41).

[74] P. Aimonen, Basic concept of Kalman filtering, 2011, URL: https://commons.wikimedia.org/wiki/File:Basic_concept_of_Kalman_filtering.svg (visited on 11/03/2018) (cit. on p. 42).

[75] R. Frühwirth and A. Strandlie, "Track fitting with ambiguities and noise: A study of elastic tracking and nonlinear filters", Computer Physics Communications **120** (1999) 197, ISSN: 00104655, URL: http://linkinghub.elsevier.com/retrieve/pii/S0010465599002313 (cit. on p. 43).

[76] J. Rauch and T. Schlüter, "GENFIT — a Generic Track-Fitting Toolkit", Journal of Physics: Conference Series **608** (2015) 012042, ISSN: 1742-6588, arXiv: 1410.3698, URL: http://arxiv.org/abs/1410.3698 (cit. on p. 43).

[77] V. Karimäki, "Effective circle fitting for particle trajectories", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **305** (1991) 187, ISSN: 0168-9002, URL: https://www.sciencedirect.com/science/article/pii/016890029190533V (cit. on pp. 43, 85).

[78] N. Berger, A. Kozlinskiy, M. Kiehn and A. Schöning, "A new three-dimensional track fit with multiple scattering", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **844** (2017) 135, ISSN: 0168-9002, URL: http://dx.doi.org/10.1016/j.nima.2016.11.012 (cit. on pp. 44, 85).

[79] R. Frühwirth, A. Strandlie and W. Waltenberger, "Helix fitting by an extended Riemann fit", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **490** (2002) 366, ISSN: 0168-9002, URL: https://www.sciencedirect.com/science/article/pii/S0168900202009117 (cit. on pp. 44, 85).

[80] M. Friedl et al., "The Belle II Silicon Vertex Detector readout chain", Journal of Instrumentation **8** (2013) C02037, URL: https://doi.org/10.1088/1748-0221/8/02/c02037 (cit. on p. 49).

[81] The APV25 0.25 /spl mu/m CMOS readout chip for the CMS tracker, vol. 2, 2000 9/113 (cit. on p. 49).

[82] P. Virtanen et al., "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python", Nature Methods **17** (2020) 261 (cit. on p. 62).

[83] P. Virtanen et al., "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python", Nature Methods **17** (2020) 261, URL: https://scipy.github.io/devdocs/reference/optimize.minimize-neldermead.html?highlight=nelder%20mead (cit. on p. 63).

[84] E. Lund, L. Bugge, I. Gavrilenko and A. Strandlie, "Track parameter propagation through the application of a new adaptive Runge-Kutta-Nyström method in the ATLAS experiment", Journal of Instrumentation **4** (2009) P04001, ISSN: 1748-0221, URL: http://stacks.iop.org/1748-0221/4/i=04/a=P04001?key=crossref.4db573cdf47b35d9171ab976705899ed (cit. on p. 107).

[85] A. Glazov et al., Measurement of the track reconstruction efficiency and fake rate with $e^+e^- \to \tau^+\tau^-$ events, Belle II Internal Note, 2020, URL: https://docs.belle2.org/record/1867?ln=en (cit. on pp. 124–127, 129).

[86] W. Waltenberger and F. Moser, "RAVE - an Open, Extensible, Detector-Independent Toolkit for Reconstruction of Interaction Vertices", 2006 IEEE Nuclear Science Symposium Conference Record, vol. 1, 2006 104 (cit. on p. 126).

List of Figures

List of Tables

Acknowledgements

First of all I would like to thank Jochen Dingfelder for the opportunity to write my PhD thesis under his supervision. I always felt valued, and I always had the freedom to try out myself and test things and basically do whatever I wanted to do. This resulted in many interesting things and findings that did not make into this thesis, and often were not even reported anywhere – but I learned a lot that I do not want to miss. Among those are the VTX and the TPC.

Second, I want to thank Florian Bernlochner, for being the second examiner for this thesis, but also for being a guide and a help during the whole time, starting with my master's thesis where he was my postdoc. This continued while became a PhD student and he was still a postdoc here in Bonn before, moving to Karlsruhe as a Professor, and since then until now, and also thanks to Julie for some valuable advice. I learned so much from both of you, Jochen and Florian, thank you so much!

Next, I want to thank Peter Lewis and Andreas Löschcke Centeno for our awesome collaboration and work on the Belle II(I) TPC. I really hope that this project makes it to reality, and I am still astonished about what we achieved within just a year, while having so much fun developing the TPC and discussing issues. Thanks a lot!

But I do not only want to thank Peter for working together on the TPC, but for helping and guiding me throughout my thesis since he arrived in Bonn. I always knew I could get help next door, or online during Covid times.

Very special thanks also go to Stephan Duell for being a support for the whole time, and very often my go-to person to discuss things, and in the end also for becoming a really good friend. I really hope to celebrate your PhD with you soon, you deserve it my friend. Stephan, Peter, Andreas: no-one bows to you, my friends ;).

Thank you also to Bruno Deschamps for sharing the ride with me developing DATCON. It took us a long time, is still not done, but in the end I hope it will be a pretty useful system. And also thank you to Botho Paschen for your support. Similar to Stephan, I hope you will finish soon, you well deserve your PhD.

I also would like to thank all of my colleagues, both in hardware (the whole Silab) and analysis. Thanks to Ralf Farkas, Ilias Tsaklidis, and Martin Angelsmark for proof reading my thesis, to Benedict Winter for the valuable discussions when he was still in Bonn, and to Eckhard von Törne for all the valuable advices I got from you over the time. Thanks a lot, it was always fun working with you. And all the others: thanks a lot, it was always fun working with you.

In addition to my local colleagues I also want to thank my external colleagues, especially the tracking group and the upgrade group. I learned a lot from being part of the tracking group, and even more on C++ programming from being part of the group, thus thanks to: Nils Braun, Thomas Lück, Simon Kurz, Giulia Casarosa, Sasha Glazov, Laura Zani, Petar Rados, Leonard Polat (the last three for their support on the di-tau study), Martin H., Thomas H., Eugenio P., and all the other (former) members of the tracking group that made it a pleasure to work with and to work on tracking, and I am

loocking forward to working with you in the future and to further improving our tracking.

Among my other Belle II colleagues, special thanks to Benjamin Schwenker for the work and incredible achievements on the VTX. We will continue and make the VTX happen. Besides that I want to thank Carlos Marinas, Francesco Forti, and Jerome Baudot for their work on and support with the VTX, and helping me to grow within the VTX project.

Second to last: Thanks to my friends since the first semester here in Bonn: Florian Beisiegel, Michael Hübner, and Katrin Kohl. Ten and a half years, and here we are. It might have been possible without you, but much harder, less fun, and very different. Thanks a lot my friends, without you I might not be here. Again: no-one bows to you, my friends.

Last but not least I want to thank my family. You know what I was going through, and you always were by my sight. I can't possibly appreciate that enough.

Of course with very special thanks to my parents and my brother. I always knew I could rely on you, and visit you whenever I wanted or needed. Where ever you lived, whatever the situation was, I always knew I could come by and spend a couple of days with you to relax. It wasn't always easy for various reasons, but I am very grateful for your support over all the years – and here I am. And to my brother: thanks for everything. If I was stuck, I always knew I could rely on you, even as "the little brother". You often enough were my backbone, and if nothing was right, geeking around with you and sharing our interests in many fields more than once save my day (or me, see it as you like ;)).

Patrick, Mama, Papa, Marion, Wolfgang: Danke für alles! Ohne euch wäre ich nicht hier!