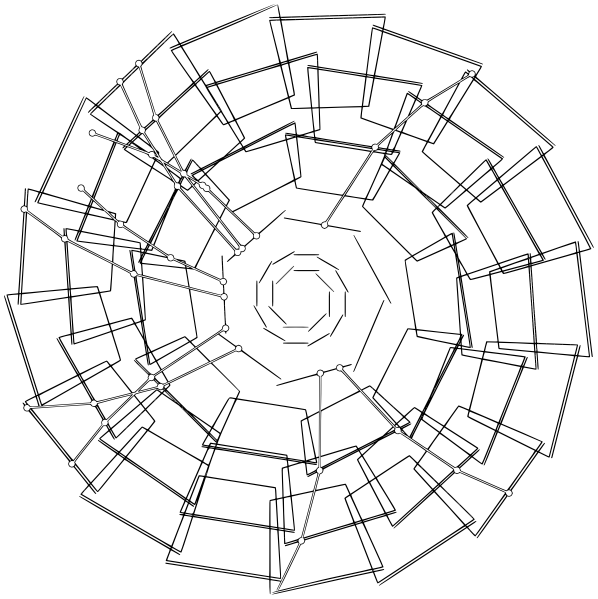
TECHNISCHE UNIVERSITÄT WIEN

Dissertation

Real-time Pattern Recognition in the Central Tracking Detector of the Belle II Experiment



Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der Naturwissenschaften unter der Leitung von

Univ.-Doz. Dipl.-Ing. Dr.techn. Rudolf Frühwirth

Institut für Stochastik und Wirtschaftsmathematik (E105) und
Institut für Hochenergiephysik der ÖAW

eingereicht an der Technischen Universität Wien
Fakultät für Mathematik und Geoinformation

von

Mag. rer. nat. Jakob Lettenbichler

Matrikel-Nr. 0207599

1220 Wien, Claretinergasse 3/3/5

Wien, am 24.10.2016

KURZFASSUNG

Das Belle II Experiment wird aktuell im Forschungszentrum KEK in Tsukuba, Japan, errichtet. Belle II ist ein Detektor am asymmetrischen Speicherring SuperKEKB, in dem Elektronen und Positronen zur Kollision gebracht werden. Die Kollisionsenergie entspricht der Masse der $\Upsilon(4S)$- bzw. der $\Upsilon(5S)$-Resonanz, das heißt etwa $10.58\,\mathrm{GeV}/c^2$ bzw. $10.86\,\mathrm{GeV}/c^2$. Diese Resonanzen zerfallen vorwiegend in Paare von B-Mesonen. SuperKEKB wird daher auch als eine B-Meson-Fabrik bezeichnet. Mit einer integrierten Luminosität von $8 \times 10^{35}\mathrm{cm}^{-2}\mathrm{s}^{-1}$ wird SuperKEKB etwa 40 mal mehr Kollisionen produzieren als sein Vorgänger, der Speicherring KEKB. Belle, das Vorgängerexperiment von Belle II, nahm über viele Jahre Messungen der von KEKB produzierten Kollisionen vor.

Aufgrund der wesentlich höheren Datenrate und der verringerten Asymmetrie der beiden Strahlen wird sich Belle II in wesentlichen Punkten von seinem Vorgänger Belle unterscheiden. Insbesonders wird parallel zur Aufrüstung des Detektors auch die komplette Rekonstruktionssoftware neu entwickelt. Diese Doktorarbeit beschreibt den VerteX Detector Track Finder (VXDTF[1]), ein Mustererkennungsprogramm für Spurrekonstruktion, das keinen Vorgänger in der Software von Belle hat. Seine Aufgabe ist es, im innersten Spurdetektor, dem VXD, Spuren von geladenen Teilchen zu rekonstruieren. Der VXD besteht aus zwei Teildetektoren, die jeder aus mehreren Lagen von Siliziumsensoren bestehen. Der innere Teil, der mit DEPFET[3]-Technologie gefertigte Pixel Detektor (PXD[4]), hat zwei Lagen und zeichnet sich durch seine hohe Auflösung und seine extrem dünnen Sensoren aus. Der äußere Teil ist der mit vier Lagen von doppelseitigen Streifensensoren ausgestattete SVD[5]. Er zeichnet sich durch seine hohe Zeitauflösung und die schnelle Auslösezeit aus. Der SVD wurde in am Institut für Hochenergiephysik in Wien entwickelt

[1] The VXD[2] Track Finder package
[3] DEPleted p-channel FieldEffect Transistor
[4] PXD: The PiXel Detector
[5] SVD: The Silicon Vertex Detector

und wird nun in Kollaboration mit anderen Instituten fertig gebaut. Von den beiden Teildetektoren kann nur der SVD zur Rekonstruktion in Echtzeit benützt werden. Die im SVD gefundenen Teilchenspuren werden dann in den PXD extrapoliert und definieren dort jene Regionen, die tatsächlich ausgelesen werden. Dieser Mechanismus der Datenreduktion ist notwendig, weil der PXD mit seinen rund 8 Millionen Pixeln mit je 8 Bit Auflösung und 50.000 "Bildern" in der Sekunde zu viele Daten aufnimmt, als dass man diese komplett auslesen und auf Dauer speichern könnte.

Der Schwerpunkt des VXDTF liegt deshalb auf der Rekonstruktion von Teilchenspuren innerhalb des SVD, wobei das Ziel ist, auch niederenergetische Spuren mit einem Transversalimpuls von lediglich 50 MeV/c zu rekonstruieren. Dazu stehen lediglich vier Lagen von Sensoren zur Verfügung, sodass die Redundanz der Messung sehr gering ist. Auch werden solche Spuren stark durch stochastische Materialeinflüsse gestört. Nicht zuletzt wird die Rekonstruktion durch einen hohen, vom Beschleuniger selbst erzeugten Untergrund erschwert. Damit der Spurfinder dieser Aufgabe gewachsen ist, wird er aus folgenden Einzelteilen zusammengesetzt:

- Zunächst wird die kombinatorische Problematik abgeschwächt, indem physikalisch unplausible Kombinationen unterdrückt werden. Ein effektiver Satz von Filtern, welche anhand von simulierten Daten trainiert werden, wird in einer sogenannten SecMap[6] abgespeichert. Die akzeptierten Kombinationen von Messpunkten werden dann in einer Baumstruktur abgespeichert.

- Diese Baumstruktur wird dann als Basis für einen zellulären Automaten benützt, welcher dann Spurkandidaten, sogenannte TC[7]s, findet.

- Die Qualität dieser TCs wird mit einer statistischen Methode abgeschätzt und gespeichert.

- Falls sich TCs einen oder mehrere Messpunkte teilen — also überlappen — wird mittels eines Hopfield-Netzes eine nicht überlappende Teilmenge von Spurkandidaten ermittelt. Dabei gehen die geschätzten Qualitätsindikatoren in die Definition der Gewichte und Schwellwerte ein.

Die am Ende übriggebliebenen Spurkandidaten werden dann für die Extrapolation zum PXD benützt.

Diese Doktorarbeit beschreibt kurz den Belle II Detektor und den SuperKEKB Speicherring. Weiters werden die zu erwartenden Eigenschaften der Kollisionsprodukte und des Untergrunds beschrieben. Die sich daraus ergebenden Konsequenzen auf die Methodik der Spurrekonstruktion werden diskutiert. Die Komponenten des VXDTF werden im Detail vorgestellt. Die Leistungsfähigkeit der Spurrekonstruktion wird anhand von simulierten Daten samt hinzugefügtem Untergrund analysiert und gemessen. Der

[6] SecMap: SectorMap
[7] Track Candidate

Spurfinder wurde auch bei einem Teststrahl im deutschen Forschungszentrum DESY[8] eingesetzt, sodass sich der VXDTFs auch anhand von echten Sensordaten bewähren musste. Die Resultate dieses Tests werden ebenfalls präsentiert. Zuletzt werden noch Ausblicke auf die nächsten Schritte gegeben, sowie der erweiterte VXDTF 2[9] vorgestellt.

Die Leistungsfähigkeit des VXDTF wird hier noch kurz zusammengefasst. Die Rekonstruktionseffizienz des VXDTF für $\Upsilon(4S)$-Zerfälle bei vollem Untergrund liegt bei einem Durchschnittswert von etwa 87%. Für niederenergetische Spuren mit einem transversalen Impuls von 100 MeV/c liegt die Effizienz noch über 80%, während sie für 50 MeV/c Spuren auf eta 65% abfällt. Alle weiteren Details finden sich in der in englisch verfassten Doktorarbeit.

[8] DESY: Deutsches Elektron Synchrotron
[9] The refactored version of the VXD Track Finder package

ABSTRACT

The Belle II[10] experiment is currently under construction in the KEK laboratory in Tsukuba, Japan. It will record events produced by the world's only second-generation B factory[11], the SuperKEKB[13] collider, an asymmetric e^+e^- storage ring with 3 km circumference. The collider will operate at the collision energy corresponding to the mass of the $\Upsilon(4S)$ and the $\Upsilon(5S)$ resonances, i.e. $10.58\,\mathrm{GeV}/c^2$ and $10.86\,\mathrm{GeV}/c^2$, respectively. With an integrated luminosity of $8 \times 10^{35}\mathrm{cm}^{-2}\mathrm{s}^{-1}$ it will produce about 40 times more events than its predecessor KEKB[14] and therefore more than any other before. Neither the detector of the predecessor experiment Belle[15] nor its reconstruction software would have been able to deal with the amount of events expected and their properties. Therefore the entire event reconstruction software was redesigned from scratch, while crucial parts of the Belle were improved and rebuilt to form the new Belle II detector.

This thesis describes the VerteX Detector Track Finder (VXDTF) which has no predecessor in the old reconstruction software. Its task is to reconstruct tracks in the innermost tracking detector, the VXD, a silicon based detector consisting of two parts, the PXD with two layers of pixel sensors using DEPFET technology, and the SVD with four layers of double-sided silicon strip sensors. The potential amount of data recorded by the PXD with its 8 million pixels with 8-bit depth each, read out with about 50,000 readout frames per seconds, cannot be stored, unless a data reduction by a factor of about 10 can be performed.

To this end, the VXDTF will be deployed to reconstruct tracks in the SVD in real time. These tracks are then extrapolated to the PXD where they define regions of interest, that are then read out. The focus of the VXDTF therefore lies on full event reconstruction

[10] Belle II: experiment of the second generation B factories
[11] B factory: a collider experiment optimized for producing and reconstructing B meson[12]s
[13] SuperKEKB: second generation B factory in KEK
[14] KEKB: first generation B factory in KEK
[15] Belle: experiment of the first generation B factories

in the SVD only, with an additional emphasis on low-momentum tracking down to a transverse momentum of 50 MeV/c. Such low-momentum tracks can be reconstructed only by a track finder operating in the innermost tracking detector. Given the small number of sensor layers in the SVD, the VXDTF hast to deal with little redundancy, with the stochastic disturbances due to material effects that are particularly strong for low-momentum tracks, and with the increased background level caused by the high luminosity of the collider.

The VXDTF consists of several steps:

- In a first step, combinatorics are reduced by discarding physically implausible hit combinations. An effective set of Filter[16]s with cuts trained on simulated data is stored in a container called SecMap. The filters reduce the hits and their accepted combinations, which are then stored as a Tree[17].
- On this Tree a CA[18] is applied, which is used for extracting track candidates, the TCs.
- These TCs then are fitted by a fast fit, which estimates their quality, stored in a quality indicator, the QI[19].
- If there are TCs that share one or more hits, a Hopfield network is used to find a subset of non-overlapping TCs. The definition of the weights and thresholds of the neural network uses the QIs, so that the final subset contains the TCs with the highest quality.

The final TCs produced by the VXDTF are then used for ROI[20] finding on the PXD, as previously described.

This thesis describes the detector Belle II, the machine SuperKEKB and their features; illustrates how typical events recorded by the SVD look like; describes the details of the VXDTF implementation; analyzes its performance on MC[21] events of $\Upsilon(4S)$ decays with realistic background; reports its performance at a combined beam test where PXD and SVD sensors were jointly tested in an electron beam at the DESY laboratory. Finally, it describes the next steps regarding the VXDTF and its successor, the VXDTF 2, which is also described in this thesis.

The performance of a current version of the VXDTF is summarized here. The reconstruction efficiency of the VXDTF for $\Upsilon(4S)$ events with full background included is about 87% for all MC tracks creating enough hits in the SVD to be able to reconstructed.

[16] Filter: filters hits combinations by calculating a value, which is then compared with predefined cuts.
[17] Directed graph wihtout loops
[18] Cellular Automaton
[19] Quality Indicator
[20] Region Of Interest
[21] Monte Carlo

For tracks with a transverse momentum of 100 MeV/c the efficiency is still above 80% and for 50 MeV/c its about 65%. The complete details can be found in this thesis.

Contents

Part I

Track Finding in the Vertex Detector of Belle II

CHAPTER 1

OVERVIEW

The emphasis of this thesis is the presentation of a low momentum tracking approach — the VXDTF — in the Belle II experiment. Belle II is situated at the SuperKEKB collider in Tsukuba, Japan.

Chapter 1 is meant to be a refresher of the basics, to explain the environment to be operated in, to describe special aspects relevant for low momentum tracking and to illustrate what tracking is. In Section 2.1 a basic introduction to the Standard model of high energy physics is given; Section 2.2 discusses the nature and impact of ***CP*** violation[22]. Section 2.3 discusses the characteristics of B factories[23] and how they can be used to study B mesons. The Belle II detector is described in Section 2.4 with an emphasis on the VXD detector, the inner silicon sensor tracking detector of Belle II which is described closely in Section 2.4.1. The relevance of material effects especially for low momentum tracking is discussed in Section 2.5, while Section 2.6 gives an overview of expected background types created by SuperKEKB or at the interaction point — the IP[24]. The basics of event reconstruction are sketched in Section 2.7 with an emphasis on track finding in Section 2.7.1. Section 2.7.2 clarifies why a stand-alone track finder is relevant for low momentum tracking.

In Chapter 3 a detailed overview of the characteristic of $\Upsilon(4S)$ events is given. This chapter discusses relevant aspects to be considered by a low momentum track finder. A short overview of the chapter can be found in Section 3.1, while the actual study is discussed in Section 3.2. An emphasis on particle start and end vertices of charged and relevant uncharged particles is given in Section 3.2.1, while the expected occupancy level with cluster- and hit-characteristics and their combinatorial issue is discussed in

[22] ***CP*** violation: charge-parity violation
[23] more details to be found in B factory
[24] Interaction Point

Section 3.2.2. Particle distributions over momentum and the direction angles θ and ϕ are shown in Section 3.2.3. A summary of the event characteristics is given in Section 3.3.

The low momentum track finder — VXDTF — is closely described in Chapter 4. An introduction is given in Section 4.1 and the basic structure of the VXDTF can be found in Section 4.2. The concept of the SecMap and its role in the VXDTF is described in Section 4.3 including key elements and their meaning in Section 4.3.1. How the SecMap is trained can be found in Section 4.3.2 and its deployment in Section 4.3.3 including the details of the filters. How track candidates — TCs — are then found is closely described in Section 4.4, including the specifics of the implemented CA algorithm in Section 4.4.1. After collecting the TCs their quality is estimated, which is described in Section 4.5, where several implemented methods are discussed, like a circle fit (Section 4.5.3) or a Kalman filter (Section 4.5.5). In Section 4.6 the procedure to find sets of TCs without overlaps is discussed, including the implemented algorithms to perform this task, like a Hopfield neural network in Section 4.6.2. The VXDTF uses different SecMaps specialized on different momentum ranges and therefore Section 4.7 describes how the output of these different momentum dependent passes can be merged in the end. A summary and an outlook of the VXDTF is then given in Section 4.8.

In Chapter 5 the performance of the VXDTF is evaluated. Important key values are defined in Section 5.1.1. A thorough check of $\Upsilon(4S)$ events without background added is discussed in Section 5.2, while events with background only are discussed in Section 5.3. How the VXDTF performs in events with $\Upsilon(4S)$ and different background levels is shown in Section 5.4 and in Section 5.4.4, where different thresholds are used to check high occupancy events as well. An overview and a summary of observed aspects relevant for interpreting the TF[25]s performance can be found in Section 5.5.

The VXDTF has been used for on-line tracking in a beam test performed at the DESY in Hamburg in January 2014. Chapter 6 discusses this beam test and the performance of the TF in that environment. An overview of the beam test can be found in Section 6.1 and the experimental setup is described in Section 6.2. The particle source is described in Section 6.2.1, the detector setup can be found in Section 6.2.2 and a description of the data acquisition (DAQ[26]) is given in Section 6.2.3. The high level trigger (HLT[27]) is described in Section 6.2.4. How track finding was used in the beam test is described in Section 6.3, where Section 6.3.1 describes the FPGA[28] based track finder of the DATCON[29] system. The settings of the VXDTF and its differences to the standard Belle II settings are discussed in Section 6.3.2. Beam test specific detectors like telescopes and its support in the VXDTF is then described in Section 6.3.3, including the performance of the off-line tracking in Section 6.3.3.2. A conclusion of the beam test performance is then given in Section 6.4.

[25] Track Finder
[26] Data AcQuisition
[27] HLT: High Level Trigger
[28] Field Programmable Gate Array.
[29] DATCON: DATa CONcentrator

The redesign of the original VXDTF, which is called VXDTF 2, and its specialties is discussed in Chapter 7. Why a redesign process was started is described in Section 7.1. An overview of the structure of VXDTF 2 is shown in Section 7.2. The first step of the VXDTF 2 is done before the TF specific modules are run: creating SpacePoint[30]s formed of clusters from the SVD and the PXD. The process is described in Section 7.3. After that a network of compatible hits is formed, which is described in Section 7.4, whereas certain important classes of that step are described in Section 7.4.1, while its deployment is discussed in Section 7.4.2. The implemented and planned algorithms which can find tracks are described in Section 7.5. The quality of the TCs are estimated in the quality estimator step, which is summarized in Section 7.6, including possible algorithms planned for that step. To find clean subsets of non-overlapping tracks, the subset finder step is applied and described in Section 7.7. The final step before repeating the process for different passes is reserving the hits for subsequent passes, which is described in Section 7.8. Preliminary results of the VXDTF 2 are shown in Section 7.9.

The final chapter of the thesis is a short outlook of next steps regarding the VXDTF and its successor, the VXDTF 2, which is described in Chapter 8.

The appendix offers some extra chapters which go into more detail about some aspects relevant for keeping them in mind.

Chapter 9 is meant for documentation of the exact formulas of the filters implemented in the VXDTF.

The Chapter 10 discusses relevant design decisions made for the VXDTF 2, among them the decision of how to define the segments used as Cell[31]s for the CA, which can be found in Section 10.1 and Section 10.2. The discussion of the relevance of different hit classes is written down in Section 10.3. Some more details about the creation of networks of hits and track segments are discussed in Section 10.4.

The settings used for the studies described in Chapter 5 are written down in detail in Chapter 11 including the hardware setup used for the studies, which is listed in Section 11.5.

Finally details about error bars shown in some plots are discussed in Chapter 12.

[30] SpacePoint: Global position of a hit in the VXD.
[31] Cell: CA-Cell.

CHAPTER 2

BASICS

This chapter is deliberately kept short to keep this thesis short as well. A more detailed look at the physics part of the chapter can be found in my diploma thesis [1] or in the most current view of the physicists of the first generation B factories [2], while an older but well written book is [3]. The accelerator- and detector-specific sections are described in more detail in [4], [5] and [6].

2.1 The Standard Model

The Standard Model of particle physics (SMHEP[32]) was developed in the early 1970s and is now the theory broadly accepted for describing matter, its antimatter counterparts, and the particles mediating the forces between them, the gauge bosons. Matter consists of quarks and leptons that can be arranged in three generations. Only particles of the first generation constitute the atoms in the known universe.

Corresponding particles in the three generations share most characteristics, for instance the charge, but have different mass and lifetime. In contrast to leptons such as electrons, which can easily be observed isolated in nature, quarks can not exist freely in nature, but are always clustered to hadrons, such as protons or pions. The interaction between quarks is described by the strong force, which affects only particles with color charge. The color charge can be compared to the electrical charge, but whereas the electrical charge has only two types, positive and negative, the color charge can be red, green or blue, plus the respective anti-colors. All objects formed by particles having color charge must have a total color of white to be observable. This can be achieved by a quark-triplet (e.g. red+green+blue = white) or a quark-antiquark-doublet, where one quark carries a color and the other one its anti-color. Hadrons consisting of three quarks are called baryons,

[32] SMHEP: Standard Model of HEP[33]

of which the proton consisting of the quarks triplet *uud* is the most common and only stable representative. Hadrons consisting of a quark and an antiquark are called mesons. Despite the fact that mesons are never stable and therefore decay within a fraction of a second, the following mesons play a central role in the studies of ***CP*** violation: the Υ-meson ($b\bar{b}$), the B mesons $B^+(u\bar{b})$, $B^-(b\bar{u})$,$B^0(d\bar{b})$ and $\overline{B^0}(b\bar{d})$. Section 2.3 describes their role they play in a B factory.

More details about the forces can be found in the list below. All leptons and quarks belong to the group of fermions, which have spin of $^1/_2$ and obey the Fermi-Dirac-statistics. Interactions between fermions are mediated by gauge bosons, which have integer spin and obey the Bose-Einstein-statistics. All the particles mentioned here are summarized in Figure 2.1.

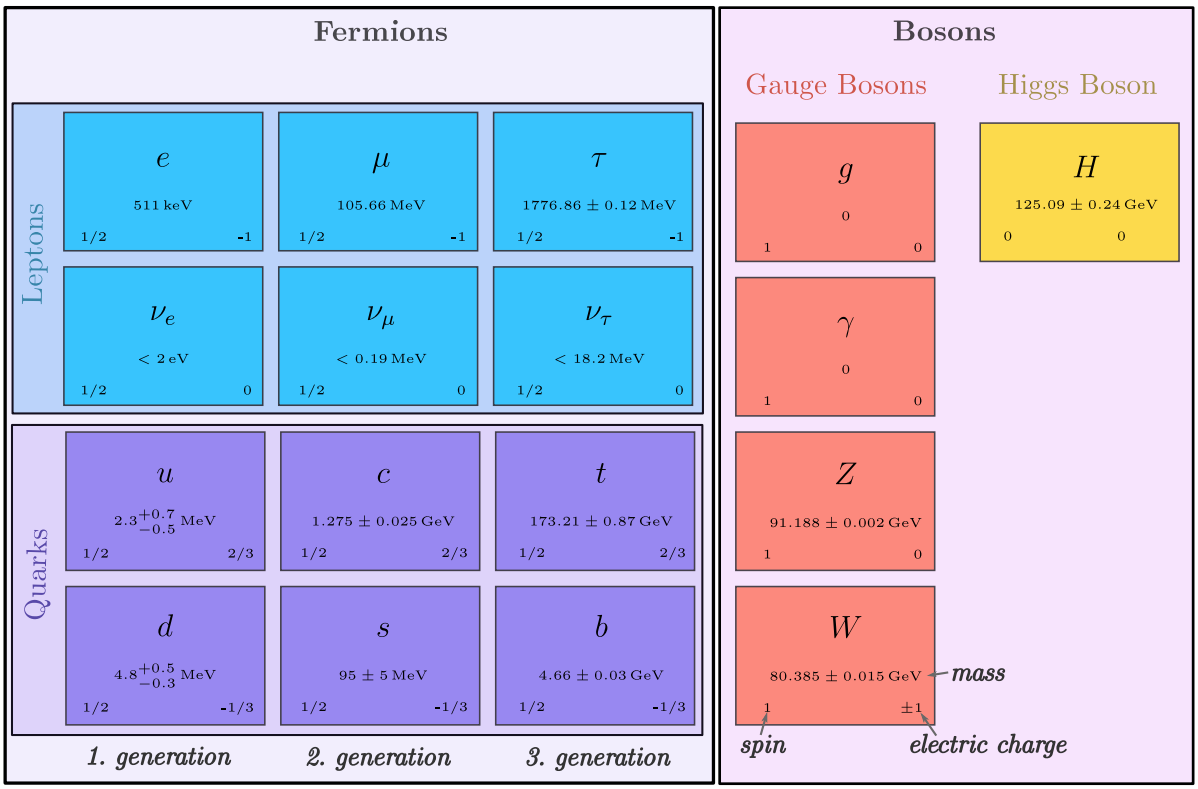


Figure 2.1: An overview of the particles of the SMHEP. The leptons and quarks are ordered in generations (column 1-3). Modified sketch courtesy of Robin Glattauer of [7]

The SMHEP is currently the most unified and accepted theory of physics since it describes the properties of three of the four fundamental forces currently known to us, which are listed below:

- **The electromagnetic force** explains interactions of particles carrying an electric charge, such as electrons and protons, but not of charge-less particles like neutrons.

The mediating gauge boson of this force is the photon (γ), which we mostly recognize as light, radio waves or gamma-rays. Its strength relative to the strong force is 10^{-2} when comparing all forces at the scale of a single proton.

- **The strong force** explains the interaction of particles carrying color charge. The gauge bosons mediating the strong force are called gluons and cannot be observed directly. This force is responsible for keeping together the atomic nuclei, although the electric charges of the protons repel each other. Although its normalized strength at proton scale is 20 times stronger than the electromagnetic force, it does not play a role in the macroscopic world, as it has only a range of a few femtometer, which is the reason why big atoms like the transuranium elements are not stable.

- **The weak force** acts on all fermions, which all carry a flavor[34]. It has the ability to change quarks into other types of quarks and thereby explains the mechanism of the radioactive beta decay. More details about the weak force can be found in Section 2.2. Its mediating gauge bosons are the $W^{\pm}$ and Z^0-bosons. It is about 10^{-9} times weaker than the strong force at the proton scale. Compared to the strong force its range is even more limited to the < 1 femtometer-range.

- **Gravitation** is omnipresent in the macroscopic world, but very weak at the proton scale. Relative to the strong force its strength is about 10^{-38} and therefore plays no role in the SMHEP. Its proposed gauge bosons are called gravitons, which have not yet been observed, neither directly nor indirectly.

The interactions of the elementary particles are sketched in Figure 2.2a. Figure 2.2b places the SMHEP in relation to other important physical theories.

Although the SMHEP is very powerful in explaining interactions of particles and is therefore a "theory of almost everything", it has some severe shortcomings, which are listed below in no particular order and in an incomplete manner:

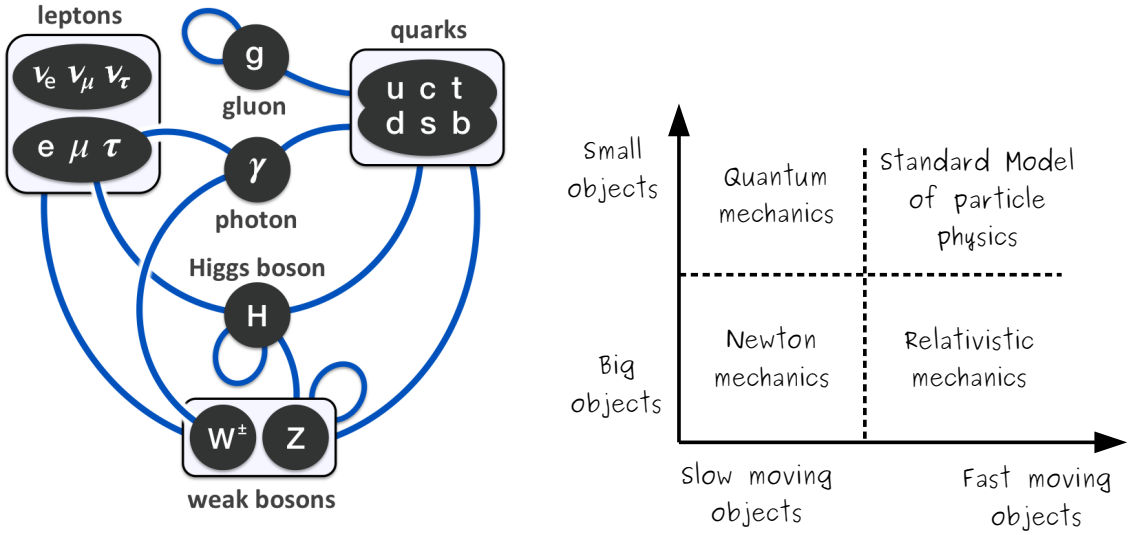
- It introduces no less than three generations of quarks (more about that in Section 2.2), but it does not explain why there are not more than three generations.

- Many essential parameters such as the masses of the quarks and leptons can be determined only experimentally.

- It does not include gravitation.

- It is not able to explain *dark matter* and *dark energy*, which constitute about 95% of our universe.

[34] there are six flavors for quarks and six for leptons.



(a) An overview of the interaction of particles. The picture has been published in [8] with a public license.

(b) The Standard Model in relation to other established theories of physics.

Figure 2.2: 2.2a summarizes the interactions relevant in the SMHEP, while 2.2b positions the SMHEP in the context of other theories.

- Although it is the only well-tested theory which explains ***CP*** violation at all (more details about that can be found in Section 2.2), it can only explain a very small fraction of the total impact of ***CP*** violation that happened right after the creation of the universe.

This indicates that the SMHEP can not be the final answer to the description of the universe. Especially ***CP*** violation is a dominating topic for the experiments at the B factories described in Section 2.3, which probe for flaws in the description of ***CP*** violation in the SMHEP. More about ***CP*** violation can be found in Section 2.2 and [2].

2.2 CP Violation

At the big bang marking the beginning of the universe equal amounts of matter and anti-matter must have been created. But nowadays no such amounts of anti-matter can be observed in the universe, but only matter. This asymmetry between matter and anti-matter is called the "baryon asymmetry", which Andrei Sakharov was the first one able to define sufficient criteria for an explanation [9]. Of these criteria ***CP*** violation was the center of interest, due to the discovery of the ***CP*** violation in 1964 [10].

It took until 1973 to find a satisfying theoretical description of the effects observed. It was published by Kobayashi and Masukawa in 1973 [11], which extended the Cabbibo-idea [12] of having more than one quark-generation to having (at least) three of them. The proposed existence of three generations allowed the introduction of a complex phase into quark mixing, which was expressed in the CKM[35]-matrix and added the possibility of ***CP*** violation to describe the matter-antimatter-asymmetry via the SMHEP. This model of quark mixing postulates that the quark eigenstates relevant to the weak interaction (called *weak eigenstates*) are not identical to the *mass eigenstates* relevant to other interactions. The CKM mechanism was soon accepted after being introduced since some of its predictions could already be observed shortly afterwards (the c-quark in 1974 [13] and the b-quark in 1977 [14]).

It was already known in the 1970s that ***CP*** violation was expected to be found also in the B meson sector. Therefore a focus was laid on $\Upsilon(4S)$ studies which produce a high quantity of B mesons. A high probability of decays $B^0 \longrightarrow J/\Psi K^0_S$, a decay which violates ***CP***, was expected. But around 1980 experimentalists searching for ***CP*** violation observed far fewer events violating ***CP*** than would be needed to fully explain the matter-antimatter-asymmetry of the universe, although the measurements were consistent with the CKM and therefore with the SMHEP. The first focused attempts on ***CP*** violation-discovery, the CESR[36]-experiment in Cornell and DORIS-II[37] in DESY, did not yield the expected results because of the technical constraints of that time, but they paved the way for the B factories PEP-II[38], KEKB and SuperKEKB.

Kobayashi and Masukawa were awarded the Nobel prize in 2008 for their paper of 1973 [11], after ***CP*** violation was successfully probed with the first generation B factories. The reason for this gap of over 30 years was that it turned out to be a long road to a sufficiently copious mass production of events in which ***CP*** violation occurs. The problems are summarized in Section 2.3.1.

2.3 B factories and SuperKEKB

2.3.1 Prerequisites for B factories

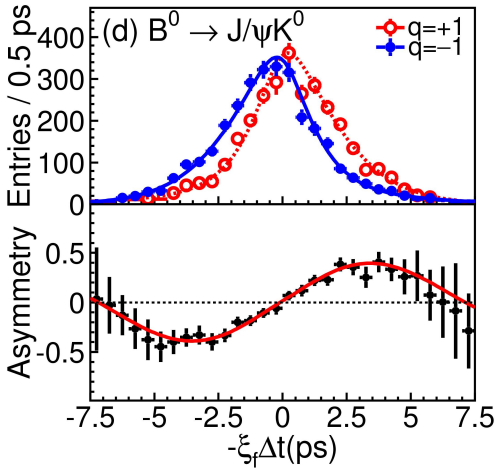
B factories are highly optimized storage rings that bring electrons (e^-) and anti-electrons (e^+) to collision at the $\Upsilon(4S)$ (a $b\bar{b}$ meson in its fourth excited state) and $\Upsilon(5S)$ ($b\bar{b}$ in its fifth excited state) collision energy of about 10–11 GeV. These energies are relevant to a B factory because the mass of the $\Upsilon(4S)$-resonance of 10.58 GeV is just above the threshold to let the $\Upsilon(4S)$ decay into two B mesons in more than 96% of the cases. This

[35] CKM: CKM mechanism
[36] CESR: Cornell Electron Storage Ring
[37] DORIS-II: DOppel-RIng-Speicher
[38] PEP-II: first generation B factory in SLAC

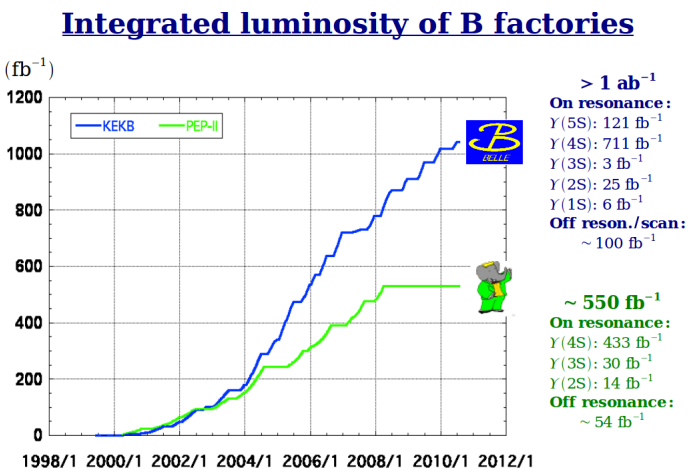
is the reason why such a collider is called a B factory. More details about B factories can be found in [2].

Of the B mesons produced, 50% are charged ($B^+ B^-$) and 50% are neutral ($B^0 \overline{B}^0$), where the neutral ones allow studies of the decay modes relevant for ***CP*** violation. Therefore SuperKEKB — like its predecessors PEP-II and KEKB — is optimized to fulfill the three essential requirements for B factories, which are a summary of the requirements listed in [2]:

- **High luminosity:** The dominant decay mode for a measurement of ***CP*** violation is $B^0 \longrightarrow J/\Psi\, K^0_S$, which only has a branching fraction of $\sim 0.04\%$ of the $B^0 \overline{B}^0$ decays, and the most easily detectable decays $J/\Psi \longrightarrow e^+e^-$ and $J/\Psi \longrightarrow \mu^+\mu^-$ occur only in $\sim 12\%$ of all J/Ψ decays. Therefore only one in about 20,000 $B^0 \overline{B}^0$ decays is relevant at all, and millions of them have to be recorded to have enough statistics for precise measurements of ***CP*** violation. Figure 2.3a shows the results measured by Belle. The total number of recorded events is usually expressed in



(a) Difference in lifetime for B^0 and $\overline{B}^0$. Plot courtesy of the Belle Collaboration.



(b) Comparison of the integrated luminosities achieved by the first generation B factories. Plot courtesy of the Belle Collaboration.

Figure 2.3

terms of integrated luminosity and is measured in units of fb^{-1}[39] or ab^{-1}[40]. BaBar[41] has recorded more than 400 million $\Upsilon(4S)$ decays, and Belle has recorded over 700 million of them. Figure 2.3b shows how the integrated luminosities of the first generation B factories has grown over the years.

[39] fb^{-1}: inverse femto-barn, $\widehat{=}$ 500,000 $B^0 \overline{B}^0$-events.
[40] ab^{-1}: inverse atto-barn $\widehat{=}$ 1000 fb^{-1}
[41] BaBar: experiment of the first generation B factories

- **Boosted $B^0\overline{B}^0$ pairs:** At a symmetric collider with the energies used to create $\Upsilon(4S)$ particles, these particles would be produced almost at rest in the laboratory frame, and therefore the ***CP*** violation-relevant B^0-$\overline{B}^0$ mixing would be next to impossible to detect. Because of that a B factory has to use asymmetric beam energies in the two rings to give the particles a Lorentz boost.

- **High resolution and hermetic detector:** All of the particles produced in a $\Upsilon(4S)$ decay have to be detected and correctly identified. Therefore a high-resolution detector is needed, which covers as much as possible of the solid angle covering around the IP. Equally important is a powerful reconstruction software, a vital part of which is the reconstruction of charges tracks. A description of the Belle II-detector and its inner tracking detectors can be found in Section 2.4 and Section 2.4.1.

2.3.2 SuperKEKB

SuperKEKB is the sole successor of the first generation B factories KEKB and PEP-II. PEP-II was built at SLAC in Stanford (U.S.A.) and was operated from 2000 to 2008. KEKB was built at KEK in Tsukuba (Japan) and was operated from 2000 to 2010. SuperKEKB is currently (2016) being built as an upgrade of KEKB and will start producing collisions at the end of 2017.

Table 2.1 summarizes some relevant specifications of the three B factories, while Figure 2.4 represents a graphical overview of the SuperKEKB collider.

SuperKEKB will produce events of accumulated 50 ab^{-1}, which will be about 50 times the number of events produced by KEKB. The plan for the recording cycles can be seen in Figure 2.5.

This will be achieved by increasing the amount of electrons and positrons stored in a larger number of bunches, leading to a higher bunch crossing frequency. The highest impact will be provided by the so-called "nano beam scheme" around the IP, which reduces the beam size at the IP by a factor of 8 in x, 20 in y and over 30 in z compared to the IP design of KEKB. This

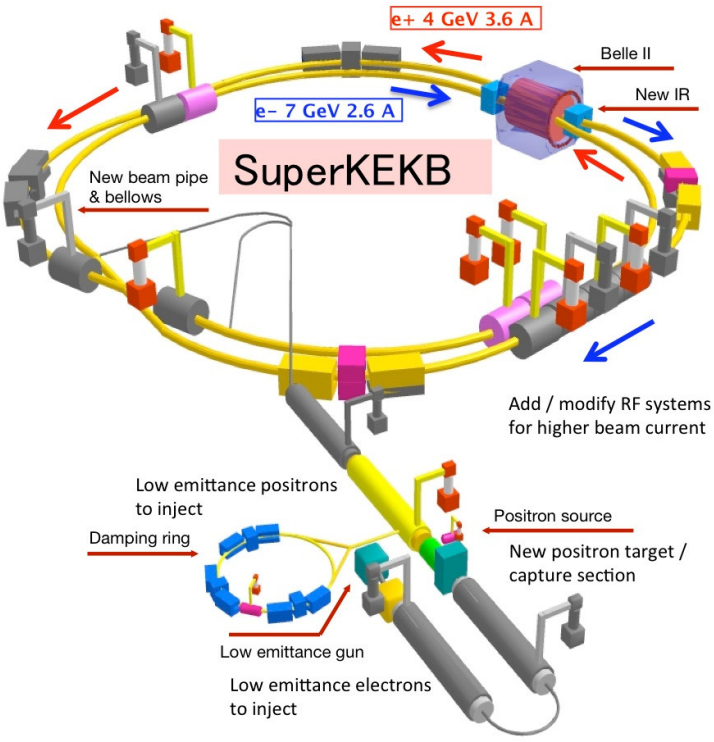


Figure 2.4: Simplified overview of the SuperKEKB-collider. Plot courtesy of the Belle II Collaboration.

Parameters	PEP-II	KEKB	SuperKEKB
Beam energy (GeV)	9.0(e^-), 3.1(e^+)	8.0(e^-), 3.5(e^+)	7.0(e^-), 4.0(e^+)
Beam current (A)	1.8(e^-), 2.7(e^+)	1.2(e^-), 1.6(e^+)	2.62(e^-), 3.6(e^+)
Number of bunches	1732	1584	2503
Bunch spacing (m)	1.25	1.84	1.2
Beam crossing angle (mrad)	0 (head-on)	$\pm$11 (crab-crossing)	83 (nano-beam)
Beam size at IP x (μm)	140	80	$\sim$ 10
Beam size at IP y (μm)	3	1	$\sim$ 0.05
Beam size at IP z (μm)	8500	5000	150
Bunch length	8500	5000	5000
Lorentz boost ($\beta\gamma$, no unit)	0.56	0.42	0.28
Luminosity (10^{34}cm^{-2}s^{-1})	1.21	2.11	80

Table 2.1: some parameters of the PEP-II, KEKB and SuperKEKB colliders. Data taken from [4], [5], [6] and [2]

will lead to an increase of luminosity dependent background, which will be discussed in Section 2.6.

As the Lorentz boost has been significantly decreased from $\beta\,\gamma\,=\,0.42$ in KEKB to $\beta\,\gamma\,=\,0.28$ SuperKEKB, the resolution of the decay vertex reconstruction had to be improved to give the same physics performance as at KEKB. Reducing the Lorentz boost was necessary to reduce an important source of background, namely the Touschek effect, which will be described in Section 2.6. The vertex resolution was improved by better detector resolution and by better track reconstruction algorithms.

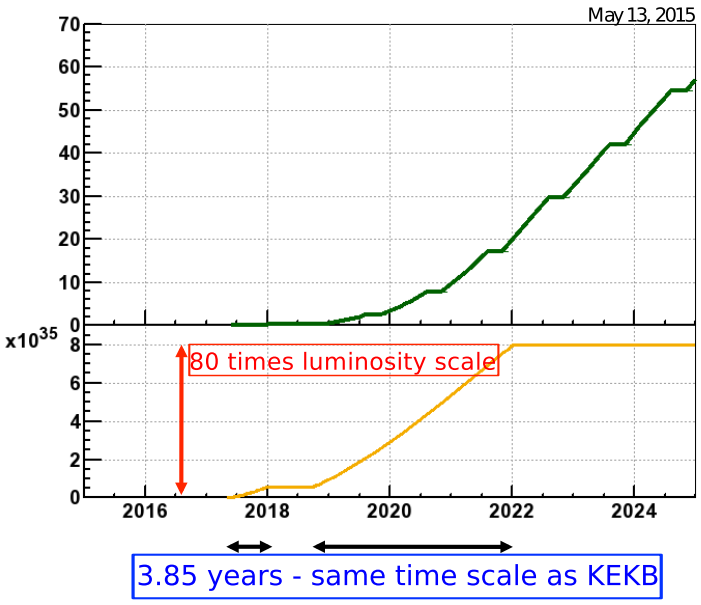


Figure 2.5: The planned amount of events recorded over the years. Plot courtesy of the Belle II Collaboration.

2.4 Belle II detector

The Belle II detector is an upgrade of Belle and re-uses some of the original parts like the magnet. A schematic picture showing the various sub-detectors of the Belle II detector can be found in Figure 2.6. The increased luminosity and the decreased Lorentz boost due

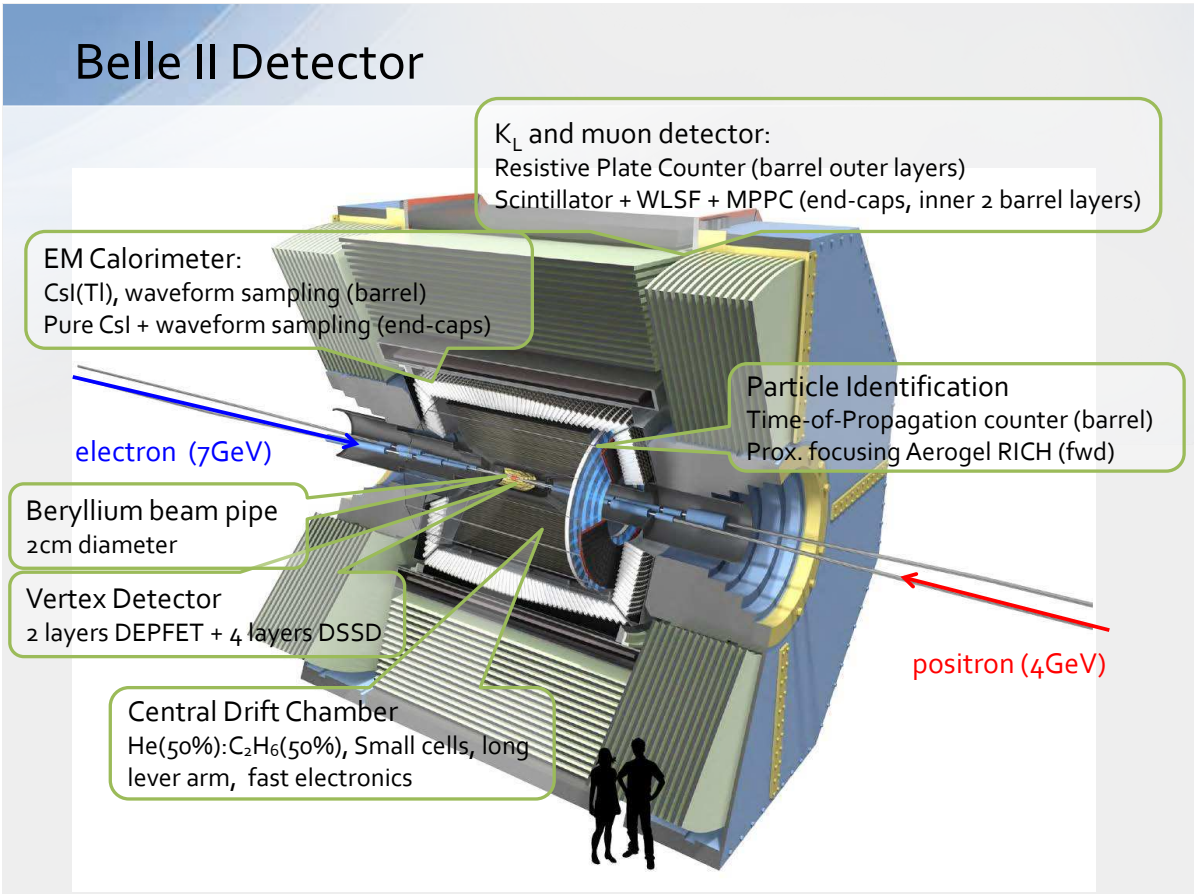


Figure 2.6: An overview of the subdetectors of Belle II, which itself has a height of 7.1 m and a length of 7.4 m, rendering courtesy of the Belle II Collaboration.

to design changes in SuperKEKB require a faster readout and a higher detector resolution. Additionally the background will increase by an estimated factor of 20. Therefore the inner tracking detectors had to be replaced completely.

The Belle II detector consists of several sub-detectors, which are described briefly in the following list:

- The new beam pipe is a double walled tube made of Beryllium. with an inner radius of 10 mm.

- The old SVD consisting of four layers DSSD[42] with shared readout, was replaced by the VXD now consisting of two sub-detectors:
  - The PXD is a pixel detector using DEPFET technology with two layers at 13 and 22 mm radius.
  - The SVD is a double-sided strip detector (DSSD) with four layers at 39, 80, 105 and 135 mm radius. The strip-wise readout allows stand-alone track reconstruction.

  More details about this sub-detector can be found in Section 2.4.1.
- The old Central drift chamber was replaced by a new one (CDC[43]) covering radii from 160 to 1120 mm. There are 56 layers in total, clustered into 9 super-layers, with smaller drift cells in the innermost super-layer. Every second super-layer consists of stereo-layers with titled wires and readout on both sides, allowing a rough z-measurement of the track position.
- Enveloping the CDC in the barrel of the detector starting at the radius of 1200 mm is the TOP, a Ring Imaging Cherenkov detector (RICH) with quartz radiator having a thickness of 20 mm.
- An aerogel RICH (ARICH) radiator with 40 mm thickness covers the endcap in forward direction for additional speed measurements.
- The Electronic CaLorimeter (ECL) envelops the barrel starting at a radius of 1250 mm to about 1620 mm and the endcap.
- The K_L and muon chambers (KLM) have got 14 layers with each 50 mm Fe + 40 mm gap where 2 resistive plate chambers are situated in each gap. The endcap is covered with 14 layers of scintillator strips.

Much more details about the Belle II detector design can be found in [4], while the VXD will be described in closer detail in the next section. Belle has been thoroughly described in [2], whereas also a comparison with the BaBar detector can be found in [5] and [15].

2.4.1 The VXD Detector

The vertex detector (VXD) of the Belle II experiment (Figure 2.7) consists of two sub-detectors using $\sim 1.2\,\mathrm{m}^2$ of silicon sensors arranged in 6 layers. The inner one is called Pixel Detector (PXD) and contains layer 1 and 2 using DEPFET-sensors with a total of about 8 million 2D-pixels on 40 sensors [16]. The outer one is called Silicon Vertex

[42] Double Sided silicon Strip Detector
[43] CDC: Central Drift Chamber

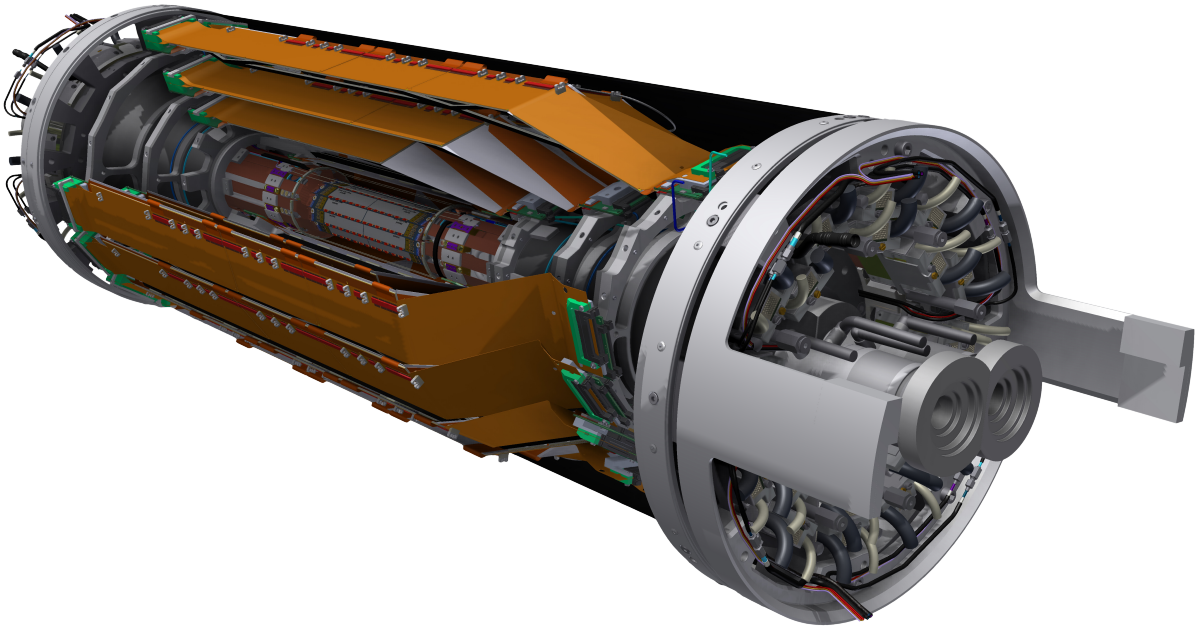


Figure 2.7: 3D rendering of the VXD with some ladders removed for better look inside. Rendering courtesy of the Belle II Collaboration.

Detector (SVD) and contains layer 3, 4, 5 and layer 6 all using double sided strip-sensors (DSSD) with 172 sensors in total [17].

The overall structure of the VXD follows a typical windmill design with a specialty in the forward region: There the standard rectangular sensors of layer 4 to 6 have been replaced by slanted sensors of trapezoidal shape, which reduces the amount of sensors needed to cover the aspired θ range of $17° - 150°$ and allows the reconstruction of particles with a low p_T but high p_z in forward direction. A comparable approach has also been used in the BaBar experiment, where the forward and the backward region was implemented using the slanted approach. In Belle II, however, for the backward region that approach was neglected since the SuperKEKB-accelerator favors the forward region through its asymmetric setup for the beam energies, as described in Section 2.3.2. When defining a coordinate system in $r - \phi - \theta$, the VXD covers mean layer radii between 14 and 135 mm, ϕ in 0°–360° and θ in 17°–150°. In Figure 2.8 the layer and sensor numbering scheme is illustrated, with slanted sensors in the forward region. Additionally one can see the focus on the forward region, which is coming from the energy difference of the HER and the LER particles. Figure 2.9 contains the ladder numbering and shows the windmill structure of the VXD More details about important parameters for the VXD layers can be found in Table 2.2 and Table 2.3, which were taken from [4] and updated using [18], [19] and basf2[46] geometry files. The slanted sensors of the SVD are slightly thinner than

[46] The Belle Analysis Software Framework 2, more details in [20]

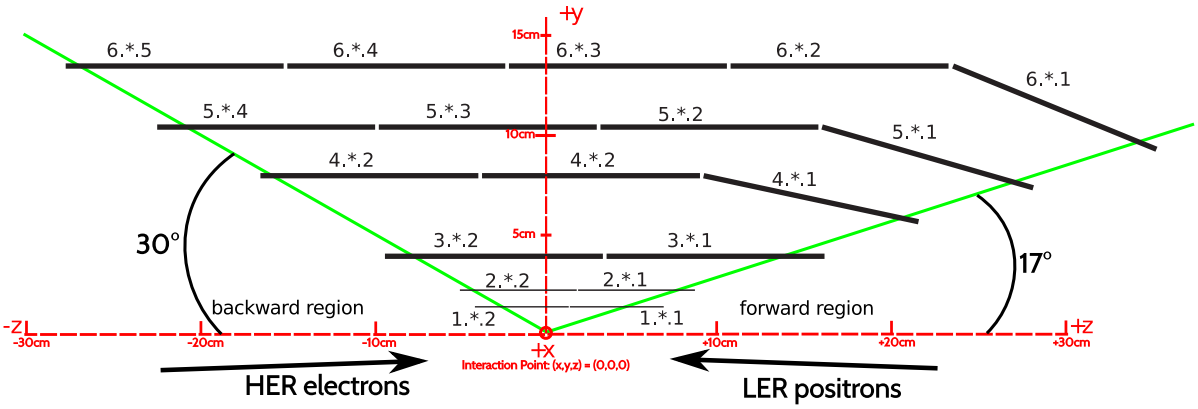


Figure 2.8: The layer and sensor numbering scheme (layerID.ladderID.sensorID) including the rough position of the sensors on the yz plane. Layer 1 and 2 are PXD layers and layers 3 to 6 are SVD layers. The HER[44] beam has 7 GeV electrons and the LER[45] has 4 GeV positrons.

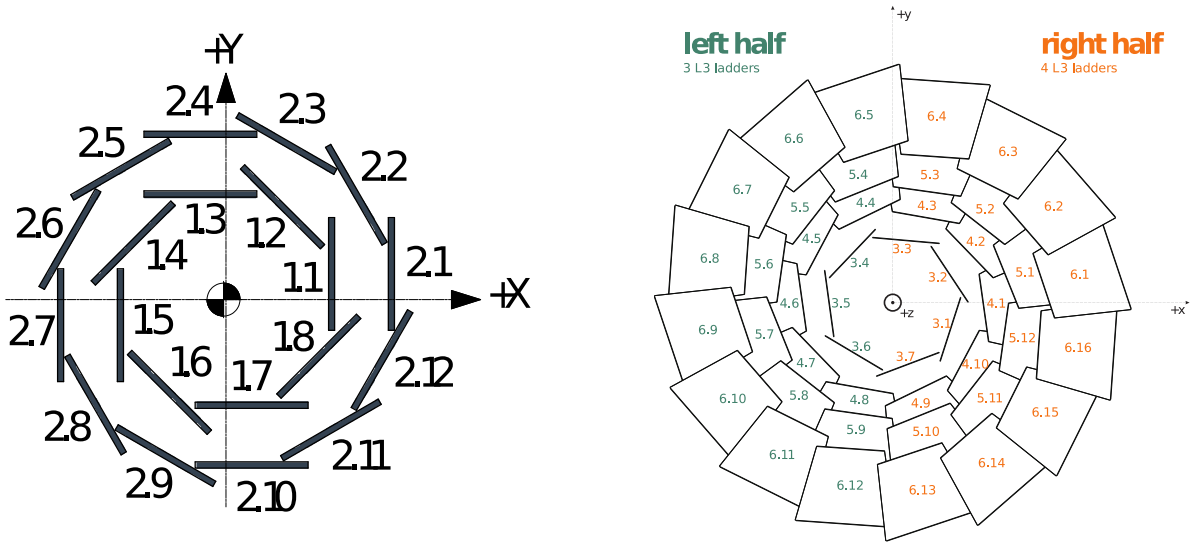


Figure 2.9: The ladder numbering scheme of the PXD (left) and the SVD (right) sketched on the xy plane. Figure courtesy of the Belle II Collaboration.

the normal SVD sensors.

Because of the windmill design the sensors overlap. The amount of overlap is calculated according to the following definition:

> Assuming straight tracks, the overlap of a layer is the fraction of tracks hitting two sensors in this layer.

The overlaps of the VXD layers can be found in Table 2.3. A nonzero overlap means

Table 2.2: Table of specifications for the PXD (L1 & L2) and the SVD (L3 – L6) layers. Layer numbers marked with extra suffixes represent only certain parts of that layer: 'i' means the inner region of the PXD sensors only, 'o' the outer regions of the PXD sensors, while 's' represent slanted sensors of that SVD layer only.

	$n_{\text{ladder}} \times n_{\text{sensor}}$	radius [mm]	sensor size u×v [mm]	thickness [µm]	pixels/strips u×v	pitch u×v [µm]
L1	8×2	14	12.5×44.8	75	250×768	50×55-60
L1i	8×2	14	12.5×14.1	75	250×256	50×55
L1o	8×2	14	12.5×30.7	75	250×512	50×60
L2	12×2	22	12.5×61.44	75	250×768	50×70-85
L2i	12×2	22	12.5×17.9	75	250×256	50×70
L2o	12×2	22	12.5×43.5	75	250×512	50×85
L3	7×2	39	38.5×120	320	768×768	50×160
L4	10×2	80	57.7×122.9	320	768×512	75×240
L4s	10×1	80	38.4-57.6×122.8	300	768×512	50-75×240
L5	12×3	105	57.7×122.9	320	768×512	75×240
L5s	12×1	105	38.4-57.6×122.9	300	768×512	50-75×240
L6	16×4	135	57.7×122.9	320	768×512	75×240
L6s	16×1	135	38.4-57.6×122.9	300	768×512	50-75×240

Table 2.3: Table of more specifications for the PXD (L1 & L2) and the SVD (L3 – L6) layers. The sensor overlap is using the tracking definition of the overlap, which is defined in this section. The windmill angle is defined in [21].

	ladders per layer	radius [mm]	δ: angle between ladders [°]	sensor overlap %	α: windmill angle [°]
L1	8	14	45°	2.15 %	13.94°
L2	12	22	30°	3.42 %	9.08°
L3	7	39	51.43°	0.92 %	7°
L4	10	80	36°	8.56 %	9°
L5	12	105	30°	2.34 %	7°
L6	16	135	22.5°	5.27 %	9°

that one can have more than one hit in a layer, which can be used for alignment and in tracking as well. This allows to increase the number of reconstructed hits per track for such cases.

Since the VXD is designed to detect low momentum particles down to $p_\top \approx 50\,\mathrm{MeV}/c$, the material budget is an important aspect. Figure 2.10 shows the accumulated material budget of the VXD detector including the beampipe. Material budget is typically defined in radiation lengths X_0[47], which is depending on the particle type. For electrons/positrons it is the mean distance when about 63% of is energy is lost by bremsstrahlung. For photons it is $^7/_9$ of the mean free path for a pair production. X_0 is dependent on density and type of the material. In Figure 2.10 one can see the increasing material budget for the forward and backward region, since the sensors are positioned parallel to the beam axis and therefore particles with higher p_z values do "see" thicker sensors than those where $p_z \approx 0$. In the forward region at the range $40° < \theta < 60° \mathrel{\widehat{=}} 0.766 > \cos\theta > 0.5$ there are bumps in the PXD resulting in higher material budget, which are coming from the joint region of the sensors, where the sensors themselves have a thicker support structure and additional ceramic inserts stabilizing that region. This bump is mirrored on the curve of the SVD, which itself has no relevant bumps adding on that spot. In the forward region of the SVD at $\theta \approx 30° \mathrel{\widehat{=}} \cos\theta \approx 0.86$ there is another bump, which comes from the transition from the standard windmill design to the slanted sensors. This effectively reduces the material budget, so that the SVD has the same thickness in X_0 for the end of the backward region with $\theta = 150° \mathrel{\widehat{=}} \cos\theta = -0.866$ as for the front tip of the forward region with $\theta = 17° \mathrel{\widehat{=}} \cos\theta = 0.956$.

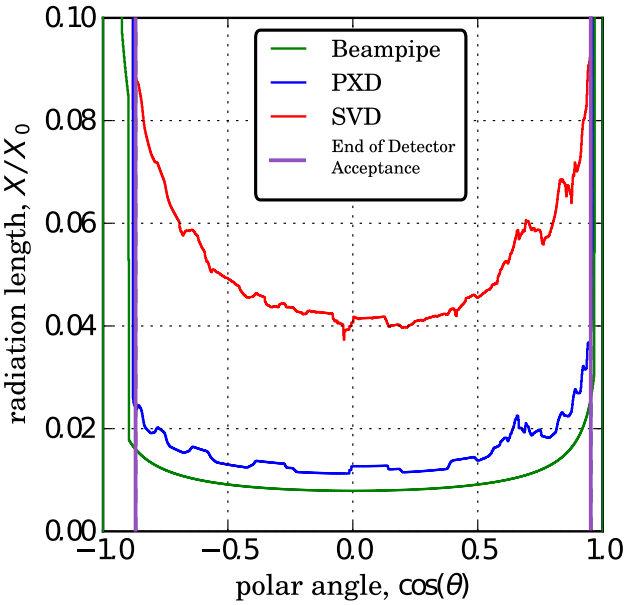


Figure 2.10: The cumulative material budget of the Beampipe, the PXD and the SVD. The violet vertical lines mark the end of the acceptance region. Plot courtesy of Martin Ritter and the Belle II Collaboration.

Particles with direction of flight of $\theta \approx 90° \mathrel{\widehat{=}} \cos\theta = 0$ face a beampipe with a material budget of 0.8% X_0, the PXD then adds 0.48% X_0. The SVD with its four layers then adds about 2.9% of X_0. The sensors of the PXD themselves add only 0.32% X_0, while the SVD sensors add 1.5% X_0 to the material. The higher numbers of the material budget for the full PXD and SVD comes from the readout, the mechanical support and the cooling

[47] X_0: Radiation length

structures. The mean value of the material budget when covering the whole acceptance range in θ is considerably higher. For the beampipe the mean material budget over θ is 1.1% X_0, for the PXD it is 0.6% X_0 and for the SVD it is about 4% X_0. The following section shows why a small material budget is important for low-momentum tracking.

The local coordinate system used for the sensors are the u and v coordinates on the sensor plane. For the SVD this physically corresponds to the n- and p-side of the sensors.

- $\boldsymbol{u} \mathrel{\widehat{=}}$ local measurement for the $r-\phi$-value (p-side on a SVD sensor)
- $\boldsymbol{v} \mathrel{\widehat{=}}$ local measurement for the z-value (n-side on a SVD sensor)

In the following chapters hits in the PXD are basically 2D-clusters on the sensor, measuring the local u and v position. In the SVD the approach is comparable, but there a hit is a combination of 2 1D-clusters, one measuring u and one measuring v. The VXDTF design was implemented mostly independent of the detector type. This requires a unification of hits, where the track finder (TF) does expect to get space points having global coordinates and an error matrix. The unified hit class for the VXD are called SpacePoints. More details about how these hits are used can be found in the following chapters. More details about the geometry of the VXD detector can be found in [1], [4], [18] and [21].

2.5 Passage of particles through matter

When particles traverse any material they interact with the atomic nuclei and the electrons in that material. This leads to a change of the direction of flight or to energy loss, while for inelastic interactions both happen at the same time. This section presents an overview of the dominant interaction types at the relevant particle energies, and their consequences for track finding. More details about the passage of particles through matter can be found for example in the chapter with the same name as this section in [22].

2.5.1 Ionization and excitation

Energy loss via ionization and excitation is the dominant form of energy loss of particles to be tracked in Belle II, with the exception of electrons and positrons, which mostly lose their energy via bremsstrahlung, which will be described in Section 2.5.2. On a macroscopic scale the energy loss via ionization and excitation is well described by the Bethe-Bloch formula (i.e. described in [22]). The curve is plotted in Figure 2.11.

The results of this formula are valid for the range $0.1 \lesssim \beta\gamma \lesssim 1000$. The minimum energy loss occurs at $\beta\gamma = \frac{p}{mc} \approx 3$. For pions and muons the minimum is around $350\,\mathrm{MeV}/c$, so that below this point the energy loss increases with decreasing momentum. This is a difficult situation for tracking, since energy loss changes the curvature of the track, which makes it more complicated to find hits belonging to the track. Moreover,

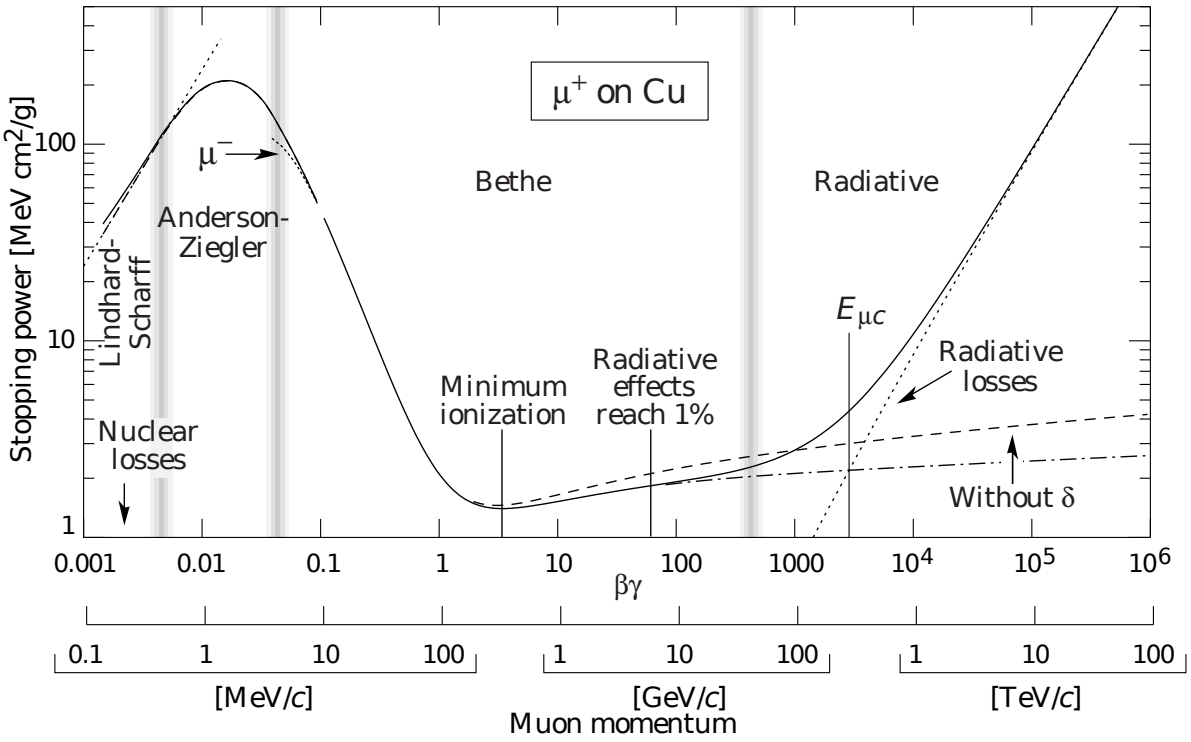
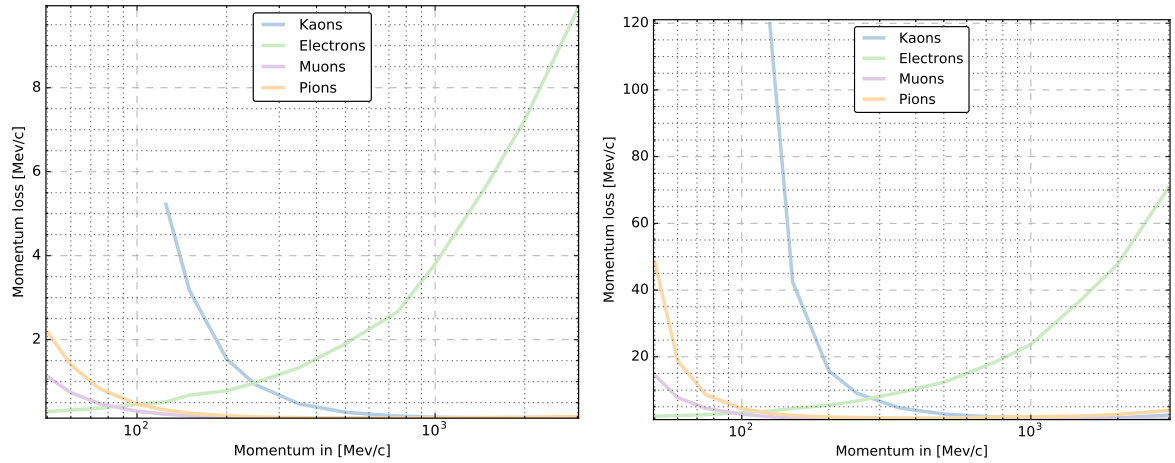


Figure 2.11: Energy loss in copper for anti-muons over a wide momentum range. Plot taken from [22].

the actual distribution of the energy loss is a Landau distribution, which means that Gaussian approximations are not accurate, because of the long right tail of the Landau distribution.

To illustrate the impact of the momentum changing material effects, a MC study was performed. 20,000 particles with different momenta, $\theta = 90°$, $\phi \in [0°, 360°]$ and zero magnetic field were followed through the geometry of the SVD, and their momentum loss was recorded. Figure 2.12 shows the momentum dependency of the momentum loss for electrons, muons, pions and kaons, where the momentum loss for the latter three is dominated by ionization and excitation, while electrons lose their momentum mostly by bremsstrahlung as discussed in Section 2.5.2. Although being too optimistic due to the fact that only particles were considered actually passing through the whole detector at$\theta = 90°$, where the material budget is lowest (see Figure 2.10), the results make clear that energy loss is an important factor in low-momentum tracking. At 50 MeV/c only about 10% of the pions even pass through the SVD, although no PXD and no Beampipe was present. Charged kaons do not pass the detector for momenta below 125 MeV/c at all.

However, the version of the formula found in [22] gives the *energy* loss $\mathrm{d}E/\mathrm{d}x$ of the particle per unit length, while for tracking purposes the *integrated* momentum loss



(a) Momentum loss when passing a single SVD sensor of layer 3.

(b) Momentum loss when passing the whole SVD detector.

Figure 2.12: The momentum loss for electrons (green), muons (violet), charged pions (orange) and charged kaons (blue). The momentum loss for electrons is dominated by bremsstrahlung (Section 2.5.2) and for the others it is dominated by the Bethe–Bloch formula found in Eq. (2.1). The curve is measured using a MC study, where only particles passing through the whole detector were considered, which is why this plot is too optimistic for lower momenta.

$\Delta p(d)$ for a particle traversing a material with thickness d is much more convenient. The Bethe-Bloch formula [23] can be modified accordingly:

$$\Delta p(d) = \int_0^d = \frac{\mathrm{d}p}{\mathrm{d}x}\mathrm{d}x = \int_0^d \frac{1}{\beta}\frac{\mathrm{d}E}{\mathrm{d}x}\mathrm{d}x = \int_0^d \frac{K}{\beta^3}\left(\ln\left(\frac{2m_\mathrm{e}c^2\beta^2\gamma^2}{I}\right) - \beta^2\right)\mathrm{d}x, \qquad (2.1)$$

In this formula β is the speed of the particle relatively to the speed of light c, K is a material constant to be found experimentally, m_e is the mass of the electron, γ the Lorentz factor, while I is the average ionization potential, which is another material dependent value. For thin layers, the integral can be replaced by a linear approximation: $\int_0^d \ldots \mathrm{d}x \longrightarrow \Delta x$, where Δx is the distance between entry and exit point of the particle through that layer.

2.5.2 Bremsstrahlung

Bremsstrahlung is the dominating contribution to energy loss for electrons and positrons. It occurs when the particles interact with the electric field of an atomic nucleus and results in the emission of a photon. This can also be seen in Figure 2.12, where electrons do not show the typical logarithmic rise of energy deposit for higher momenta coming from Eq. (2.1), but a much faster rise wich is due to bremsstrahlung. The strength of bremsstrahlung is $\propto \frac{1}{m_p^2}$ where m_p is the mass of the particle. Therefore electrons are

experiencing this effect about 200^2 times stronger than muons with the same momentum, since muons are 200 times heavier than electrons. The full equation for bremsstrahlung can be found in [22] and its deployment on track fitting in [23], but for here it is sufficient to know that for electrons and positrons in the relevant energy range the energy loss by bremsstrahlung can be simplified to:

$$-\frac{\mathrm{d}E}{\mathrm{d}x} \approx \frac{E}{X_0} \tag{2.2}$$

Bremsstrahlung also occurs as *synchrotron radiation*, which is a relevant background source for Belle II and is described in more detail in Section 2.6.1.

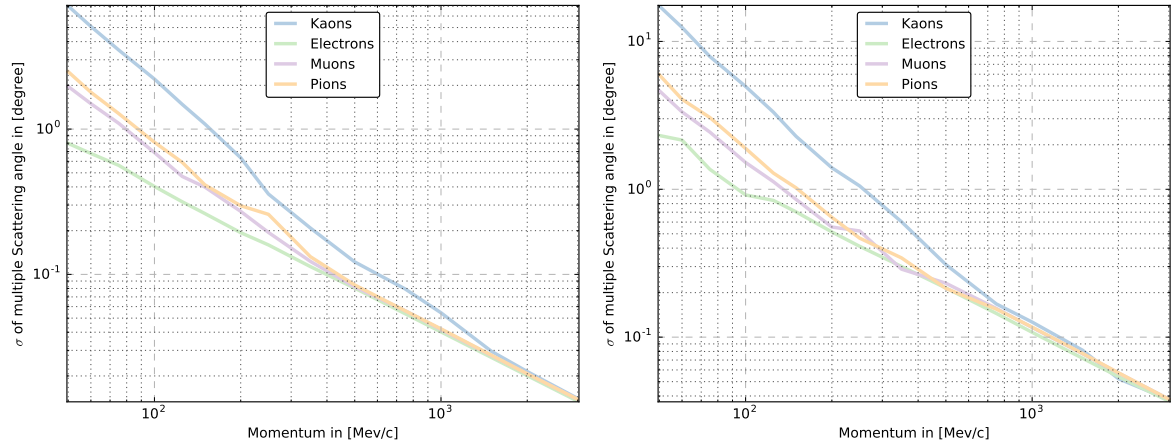
2.5.3 Multiple scattering

Even if no energy is lost, the interactions with matter change the direction of the particle. This change is a random process, which can be characterized by the standard deviation of the scattering angle θ. It is inversely proportional to the particle momentum:

$$\sigma_\theta = \frac{13.6\,\mathrm{MeV}}{\beta cp} q\sqrt{x/X_0}\left[1 + 0.038\ln\left(x/X_0\right)\right] \tag{2.3}$$

The Highland formula (2.3) was taken from [22] and is valid for relativistic particles (except electrons/positrons, which are dominated by bremsstrahlung) in all materials relevant for our purposes. In this formula β is again the speed of the particle relatively to the speed of light c, p the momentum in MeV/c, q the charge number of the particle ($q = 1$ for positively charged particles, $q = -1$ for negative ones) and x/X_0 is the thickness of the medium in radiation lengths X_0. In Figure 2.13 the effect of multiple scattering versus the momentum of some relevant particles has been plotted. The strong impact of low momentum on the standard deviation of the scattering angle is coming from the denominator of Eq. (2.3), where small β and p both increase the value of the result. Therefore 50 MeV/c pions have a standard deviation (σ_θ) from their original trajectory by more than 2 degrees when passing a single layer (Figure 2.13a), while 1 GeV pions have a σ_θ of only 0.04 degrees, which is more than 50 times smaller. In Figure 2.13b one can see the impact of extrapolating the same tracks through the whole SVD, which results in a σ_θ of $4.5° - 6°$ for 50 MeV/c muons and pions. For muons and pions above 130 MeV/c the standard deviation σ_θ is smaller than one degree; kaons reach this mark at about 250 MeV/c. While barely having an impact for higher momenta, multiple scattering can not be neglected for low-momentum tracking and has therefore to be considered by the tracking algorithms.

The Highland formula Eq. (2.3) uses a Gaussian approximation to the distribution of the scattering angle and is not valid for very thin layers. In the case of the very thin PXD sensors, Eq. (2.3) therefore does not properly take into account the tails of the distribution.



(a) σ_θ when passing through a single layer. (b) σ_θ when passing through the whole SVD.

Figure 2.13: The standard deviation of the residual ($|pos_{unscattered} - pos_{scattered}|$) plotted for electrons (green), muons (violet), charged pions (orange) and charged kaons (blue) without magnetic field through layer 3 with a thickness of 320 µm (Figure 2.13a), and for extrapolating a particle through the whole SVD (Figure 2.13b). The multiple scattering plotted is described by the Highland formula in Eq. (2.3). Values were found via a MC study of 25,000 tracks per particle type, all started with $\theta = 90°$ and over the full ϕ range.

2.6 Background sources

This section gives a short overview of the background sources relevant for the VXD detector, while Chapter 3 will discuss how much background the SVD registers in form of hits, occupancy and other points of view.

There are multiple sources of background creating hits in the VXD detector, which can be categorized by their production source. Their exact behavior is not trivial to understand and those who like to read about background sources for the VXD in more detail, will find a thorough discussion on that topic in [18], which also served as one of the sources for this section. All background sources listed here effectively reduce the amount of particles placed into the bunches of the collider beams. This leads to a shorter beam lifetime, which is the amount of time it takes to lose so many particles of a beam that it cannot effectively be used anymore for data taking. The beams of SuperKEKB are constantly refilled with newly accelerated e^+e^- particles, to keep the beams circulating with sufficiently high numbers of particles.

The background sources can be categorized into two different groups: the beam-induced and the luminosity dependent ones. Details about their characteristics are described in the following subsections.

2.6.1 Beam-induced background

Beam-induced background types are created by the beam interacting with itself (Touschek effect), with residual gas in the beam tube (beam-gas scattering), or by being bent by magnetic fields (synchrotron radiation). Therefore these background types are created by the electron (HER) and positron (LER) beam independently from each other. This means that these background types, for the case of interacting with the VXD, create particle tracks not coming from the IP/origin, and are therefore generally easier to deal with than luminosity dependent background, if compared on the same dosage level. Nonetheless they create a considerable amount of background hits and have a negative impact on the beam lifetime. This results in a considerably shorter lifetime than in the first generation B factories. Beam-induced Background sources being are described in more detail in alphabetical order in the following list :

- **Beam-gas scattering:** The beam pipe can never be completely evacuated, therefore residual gas molecules remain in the beam pipe, leading to interactions with the beam. Scattering of the beam at the gas molecules leads to energy loss through inelastic bremsstrahlung and a change of the direction of flight via elastic Coulomb scattering. This results in additional particle showers created by the deviated particle when they hit the beam pipe or other material such as magnets and collimators. The primary particles therefore are e^+e^- kicked out of the HER and LER with high energies and those reaching the VXD are typically created within a few meters from the IP. The secondaries are e^+e^- too and mostly are created in the material near the detector or in the detector itself.

- **Synchrotron radiation:** e^+e^- of the LER and HER beams are transversely accelerated by the bending and focusing magnets of the accelerator, which leads to the emission of synchrotron radiation (photons). Due to the beam correction magnets and the QCS[48], which located about 1 m off the IP, the emitted photons can reach the VXD. The energy loss via synchrotron radiation scales with *beam current* $\times$ *B-field*^{2} $\times$ *particle energy*^{2}, and therefore the HER is expected to dominate this background type. For the HER the emitted photons can have up to 200 keV, while LER photons can reach up to 70 keV. This is not enough for e^+e^- pair production, but the photons can still create hits in the silicon sensors.

- **Touschek effect:** Particle bunches of the beams of SuperKEKB are repeatedly squeezed horizontally, vertically and in direction of flight to achieve a high particle density for all the bunches, which results in the situation that the particles of the bunches oscillate perpendicular to the direction of the beam. This leads to intra-bunch interactions, where particles of the same bunch collide with each other and frequently kick each other out of the bunch, resulting in particle showers after hitting the beam pipe or other materials. The higher the mean particle energy of a

[48] Final beam focussing magnets before the IP.

bunch, the lower is the probability of intra-bunch interactions. Therefore bunches from the LER are the main source for background particles created by the Touschek effect. This is one of the reasons why the asymmetric beam energies fof the HER and the LER, as described in Table 2.1, differ less in SuperKEKB than in KEKB and PEP-II, as the new collider is designed to achieve higher luminosities and therefore squeezes more particles per bunch in a smaller space per bunch compared to its predecessors. The Touschek effect is counted as beam-induced background despite the fact that it is also luminosity dependent, since the beams experience this effect without needing to interact with each other.

2.6.2 Luminosity dependent background

Luminosity dependent background types are created at the IP by "unwanted" physical effects leading to particle tracks not relevant for the actual experimental goals. From the point of view of tracking, these particles produce tracks in the tracking detectors that are coming from the IP and therefore are geometrically indistinguishable from "interesting" tracks. Event triggers, energy deposit and hit time can be used to reduce the effects of these background sources, but still they can not be completely filtered out.

- **Radiative Bhabha scattering:** (RBB[49]) happens when particles from both beams interact, but do not create new particles: $e^+e^- \longrightarrow e^+e^-$. This is the most often produced background type and in the case of SuperKEKB occurs about 75 times more often than relevant physics processes like $\Upsilon(4S)$ events. Fortunately for tracking, the scattering angle is very small, and therefore most particles that have been scattered leave the beam pipe far from the VXD. The rate of RBB depends only on the luminosity and can therefore be used as a benchmark for measuring the luminosity of the B factory. It can easily be filtered by the event triggers, as only two particle tracks enter the detector, but the sheer amount of events is a problem for the PXD, which is hit many times within a ROF[50] and therefore collects a high number of background hits. The particles kicked out of the beam by this effect can also create showers in the beam pipe and additionally create a bremsstrahlung photon. This photon can interact with matter and for those being in the range of $8\,\mathrm{MeV} < E_\gamma < 30\,\mathrm{MeV}$ have a high cross section to excite the nuclei of atoms, which often leads to a neutron being kicked out of the nucleus in the material hit. Therefore this is the major source for free neutrons interacting with the detector. They slowly saturate the depleted silicon sensors over the years of operation and therefore reduce the signal detectable by the sensors of the VXD.

- **Two-photon process:** Often called "QED[51] background", since its production is mediated by the electroweak force. Although the production rate is only 50% of

[49] Radiative Bhabha scattering
[50] ReadOut Frame
[51] Quantum Electro Dynamics

the RBB, particles created by the QED background hit the PXD more frequently than RBB background. The two-photon process creates in most cases four low-momentum electrons and positrons: $e^+e^- \longrightarrow e^+e^-\gamma\gamma \longrightarrow e^+e^-e^+e^-$. They are responsible for most of the background hits seen in the PXD and the innermost SVD layer. Since the particles created have low momentum they rarely leave the VXD to reach the CDC and therefore only the inner layers suffer from this background type.

2.7 Event Reconstruction

The purpose of event reconstruction in high-energy physics is described very well in [24] and, more concisely, in [23] and [25]. Therefore only a brief overview is given here.

High-energy physics experiments need to record huge amounts of events to be able to probe theories and hypotheses. These events are barely ever free of background, so the recorded data needs to be filtered in order to find interesting events and to discard uninteresting ones. To be able to do this, hits must be combined to tracks and correctly identified, which is the task of event reconstruction.

There are several steps in this process, which are summarized in the following list, adapted from [1].

1. Start by reducing the amount of raw data to processable levels by suppressing events consisting purely of background effects and enhancing the rate of desired physics events. This step is called *triggering*.

2. Distinguish between real data and background hits within the same event and bundle the hits to track candidates. This is done in the *track finding* step.

3. Estimate the geometric and kinematic parameters of the particle tracks found, and discard bad track candidates. This is the *track fitting* step.

4. Determine the types of the particles creating the tracks. This is the *particle identification* step, sometimes performed in parallel with the track fitting step.

5. Find the production points (vertices) of one or several particles and estimate their location. This is the *vertex finding and fitting* step.

Although all of the listed steps are essential for a successful event reconstruction, this thesis focuses on the track finding step, where TFs are the main tools to be developed.

2.7.1 Track Finding

The task of track finding is to assign detector information to particle tracks. The information consists of position measurements in the tracking detectors and energy

deposits in the calorimeters. TFs typically use only tracking detectors for their task. In the case of Belle II these are the PXD, the SVD and the CDC, while the information of the calorimeters is added in the particle identification step.

There exists a wide variety of algorithms, which are specialized on finding tracks and rejecting background hits. Which one is the best, strongly depends on the goals to achieve and the boundary conditions to meet. Therefore one can not build a single best suited algorithm for every task.

In the context of Belle II low-momentum tracking, where material effects and considerable amounts of background hits a selection of existing algorithms including their strengths and weaknesses will be discussed in the following sections.

2.7.1.1 Conformal Mapping

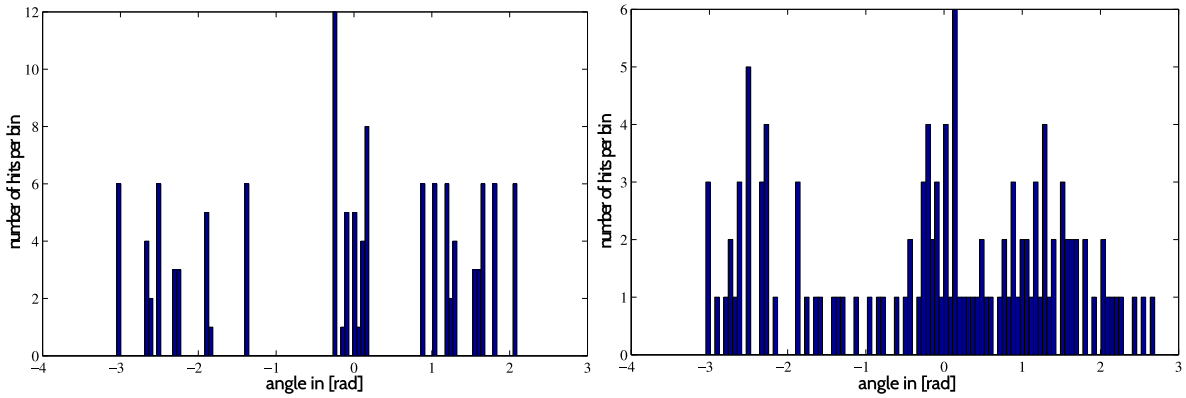
Conformal mapping is frequently done in the preprocessing stage of a TF, but can also be applied stand-alone. It transforms circles through the origin in a plane to lines in the plane. It can be applied if the magnetic field is homogeneous and parallel to the beams, so that in the projection the hits of a track lie close to a circle. Assume that the detector hits are given in Cartesian coordinates $(x_i, y_i), i = 1, \ldots, N$, the transformation is given by:

$$\begin{aligned} u_i &= \frac{x_i}{x_i^2 + y_i^2} \\ v_i &= \frac{y_i}{x_i^2 + y_i^2} \end{aligned} \tag{2.4}$$

It can be shown that hits lying on a common circle are transformed to points on a straight line. The distance of the line from the origin depends on the radius of the circle. If the track multiplicity is sufficiently small, the azimuthal angle ϕ of the transformed hits can be filled into a histogram in ϕ. In the case of six layers and therefore six hits per tracks, one then gets bins with ideally six entries, indicating that hits belonging to that bin are coming from the same circle and therefore from the same track. The algorithm is very fast, since the processing time increases linearly with the number of hits. It ignores, however, the measurement of the z positions of the hits, so that an application to high-multiplicity events is problematic.

In order to see how conformal mapping performs in the VXD of Belle II, a small MC study has been performed, using a simplified setup with six cylindrical layers without gaps and overlaps. In Belle II the IP is defined to be at the origin of the coordinate system, and the magnetic field forces charged particles into a helicoidal trajectory, which looks like a circle if projected to on the x-$y \mathrel{\widehat{=}} r$-$\phi$ plane. So the basic conditions are actually met. When running 20 tracks over 6 layers with about 1 GeV/c each, this results in 120 hits to be transformed, if no background is added. Figure 2.14a shows the results of a typical event after transforming the hits and histogramming them in ϕ, where by simple clustering of bins one can identify 16 of 20 tracks, meaning 80% of the tracks can

be fully recovered. But in the case of VXD tracking in Belle II one needs to find tracks



(a) Histogram over hits of 20 tracks with 1 GeV/c each after conformal mapping.

(b) Histogram over hits of 20 tracks with 80 MeV/c each after conformal mapping.

Figure 2.14: Plots taken from [1] showing histograms over ϕ for 20 muon tracks on a 6 layer configuration without overlaps and therefore 6 hits per track.

down to 50 MeV/c, where material effects play an important role, as already discussed in Section 2.5. The result of histogramming an event of 20 tracks with 80 MeV/c is shown in Figure 2.14b, where due to the material effects no tracks can be fully recovered. The deviations from a perfect circle due to multiple scattering and energy loss therefore makes this algorithm unsuitable for low-momentum tracking.

Moreover, in Belle II a significant amount of tracks do not come from the origin and therefore cannot be found by this approach. Also, the fact that only two of three dimensions are used neglects essential information especially relevant in the case of many background hits, which is again the case for Belle II. More details about the amount of background and the origin of tracks to be reconstructed can be found in Chapter 3.

2.7.1.2 Hough Transform

The Hough transform is another commonly used algorithm. In Belle II it is also used for ROI finding using SVD information. More details can be found in [26]. Compared to conformal mapping, which solely works with circles passing through the origin and can only use 2D, the Hough transform can work with any analytical function and is not restricted to tracks passing through the origin. To perform the Hough transform one needs an analytical function describing the track model, on base of which one intends to find tracks. In the case of a straight line this works as follows for a given hit $h_i = (x_i, y_i)$:

$$y_i = kx_i + d \tag{2.5}$$

Eq. (2.5) describes a straight line in the x-y plane, where (x_i, y_i) are the hit coordinates. This coordinate system is called the *image space*. This equation then is transformed into

another straight line in the *parameter space* of the parameters k and d, in the k-d plane, where (k_j, d_j) are the coordinates. The formula 2.5 can not be solved if only one hit is used, since one equation with two unknown parameters results in an infinite number of compatible $k - d$ pairs. But by transforming it into the parameter space every hit — being a point in image space — becomes a line in the parameter space, where x and y become the parameters then resulting in a new line equation 2.6:

$$d_j = y_i - k_j x_i \tag{2.6}$$

After transforming all hits to the parameter space, hits being on the same line in the image space intersect as lines on the parameter space at the common k and d values. Therefore one simply needs to perform a 2D histogram on the parameter space to find entries with enough intersects, which then indicate that hits forming these lines come from the same line. This approach works with other track representations too, as long as one can describe the path of the particle by an analytical function. Unfortunately the flexibility comes with the cost of introducing a combinatorial issue: Although the transformation scales linearly with the number of hits, binning the parameter space scales with n_{hits}^c, where c is the number of parameters. For a line one needs two parameters, for a helix representation bound to start form the IP one needs 5 parameters, which would be far too slow for realistic processing times. On the other hand faster implementations of the algorithm, namely the "fast Hough transform", which is also discussed in [26], can considerably speed up the code. Still being too slow for 3 parameters, at least 2 parameters can perform well for amounts of hits occurring in Belle II.

Although being much more flexible than conformal mapping, the combined impact of the major downsides of being two dimensional again — although this time only for the necessity of being fast enough — and the fact that material effects (Section 2.5) do break this approach for low-momentum tracking.

2.7.1.3 Cellular Automaton

The cellular automaton (CA) was introduced for tracking purposes in [27] will be described in close detail in Section 4.4.1. Therefore only a short summary is given here.

To apply a CA, one defines compatible combinations of hits as Cells. Cells carry an integer state, which is initialized to 0. Adjacent Cells are defined as neighbors, if they pass another compatibility criterion. The result is a directed graph, where the Cells are the nodes (vertices) and neighbors are connected by edges. This is followed by an iterative procedure. In each step of the iteration all cells are independently and simultaneously checked whether they have an inner neighbor with the same state. If so, the states of these Cells are simultaneously incremented by one. This check-and-increment procedure is iterated until a steady state is reached. If the graph is a tree, this is guaranteed by the fact that the state of a Cell is equal to the longest chain of Cells connecting it to the innermost nodes of the graph and is thus bounded from above.

This approach can inherently consider material effects and can use 3D information. Its performance depends on the proper definition of the compatibility criteria. These are realized by a set of filters that are trained on various samples of MC tracks. Therefore this approach is much more suitable for low-momentum tracking than the other algorithms presented so far. In its basic form the CA can only deal with simple geometries, since overlaps lead to a variable number of hits per layer of a track. In addition, its performance improves with the number of layers. Nonetheless, its speed and its capability of dealing with low-momentum tracks were the reason to choose this approach for track finding in the VXD. Its disadvantages were overcome by important modifications. The details are described in Chapter 4.

2.7.1.4 Combinatorial Kalman Filter

The CKF[52] is an extension of the basic KF[53] idea [28] for track finding and has been introduced in [29].

The CKF starts from a seed, which is typically a combination of three compatible hits that are used to calculate a momentum estimate. The seed is propagated in the direction defined by the seed, and the n ($n \geq 1$) best next inner hits are collected for extending the seed to a TC. Each of these valid inner hits is added to a clone of the current TC, which is then independently followed on until no more hits are found or a χ^2/QI-threshold is exceeded. In the end, for each seed, the m ($m \geq 1$) best TCs for are retained for further processing.

The CKF takes into account the hit errors and material effects that can be described by a Gaussian approximation. In addition, it provides a quality indicator of the TC. The algorithm behind the KF is an iterative version of the least-squares fit and therefore delivers a best estimate of the TC quality. It is very powerful, but also considerably slower than the methods considered so far. A fast preselection of a sufficiently small number of seeds is therefore mandatory.

2.7.2 Motivation for low-momentum tracking

This section explains why stand-alone tracking in the VXD is relevant to the experimental program of Belle II. Additional details can be found in [2].

A basic motivation for low-momentum tracking is to increase the effective sample size usable for physics analyses. Of the many decay channels which are relevant in this context, the decays $D^{*\pm} \to D^0 + \pi^\pm$ are of particular interest, since they contain a very distinctive slow pion, which can be used as an identifier for these channels. $D^{*\pm}$ are common decay products of $\overline{B}^0$ ($\to D^{*+}$) and B^0 ($\to D^{*-}$). There are decay channels where $D^{*\pm}$ decays are an intermediate state, which are probes for searches beyond standard model effects. As described in [30] with more detail, there are already some deviations detected by BaBar,

[52] Combinatorial Kalman Filter
[53] Kalman Filter

Belle and LHCb, having a combined deviation of 3.9σ from the SMHEP predictions, which could be described by a charged Higgs interaction instead of a W boson, whereas the charged Higgs can not be explained by SMHEP but by supersymmetry or other models predicting charged Higgs bosons.

$$\mathcal{R}(\mathrm{D}^{(*)}) = \frac{\mathcal{B}((\overline{\mathrm{B}} \longrightarrow \mathrm{D}^{(*)}\tau^{-}\nu_{\tau})}{\mathcal{B}((\overline{\mathrm{B}} \longrightarrow \mathrm{D}^{(*)}\ell^{-}\nu_{\ell})} \quad (\ell = \mathrm{e}, \mu) \tag{2.7}$$

Measurements of the branching ratio shown in Eq. (2.7) suffer from background of $\mathrm{B} \to \mathrm{D}^{**}\ell\nu_{\ell}$ decays, which is expected to often decay into more than one slow pion. Therefore it is vital to have low-momentum tracking to be able to identify background by the number of the slow pions. Belle II including low-momentum tracking will therefore be able to keep the background lower while having much more events measured than its predecessors.

The positive effect of low-momentum tracking for slow pions compared to Belle was already shown in [31], where a MC study using the VXDTF for reconstructing the decay channels $\mathrm{D}^{*\pm} \to \mathrm{D}^{0} + \pi^{\pm}$ found an increase in reconstruction efficiency from 40% of the slow pions below $100\,\mathrm{MeV}/c$ in Belle to about 80% when using a Belle II including low-momentum tracking. Roughly a third of all slow pions in that study could only be found by tracking in the SVD.

CHAPTER 3

STUDY OF EVENT CHARACTERISTICS

3.1 Introduction

After discussing the facility in Section 2.3.2, the overall detector in Section 2.4, the VXD tracking environment in Section 2.4.1, the aspects of material effects especially for low momenta in Section 2.5 and the sources of background in Section 2.6, the characteristics of $\Upsilon(4S)$ events are discussed in this section in order to show what the VXDTF has to deal with. For non-physicists this also includes some basic knowledge needed to grasp the situation to be faced.

The general purpose of a TF working in the Belle II environment is to reconstruct charged particle tracks created in $\Upsilon(4S)$-events and other important physics channels. These events are generated by colliding pairs of e^+e^- at the IP, which is situated at the origin of the Belle II coordinate system. The interaction region is rather small and its size is on the µm level. Detailed specifications about the IP can be found in Section 2.3.2 and will not be discussed again here.

$\Upsilon(4S)$ particles are created by the SuperKEKB collider at a rate of about 1,000 Hz, while the total physics rate is expected to be 20–30 kHz, which then includes other relevant decays with $c\bar{c}$ and $d\bar{d}$ mesons and background only events like Bhabha or QED background. Because of the asymmetric beams, they are generated with a momentum boost (called "Lorentz boost", more details in Section 2.3.2) in the forward direction of the detector. They decay almost instantly, their average lifetime being only about 1.2×10^{-20} s [22]. Since particles near the speed of light travel only about 30 cm in a nanosecond ($\hat{=}\ 10^{-9}$ s), the $\Upsilon(4S)$ particles themselves do not travel a significant distance from the interaction region before they decay, and thus do not enter the detector at all. Their most probable decay products, a pair of charged B mesons ($B^+ B^-$) produced in about 48.2% and a pair of neutral B mesons ($B^0 \overline{B}^0$) in about 46.8% of the cases, live for about 1.5×10^{-12} s. Taking the relativistic time dilation due to the Lorentz boost into

account, the result is a typical flight distance of about 100 µm. Although this difference is still not enough to be tracked by the PXD, its value is measurable by vertex finding algorithms used in the RAVE package [32] implemented in GENFIT2 [33], which uses the fit information of tracks produced by TFs such as the one described in this thesis. While the dominant decay modes of the $\Upsilon(4S)$ meson are very characteristic, the decay modes of the various B mesons, while being essential for many physics studies, are not mentioned here to keep this chapter at a basic level.

Below follows a study of what these $\Upsilon(4S)$ events look like and what are the challenges for a TF to deal with. To give a realistic insight into the situation, a simulated sample of 30,000 $\Upsilon(4S)$ events was prepared, which is described in further detail in Section 11.1.

3.2 Study

For the event characteristics it is sufficient to know that only a small number of particles (and their antiparticles) actually reach the detector. Listed in alphabetical order, they are: electrons, kaons (charged and neutral), muons, neutrinos, neutrons, photons, charged pions and protons. Of those only the charged ones and photon conversions can be detected in the silicon tracking detectors, which leaves a small number of relevant particles for tracking purposes itself.

In the SVD the fractions of occurrences of the particles produced by $\Upsilon(4S)$ decays is listed in Table 3.1. It shows the fractions of the particles relevant for physics processes

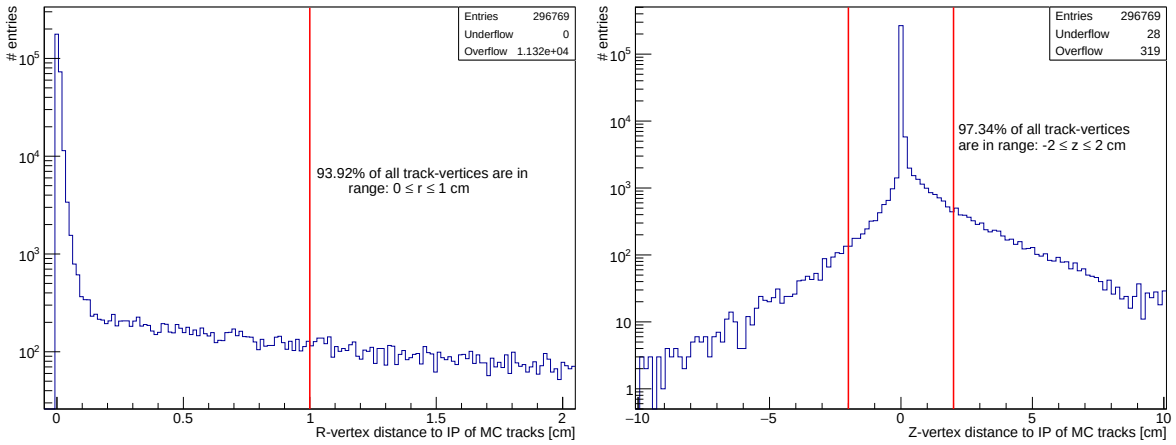
Particle name	electrons	muons	pions	kaons	protons	others
PDG code	—11/11	—13/13	—211/211	—321/321	—2212/2212	—
Frequency [%]	2.61/2.62	2.08/2.10	37.18/37.19	7.16/7.19	0.96/0.93	5×10^{-3}

Table 3.1: Percentage values of the most typical particles and antiparticles leaving hits in the SVD. "Others" are typically rarely occurring particles like Lambdas or Sigmas of the "strange baryon" group.

in the detector, but ignores the particles produced by the various types of background described in Section 2.6, which are mostly electrons, positrons and photons. The list in this table was extracted from the simulation sample described in Section 11.1 and contains only particles leaving enough hits to be able to be found by a TF. As can be seen in the table, about 74% of all relevant particles seen in the SVD are pions, while about 14% are kaons; together with protons ($\approx$ 2%) these hadrons dominate the number of trackable particles by 90%. Leptons in the form of electrons and muons represent only about 9% of the particles coming from $\Upsilon(4S)$ decays.

3.2.1 Particle vertices and d_0

Next to the type of the particles, their origin — the so-called primary or start vertex — is relevant, since many tracking algorithms rely on the fact that the majority of trackable particles is generated at the IP, i.e. at the origin of the coordinate system. Therefore a look at the distributions of the primary vertices of relevant particle tracks is worthwhile. Plots showing their distance from the origin in $r = \sqrt{x^2 + y^2}$ and z can be found in Figure 3.1. First of all, although there is a sharp peak around zero in the radial distance,



(a) r vertex distribution.

(b) z vertex distribution.

Figure 3.1: Distributions of r vertices (Figure 3.1a) and z vertices (Figure 3.1b) of MC tracks with at least 6 clusters in the SVD - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

the total number of tracks leaving at least 6 clusters in the SVD and starting within a distance of 1 cm from the origin is significantly below the 100%-mark. In fact, only 95.46% of all trackable particles originate in the range $-1\,\text{cm} \leq z \leq 1\,\text{cm}$ around the origin, and only 93.92% originate in the range of $r \leq 1\,\text{cm}$. This means that algorithms which require the particles starting at the origin as a strong prerequisite will automatically deliver efficiencies far off the 100%-mark too. The boost of the collider in the forward direction (positive z values) can be seen in Figure 3.1b. Again a significant fraction does not come from the origin, and this has to be taken into account when choosing the best tracking approach.

A different point of view is provided by Figure 3.2, which shows the distribution of the MC tracks as a function of d_0 — one of the helix parameters used to describe a track in Belle II. It describes the *signed* distance of the P.O.C.A.[54] with respect to the z axis.

[54] Point Of Closest Approach

Since the definition is experiment dependent, the version used for Belle II can be found in Eq. (3.1), which is defined in the SI system:

$$
\begin{aligned}
d_0 &= \operatorname{sign}(B_z q)\left(\sqrt{\left(\frac{P_y}{B_z q} + X_x\right)^2 + \left(X_y - \frac{P_x}{B_z q}\right)^2} - \sqrt{\frac{P_x^2 + P_y^2}{B_z^2 q^2}}\right) \\
q &\mathrel{\widehat{=}} \text{ charge of the particle} \\
\vec{B} &= \begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} \mathrel{\widehat{=}} \text{ magnetic field vector} \\
\vec{P} &= \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} \mathrel{\widehat{=}} \text{ particle momentum vector at the P.O.C.A.} \\
X &= (X_x; X_y) \mathrel{\widehat{=}} \text{ the 2D position of the P.O.C.A. of the particle, which lies on} \\
&\quad \text{a plane which is defined by the position of the origin and the vectors:} \\
&\quad \text{the } z \text{ axis vector and the vector pointing from the origin to the P.O.C.A.}
\end{aligned}
\tag{3.1}
$$

The sign of d_0 therefore is the sign of the z component of the angular momentum and the P.O.C.A. with respect to the origin, which depends on the charge of the particle.

In short, d_0 is a signed distance to the origin of the particles P.O.C.A. with respect to the z axis and incorporates the charge of the particle. While the particle distribution over the primary vertex of the particles encodes the actual position in space, d_0 adds the charge as additional information and also does not use the vertex, but rather the P.O.C.A. which is generally not exactly the same point. Next to the actual particles' start vertices d_0 is therefore a relevant parameter to be checked for a performance estimation of the VXDTF since it captures the charge dependency and how near the particle trajectory passes to the origin. Considering that the bin width of Figure 3.2 is 100 µm, one clearly sees that over 80% of all tracks come from a distance to origin of $< 100\,\mu\text{m}$, and 2.9% come from further away than 1 cm of the IP, while about 2% lie in the range $0.5\,\text{cm} \leq r \leq 1\,\text{cm}$. Additionally no charge dependent bias can be seen, which would lead to a diverging shape between the left and right tail of the plot.

While Figure 3.1 focuses on tracks actually leaving a sufficiently high number of hits in the detector to be able to be found, it is also instructive to analyze the positions of the end points of the tracks, i.e. the end vertices. Figure 3.3, Figure 3.4, Figure 3.5 and Figure 3.6 show the distribution of the end points of the particles of the various types, as a function of the radial distance to the origin of the IP. Only end points within the volume of the VXD have been considered. It has to be noted that these plots contain the particles mentioned and their antiparticles with the exception of photons. An end point is either a decay or an interaction vertex, or the point where the particle is absorbed by the material of the detector. Although neutrons get stuck in the VXD volume about once in three events, no plot for neutrons was added since they do not produce hits but

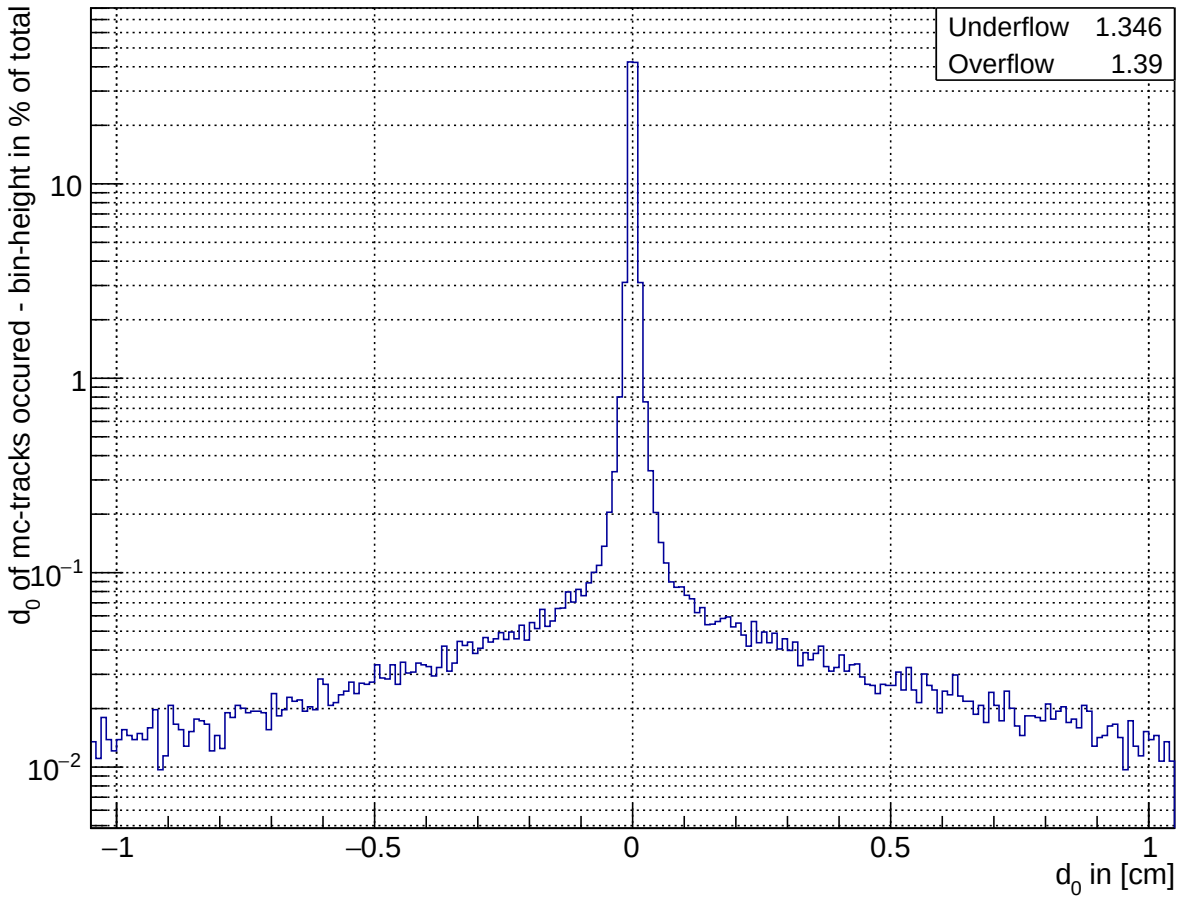
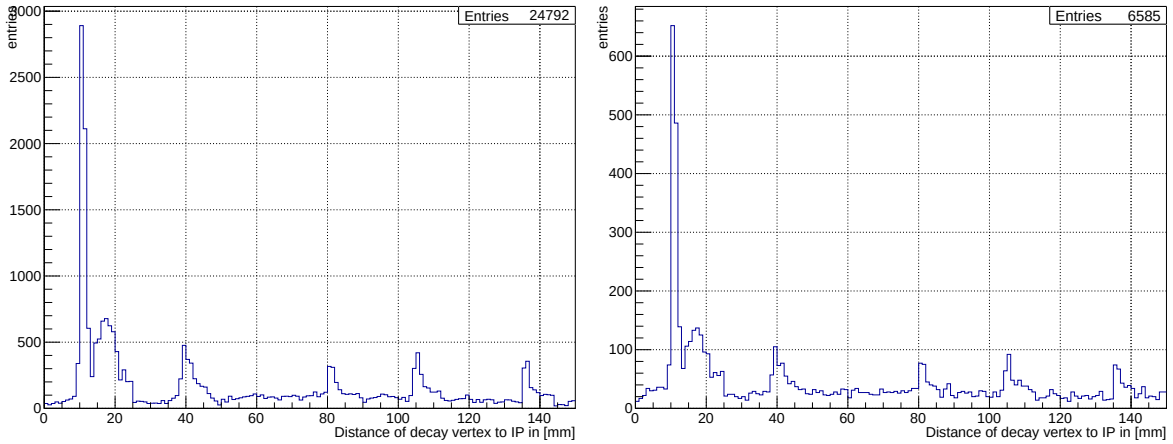


Figure 3.2: Distribution of of MC tracks having at least 6 SVD clusters over d_0 - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

simply degrade the material of the detector over the years.

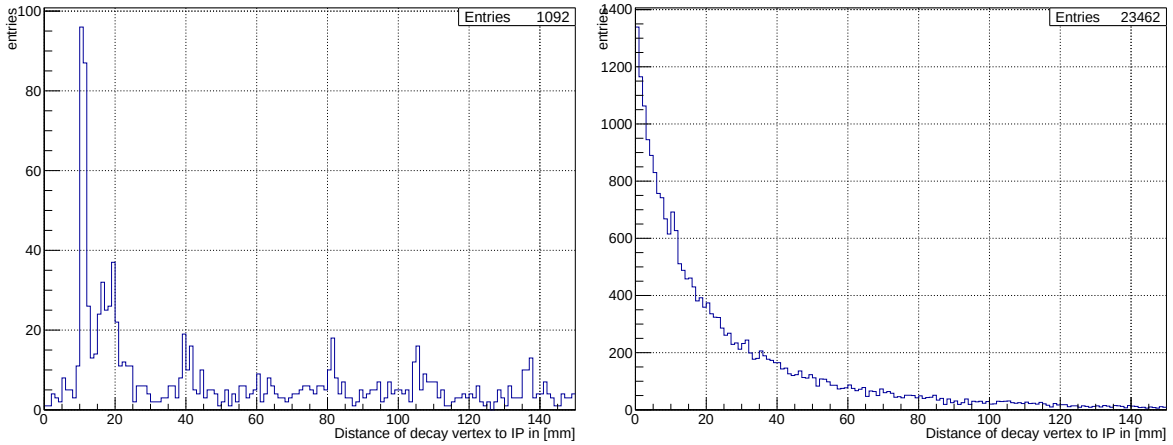
First, a detailed look at the dominant particles to be tracked in the VXD can be found in Figure 3.3. Although the distribution of the end vertices look virtually identical for charged pions and kaons, the total number of occurrences has to be noted. While pions get stuck nearly once each event in the VXD volume, the same happens for charged kaons only about once in five events. Additionally relevant are the clearly visible peaks to be found in the plots, which can be explained by considering the radial positions of the beampipe (at 1 cm), the PXD layers around 1.4 cm and 2.2 cm and the SVD layers situated at around 3.9 cm, 8 cm, 10.5 cm and 13.5 cm. The interaction of the particles with the detector clearly results in a non-negligible number of particles absorbed inside the volume. The material effects explain the high numbers of end vertices of those particles, since their lifetimes found in [22] tell us that their decay inside the volume of the VXD is very unlikely.



(a) Pion end vertex distribution.

(b) Charged Kaon end vertex distribution.

Figure 3.3: Distribution of the radii of the end vertices of charged pions (Figure 3.3a) and charged kaons (Figure 3.3b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

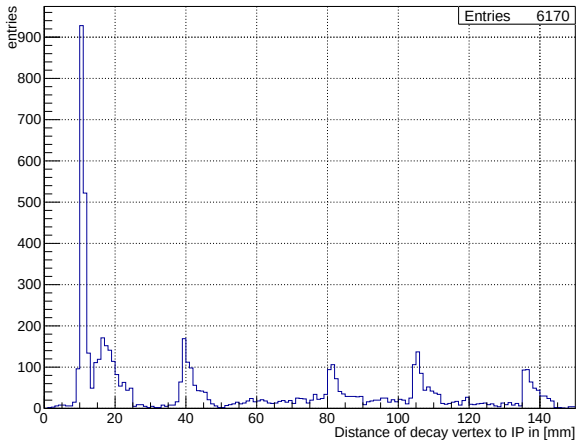


(a) K_{Long} end vertex distribution.

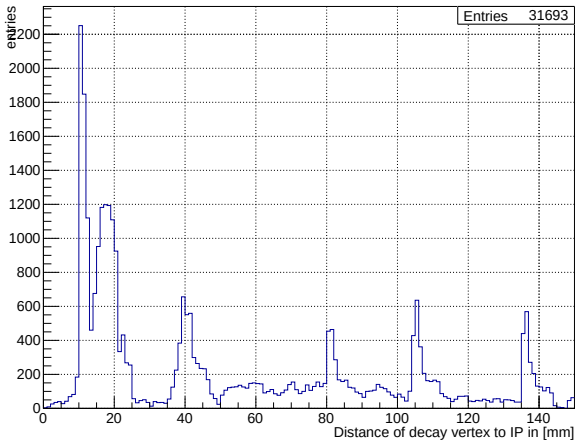
(b) K_{Short} end vertex distribution.

Figure 3.4: Distribution of the radii of the end vertices of neutral K_{Long} particles (Figure 3.4a) and neutral K_{Short} particles (Figure 3.4b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

The same pattern of end vertices can be seen in Figure 3.4a, where the long living K_{L} particles (at least in respect to the VXD volume) again preferably are absorbed by the material of the detector material. Absorption of K_{L} is, however, relatively infrequent and occurs only once in 30 events. Decays of K_{S} are more frequent. They reach the detector only because of the relativistic time dilation, since their lifetime is very short: 8.954×10^{-11} s ([22]). Nearly one K_{S} in two events decays outside the beampipe. Their decay products such as charged pions have start vertex far from the IP. The daughter particles of the K_{S} are therefore difficult to reconstruct.

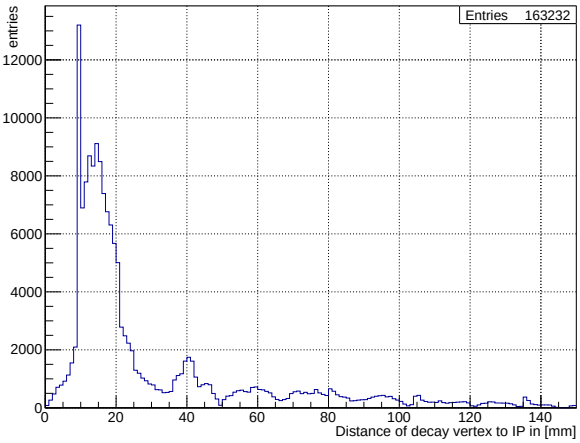


(a) Muon end vertex distribution.

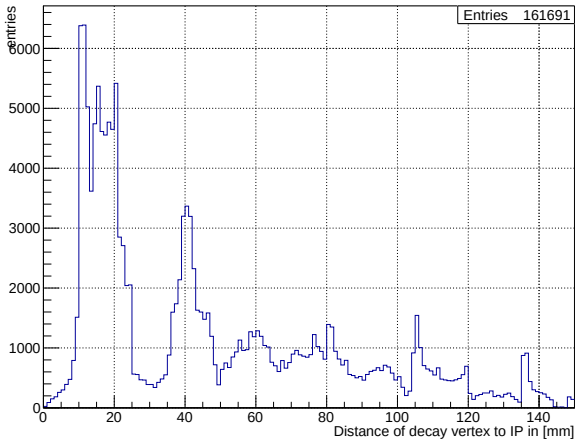


(b) Proton end vertex distribution.

Figure 3.5: Distribution of the radii of the end vertices of muons (Figure 3.5a) and protons (Figure 3.5b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.



(a) Photon end vertex distribution.



(b) Electron end vertex distribution.

Figure 3.6: Distribution of the radii of the end vertices of photons (Figure 3.6a) and electrons (Figure 3.6b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Muons and protons, as shown in Figure 3.5, again clearly show the end vertex characteristics typical for (quasi) stable particles, reflecting the layers of the VXD. In their frequency they differ noticeably: muons are absorbed by the material of the VXD once in five events, while protons are absorbed about once per event.

The next plots shown are those of photons and electrons in Figure 3.6, which are listed for the sake of completeness. In the MC simulation a lot of photons and electrons are created when particles pass through matter. These photons and electrons do exist only for a very short span of time and typically do not leave the region significantly they were created in, before they get absorbed by the material again. In order to be able to

show the typical structure of the peaks of the end vertices these short living particles are filtered by plotting only those existing for more than 10 ps. Photons can travel 2–3 mm in that time range (which is also the upper boundary for electrons) and therefore many "secondaries" can be filtered out that way. Still, the results seem to be very crowded as the total numbers of about 160,000 each suggest. Since Table 3.1 suggests that electrons form only a minor part of the track-able particles seen in the VXD, one has to accept the limited amount of interpretable information of the plots shown in Figure 3.6.

So what can be understood from these plots? The fact that even quasi stable particles like pions and protons interact non-negligibly with the VXD emphasizes that discussions about the material effects of the VXD are relevant. As the plots in Figure 3.1 suggests, tracking has either to consider tracks not coming from the origin, or accept that 100% efficiency in track reconstruction is not a realistic goal to achieve.

3.2.2 Clusters, SpacePoints, Occupancy and combinatorics

The geometrical key aspects of the detector are relevant for tracking purposes. As already sketched in Section 2.4.1, the VXD covers the full ϕ range at each layer, where the ladders of the layers have overlapping regions of about 1–9%. In θ the VXD covers the region $17° < \theta < 150°$ around the IP, which itself is specified in Table 2.1. The slanted parts in the forward region (as can be seen in Figure 2.7 and in Figure 2.8) are designed to reduce the material effects and allow a higher hit efficiency of the tracks boosted in that direction.

The VXD will operate in an approximately constant magnetic field of 1.5 T which is parallel to the z axis. Charged particles will therefore propagate along a helix whose axis is parallel to the z axis. If the helix is projected on the x-y plane, the projected curvature of the helix depends on the momentum of the particle according to the following relation:

$$\begin{aligned} p_\top[\,\mathrm{eV}/c] \mathrel{\widehat{=}} c[\mathrm{m/s}] \cdot B[\mathrm{T}] \cdot r[\mathrm{m}] \quad \xrightarrow[\mathrm{eV}/c \longrightarrow \mathrm{GeV}/c]{B=1.5\,\mathrm{T}} \quad p_\top[\,\mathrm{GeV}/c] \approx 0.3 \cdot 1.5 \cdot r \\ \leftrightarrow \quad r \approx \frac{p_\top}{0.45} \end{aligned} \tag{3.2}$$

This means that the trajectory of a particle is dependent on its momentum and not its overall energy. While particles with higher momenta (near 1 GeV/c) produce tracks which are nearly straight in the VXD volume, tracks of particles with lower momenta bend visibly and can start curling back into the VXD volume (for $p_\top < 500\,\mathrm{MeV}/c$) or never leave it at all. Using the formula given Eq. (3.2), one can calculate the minimal transverse momentum needed to get curlers which never leave the VXD before decaying or getting stuck: the outer radius of the VXD is 15 cm, therefore the helix radius of the particle is < 7.5 cm, corresponding to $p_\top \leq 34\,\mathrm{MeV}/c$ when neglecting material effects.

Unfortunately the material effects, namely energy loss and multiple Coulomb scattering, are non-negligible at low momenta. Details about material effects and their connection to

$\beta\gamma$ — the speed of the particle in relation to its mass at rest ($\beta\gamma \mathrel{\widehat{=}} p/m$) — can be found in Section 2.5. For here it is sufficient to know that at low values of $\beta\gamma$, i.e. $\beta\gamma < 1.5$ of $p_\top < 200\,\mathrm{MeV}/c$ for a pion, the ideal helix as a particle trajectory is not fully valid any more and the path of the particle becomes more and more dominated by its interaction with matter when $\beta\gamma$ decreases. For the case of the VXD the diploma thesis of Christian Pulvermacher [34] gives an overview of the situation to be faced. There it was shown that at $\theta = 90°$ pions below $p = p_\top \approx 40\,\mathrm{MeV}/c$ do not even reach layer 1 of the PXD, while about $\approx 55\,\mathrm{MeV}/c$ are needed to reach layer 5 and produce enough hits to be tracked by a TF solely relying on SVD input. The covered θ range extends these values for reaching layer 5 down to $p_\top \approx 25\,\mathrm{MeV}/c$ ($\longrightarrow$ up to p$\approx 80\,\mathrm{MeV}/c$) for the forward region and down to $p_\top \approx 40\,\mathrm{MeV}/c$ ($\longrightarrow$ up to p$\approx 75\,\mathrm{MeV}/c$) for the backward region. This means that the theoretical value of about $34\,\mathrm{MeV}/c$ is not very accurate for the situation faced, and an algorithm relying on an "ideal" helix as the track model might not be very reliable due to the heavy impact of material effects. Section 4 describes how these effects are dealt with in the implemented method for low-momentum tracking.

The properties of the hits that are produced by the VXD sensors depends on the detector type. While the PXD always delivers 2D clusters, the SVD is a double-sided strip detector delivering a 1D cluster on each side. The u side measures the position in $r - \phi$ using the long strips, the v side measures the z coordinate using the short strips. Since these measurements are independent from each other, the u and v coordinates have to be combined to 2D hits, called SpacePoints. This leads to the issue of ghost hits, of which more details are given in Section 2.4.1.

Tracks have to generate at least three SpacePoints to be able to be found, which leads to a lower threshold for track momenta of around $50\,\mathrm{MeV}/c$, as described above. Therefore tracks which produce at least three hits are the focus of this chapter.

To give an overview of some relevant characteristics of typical $\Upsilon(4S)$ events, the sample described in Section 11.1 has been used for illustration. About 10 particles per event leave enough SpacePoints to be found, and background hits have to be discarded; for more details about background and its sources, see Section 2.6. The actual spread of the number of tracks per event can be seen in Figure 3.7, which shows that most of the events contain about 5–15 tracks that have to be found by the TF.

Next to the number of tracks per event the number of clusters per track is relevant, which is shown in Figure 3.8. Tracks with 8 clusters are the most frequent ones, because the SVD has 4 layers and the double-sided sensors deliver typically two clusters per sensor, one u and one v cluster. 8 clusters and therefore 4 SpacePoints for a track occurs in only about 68% of the cases. In about 23% of the cases at least one overlapping part is hit, which results in a higher numbers of SpacePoints than there are actual layers. This number is so high that the overlapping parts have to be incorporated into the tracking approach.

Another reason for having more SpacePoints than layers are curling tracks. The geometry and the magnetic field of Belle II force particles to curl within the tracking detectors if the particle has a transverse momentum of $> 500\,\mathrm{MeV}/c$ and its θ angle is

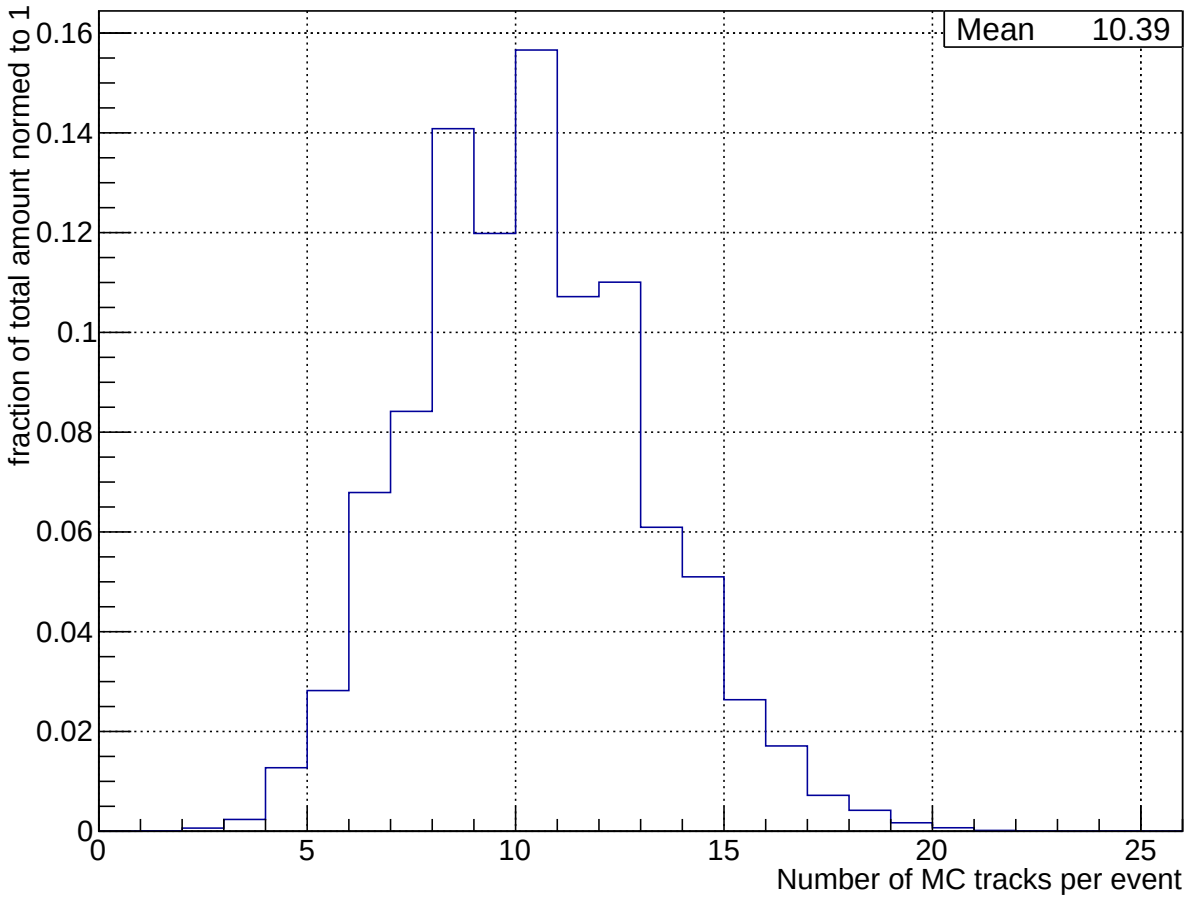


Figure 3.7: Distribution of the number of tracks per event normed to amount of total occurrence - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

near 90°. The lower the transverse momentum, the more the θ angle can differ from 90° to still allow curlers, which is especially relevant for tracks having transverse momenta of $> 90\,\mathrm{MeV}/c$, which leads to curling tracks not leaving the VXD before getting stuck in the forward or backward shields. In the setup used for the simulation no curlers are coming back from the CDC — as described in Section 11.1 — since the magnetic field is deactivated outside the VXD volume, which is not the case in the actual Belle II environment. The track length can be used as an indirect indicator for curling tracks within the VXD volume. About 0.3% of the tracks have got more than 16 clusters, which can be achieved only by curling particles inside the SVD. This number has to be regarded as the lower limit of curling particles, since hitting the overlapping parts of all layers with the same track is highly improbable. Another aspect has to be noted here: there is a significant number of tracks having an odd number of clusters. This typically comes from sensor inefficiencies due to energy deposits below the signal threshold, while

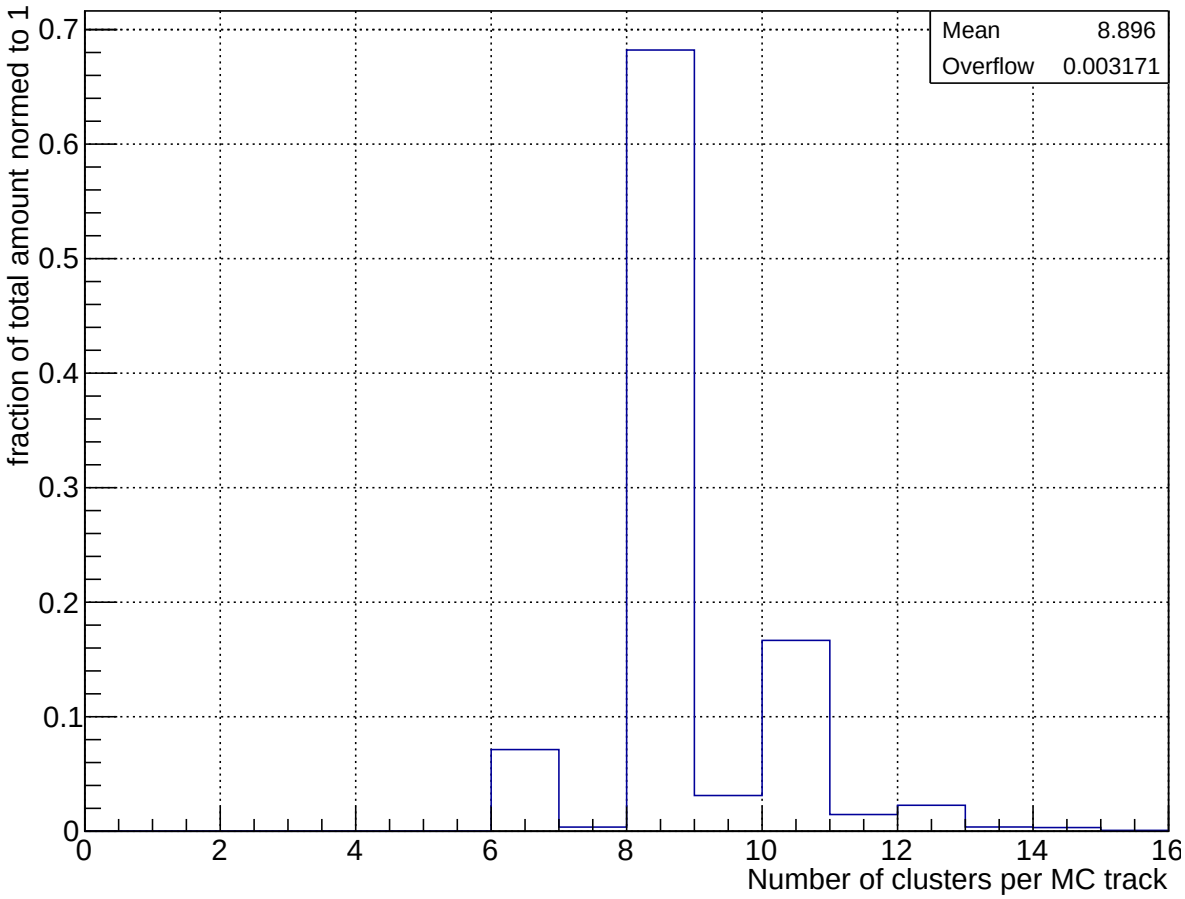


Figure 3.8: Distribution of the number of clusters per track normed to amount of total occurrence - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

another typical source, namely dead strips, was not part of the simulation. These single clusters are difficult to deal with when working with SpacePoints and seem to occur in at least 6% of the cases. The actual number is likely to be higher since tracks with an even number of SpacePoints can contain several single clusters.

In Table 3.2 the number of clusters, strips and SpacePoints is listed layer-wise and in relation to their provenance, along with the strip occupancy. Table 3.2 shows that numbers of clusters that are coming from the background are dominating. This means that the signal to noise ratio on cluster level is about 1:4, while it is 1:8 on the SpacePoint level, which indicates domination by ghost hits. The issue of ghost hits manifests itself in the ratio between clusters and SpacePoints: if each sensor with clusters would be hit only once, the number of SpacePoints should be exactly half the number of clusters (since two SVD clusters form one SpacePoint). But especially for the realistic case of having $\Upsilon(4S)$ with background the number of SpacePoints is about twice the number of clusters. This

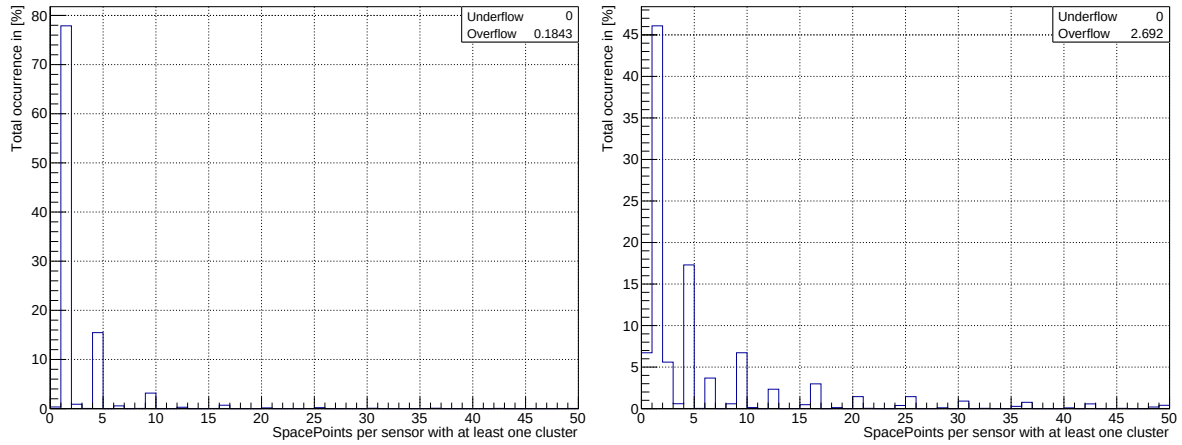
Particle name	$\Upsilon(4S)$-only	BG-only	$\Upsilon(4S)+\text{BG}$	$\Upsilon(4S)+2\times\text{BG}$
L3 strips u/v	49.2/36.7	260.0/121.7	308.1/158.0	562.2/278.8
L3 occupancy u/v [%]	0.46/0.34	2.42/1.13	2.87/1.47	5.23/2.59
L3 clusters u/v	11.8/11.8	39.0/37.9	50.3/49.3	87.0/86.1
L3 SpacePoints	26.1	233.9	318.0	791.0
L4 strips u/v	39.4/29.1	120.3/61.2	159.1/90.1	277.8/150.6
L4 occupancy u/v [%]	0.17/0.19	0.52/0.40	0.69/0.59	1.21/0.98
L4 clusters u/v	12.7/12.6	29.9/26.7	42.5/39.2	71.8/65.3
L4 SpacePoints	22.5	100.5	143.1	320.4
L5 strips u/v	37.3/28.5	122.7/67.2	160.1/95.8	282.7/162.9
L5 occupancy u/v [%]	0.10/0.12	0.33/0.27	0.43/0.39	0.77/0.66
L5 clusters u/v	12.3/12.1	35.0/30.5	47.3/42.7	82.0/72.9
L5 SpacePoints	19.2	99.3	132.3	299.3
L6 strips u/v	38.3/28.6	134.6/76.8	172.9/105.4	307.1/182.0
L6 occupancy u/v [%]	0.06/0.07	0.22/0.19	0.28/0.26	0.50/0.44
L6 clusters u/v	12.4/12.2	42.1/36.3	54.4/48.5	96.2/84.5
L6 SpacePoints	17.0	100.8	127.9	283.1
Total strips u/v	164.3/122.8	637.6/326.8	800.3/449.3	1429.8/774.4
Total occupancy u/v [%]	0.12/0.14	0.48/0.37	0.61/0.51	1.08/0.88
Total clusters u/v	49.2/48.7	146.0/131.3	194.4/179.6	337.1/308.9
Total SpacePoints	84.8	534.6	721.3	1693.8

Table 3.2: Mean number of clusters and SpacePoints per Layer (L3-L6) and total of relevant cases - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

problem becomes even more severe in layer 3, which is the layer of the SVD closest to the IP, where for each cluster there are more than three SpacePoints. The fact that nearly half the number of all SpacePoints can be found on layer 3 means that combinatorial issues are likely to be problematic for special constellations of background and $\Upsilon(4S)$ tracks. Therefore powerful filters have to be used to successfully catch such cases. This is also clearly visible in the occupancy levels: while the u and v strip occupancies of layer 3 easily reach up to 3%, layer 6 stays at a low level of about 0.5% for the case of $\Upsilon(4S)$ events with realistic background. When comparing the values of u/v clusters versus the u/v strips and u/v occupancies, one can see that for signal clusters the number of u and v clusters barely differs. For background clusters, however, there are significantly more u than v clusters, which leads to the conclusion that single cluster SpacePoints may not be worthwhile to consider since cases where only a single u cluster can be found on a sensor, the probability is higher that it is a background hit than coming from a signal hit. But not only the number of clusters starts to differ for u and v clusters in case of

background, but also the cluster sizes (in number of strips per cluster). This indicates another possible way to distinguish between signal and background clusters and might allow the development of intelligent algorithms for combining clusters to SpacePoints.

While Table 3.2 focuses on conclusions that can be drawn from the mean values of strips, clusters and SpacePoints, which was discussed in the previous paragraph, Figure 3.9 only plots the number of SpacePoints registered for all sensors having at least one cluster, which allows to neglect empty sensors and reveals some more details. First the focus on



(a) $\Upsilon(4S)$ events without background.

(b) $\Upsilon(4S)$ events with background.

Figure 3.9: Distributions of number of SpacePoints (being a combination of an u and a v cluster on the same sensor) per sensor which have at least one cluster (sensors without clusters are ignored here); the first bin represents the case of having no SpacePoint for a sensor with at least one hit. The overflow value is in percent of the whole dataset - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 3.9a reveals that the dominating case is one SpacePoint per sensor, which about represents 78% of all measurements. Nearly 16% have four SpacePoints and nearly 4% have nine SpacePoints, while the rest of the cases stay below the 1% level. Although very small in occurrence compared to the total amount of measurements, some of the barely populated bins contain additional information. The first bin, representing the case of having clusters on a sensor but no SpacePoint, occurs in about 0.5% of the cases. This value and the ones collected for the case of two, six and twelve SpacePoints accumulate to about 2%, which means that for the case of no dead strips the detector efficiency is about 98%, since these cases can only occur if at least one hit had only one cluster instead of two. This value only seems to be better than the conclusion which can be drawn of Figure 3.8, since the value of 2% hides the fact that both cluster types can be missing. If for 0.18% of the cases more than 50 SpacePoints for a single sensor were recorded, which means that in about 2 out of 1000 sensors hit, the sensors are flooded with ghost hits. These numbers change dramatically when background is added, which is shown in Figure 3.9b. About 2.7% of all sensors hit then have more than 50 SpacePoints

and about 1% have more than 100 SpacePoints, a fact which demands special care for combining clusters to SpacePoints. Additionally relevant is the fact that in nearly 7% of all cases there were no SpacePoints produced of clusters. This strongly indicates that the typical energy deposit of background hits is far lower than for "signal" hits, which indicates a potential solution for keeping background and primary hits separated. It also supports the approach of *not* creating SpacePoints out of single clusters, since it is highly probable that they come from background hits. Although the typical case of having a single SpacePoint drops from 78% in the case without background to about 46% in the case of having background, the value for four SpacePoints per sensor nearly stays the same. This indicates that primary and background hits are separated in space and therefore should be able to be filtered from normal tracks more easily. Unfortunately this can only be supported for background tracks not coming from the origin, since those leaving realistic tracks in the VXD will probably be detected as such by typical TF algorithms. An example for such a type of background is the QED background. More details about the relevant background types can be found in Section 2.6.

Next to the mean values of SpacePoints and occupancy per layer — as presented in Table 3.2 — the distribution of these characteristics is of interest as well. Figure 3.10 and Figure 3.11 show the data in six box plots visualizing the min and max values by the "whiskers", the range between the 25% and the 75% quantile by the box, the median as a red line and the mean value as a blue dot. In direct comparison one again can see the effect of ghost hits, since for example in layer 3 the mean number of SpacePoints increases by a factor of 15, while the occupancy, which is not affected by ghost hits, increases only by a factor of five. A focus on the SpacePoints reveals again that layer 3 is taking most of the hits, which can add up to 3,000 in this single layer with the data used.

Layer 3 is suffering most from the background added since some background types create mostly low-momentum particles (as is the case for QED background). Additionally the tungsten shield protecting the SVD from beam dependent background types like bremsstrahlung and Touschek effect can not shield the inner layers L1–L3, as the power and readout cables of the PXD block the region nearest to the beam pipe. Compared to the outer layers, layer 3 has a relatively small area of active sensor elements. This means that the same number of clusters evenly spread on higher layers would result in a smaller number of SpacePoints there compared to those created in layer 3.

After discussing the combinatorial issues on the SpacePoint level, another essential marker for combinatorial issues is the possible number of two-hit combinations. This is a relevant number since two and three-hit combinations are often used as so-called *seeds* for track finding, where the seeds are then used in track following algorithms to find and add compatible hits to their TCs. To present a straightforward estimate of the number of two-hit combinations, the following equation 3.3 is used to determine an unfiltered number of two-hit combinations:

$$2HC = \sum_i n_{\text{Hits of Layer } i} \times n_{\text{Hits of Layer } i+1}, \quad i \in 1, 2, 3 \tag{3.3}$$

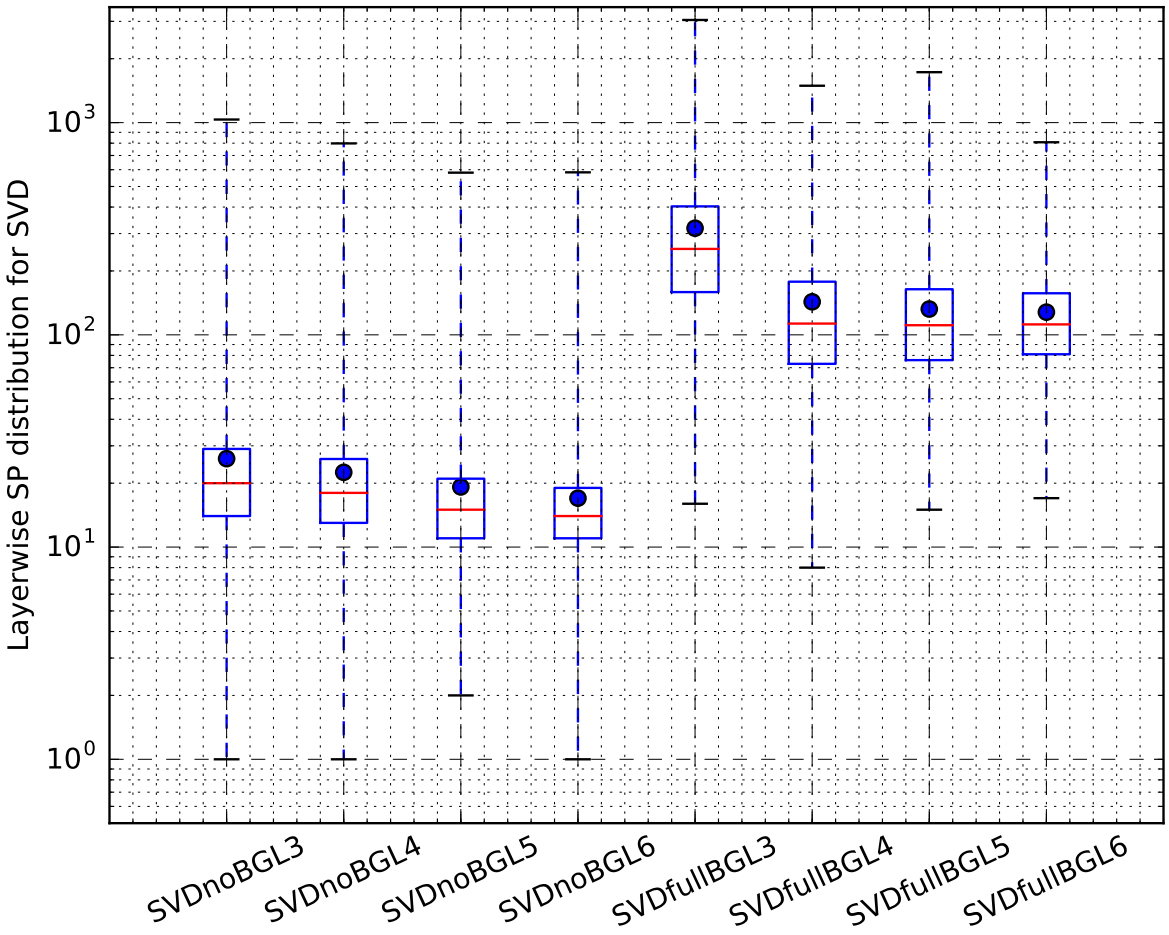


Figure 3.10: Compares numbers of SpacePoints per layer for the case of $\Upsilon(4S)$ events without (left) and with (right) background. The whiskers of the boxplots represent the min (lower) and max (upper) values found, the boxes enclose the 25% and 75% quartiles, while the red bar in the middle represents the median and the blue dot the mean of the values - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1

The formula underestimates the combinatorial issue since it neglects the overlapping parts of the detector, which results in more than one hit per layer of a single track in many cases. Additionally the formula does not take into account that the sensors themselves do not have a100% hit efficiency and therefore layer jumps of neighboring track hits — if a particle leaves a hit in layer 3 and layer 5 but none in layer 4, this is called a layer jump — have to be taken into account as well. A similar formula can be constructed for three-hit combinations, which estimates the rough numbers of naive seeds needed for tracking algorithms which require a seed, like the KF:

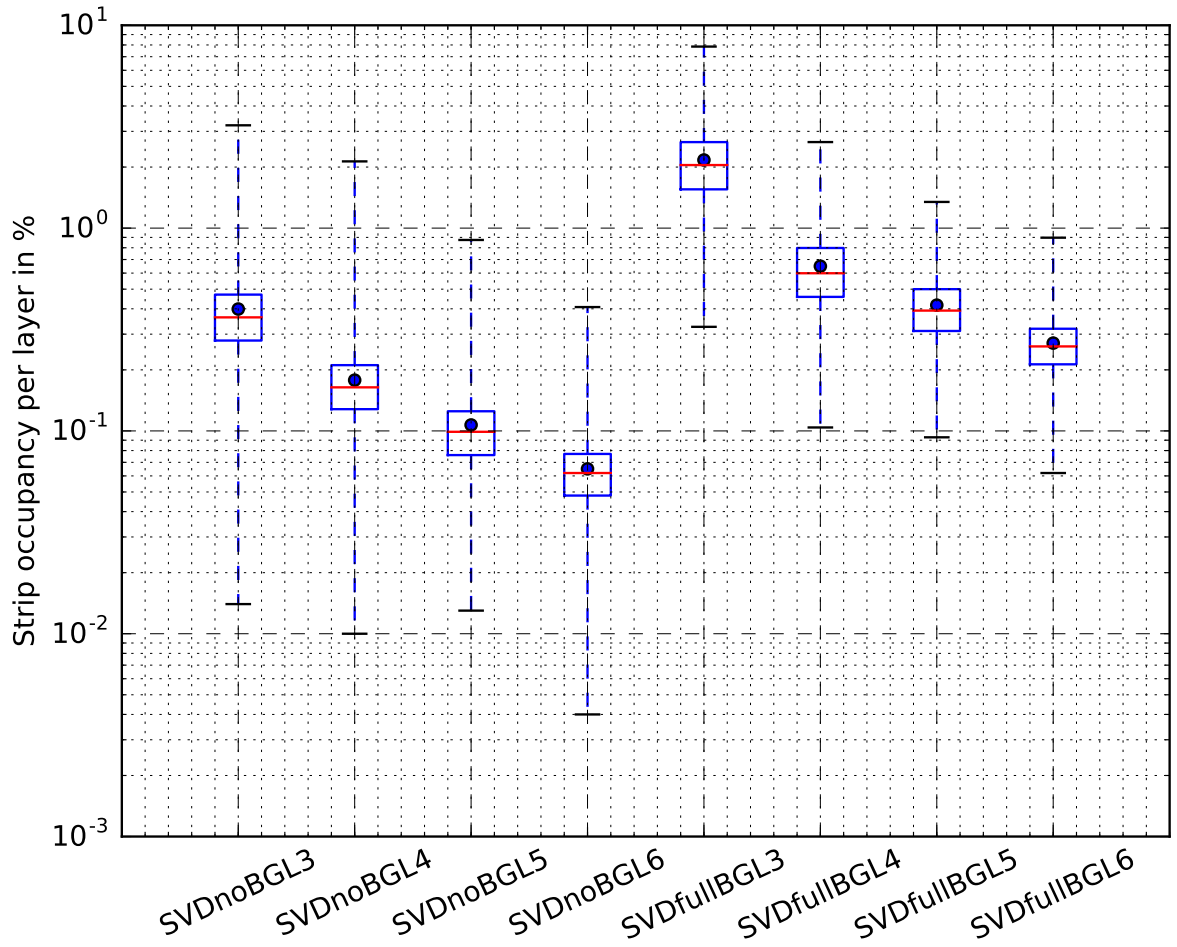


Figure 3.11: Compares strip occupancy per layer for the case of $\Upsilon(4S)$ events without (left) and with (right) background.The whiskers of the boxplots represent the min (lower) and max (upper) values found, the boxes enclose the 25% and 75% quartiles, while the red bar in the middle represents the median and the blue dot the mean of the values - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1

$$3HC = \sum_{i} n_{\text{Hits of Layer } i} \times n_{\text{Hits of Layer } i+1} \times n_{\text{Hits of Layer } i+2}, \quad i \in 1, 2 \tag{3.4}$$

Applying the formulas 3.3 and 3.4 to the number of SpacePoints per event found in the test sample used in this section leads to the numbers summarized in Table 3.3, which clearly shows that it is not advisable to ignore the combinatorial issue for tracking algorithms which need seeds.

While the case of $\Upsilon(4S)$ events without background seems still practicable for efficient

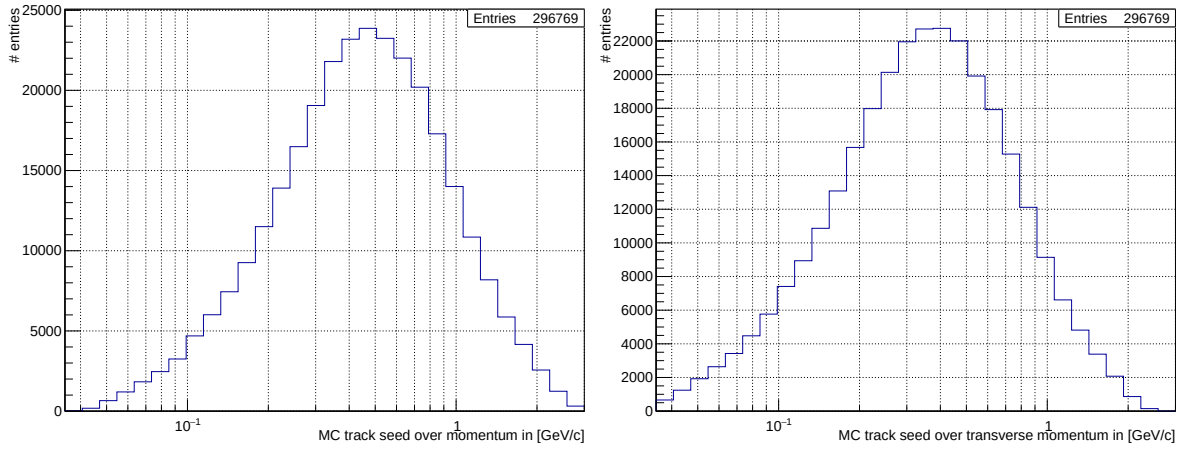
implementations of e.g. the KF, this is definitively not the case for 100% background, and a fast preselection of good seeds is mandatory.

Setup	$\Upsilon(4S)$+ 0% BG	$\Upsilon(4S)$+ 50% BG	$\Upsilon(4S)$+ 75% BG	$\Upsilon(4S)$+ 100% BG	$\Upsilon(4S)$+ 125% BG	$\Upsilon(4S)$+ 150% BG	$\Upsilon(4S)$+ 200% BG
2HC mean	1,348	20,829	44,471	81,340	133,734	207,941	434,064
3HC mean	18,619	1.1×10^6	3.4×10^6	8.4×10^6	1.7×10^7	3.4×10^8	1.0×10^8

Table 3.3: Straightforward estimation of two-hit combinations (2HC) calculated using equation 3.3 and three-hit combinations (3HC) using equation 3.4 for mean value of SpacePoints per layer and event - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

3.2.3 Momentum, θ and Φ

Another relevant aspect is the distribution of the tracks over (transverse) momentum, which is plotted in Figure 3.12, where the plots are split into tracks versus momentum (Figure 3.12a) and its transversal component (Figure 3.12b). The tracks are mostly



(a) Momentum distribution).

(b) Transverse momentum distribution.

Figure 3.12: Momentum (Figure 3.12a) and transverse momentum (Figure 3.12b) distribution of MC tracks with at least 6 clusters in the SVD on a logarithmic scale - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

tracks below $1\,\mathrm{GeV}/c$ and are additionally described in Table 3.4. Although only a small percentage of all tracks are below $100\,\mathrm{MeV}/c$, they play an important role for certain decay channels, like the D^* decay and therefore low-momentum TFs which can operate independently from the CDC are needed. Tracks below $500\,\mathrm{MeV}/c$ bend visibly in the VXD, and those below $300\,\mathrm{MeV}/c$ have considerably higher energy deposit (Section 2.5),

which can be concluded from the facts that pions are dominating the particle list and that particles in general (except electrons) increase their energy deposit at $\beta\gamma$ below three; more details about that can be found in Section 2.5.

Range	0-100 MeV/c	100-300 MeV/c	300-500 MeV/c	500-1000 MeV/c	>1000 MeV/c
$\lvert p\rvert$	3.3%	26.2%	26.0%	31.3%	13.2%
$\lvert p_\top\rvert$	4.5%	35.0%	26.2%	24.4%	7.4%

Table 3.4: Table form representation of the $|p|$ and $|p_\top|$ distribution of tracks - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

In Figure 3.13 the distribution over momentum for all particle types listed in Table 3.1 — normalized to one to allow direct comparisons — is shown. First one has to consider

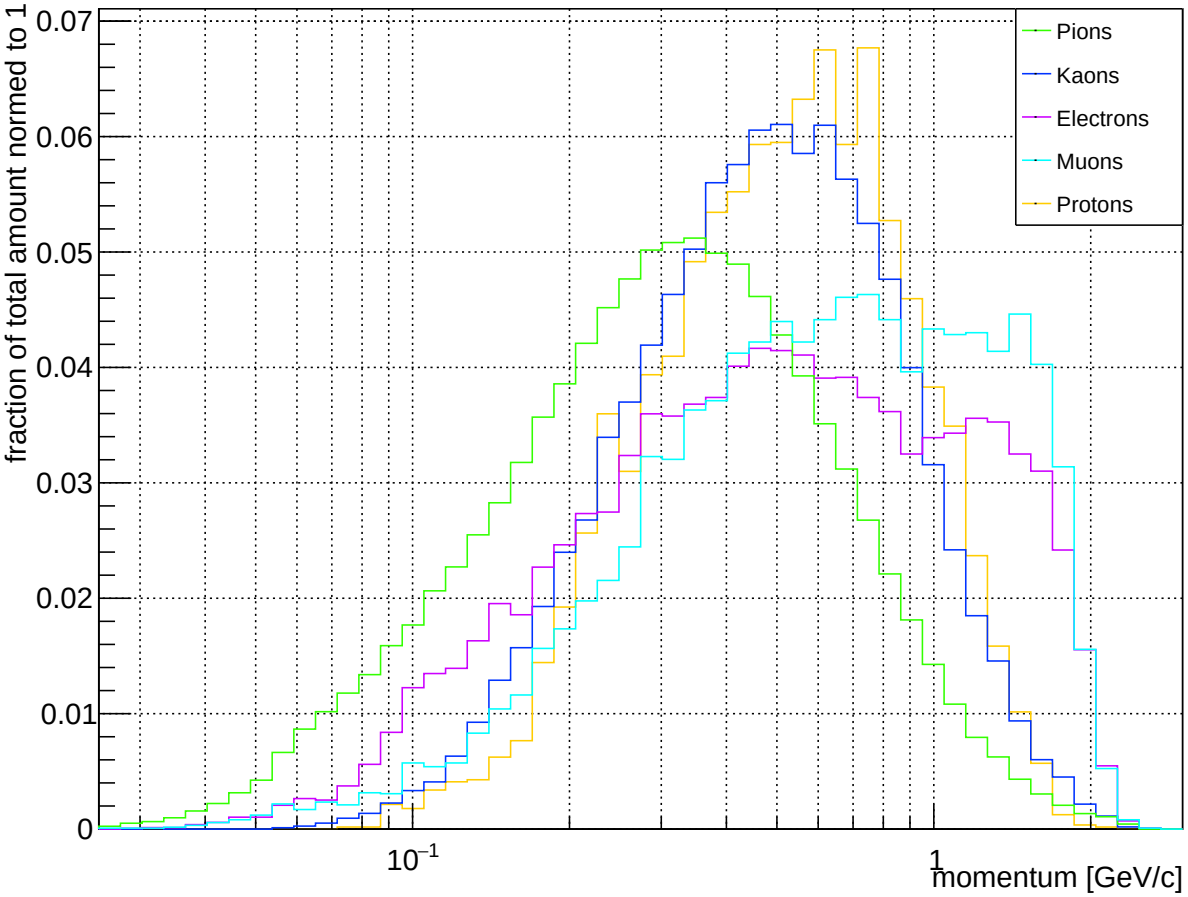
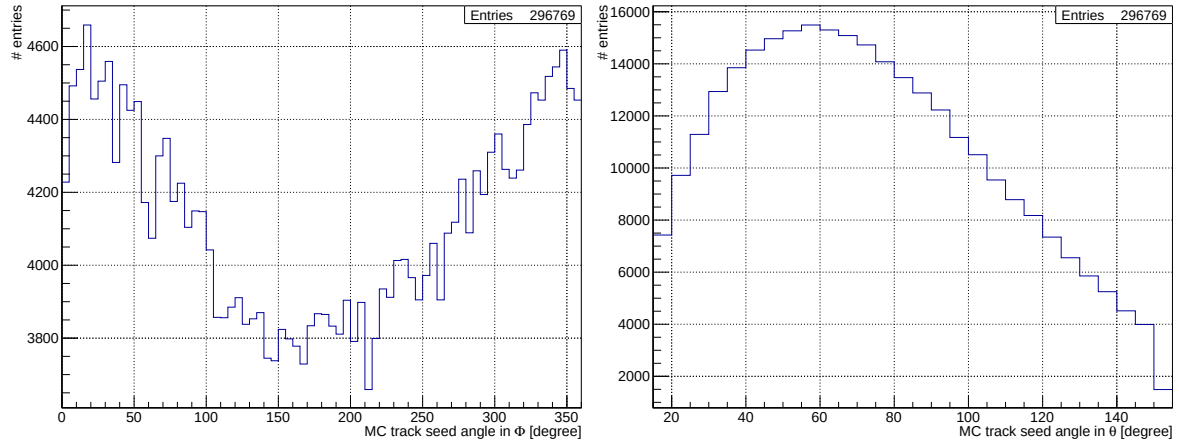


Figure 3.13: Distribution over momentum of the particle types listed in Table 3.1, taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1

that these particles were selected by requiring at least 6 clusters in the SVD, which means that the sample is not identical with the decay plots in Figure 3.3, Figure 3.5 and Figure 3.6 — especially background particles and secondaries are practically nonexistent in this plot. The distribution of pions over momentum shows that their expected value is the lowest and their left tail stretches farthest into the low momentum range, while kaons typically have higher momenta. Unsurprisingly, protons are rarely occurring in low momentum regions, since protons with $p \approx 100\,\mathrm{MeV}/c$ have a $\beta\gamma$ value of ≈ 0.1 and therefore very high energy deposit in the range of MeV per SVD layer; nonetheless there are still some protons surviving long enough to leave tracks in the SVD. The bumps seen in the distribution for muons and electrons in the region of $1\,\mathrm{GeV}/c < p < 2\,\mathrm{GeV}/c$ are probably from the same decay channels allowing electrons or muons to be the final product. A probable candidate for this would be the decays of J/Ψ which result in two muons or two electrons in 6% each of the cases.

The distributions of momentum directions of the particle vertices are dominated by the SuperKEKB and IP design (described in Section 2.3.2 and Section 2.4) and can easily be spotted in the plots of Figure 3.14. The ϕ values have their minimum at about 180°, which



(a) Phi distribution.

(b) Theta distribution.

Figure 3.14: Phi (Figure 3.14a) and Theta (Figure 3.14b) distribution of MC tracks with at least 6 clusters in the SVD on a logarithmic scale - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

comes from the fact that both beams are lying in the horizontal plane ($\phi = 0° \,\&\, \phi = 180°$), come from the direction of $\phi = 180°$ on the r-ϕ plane and therefore boost their particles in the region around 0°/360°. This asymmetric boost is also visible in Figure 3.14b where one can easily see the boost in forward direction ($0° < \theta < 90°$), caused by the asymmetric beam energies of the collider, forcing the majority of the particles in that direction. Therefore not all sensors are populated uniformly and hotspots will be formed, which complicates the situation for tracking.

3.3 Summary of the event characteristics

Although having only 10 tracks per event would seem to suggest otherwise, geometrical and technical aspects of the VXD, the IP and SuperKEKB design as well as the background sources provide a challenging situation for tracking purposes. To summarize: a good TF for the VXD has to deal with kinks in tracks due to a high energy deposit of low momentum tracks, tracks not coming from the origin and a dominating influence from ghost hits which results in exploding combinatorics when background and event hits occur in unlucky constellations. Paired with missing hits and overlapping parts in the layers, many basic tracking approaches will likely have trouble dealing with the situation presented in this chapter. Additionally several hints for further ways of preselecting relevant hits where sketched and support that further efforts should be invested in that topic. Unfortunately the picture is still missing some relevant pieces, since dead strips, mis-alignment and the impact of back curling particles from the CDC could not be considered in this study. Nonetheless many pitfalls and side effects could be identified, and therefore this chapter serves as an introduction to low-momentum tracking in the Belle II VXD.

CHAPTER 4

THE VERTEX DETECTOR TRACK FINDER (VXDTF)

4.1 Introduction

The core of this thesis is the silicon-only track finder VXDTF, which is the general purpose TF implemented for the VXD of Belle II. A conceptual study of this TF was already presented in [1], and its current implementation is described in this chapter in thorough detail. Along with the VXDTF there are several other approaches for track finding in the VXD under development: a FPGA-based implementation of a fast Hough transform described in [26]; a track extrapolation approach using CDC TCs as seeds; and a combinatorial Kalman filter [29] (CKF) as track collector and another Hough transformation implementation searching for ROIs in the PXD using only two SpacePoints of the SVD. Unfortunately none of the other approaches mentioned are currently in a shape that would allow a fair comparison of the implementations. Therefore no further details of these implementations will be given here.

While the alternative approaches listed above are implemented for special tasks only, the VXDTF is to be used for two main tasks:

- In the HLT the VXDTF is used as a SVD only TF running on the small computing cluster of the HLT, with the aim to construct TCs that are used for ROI finding on the PXD. The focus in this mode of operation lies in a fast on-line reconstruction in 3 to 4 layers dealing with the combinatorial problem of the SVD background with limited hardware power.

- In the Express Reco[55] the VXDTF is used for 6 layer VXD tracking with reduced ROFs of the PXD. Although the number of hits at that point is already reduced

[55] Express Reconstruction system

by about 90%, the total number of clusters per event still easily outmatches the number of SpacePoints expected in the SVD. This increased combinatorial issue is treated with considerably larger computing power of the Express Reco system, which still runs on-line.

Both approaches use the same modules written in the `C++11` programming language and implemented in the basf2 framework, but with differing settings. Although the focus of the VXDTF lies on reconstruction of low-momentum tracks, since they can not be recovered via extrapolation techniques using the CDC, TCs of the full momentum range are expected to be reconstructed with solid efficiencies. The lower limit imposed by the geometry of the detector, the strength of the magnetic field and the material effects of low-monentum particles, is a transverse momentum of 50 MeV/c. Chapter 5 shows how the implementation described in the following sections of this chapter fares with realistic simulated input.

4.2 Basic structure

This section gives an overview of the main parts of the VXDTF, which is then described in detail in the following sections. Figure 4.1 sketches the most relevant parts of the VXDTF as a flowchart. The VXDTF takes hits from the silicon detectors, namely

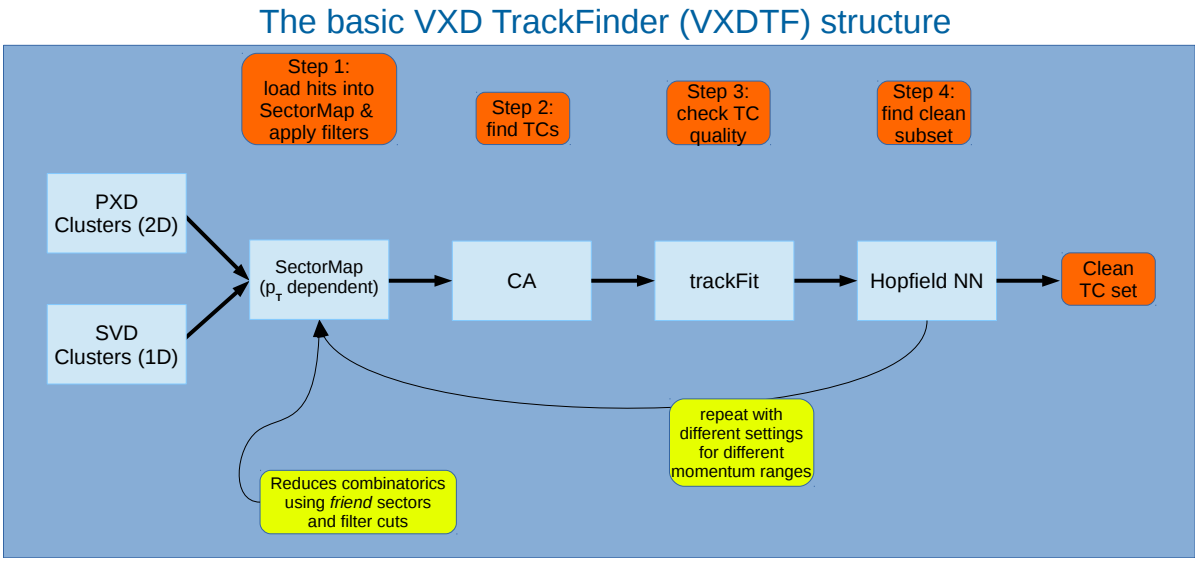


Figure 4.1: The basic structure of the VXDTF

PXD and SVD, and converts them into a common hit format, the so-called SpacePoints. These SpacePoints are then sorted into a SecMap, a tool that is described in close detail in Section 4.3, which then is used for filtering hit combinations. Compatible two-hit

combinations are stored as Segment[56]s, which are then treated as Cells for the CA algorithm. Compatible Cells are stored as Neighbour[57]s; more details about that are given in Section 4.3 and Section 4.4.1. The network of neighboring Cells created by applying two and three-hit Filters is then used by the CA to define chains of compatible hits, which will be collected as TCs afterwards.

After collecting the preliminary TCs, they are filtered again using four-hit Filters and then are fitted by a fitting algorithm described in Section 4.5. The fitting algorithm assigns a quality indicator to each of the TCs, which are then checked for overlaps, meaning whether or not they share clusters. All overlapping preliminary TCs are then fed into an algorithm to find a clean subset of final TCs, a process which is described in Section 4.6. This process is repeated several times, using settings for different momentum ranges. It starts with the easier high-momentum pass and reserves the best TCs — and their hits — found in that pass to reduce the combinatorial issue for the following passes.

4.3 The SecMap

The basic idea of the SecMap is to reduce the combinatorial issue in track finding using information, for instance Filter cuts, which is first collected off-line, then applied on-line. The problem with combinatorics emerges from the fact that local algorithms do always need to combine suitable hits to chains, and pre-selecting possible combinations done in a straight-forward approach — like combining hits from neighboring layers — leads to a high number of possible combinations. This issue has been illustrated in Section 3.2.2 and an estimation of the typical number of two- and three-hit combinations expected in our events is given in Table 3.3. The geometry of the VXD increases the combinatorial effect because of considerable overlaps of sensor ladders within the same layer.

To handle the situation of having more than one hit of the same track at the same layer, the possibility of missing hits and the issue of many ghost hits and therefore a high probability of combining a bad pair of hits, a virtual subdivision of the sensors was introduced, the Sector[58]s. The impact of the Sectors on the reconstruction speed is described in the following section.

4.3.1 Sectors, Friends and Filters

Some essential terms related to the SecMap have to be defined before a discussion of the characteristics of the SecMap is feasible.

Sectors are virtual subdivisions of the sensors of the VXD. Each sensor is subdivided both in the $r - \phi$ direction and in the z direction. The typical number of subdivisions per sensor used in the VXDTF is $3 \times 3 = 9$ and allows to decouple sensor areas having

[56] Segment: Two compatible hits form a Segment.
[57] Neighbour: Neighbour-Cell
[58] Sector: Subdivision of a VXD-sensor which is able to store Filter cuts.

overlaps with neighboring ladders from those having none. To each of these Sectors then compatible *inner* Sectors — being called Friend[59]s — are stored. Figure 4.2 shows an illustration of such a Sector having several Friends.

By analyzing a given set of tracks for training such Sector Friend combinations can be stored and all Sector Friend combinations together form a network. The SecMap — the name of the network of Sectors — where each Sector is connected to its own Friends, is a directed graph, because of the fact that only *inner* Sectors can become a Friend of a Sector. Additionally this graph has no loops, a feature being made possible due to the subdivision of sensors; something which would not work for a network of full sensors, since they would then connect all sensors on neighboring ladders to a complete loop because of the closed windmill design of the VXD. The SecMap is therefore a Tree. An abstract illustration of the SecMap is sketched in Figure 4.3a.

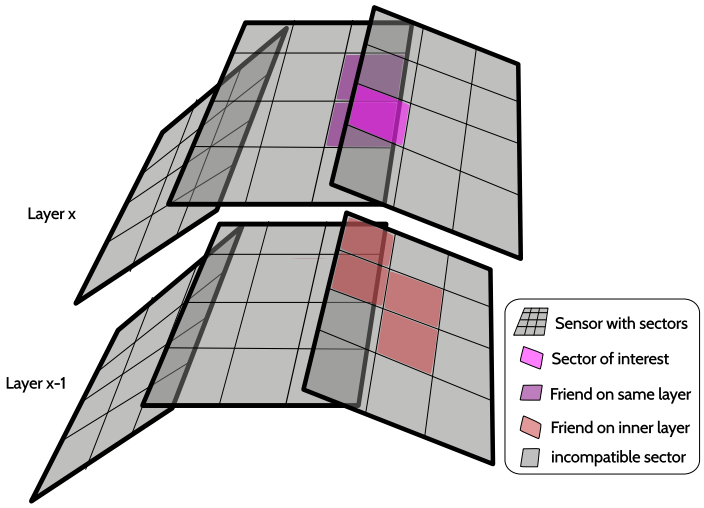
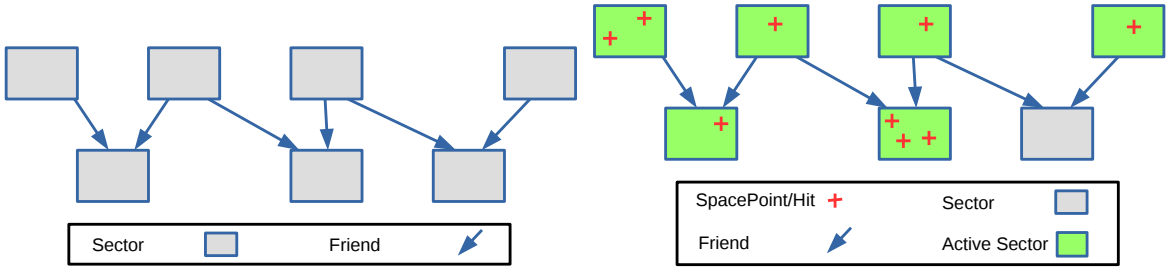


Figure 4.2: For each Sector its own set of compatible Friends is stored. In this case a 3×4 Sector per sensor setting is used and for the Sector of interest shown six Friends were found. Two of them are on the same layer on a neighboring sensor, the other ones are on the next inner layer.



(a) Sectors and their Friends including their individual Filter cuts are static and stored off-line to be used on-line.

(b) For each event SpacePoints are sorted into Sectors which become active for the current event.

Figure 4.3: Abstract illustration of how the SecMap works.

The separation of regions in the VXD by the introduction of Sector-Friend relations allows to filter SpacePoint-combinations too. This is done for each event by sorting

[59] Friend: A compatible inner Sector connected to an outer one.

SpacePoints to their corresponding Sectors in the SecMap. After the sorting step only SpacePoints of compatible Sectors are then checked for valid SpacePoint combinations. This reduces the combinatorial issue, since only a small number — typically around five — of Sectors can become Friends of any given Sector. This makes the SecMap a powerful tool for preselecting hit combinations, since only an area of less than a sensor size has to be considered for compatible inner hits instead of the full inner layer.

The prerequisite of the SecMap to be a Tree is needed to allow finding chains of compatible SpacePoints without entering any infinite loop state. The hit chaining algorithm used in the VXDTF — the CA — is described in Section 4.4.

An essential task of the SecMap is to deal with the geometrical specialties of the detector and allows to increase the mean track length compared to a layer based approach. This works via the introduction of Sector combinations allowing both Sectors to be at the same layer, which is a common situation for overlapping sensors.

Additionally a *virtual* Sector is used, which is not attached to any sensor but is placed at the IP. Since no sensor of the VXD is positioned there, no real hits can be found to be on that Sector. Therefore the virtual Sector carries nothing but a virtual hit with the coordinates of the IP. For the error of the virtual hit the spatial volume of the IP region is assumed. This virtual hit is used as the most probable origin of the particle and is treated like a normal hit in reconstruction. This feature is needed to compensate the small number of layers available for on-line tracking and effectively increases the number of hits for a reconstruct-able track segment by one hit. Especially for low-momentum particles, where the curvature is significant, the reconstruction efficiency profits from the extra hit introduced. Of course the actual start vertex of the particles is not known, but since practically all particles to be tracked come from a region "near" the origin, the assumption of an extra hit, where the position error is sufficiently high, is valid. Since the SecMap is forming a Tree, the virtual Sector forms the typical inner end of the sub-networks of Sectors in the SecMap.

The existence of a network of Sectors and their Friends allows to do fine-grained filtering combinations of SpacePoints by storing Filter cuts individually for each Sector combination. Filters are simple tools, which are used to filter SpacePoint combinations lying on compatible Sectors. In the context of the VXDTF they occur as two-hit Filters as described in Section 4.3.3.2, three-hit Filters (Section 4.3.3.3) and four-hit Filters (Section 4.3.3.4). A typical example of a *two-hit* Filter is "Distance3D", which measures the distance between two SpacePoints and neglects the combination if they are too near or too far away from each other. The effectiveness of such a Filter is strongly depending on the correct choice of *when* hit combinations have to be neglected. Therefore is a common cut not very effective for such Filters on all SpacePoint combinations and would only work for simple geometries like equidistant sensor disks as used in bigger experiments like the forward and backward regions of the upcoming CMS[60] upgrade or the ILD[62] detector.

[60] CMS: Compact muon solenoid at the LHC[61].
[62] ILD: International linear detector

For the case of the comparably small Belle II VXD, where a complex windmill design with slanted parts and 5 different sensor types is used, specialized cuts for different regions of the detector are highly recommended. The SecMap therefore is capable of storing such cuts individually for each Sector combination and therefore provides such specialized cuts. This is why the SecMap is not only used to prevent unrealistic SpacePoint combinations due to their geometrical distances, it is also used for storing Filter cuts dependent on the Sector combinations. Having several SecMaps additionally allows to focus on different cases like the reconstruction of curling tracks or a single SecMap for positively charged particles only.

Having Sector combination dependent cuts and pass dependent settings is of course also valid for the virtual IP and its virtual Sector. For each pass such a virtual hit on the virtual Sector can be added and is then treated as a normal hit, while the virtual Sector is connected to its outer Sectors.

The following section describes how Sector-Friend relations are found and how Sector combination specific cuts for Filters are determined.

4.3.2 Training

The SVD has a total number of 172 sensors arranged in a windmill design with slanted extensions in the forward region. It is built of three different sensor types, of which two are rectangular and one has a trapezoidal shape. The four layers have individual configurations regarding the number of ladders and the number of sensors per ladder. The outer three layers are extended into the forward region using slanted sensors, which are the trapezoidal ones mentioned above. Each ladder of the SVD has a region of overlap with its adjacent ladders, leading to very small distances between sensors of neighboring ladders of about 1 mm and more, while neighboring layers can be as far away as 40 mm or as near as 20 mm. The number of sensors per ladder is layer specific, as is the number of ladders per layer and the percentage of overlapping sensors. Together with the PXD, which consists of 40 sensors in two layers with two different sensor types, the VXD forms a complex geometrical structure, that strictly forbids to find Filter cuts or even which Friends are connected to which Sector by hand. Therefore an automated training of the settings of the SecMap was implemented in the basf2 framework.

The task of the SecMap training is as follows:

- Determine the compatibility list (the list of Friends) for each Sector in the SecMap.
- Determine the cuts for all Filters for each Filter type and Sector combination.

These tasks are achieved by using MC simulated events of $\Upsilon(4S)$ decays, the main type of events to be expected. These events result in about 10 tracks per event, distributed in a momentum range of below 50 MeV/c up to about 3 GeV/c, which are described in close detail in Chapter 3. The input for the training are the tracks of such events, formed by a MC based TF. The SecMap trainer then analyzes each track of each event given

and attaches the track hits to their Sectors accordingly. One special treatment has to be mentioned for the virtual hit, which is not part of the actual track, but is added during the training process to allow the trainer to consider its results in the SecMap stored in the end. The neighboring hits of the same tracks then define which Sectors are compatible: only Sector combinations having neighboring hits of the same track are allowed to be marked as Friends. This automatically only allows physically possible SpacePoint combinations to be checked in the VXDTF. For the trainer it does not matter if the Sector combination lies on the same layer, on neighboring layers or on two layers with another one in between, it only checks if neighboring hits of the same track are not on the same sensor, while curling tracks are split in ingoing and outgoing arms before being fed into the trainer. That approach automatically considers specialties in the layer setup, like gaps between neighboring sensors, missing or tilted sensors. This concept is not working, however, if particles are to be tracked which do not come from the same point in space, e.g. the IP. For example tracking daughter particles of K_{short}s can not easily found this way, since the K_{short} decays far off the IP. The restrictions are coming from the prerequisite of the SecMap to be a Tree, which expects a common inner end for all particles in all events. Consequently tracking for cosmic particles can not be done by using this SecMap design. Additionally wire-based trackers like the CDC request other approaches for tracking, while the environment formed by the VXD when analyzing $\Upsilon(4S)$ events are ideally covered by the SecMap approach.

Right after the neighboring hits of the same track have been used to find compatible Sectors, the same hits are then used to determine the cuts of all Filters to be trained. This means that for each Sector combination and each Filter all calculated values occurred are collected for a final analysis after the collecting run. This results in a distribution of Filter values for each Sector combination and Filter in the end, which then are sorted by their value. Then a minimum and a maximum cut — typically this is the 0.1% and the 99.9% quantile of the distribution — is determined by analyzing the sample. In the same step when min and max cuts are determined, the Sector combinations are checked for their relative frequencies. In detail this means: for each Sector its total frequency is determined from the collected sample, then the relative frequency for each Friend of that Sector is measured. The combinations occurring only in rare cases — the standard setting used is in less than 0.5% of the cases — are then neglected.

Although many values stored in the SecMap are found by the automated training described above, the training has yet to be optimized. In the current implementation still many parameters have to be chosen by hand, a situation that should be changed to increase the overall performance of the VXDTF. Among the manually set parameters is the momentum range of particles covered by a single SecMap, the numbers and size of the Sectors per sensor (one setting is currently used for all sensors), and the choice of the training sample. Additionally chosen by hand is which set of Filters are actually used and which quantiles of the training sample shall be stored as minimum and maximum cuts for the Filters in the Sector combinations. The choice and values of these parameters are crucial for the VXDTF performance and have a severe impact on the time consumption

of the TF and therefore should be determined with Machine Learning[63] or comparably performing techniques

Something that has not yet been tested, but should be possible in the SecMap approach, is to train a SecMap for finding curlers, especially low-momentum ones that curl inside the VXD volume and therefore do not create enough hits for each out- and ingoing track arm individually. While the concept of the SecMap itself does support such a network, the CA, the algorithm used for the hit chaining is in its current implementation not compatible with loops in the network, which undoubtedly would occur when passing curlers to the SecMap trainer. This therefore represents another task, being a possible follow-up project of this one.

4.3.3 Deployment

This section describes how a trained SecMap is applied during the reconstruction of an event. Additionally a full list of Filters, which is implemented in the VXDTF package, can be found in the following subsections. First one has to define what a Filter is: a Filter in the context of this TF does not filter hits directly, but filters hit combinations. The reason for this is simple: the same hit can be relevant for several hit chains and therefore only the combinations are filtered to prevent the TF to throw away valid hits before all possible combinations have been checked. In the end of each Filter level the possible numbers hit combinations for the following Filter levels have been reduced. Of course single hits can drop out of the filtering process that way too, if all possible combinations of these hits have been discarded.

Of particular importance again is the virtual hit at the virtual IP, which is treated as a normal hit for each Filter type applied.

The combinatorial issue manifests itself by the step of combining hits to hit chains. Therefore Filters are applied together with other Filters of the same complexity level. These steps, where a cluster of Filters is applied, are executed with an order of increasing complexity. This is why our first Filter step is a one-hit Filter, which is not sensitive to the combinatorial issue at all.

4.3.3.1 One-Hit filter

One-hit Filters can be applied on single hits and are therefore very good for reducing the total combinatorial problem since this step scales linearly with the number of hits. The SecMap itself already is a one-hit Filter, since it uses the sorting of single hits into the correct Sector to filter possible hit combinations. The number of hits filtered this way is small, but the possible number of combinations drastically drops by restricting possible combinations of hits to those lying in compatible Sectors.

The basic idea is to have a previously trained SecMap which stores the geometrical compatibility of neighboring Sectors, which are a subdivision of sensors of the tracking

[63] Automated training of parameters for algorithms using multivariate analysis.

detector. This network of Sectors therefore dictates which hits can be combined since only neighboring Sectors are allowed to be tested for compatible hits later-on. This results in 3 different cases possible for a hit after this Filter stage:

1. There was no compatible Sector found for given hit: hit is neglected
2. There was a compatible Sector, but that Sector has no hits in its Friends: hit is neglected
3. There was a compatible Sector and at least one of its Friends got hits too: hit is accepted.

Using an approach where no SecMap is included, this would mean that one has to combine each hit with any other hit "near" of him (e.g neighboring layers, the same layers if overlapping parts exist or even next-to-neighboring layers if one has to consider broken sensors). Even if the tests for a pair of hits are fast (like Distance3D), the combinatorial issue grows with the second power of the number of hits and therefore can be a bottleneck for reconstruction. Figure 4.4 illustrates the principle of the one-hit Filter.

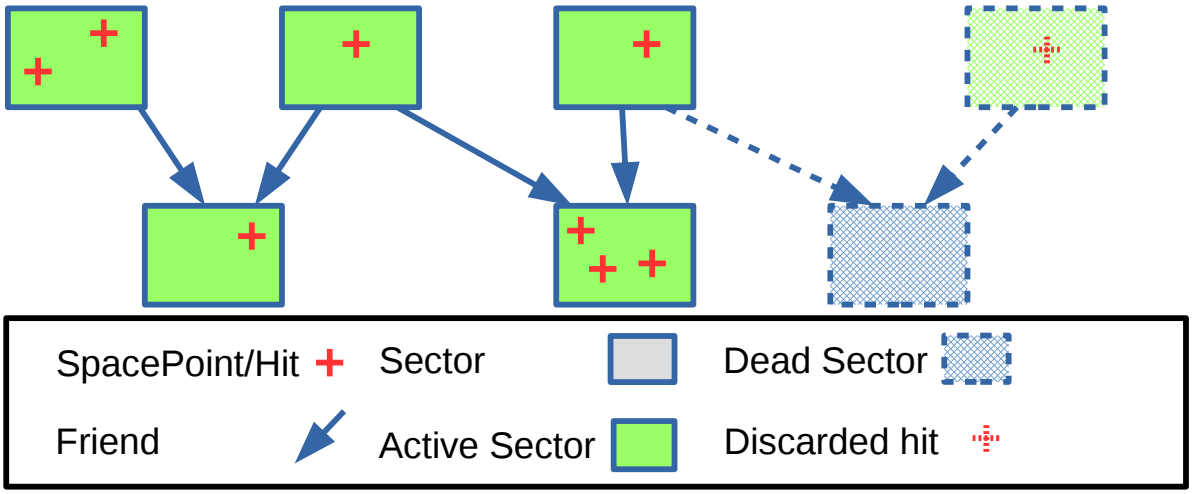


Figure 4.4: After sorting the hits into the SecMap, only the Sectors which inhabit hits are active and therefore only a sub-network of the SecMap is relevant in each event, respectively. But only hits in active Sectors which are actually connected to other active Sectors are considered in following steps, additionally only hits in a Sector-Friend Sector combination are tested for valid combinations.

During each event for each SecMap, the sorting of hits into the current SecMap is done independently. This forms different networks for different SecMaps which then are the base for the following Filter steps.

The advantage of that approach is obvious: using a SecMap trained on high momentum data in the first iteration, allows to get a network of hits, which favors straight TCs.

Since the best TCs found in an iteration can reserve their hits and therefore prevent them to be re-used in the following iteration, this reduces the combinatorial issue for subsequent iterations, where looser hit combinations are allowed to include tracks with stronger curvature and therefore represent a bigger challenge in combinatorial aspects.

4.3.3.2 Two-hit filters

Two-hit Filters are similar to the ones used in CA algorithms like those found in [35]; they have been extended in the VXDTF implementation. They build on the information collected by sorting the SpacePoints into the SecMap. Therefore only SpacePoints, which lie in compatible Sectors, are tested. Figure 4.5 illustrates the principle of the two-hit Filter. The Filters are applied as follows: for a given pair of hits a parameter is calculated,

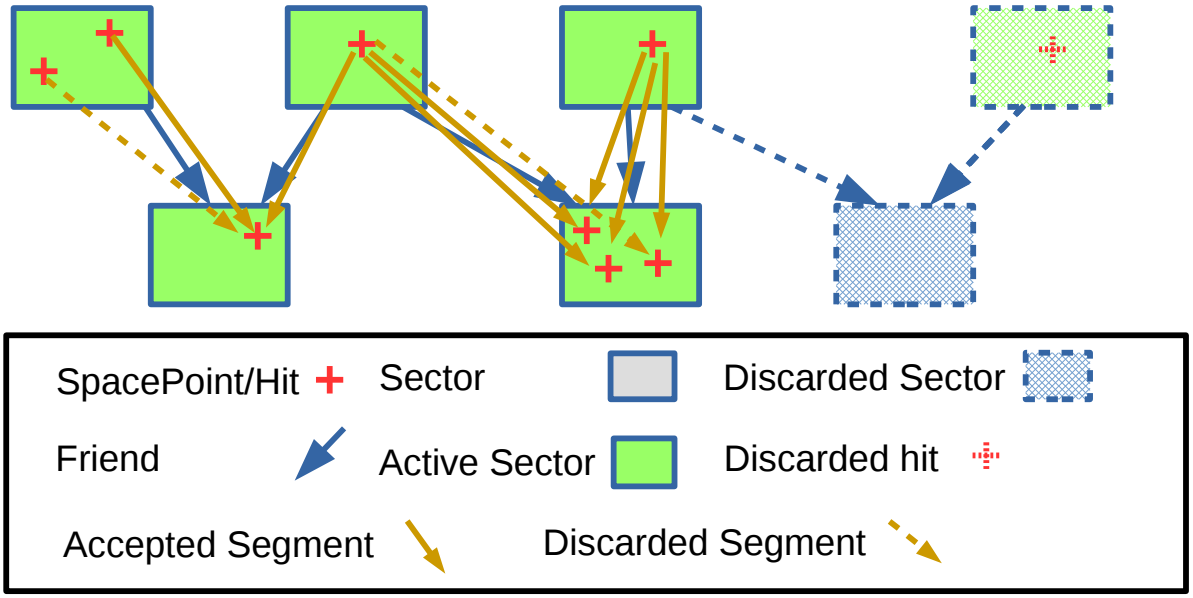


Figure 4.5: Only active Sectors with Friends being active Sectors two are checked for compatible SpacePoint combinations. Compatible SpacePoint combinations, which are accepted if all Filters accepted the combination, are stored as Segments, whereas incompatible ones are discarded.

which is then compared with min and max cuts stored for the attached Sectors of the hits in the SecMap. This means that for each Sector combination in which hits are stored the cuts are different. The calculated parameter is depending on the Filter used. The formulas used for the Filters implemented can be found in Table 4.1. The names of the Filters indicate what they check for but the implemented formulas are simplified for faster computation speed and therefore are not calculating the expected results indicated in the Filters' names. For example the distance Filters do actually calculate the squared distance to get rid of the time consuming calculation of the root. In the case of the

Table 4.1: All the two-hit Filters implemented in VXDTF are described in this table. The exact formulas used can be found in the appendix in Section 9.1 and are additionally mentioned in the table. $f_{...}$ represents the result of the formula of .... The domain describes the range where the Filters described are producing valid results.

Name	formula	description
distXY	see Eq. (9.1)	distance between two hits in the x-y plane, $f_{XY} \in [0; \infty[$
dist3D	see Eq. (9.2)	distance between two hits in 3D, $f_{3D} \in [0; \infty[$
distZ	see Eq. (9.3)	the signed distance between two hits in z, $f_Z \in]-\infty; \infty[$
slopeRZ	see Eq. (9.4)	the angle of the slope of the hits in the rz plane $\widehat{=} \theta$, $f_{sRZ} \in \{[0; \pi[\setminus z_1 - z_2 = 0\}$
nDist3D	see Eq. (9.5)	normed distance between two hits, $f_{n3D} \in \{[0; 1] \setminus f_{3D} = 0\}$

VXDTF a two-hit combination which was accepted by all Filters activated for the given pass, is stored as a two-hit Segment. This two-hit Segment is then treated as a Cell by the CA which explains the term and is itself described in Section 4.4. How the two-hit Filters perform in realistic environments is described in Chapter 5.

4.3.3.3 Three-hit filters

All pairs of accepted two-hit combinations that share a middle hit — and thus form a three-hit combination — are then checked by the three-hit Filters. The procedure is similar to the two-hit Filters approach: the Sectors, on which the hits are lying, define the cuts that are used by the respective Filters. The SecMap design of the VXDTF cannot store the cuts for three-Sector combinations, but only for two-Sector combinations. This reduces the effectiveness of the three-hit Filters a bit, since only the *outer* two hits lying in the outer two Sectors of a three-Sector combination define a cut stored.

Compared to the number of two-hit Filters the three-hit Filters are more numerous, as can be seen in Table 4.2. The Filters do not actually calculate what their names indicate but do a simplified (and therefore non-equivalent) version of the indicated result. The descriptions in the table try to describe the actual meaning if possible.

Their performance in realistic conditions is described in Chapter 5. Due to the pre-selection of possible three-hit combinations by the two-hit Filters, the effect of increased complexity and therefore the increased computation time is reduced. Like storing accepted two-hit combinations as Segments or Cells in the context of two-hit Filters, the compatibility of two connected two-hit combinations as one accepted three-hit combination is stored as a Cell Neighbour relationship in the network of created Cells. This two-hit Segment is then treated as a Cell by the CA which explains the term and is itself described in Section 4.4.

Table 4.2: All the three-hit Filters implemented in VXDTF are described in this table. The exact formulas used can be found in the appendix in Section 9.2 and are additionally mentioned in the table. $f_{...}$ represents the result of the formula of ...; h_i represents hit i. The domain describes the range where the Filters described are producing valid results.

Name	formula	description
angleXY	see Eq. (9.6)	in xy: angle between outer ($\widehat{=}$ outer and center hit) and inner ($\widehat{=}$ center and inner hit) Segment, $f_{aXY} \in [0;\infty[\setminus \{h_i = h_j\}$
angle3D	see Eq. (9.7)	in 3D: angle between outer and inner Segment, $f_{a3D} \in [0;\infty[\setminus \{h_i = h_j\}$
angleRZ	see Eq. (9.8)	in rz: angle between outer and inner Segment, $f_{aRZ} \in [0;\infty[\setminus \{h_i = h_j\}$
circleDist2IP	see Eq. (9.9)	the circle center and radius of the hits in the x-y plane is determined and the distance from the origin to the P.O.C.A. computed
pT	see Eq. (9.10)	transverse momentum of a hypothetical particle creating the hits, using c as the speed of light in S.I. units, $\left\|\vec{B}\right\|$ the magnetic field in Tesla, $f_{p_\top} \in [0;\infty[$
deltaSlopeRZ	see Eq. (9.11)	difference between the slope angle θ in rz of the outer Segment and the inner one. $f_{dsRZ} \in \{[0;\pi[\setminus (z_1 - z_2 = 0) \vee (z_2 - z_3 = 0)\}$
deltaSoverZ	see Eq. (9.14)	helix parameter describing the deviation in arc length per unit in z
deltaSlopeSoverZ	see Eq. (9.15)	compares the "slopes" z over arc length. deltaSlopeSoverZ is invariant under rotations in the rz plane
helixParameterFit	see Eq. (9.16)	calculates the helix parameter describing the deviation in z per unit angle

The list of Filters described in Table 4.2 only check for geometrical specifics of SpacePoints in three dimensions. An attempt of how machine learned Filters perform for the case of the VXDTF when using a nine dimensional parameter space can be read in [36].

4.3.3.4 Four-hit filters

The four-hit Filters have got a special standing among the Filters since they are not applied in the same way as the two and three-hit Filters. Although they store the cuts again depending on the Sector combinations of the hits of interest, they are not applied

Table 4.3: All the four-hit Filters implemented in VXDTF are described in this table. The exact formulas used can be found in the appendix in Section 9.3 and are additionally mentioned in the table. $f_{...}$ represents the result of the formula of ...; h_i represents hit i. The domain describes the range where the Filters described are producing valid results.

Name	formula	description
deltaPt	see Eq. (9.17)	difference of $p_\top$ of h_1, h_2, h_3 and $p_\top$ of h_2, h_3, h_4, $f_{dp_\top} \in [0; \infty[$
deltaDistCircleCenter	see Eq. (9.18)	the distance between the circle center of h_1, h_2, h_3 and the circle center of the h_2, h_3, h_4, $f_{ddCC} \in [0; \infty[$

until preliminary TCs have been generated by the CA section. This means that they are applied on the TCs instead. Since TCs collected by the VXDTF can have any length of three or more hits, the four-hit Filters are not applied to three-hit TCs. For TCs having more than three hits they are applied once for each consecutive hit triplet. A list of the four-hit Filters implemented can be found in Table 4.3. Among the four-hit Filters there are also some Filters which do not depend on cuts at all, which is why they are not listed in the table mentioned above. These Filters are "ziggZaggXY" and "ziggZaggRZ", which only check if the TCs are zigzagging in the x-y plane or in the rz plane. This is checked by using two combinations of three consecutive hits of a TC, where two of the hits are shared for both combinations. If the sign of the curvature of these combinations differ, the TC is ziggzagging, and will therefore be neglected. Only in low-momentum passes the Filter for ziggzagging in rz is applied, since the magnetic field is bending tracks far less prominently in rz than in xy.

Again a specialty of the VXDTF design has to be mentioned: like the three-hit Filters the current design can only store cuts up to two Sector chains and therefore only the two outermost hits determine which cuts to be used.

For the VXDTF there are no Filters using five or more hits, since the need of such Filters is very limited in a four-layer detector geometry. Moreover, complex Filters would use χ^2-based cuts, and the SecMap would not be needed for checking the validity of such a hit combination.

4.3.3.5 High-occupancy mode

Next the concept of the high-occupancy mode is described here. The high occupancy mode is an extra Filter level, which can be applied after the two-hit Filters and after the three-hit Filters. The basic idea is connected to the virtual IP and its de facto boundary condition for the VXDTF. If one can assume that any track to be found has to come

from the origin, one can add the hit of the virtual IP to gain an "extra" hit not only on the overall level of extending the maximal chain length, but also to apply three-hit tests on any two-hit combination. This means that for any two-hit combination which has been accepted as valid, a three-hit test, where the virtual hit is added as a third and innermost hit to the accepted pair of hits, can be checked too. The SecMap stores extra cuts for that mode, but uses the normal Filter algorithms of the three-hit Filters for that step. Therefore a two-hit combination plus the virtual hit can be checked for having a valid p_{T} estimation, which is one of the three-hit Filters listed in Section 4.3.3.3.

The same idea of course can also be used on three-hit combinations. In this case the virtual hit is then added to a three-hit combination, which allows to deploy four-hit Filter tests.

Of course the information gain in the high occupancy mode is not as high as having an actual three or four-hit combination for a three or four-hit Filter, but again this allows to filter combinations more effectively than simple two and three-hit Filters and therefore reduces the combinatorial issue in cases where it is needed most, namely in high occupancy cases, hence the name. On the downside it also slows down the reconstruction process when the occupancy is low and therefore it is not advisable to use them all the time. But for high occupancy cases this approach does help to keep the number of preliminary TCs down.

4.4 Collecting Track Candidates

After discarding numerous two- and three-hit combinations, the accepted combinations form a network of compatible hits, which is a directed graphs without loops and therefore a Tree. This network is then used for finding TCs which can be simply collected by starting at all the outermost nodes of this network and collecting all possible paths to the innermost nodes of the network as individual TCs. This straightforward approach has the disadvantage that the information of three-hit Filters is neglected: a hit only knows its next inner hits and not what is attached to its inner hits. The reason for this is simple: assume one center hit which has several outer and inner hits, which have been approved by the two-hit Filter. Applying the three-hit Filter leads to the situation that still all two-hit Segments could survive, but not all possible combinations of two-hit Segments do survive. Therefore the information which three-hit combination is valid and which is not, can not be stored without further adaptations. A way to adapt such a network is described in the following section.

4.4.1 Cellular automaton

The CA is a very versatile algorithm, which has been introduced to track finding by A. Glazov and I. Kisel about 25 years ago [27]. The Cells of the automaton are compatible track Segments that have passed the filters described in the previous subsection and are

stored as a directed graph. Each Cell is assigned a state $s \geq 0$ which is a nonnegative integer and initialized to zero. The Cell states are updated by the following iterative procedure:

- Check step: Each Cell is checked independently if there is at least one inner connected with the same state. If yes, mark the Cell to be updated.
- Update step: for each Cell marked for update the state is incremented by one.

Figure 4.6 illustrates the general principle of the CA.

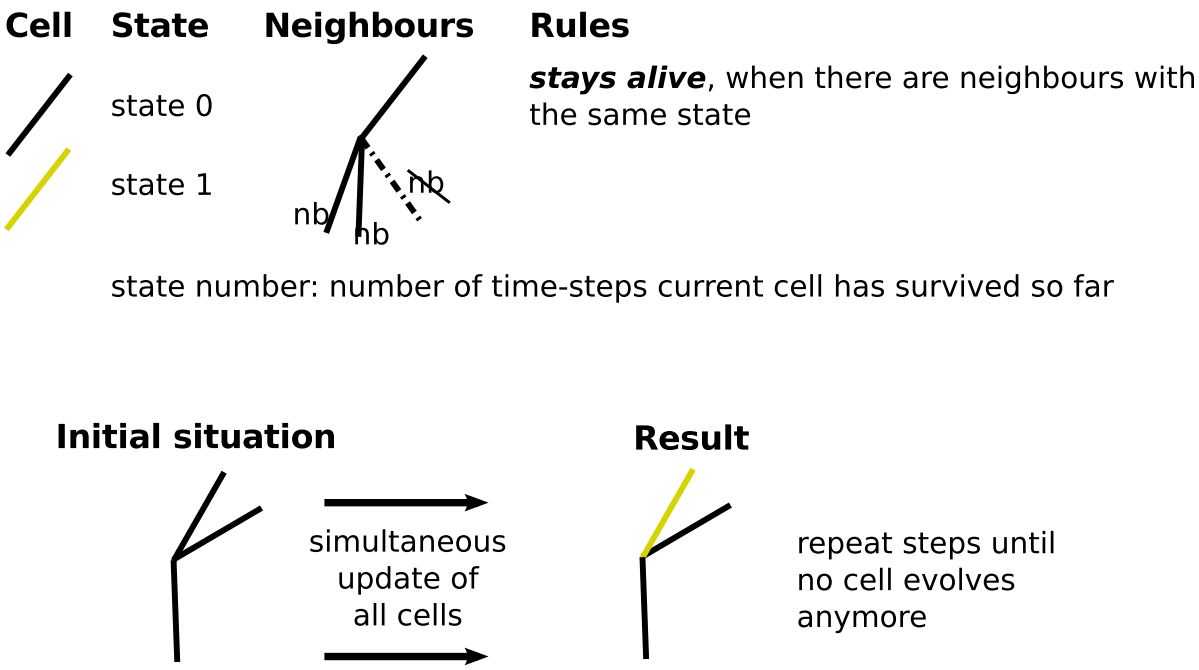


Figure 4.6: Summary of the basic aspects of a CA used for track finding.

The two steps are repeated until a stable state has been found for all Cells in the network. The final value state of a Cell encodes the the length of the longest chain of compatible inner Cells that is connected to it. Higher states indicate longer chains, which are less likely to be random combinations of hits. This is why the SecMap is designed for allowing overlapping sensors in the same layer and and introduces the IP as an extra virtual hit, thereby increasing the largest possible Cell state. The difference between a basic implementation where layer overlaps and virtual IPs are not considered and the VXDTF implementation of the CA is illustrated in Figure 4.7.

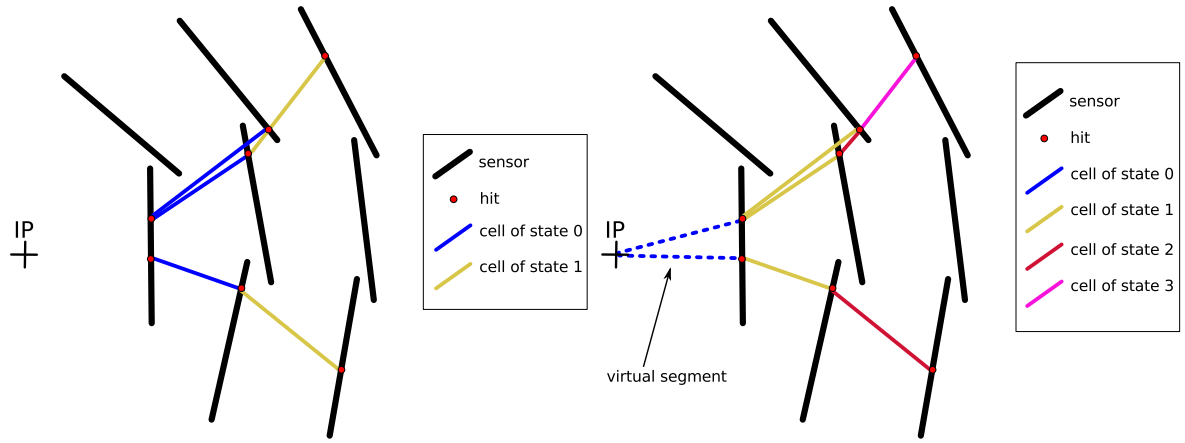
As already indicated, the VXDTF applies the Filters before the CA, which is normally combined to one step, where the two-hit Filters build the Cells, the three-hit Filters then

define the Neighbours and therefore the full network in the first time step. then the following time steps are not using any Filters but simply apply the CA algorithm. In the case of the VXDTF the network is built before, which is simply a design decision without major impact on the general working principle.

There are other implementations of the CA used for other experiments, e.g. [27], [35], [37] and [38] where different definitions of Cells are used. Very popular in more modern implementations is to define the Cells of a CA in a more flexible way. While the VXDTF uses only two-hit Segments as Cells, e.g., the implementation of the CA in [37] uses the CA several times with different Cell types, starting with one-hit Cells up to many-hit Cells. The reason why this was not implemented in the VXDTF as well, is the small number of layers of the SVD. Cells having more than two hits can implicitly store information of Filters with more than three hits and therefore are to be preferred for the case of high number of layers. If the shortest TCs to be found can have three SpacePoints, which is the case in the VXDTF, such Cells do not result in better samples of TCs. In this case this only leads to more overhead, since three-hit Cells need to share more than one hit, and therefore carry a considerable overlap, to be able to consider all possible hit combinations without neglecting ones.

4.4.2 Track Candidate collector

The CA described in the last section does not collect TCs, but just marks long chains of compatible Segments and their corresponding hits. For collecting TCs a recursive function was implemented, which collects all chains of compatible hits for all Cells lying at or above the threshold state of 2. Using a minimal state of two results in minimal TC lengths of three hits and the virtual IP or four hits for TCs which are not connected to



(a) Basic approach. (b) VXDTF approach.

Figure 4.7: Difference between a basic implementation of the CA (left) and the VXDTF version (right) for the case of three layers (side view).

the IP. The latter case does occur sometimes e.g. for tracks of decay products of K_S^0 particles.

The principle of the Track Candidate Collector is simple: loop over all Cells, and if the current Cell has a state $x \geq y$, where $y \mathrel{\widehat{=}}$ the threshold value, then start collecting additional Cells (and with it their hits) by adding the next inner Neighbour with a state of $x - 1$. If there is more than one Neighbour fulfilling that requirement, then the chain collected so far is copied and added to the stack for later propagation. This procedure is repeated for the Neighbour too until the chain comes to an end and the TC is completed and the next chain from the stack is taken or the next Cell with sufficient state value is found. Following this process it is common for longer chains (>3 hits) that their sub-chains are collected too. This is done on purpose, as one of the hits might be an outlier, which can be identified by the quality estimation step applied after collecting the TC.

In the implementation of the VXDTF the preliminary TCs are checked using four-hit Filters described in Section 4.3.3.4, to filter obvious bad candidates before passing the preliminary set of TCs for the more complex and time consuming fits available for quality estimation. They are described in the following section.

4.5 Quality estimation

The implementation of the VXDTF contains several methods to determine the quality of the preliminary TCs collected by the Track Candidate Collector. Some of them can estimate the *momentum seed* of the track, which is a prerequisite to find ROIs in the PXD. The momentum seed is a momentum vector in three dimensions (p_x, p_y and p_z), in units of GeV/c. It is estimated at the position of the innermost hit of the TC and is stored as a parameter for every TC.

In this section all possible estimators are listed, although only some of them are actually used in current VXDTF setups. A common output of all estimators is a *quality indicator* (QI), which is a floating point number carrying a value in the range $[0; 1]$. A value of 0 indicates a very bad TC, while a value of 1 means a very good candidate. It serves as an essential input parameter for the subset finding algorithms discussed in Section 4.6.

4.5.1 Track length

The quality estimator using the track length was the first one implemented. It rates the TCs by their track length, where longer ones get better QIs as shorter ones. In this implementation a QI of 1 can only be achieved for tracks with 8 hits, which is the maximum length of a TC due to the geometry of the SVD. This estimator was implemented for testing purposes only and was never used in productive environments.

4.5.2 Straight line fit

The straight line fit was implemented for the combined beam test at DESY in 2014. More details about the beam test can be found in Chapter 6. It is needed for quality estimation in runs without magnetic field. In case the fit can also be used to get a valid momentum seed for the TCs, but only with a pre-determined $|\vec{p}|$ value, since this value can not be estimated by the VXDTF without the presence of a magnetic field. It is a weighted least-squares fit that uses the hit errors to determine the weights of the positions. The fit estimates the slope and the intercept of a given track from hits in two planes, as listed in equation 4.1:

$$\begin{aligned} Y_i &= k_{xy} \cdot X_i + d_y \\ Z_i &= k_{xz} \cdot X_i + d_z \end{aligned}$$

$$\begin{aligned} k_{xy} &\mathrel{\widehat{=}} \text{slope in } x-y \text{ plane} \\ d_y &\mathrel{\widehat{=}} \text{intercept in } y \text{ axis} \\ k_{xz} &\mathrel{\widehat{=}} \text{slope in } x-z \text{ plane} \\ d_z &\mathrel{\widehat{=}} \text{intercept in } z \text{ axis} \end{aligned} \tag{4.1}$$

The parameters were chosen according to the sensor arrangement in the beam test, which will be described in Section 6.2. A schematic view can be found in Figure 6.1. The choice of the fit parameters allows to use the errors of the SpacePoint in local u ($\mathrel{\widehat{=}} y$) and v ($\mathrel{\widehat{=}} z$) without transformation and therefore the y and z hits as the dependent variables. x is treated as a quantity without errors, which in this case is not measured but assumed to be exactly known by choice of the geometrical setup described in Section 6.2. The χ^2 is then obtained using the estimated values for k_{xy}, d_y, k_{xz} and d_z:

$$\chi^2 = \sum_{1 \le i \le n_{\text{hits}}} \left(\left(\frac{Y_i - k_{xy} \cdot X_i - d_y}{\sigma_{Y_i}} \right)^2 + \left(\frac{Z_i - k_{xz} \cdot X_i - d_z}{\sigma_{Z_i}} \right)^2 \right) \tag{4.2}$$

The χ^2 then is used to determine the value p of the cumulative distribution function of the χ^2 distribution with the correct number of degrees of freedom (DOF[64]), which is $2 \cdot n_{\text{hits}} - n_{\text{parameters}}$. The four estimated parameters are the slopes and intercepts mentioned above; therefore the fit has two DOF for the case of three hits, and two additional DOF for every further hit in the track. p is the probability that for a straight line the value for the χ^2 will be lower than the one obtained. p is then used to calculate a QI for the TC just checked.

Although the fit is correctly implemented, it cannot be used for the Belle II VXD geometry, since there the slope parameter can become ∞ for tracks going straight up, which is not possible in the geometry setup used in the combined beam test of 2014.

[64] Degrees Of Freedom

When $k_{xy} \longrightarrow \infty$ the fit aborts the program. This means that for a full Belle II geometry setup this fit can not be used and therefore the fit should be replaced by a version which includes well-behaved fit parameters.

4.5.3 Circle fit

The circle fit implemented for the VXDTF is taken from [39], where a non-iterative fit for the following parameters is applied (Eq. 4.3).

$$\begin{aligned}
\phi &= \frac{1}{2}\arctan\frac{2(C_{r^2r^2}C_{xy} - C_{xr^2}C_{yr^2})}{C_{r^2r^2}(C_{xx} - C_{yy}) - C_{xr^2}^2 + C_{yr^2}^2} \\
\rho &= \frac{2\kappa}{\sqrt{1-4\delta\kappa}} \\
d &= \frac{2\delta}{1+\sqrt{1-4\delta\kappa}}
\end{aligned}$$

$$\begin{aligned}
\phi &\mathrel{\widehat{=}} \text{the } \phi \text{ value of the P.O.C.A. of the fitted circle to the origin} \\
\rho &\mathrel{\widehat{=}} \text{the signed curvature } (1/r) \text{ of the fitted circle in the } x\text{-}y \text{ plane.} \\
&\quad \text{It is positive for particles propagating along the circle clockwise} \\
&\quad \text{and negative for counter clockwise propagation.} \\
d &\mathrel{\widehat{=}} \text{the distance of the P.O.C.A. to the origin, which is known as the} \\
&\quad \textit{impact parameter}, \text{ which in this case is signed if the vector pointing} \\
&\quad \text{from the origin to the P.O.C.A. and the track direction form a} \\
&\quad \text{right-handed system and is negative if otherwise.}
\end{aligned} \tag{4.3}$$

The covariances for the measurements x_i, y_i and $r_i^2 = x_i^2 + y_i^2$ written as $C_{xx}, \ldots$ and the helper variables κ and δ are defined as follows (Eq. 4.4):

$$\begin{aligned}
C_{xx} &\mathrel{\widehat{=}} < x^2 > - < x >^2 \\
C_{xy} &\mathrel{\widehat{=}} < xy > - < x >< y > \\
C_{yy} &\mathrel{\widehat{=}} < y^2 > - < y >^2 \\
C_{xr^2} &\mathrel{\widehat{=}} < xr^2 > - < x >< r^2 > \\
C_{yr^2} &\mathrel{\widehat{=}} < yr^2 > - < y >< r^2 > \\
C_{r^2r^2} &\mathrel{\widehat{=}} < r^4 > - < r^2 >^2 \\
\kappa &\mathrel{\widehat{=}} \frac{(\sin(\phi)C_{xr^2} - \cos(\phi)C_{yr^2})}{C_{r^2r^2}} \\
\delta &\mathrel{\widehat{=}} -\kappa < r^2 > + \sin(\phi) < x > - \cos(\phi) < y >
\end{aligned} \tag{4.4}$$

Next to being very fast, this fit behaves well for the straight line case. This fit allows to calculate the p_x and p_y components for the momentum seed, while the p_z components have to be calculated separately. To retrieve the χ^2 the following formula (Eq. 4.5) is deployed:

$$\chi^2 = \left(\sum_i w_i\right)(1+\rho d)^2(\sin^2 C_{xx} - 2\sin(\phi)\cos(\phi) + \cos^2(\phi)C_{yy} - \kappa^2 C r^2 r^2) \tag{4.5}$$

$$w_i \mathrel{\widehat{=}} \frac{1}{\sigma_{x_i}^2 + \sigma_{y_i}^2} \mathrel{\widehat{=}} \text{ weight of the } i^{th} \text{ hit}$$

Although the z information is ignored, the circle fit is used as standard quality estimator for the VXDTF because of its stability for ill-fit data and its high computation speed. For the momentum seed it is not used since the z information is not considered ideally.

4.5.4 Helix fit

The helix fit used in the VXDTF is taken from [40] and combines a circle fit using a Riemann sphere with a straight line fit. It is slower and less stable for ill-fit input but delivers better momentum seeds as the Karimäki circle fit presented in the section above.

Due to its instability with "bad" TCs and some coding issues, this fit is not used for QI estimation, but only for creating the momentum seeds of the final TCs. The number of final TCs is always very small — more details on that can be found in Section 5.4.4 — and therefore the slower execution speed compared to the Karimäki fit discussed in the preceding section is not relevant.

4.5.5 Kalman filter

The Kalman filter (KF) was first introduced to high energy physics in [28] and has become one of the standard tools for track fitting. It is a statistically optimal fitter and mathematically equivalent to the least-squares fit. A global track fit requires the inversion of a potentially large matrix, which is very time consuming. On contrast, the KF relies on a track following approach, which inverts a smaller matrix whenever a hit is added. Its time consumption scales linearly with the number of hits. This approach can additionally consider effects of multiple scattering on the track, while energy loss has to be approximated by a Gaussian model if it is to be included in the filter.

The VXDTF implementation does not contain its own KF, but uses a standardized interface to the GENFIT2 package [33] that is used for fitting in the basf2 framework.

Although KF of GENFIT2 is the best fitter adapted for usage in the VXDTF, it is too slow for using it on all the preliminary TCs. This emerges from the fact that GENFIT2 is not optimized for speed since its focus lies on fitting in an off-line environment. However,

optimization of the GENFIT2 package is ongoing, and it is therefore possible that it will be used as standard again sometime in the future.

4.6 Subset finding

Compared to the CDC, in the VXD the probability is very small that two tracks actually share a cluster. Therefore a requirement for a final set of TCs is that no cluster is shared by two or more TCs. To verify this, a dedicated function finds all TCs sharing clusters with other TCs and informs them of their overlapping status. In the end each TC knows with which other TCs it shares clusters. Since the execution time of this procedure is very sensitive to the total number of TCs, it is very important to have an efficient implementation. The version currently used in the VXDTF is not optimized for speed yet, which will be discussed in Section 5.4.3.

Finding overlapping TCs is essential for the application of algorithms which then can find a good subset of the TCs which are not overlapping. The VXDTF provides at the moment two ways to clean the TC set, the straightforward approach called the greedy algorithm, discussed in Section 4.6.1, and a more sophisticated one, described in Section 4.6.2.

4.6.1 Greedy algorithm

The greedy algorithm is a straightforward approach to get rid of overlapping TCs. It is executed as follows:

1. Sort all overlapping TCs by descending QI, so that the TC with the best QI is on top of the list.
2. Kill all TCs overlapping with current one.
3. Descend down the list until another TC is found which still has overlapping TCs and is not dead itself. If the end of the list is found, stop procedure.
4. Goto step 2.

The algorithm is fast, although it is sensitive to combinatorial effects. But its efficiency is so much worse than the algorithm presented in the following section, that the greedy algorithm is not used in any standard setting.

4.6.2 Hopfield Neural Network

The HNN[65] is a feed-back neural network without hidden layers and belongs to the group of adaptive methods [25] used in in track reconstruction. The neural network is used to

[65] HNN: Hopfield Neural Network

solve a global optimization problem by selecting optimal subsets of non-overlapping TCs. The version used for the VXDTF is an implementation from [41], which will be sketched here describing the main characteristics.

The situation of overlapping TCs can be mapped on a graph, where each TC represents a node and overlapping TCs are connected by an edge. Nodes connected this way are incompatible, while nodes without edge connecting them are compatible with each other.

The structure of the graph is a template for the neural network, where one neuron per overlapped TC is created and the compatibility between neurons is mapped with a weight matrix of $n \times n$ entries, where n is the number of nodes/neurons/TCs. The entry w_{ij} is then describing the weight between the i^{th} and the j^{th} neuron:

$$w_{ij} \begin{cases} -1, & \text{if } i \text{ and } j \text{ are } \textit{not} \text{ compatible,} \\ (1-\omega)/n, & \text{if } i \text{ and } j \text{ are compatible,} \end{cases} \tag{4.6}$$

The weights are used to steer the behavior of the HNN and are chosen to guarantee that the absolute value of a weight for an incompatible neuron connection is higher than for a compatible one. Additionally a tuning parameter $\omega \in [0;1]$ can be used to favor sets of TCs with the highest number of compatible TCs ($\omega \approx 0$), or sets with the best sum of QI ($\omega \approx 1$). In the setup used for the VXDTF the standard value for $\omega = 0.5$, which is a solid trade-off between the two cases.

In the beginning each neuron is assigned a random state drawn from a uniform distribution in the interval $[0;1]$. Then the HNN is run iteratively where in each round the states x_i of all neurons are updated in random order. The updated state is computed according to:

$$x_i = \frac{1}{2}(1 + \tanh(a_i/T_k)) \tag{4.7}$$

where a_i is the input to the neuron and T_k is an iteration dependent temperature parameter. The input a_i is a weighted sum of the states of all neurons plus a bias depending on the QIs:

$$a_i = \sum_{j \in n} w_{ij} x_j + \omega q_i \tag{4.8}$$

The temperature parameter T represents a mean-field approximation of an artificial thermal noise, which is used to prevent the overall state of the neural network to get stuck in a local minimum of the optimization problem at hand. The temperature parameter starts with an initial value used for this implementation of $T_{start} = 3.1$ and is then lowered in each iteration of the network using the formula for iteration k: $T_{k+1} =^{1} /_{2}(T_k + T_{min})$ with $T_{min} = 0.1$. This procedure is known as mean-field annealing and is also used in some other advanced track reconstruction algorithms.

The update procedure is asynchronous, so that the neurons are not updated simultaneously but sequentially, in order to prevent the neural network from oscillating between two global states (see [42]).

The updating algorithm is iterated by using the procedure mentioned above, until no neuron changes the value of its state by more than 0.01. All the neurons having a higher state than 0.75 are then rounded up to 1, meaning activated, while all others are being deactivated (state 0). In the end all TCs corresponding to activated neurons are then accepted as final and have no overlapping TCs being accepted as well. This behavior is guaranteed by the choice of the weights discussed above.

The performance of the algorithm is much better than the greedy algorithm described in the previous section and is therefore used as the standard method for obtaining a non-overlapping subset of TCs in the VXDTF.

4.7 Multiple passes

The VXDTF can run using a single pass per event, but to reduce the combinatorial issue, the typical setup executes three — mostly independent — passes consecutively, starting with the high momentum pass, where the combinatorial issue is the smallest. The idea behind this is to reduce the possible number of combinations for low-momentum checks, where wide areas of neighboring Sectors and therefore a high number of hits have to be considered in order not to lose any low-momentum tracks. The best TCs of the preceding passes can reserve their hits, which therefore do not have to compete again with the TCs of the current pass. In the end of each pass, the final TCs of the pass are sorted by QI and then the parameter "reserveHitsThreshold" is used for reserving the best x% of the TCs. This cut is pass dependent and typically set to 60%.

4.8 Outlook

The VXDTF is a feature-complete implementation of the chosen low-momentum tracking approach capable of reconstructing the full momentum range of particles produced by the SuperKEKB in the Belle II VXD. It has been used for several years in the official reconstruction chain of Belle II, including several preliminary MC studies preparing the physics group for the real data. Its performance is solid but not without flaws, which will be discussed in detail in Chapter 5. Its biggest flaw is the code design, which — although fulfilling its reconstruction task — it is not of "production quality", where it can be maintained and improved for the next 10–15 years. The code base of the VXDTF currently contains over 44,000 lines of code (not counting comments or empty lines) in C++, which is too much to be packed in a monolithic design. Therefore a redesign process was started two years ago, which unfortunately did not yet result in a track finder capable of replacing the VXDTF I. But the concept has been finished and first tests are very promising. The new design is much more versatile and allows for instance to replace the CA by a CKF by changing a few lines in the steering file, which has been made possible by switching to a highly modular design with standardized interfaces connecting

the different reconstruction steps in the track finder. Since the writer of this thesis will not be able to finish the rewrite of the VXDTF code, but several groups of the Belle II community shall take over, this thesis will serve as a design documentation of the current status of the old code — as has been sketched in this chapter — and of the redesigned version, which the reader can find in Chapter 7.

The VXDTF has served as a playground to test a lot of essential aspects of track finding in realistic conditions Chapter 5 and real setups Chapter 6. In addition, unexpected bottle necks could be identified, which will enhance the quality of its successor, the VXDTF 2.

CHAPTER 5

STUDY OF FINDING EFFICIENCY AND TIME CONSUMPTION

5.1 Overview

This chapter requires knowledge of Section 2.4.1, which describes the detector-specific aspects of the working environment; Chapter 4, which discusses the detailed structure and expected strengths and weaknesses of the VXDTF; Section 2.6, which explains the nature and origin of the background sources to be expected; and Chapter 3, which describes many essential aspects of the situation to be faced by the VXDTF.

The performance of the VXDTF is evaluated using MC data in a perfectly aligned and fully working detector, which is not identical, but close to the actual status of the Belle II experiment once it is operating. The magnetic field is set to a constant value of 1.5 T, which is close to the final value. First tests have shown that the deviations from the ideal value are at the per mill level. More details about the geometrical setup used can be found in Section 11.1. Although the event data is coming from MC, the events represent realistic $\Upsilon(4S)$ conditions, where results are shown for various expected background levels. Several simulation setups using different background levels mixed with $\Upsilon(4S)$ events have been used to form a useful picture of the performance of the VXDTF. There is one notable simplification of the simulation settings compared to a realistic setting: the range of the magnetic field was limited to the volume of the VXD, which prevents particles from curling back from the CDC into the VXD. More details about the VXDTF settings used for this study can be found in Section 11.2, Section 11.3 and Section 11.5.

The following Section 5.1.1 defines the most important markers that are needed to evaluate the performance of the VXDTF. In Section 5.2 the general behavior of the VXDTF in an $\Upsilon(4S)$ event environment without background is shown. The behavior of

the VXDTF for the case of background only events is illustrated in Section 5.3, and the case of $\Upsilon(4S)$ events with different background levels is discussed in Section 5.4. The impact of the VXDTF parameter "killEventsForHighOccupancy" — used for filtering high occupancy events — on efficiency and time consumption is discussed in Section 5.4.4.

5.1.1 Definition of important markers

Since every experiment has its own definitions of important markers, the ones used for the VXDTF are defined here.

The definition of the parameters efficiency and purity depends on the precise meaning of SpacePoints, clusters and TCs.

- A "cluster" is a detector dependent hit type. In the SVD a cluster is a 1D hit consisting of one or more strips of a sensor above the threshold of the energy deposition. It can lie on the u or the v side of the sensor and carries the local position on the sensor, a position error estimation, the total energy deposit and a hit time information with a resolution of 2 ns in the ideal case. It counts as a single measurement. In the PXD a cluster is a 2D hit consisting of one or more pixels having energy deposit above threshold. Like the SVD-cluster it stores the local position, the position error estimate and the total energy deposit, but lacks a usable hit time information because of the detector limitations in the ROF size of 20 µs. It counts as two measurements. More details about the SVD and the PXD can be found in Section 2.4.1.

- A SpacePoint is a detector independent hit type. It stores the hit position in global coordinates with an error matrix. The SpacePoint can be formed from a single PXD cluster or from a pair of u and v clusters in the same SVD sensor. In both cases it counts as two measurements.

- A TC consists of several clusters or SpacePoints — depending on the container type used — which are assumed to be created by the same original particle. The TC is found by a TF — in this case the VXDTF — and has no information about the MC truth. To check whether a TC is valid, a separate module is invoked after the track finding step. It reconstructs the MC information, which then is used to validate the VXDTF.

The studies presented here are based on MC data, which has to be used for a performance analysis since real data is not yet available. Not all MC tracks are actually used for the performance checks. To be part of the reference sample of MC tracks used in this chapter, a MC track needs to fulfill all of the following conditions:

- The MC track has to create at least six clusters ($\widehat{=}$ six measurements) in the SVD. This means only charged particles in the acceptance region of the SVD are considered.

- The MC track needs to be a *primary* particle, which essentially means all particles coming from the event generator. Particles then created by the simulation (secondaries) and background particles do not count as primary.

There is no additional cut on momentum or start vertex position. These conditions do not require that tracks have their clusters on the outgoing arm, so for example curlers which only hit layer three and four twice are not excluded. There is also no requirement on the number of SpacePoints existing for these tracks, which means that there are tracks with six or more clusters that have only two SpacePoints. In such cases, the VXDTF can not reconstruct this track at all and therefore one has to keep in mind that in all efficiency plots shown in this chapter, the VXDTF actually can not reach 100% efficiency.

The definition of the efficiency ε relies on the MC truth:

$$\varepsilon \mathrel{\widehat{=}} \frac{\text{number of reconstructed MC tracks}}{\text{total number of MC tracks}} \tag{5.1}$$

Although not applicable to real data, the information from the MC truth allows better insight into the bottlenecks and weak points of the TF, which itself does not see any MC information. In MC data one can determine which cluster is assigned to which particle track, and this allows to check whether the clusters all belong to the same track. The fraction of hits in a TC that belong to the same track is called "purity". It is 100% if all clusters of a TC belong to the same track. For a TC to be accepted as a valid reconstruction of a track, two conditions have to be fulfilled:

- The TC has to have at least 3 SpacePoints coming from the original track.
- More than 70% of the clusters of the TC must come from the original track. This therefore means that the purity has to be higher than 70%.

TCs found by the VXDTF are classified into five categories:

- *Perfect* TCs: contain *all* clusters from the original track and no additional clusters from other sources. All perfect TCs therefore must fulfill the necessary condition to have 100% purity. Additionally it must fulfill the conditions of a valid reconstruction.
- *Clean* TCs: contain a subset of the clusters from the original track and no additional clusters from other sources. All clean TCs therefore must fulfill the necessary condition to have 100% purity. Additionally it must fulfill the conditions of a valid reconstruction.
- *Contaminated* TCs: more than 70% of the clusters come from the dominating original track, but it also contain clusters from other sources. Additionally it must fulfill the conditions of a valid reconstruction.

- *Clone* TCs: could be of any category above, if no other TC already would have been assigned to the same track with a higher purity or quality. For example if a perfect and a clean TC to the same track is found, only the perfect one is marked as a valid reconstruction, the clean one will be marked as clone. Since the "clean overlaps" part of the VXDTF (described in Section 4.6) does not allow overlapping TCs, there are virtually no clone TCs in the sample of final TCs.
- *Ghost* TCs: all TCs which can not be categorized as any of the other TC types.

Tracks for which no perfect, clean, or contaminated TC exists are marked as "lost". The ghost rate ρ is defined by:

$$\rho \mathrel{\widehat{=}} \frac{\text{number of TCs marked as ghost}}{\text{total number of TCs}} \tag{5.2}$$

If three SpacePoints in a TC are from the same track and the fourth one is wrong, the TC is marked as contaminated, but the track is counted as reconstructed, since the purity is 75%. If only two SpacePoints are from the same track and a third SpacePoint has only one of its clusters belonging to this track, the purity is 83.3%, but the criterion of three SpacePoints from the same track is not satisfied. Such TCs are marked as ghosts as well.

5.2 $\Upsilon(4S)$ events without background

In this section the performance of the VXDTF is analyzed on a sample of 30,000 $\Upsilon(4S)$ events without background, which is described in more detail in Section 11.1. As already discussed in Chapter 3, these events contain typically 5–15 charged tracks with an average of about 10 tracks per event. In the simulation phase, the particles are tracked through the detector described in Section 2.4.1 and produce clusters on the u and the v side of the SVD sensors, which are then combined to SpacePoints if both sensor sides have recorded a hit. As mentioned in Section 3.2.2, SpacePoints can be formed in only about 98% of the cases where a passing track generated at least one cluster at a sensor. This certainly has an impact on the reconstruction efficiency. Ghost hits, kinks in the tracks due to material effects, energy loss — which is especially relevant for low-momentum tracks below 300 MeV/c — and primary vertices far from the IP region are also expected to lower the efficiency of the VXDTF. Further details of the situation faced can be found in Chapter 3.

The reconstruction efficiency is analyzed as a function of various parameters in order to get a detailed picture of the performance of the VXDTF. The most relevant plot can be found in Figure 5.1. The focus of the VXDTF is on low-momentum tracking, therefore a logarithmic scale of the (transverse) momentum was chosen, which allows a more detailed view of the low-momentum region. As mentioned in Section 5.1.1, the efficiency

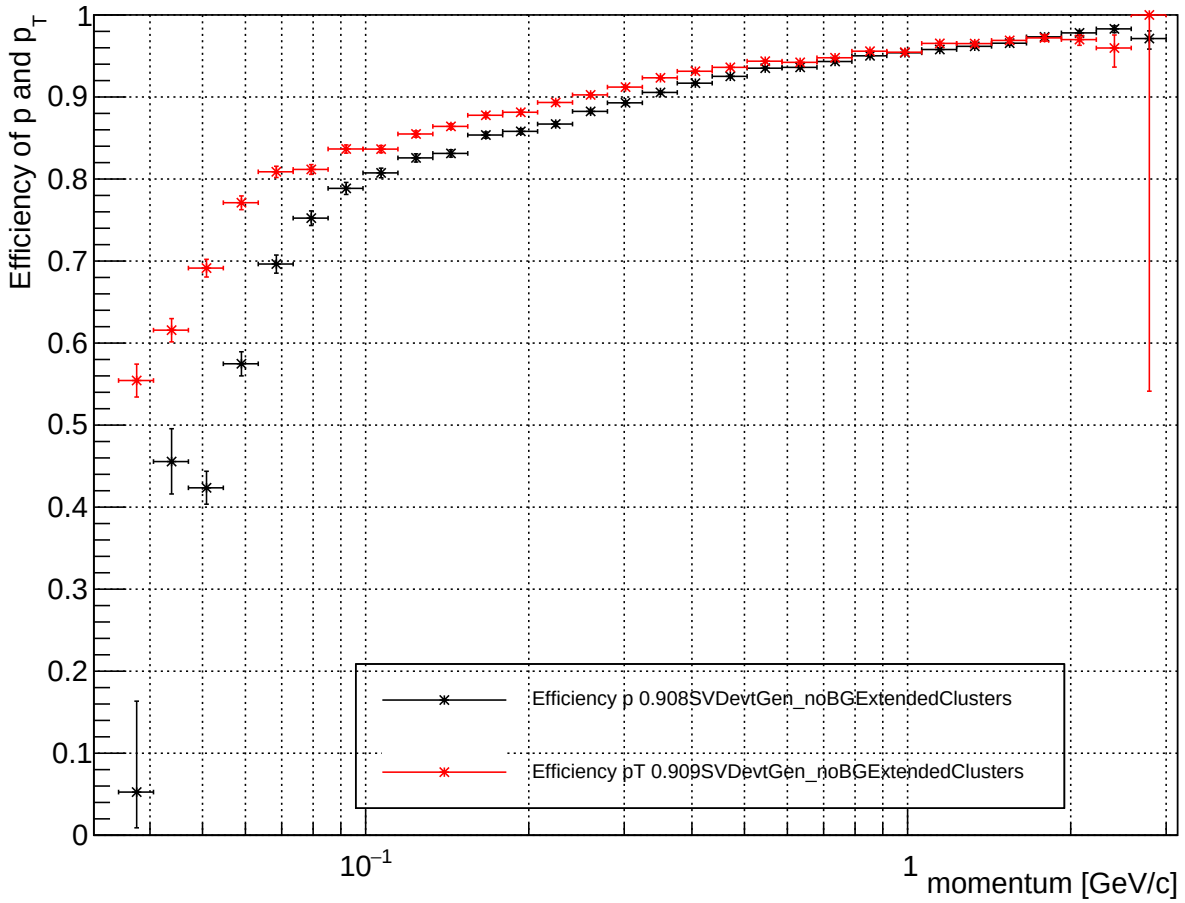
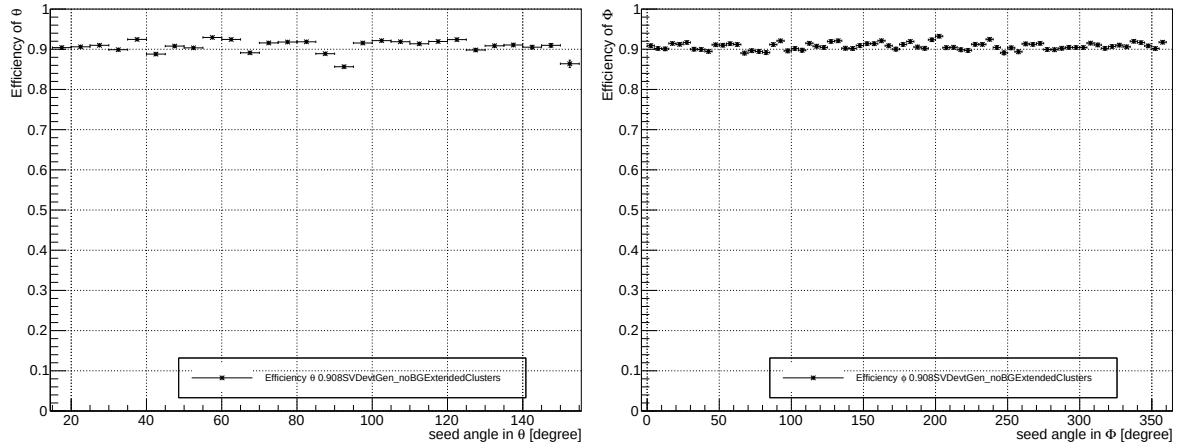


Figure 5.1: Efficiency versus momentum (black, total efficiency over the p range displayed = 90.8%) and transverse momentum (red, total efficiency over the p_T range displayed = 90.9%) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

is measured relative to the number of MC tracks having at least 6 clusters in the SVD, which does not mean that every MC track had 3 SpacePoints or more. The plot shows that the efficiency is about 96% for 1 GeV/c tracks and drops to 84% at $p_T = 100\,\mathrm{MeV}/c$ and to 81% at $p = 100\,\mathrm{MeV}/c$. In Belle no tracks below $100\,\mathrm{MeV}/c$ could be reconstructed since the SVD of Belle had no standalone tracking system, which means that such tracks are now available for physics channels which were not accessible before. For low-p_T tracks the efficiency stays above 80% down to about $65\,\mathrm{MeV}/c$, and is around 55% even for $35\,\mathrm{MeV}/c \leq p_T \leq 40\,\mathrm{MeV}/c$. For low p the situation is worse, since here the $\beta\gamma$ issue described in Chapter 3 can be identified by the gap between the p and p_T efficiency. Therefore for $p < 70\,\mathrm{MeV}/c$ the efficiency drops below 70% and for $p < 55\,\mathrm{MeV}/c$ the efficiency drops below 50%. Still, this pronounced drop has no strong impact on the overall efficiency value of 90.9% for tracks in the range $35\,\mathrm{MeV}/c \leq p_T \leq 3\,\mathrm{GeV}/c$ and

90.8% for tracks in the range $35\,\mathrm{MeV}/c \leq p \leq 3\,\mathrm{GeV}/c$. Due to the small number of low $p_\top$ tracks — as shown in Figure 3.12 — the overall efficiencies of p and $p_\top$ differ since they are not calculated using the same sample, because of the (transverse) momentum cuts applied. Overall one can clearly determine a strong dependency of the efficiency on the (transverse) momentum, which is not very surprising, since this is the case for practically any tracking algorithm. As possible causes one can suspect the increasing impact of material effects for low momenta and the increasing curvature for lower momenta. Other dependencies are checked in the following paragraphs.

Keeping the non-trivial layer layout of the SVD in mind, geometrical influences on the efficiency should of course be investigated, since the CA — and therefore the VXDTF — might have troubles here due to its semi-global tracking approach, which makes the algorithm at least possibly vulnerable to geometrical peculiarities — something which does not occur when using global algorithms like conformal mapping. Figure 5.2a shows the efficiency as a function of the angle θ. The error bars are not missing, but are too small



(a) Efficiency versus Theta, total efficiency over the θ range displayed = 90.8%.

(b) Efficiency versus Phi, total efficiency over the Φ range displayed = 90.8%.

Figure 5.2: Efficiency versus Theta (Figure 5.2a) and Phi (Figure 5.2b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

to be seen here. This plot is able to reveal possible issues with the slanted parts in forward region ($\hat{=}$ small θ angles) and other issues which might occur at $\theta \approx 90° \Leftrightarrow \cos\theta = 0$. First of all no severe effects of the slanted parts covering the range $17° \leq \theta \leq 43°$ can be detected, except that the transition between slanted and non-slanted parts is not smooth, which means that there the probability is higher to have missing clusters. Therefore the TF has to face the situation of missing SpacePoints more often and subsequently has less chances to find its tracks. Another dip can be found at $\theta \approx 65° - 70°$, which is explicable when checking the sensor distribution per layer over θ, which can be seen in Figure 2.8. There one can see that the gap between the sensors in layer 3 and of some of the other layers coincide and therefore again produce a higher probability of missing

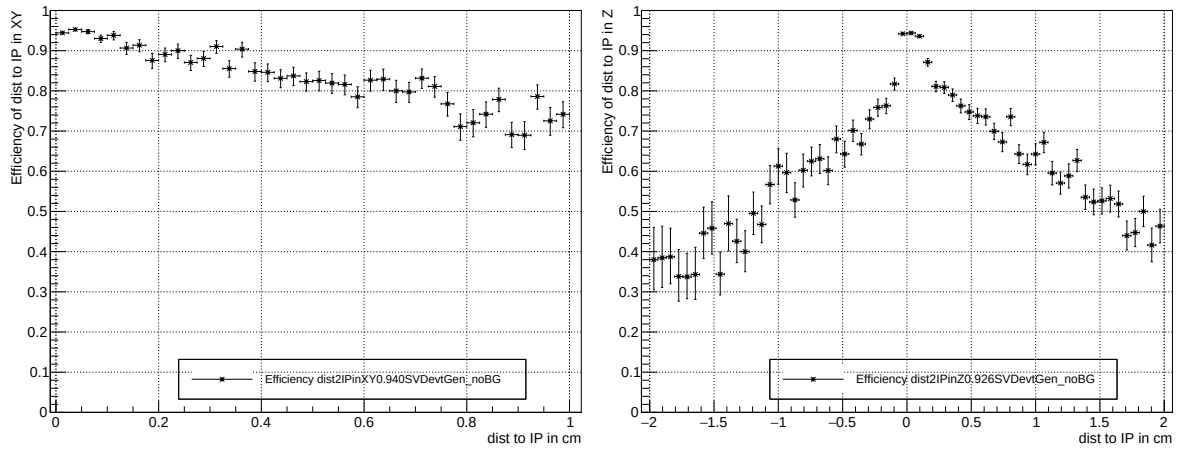
clusters. The conspicuous dip in $\theta \approx 85° - 95°$ has at least two causes. First, tracks with $\theta \approx 90°$ see the lowest material budget in terms of sensor thickness and therefore have the lowest energy deposit. The discussion in Section 3.2.2 indicates that some hits stay below the threshold, which leads to missing clusters and therefore to missing space points. Additionally at $\theta \approx 90°$ layer 6 has a gap between sensors 3 and 4 in all ladders. Figure 3.14b shows the particle flux, as a function of θ, has no visible effect on the efficiency, although the forward region sees nearly twice as many tracks as the backward region. This indicates that the VXDTF does not run near its limits regarding the particle and hit density here.

Φ is the second essential angle to be investigated for geometry dependent issues. Its study it can reveal problems that come from the overlapping parts of the ladders of the layers. Figure 5.2b shows the efficiency versus the Φ angle of the momentum vector at the vertex of the tracks recorded, covering the entire Φ range. Compared to Figure 5.2a, the various dips in efficiency are far less pronounced and the nature of the parameter makes it less obvious to name possible suspects for being the cause. While the θ angle of the momentum vector is virtually constant along the track, Φ changes according to the curvature and the charge. High momentum tracks are nearly straight and therefore even issues with sensors lying at a certain Φ region on outer layers can be identified. For low-momentum tracks, however, the situation is different, since they bend away from a straight line extrapolated from the momentum vector at their start vertex position. Additionally the overlapping parts of the ladders are on different Φ positions on each layer, and it is therefore difficult to pinpoint any special cases here. A conclusion which is safe to form here is that the overlapping parts do not have a severe impact on the efficiency. The particle flux as a function of Φ (as can be seen in Figure 3.14a) has no visible impact either.

Figure 5.3 plots the efficiency versus the x-y position and versus the z position of the vertex of the particles. This allows to focus on issues which might occur because of cuts set for training the SecMap for the VXDTF (detailed settings are described in Chapter 4 and Section 11.2). For these plots the cuts for selecting the training sample are relevant, where basically two parameters may have an effect on the results:

- 'maxXYvertexDistance' $= 1\,\text{cm} \longrightarrow r_{vertex} \leq 1\,\text{cm}$
- 'maxZvertexDistance' $= 2\,\text{cm} \longrightarrow |z_{vertex}| \leq 2\,\text{cm}$

The main goal of these cuts is to define a trade-off between getting as much tracks as possible and keeping the combinatorial issue low. These two goals cannot be achieved at the same time, since looser cuts for the training sample — although increasing the possibility to reconstruct interesting tracks — force the final cuts to be looser themselves and therefore the essential filtering done by the SecMap, the two- and three-hit Filters is weakened. Since this is one of many parameters influencing each other, a detailed machine learning procedure should be implemented to allow an automated scan in in this multi-dimensional parameter space for the best set of cutoffs. Coming back to the plots



(a) Efficiency over vertex position in xy, total efficiency over the xy range displayed = 94.0%.

(b) Efficiency over vertex position in z, total efficiency over the z range displayed = 92.6%.

Figure 5.3: Efficiency versus distance of primary vertex from origin in xy (Figure 5.3a) and in z (Figure 5.3b) - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

of Figure 5.3 one immediately recognizes a strong dependency of the efficiency on the vertex position. This correlation is much more visible than possible correlations between the efficiency and θ or Φ. At second glance there are several interesting things to see:

- Even for the quasi perfect situation of tracks coming from the origin, the efficiency is only about 95%. This indicates a global effect and can be traced back to the hit efficiency being less than 100%.
- Sharp boundary effects are not visible, but the decline of efficiency is steady up to the position of the cuts.
- The particles within the xy cut range are reconstructed with a higher efficiency (94%) than for the z cut range case (92.6%).
- The efficiency depends not only on the distance from the IP, but is also correlated with the number of tracks in a bin. This can especially be seen in Figure 5.3b, where the forward region (with positive z values) has a higher efficiency while having the smaller error bars, which themselves are directly derived from the sample size in that bin. This can be explained by the fact that the cuts in theSecMap do not cover the entire range of physically valid combinations, so that "typical" combinations are accepted more often than "rare" combinations.

These problems clearly require further study.

A closely related perspective can be investigated in Figure 5.4, where the efficiency is plotted versus the calculated d_0 value, which has been defined in Eq. (3.1). First, one

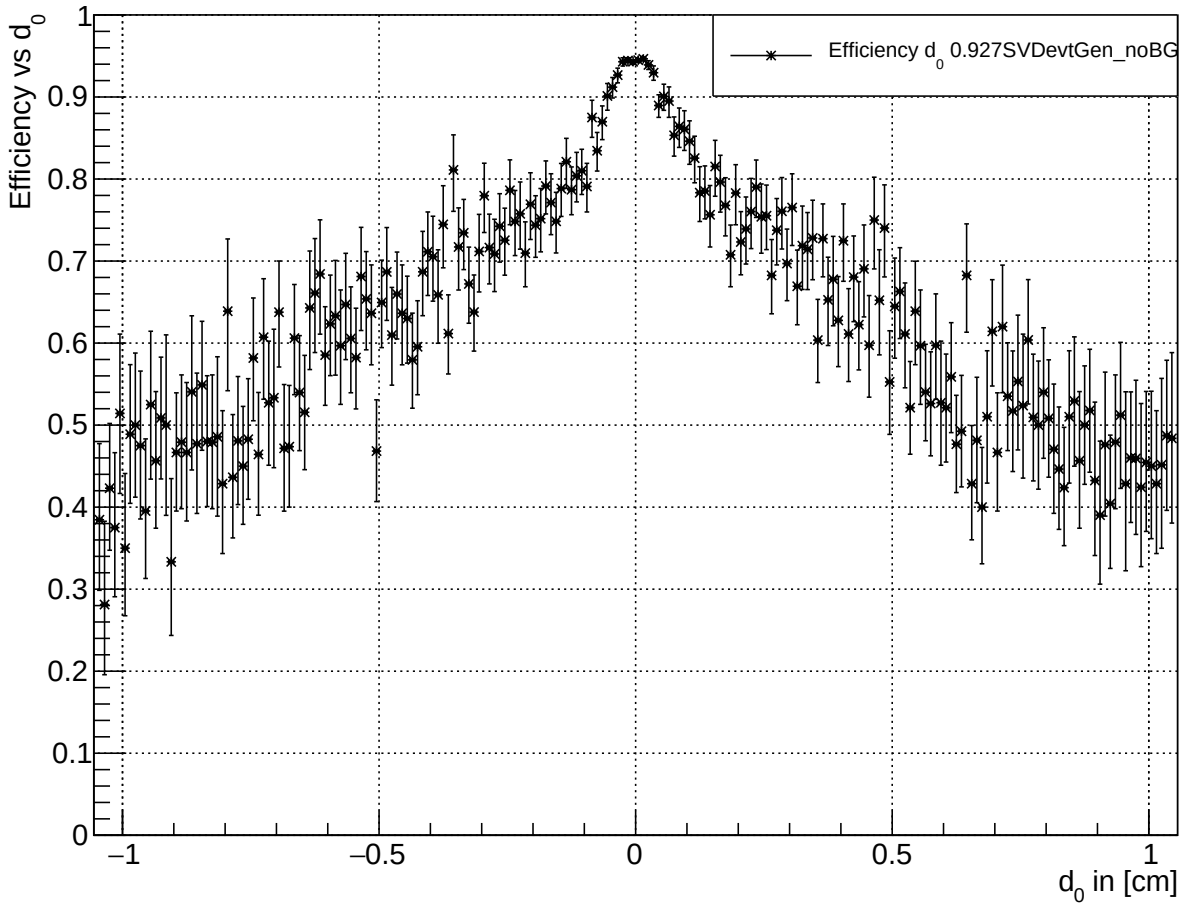
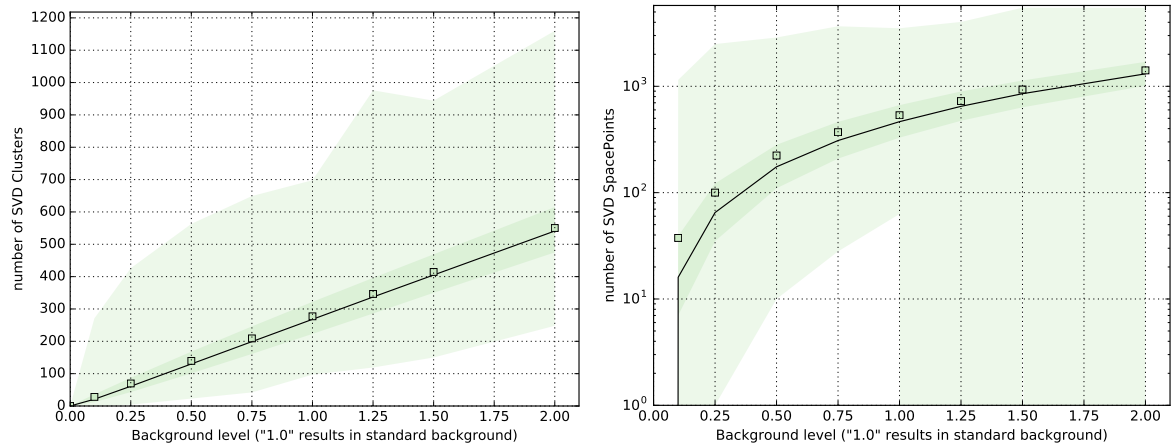


Figure 5.4: Efficiency versus d_0 in percent per bin, total efficiency over the d_0 range displayed = 92.7%. Computed from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

can again see a clear dependence of the efficiency on d_0. It is even more pronounced than in Figure 5.3a. No clear difference between positive and negative signs (charges) can be observed, with the possible exception of the range $|d_0| > 1$, which however is dominated by the size of the error bars.

5.3 Behavior with background

After interpreting the performance of the VXDTF for the case of $\Upsilon(4S)$ events without background, the behavior with background only events is analyzed. The background types described in Section 2.6 are coming from different directions, which is why only a fraction of them produce trajectories which seem to come from the origin, notably the two photon background, which creates e^+e^- pairs coming from the IP.



(a) Clusters. (b) SpacePoints.

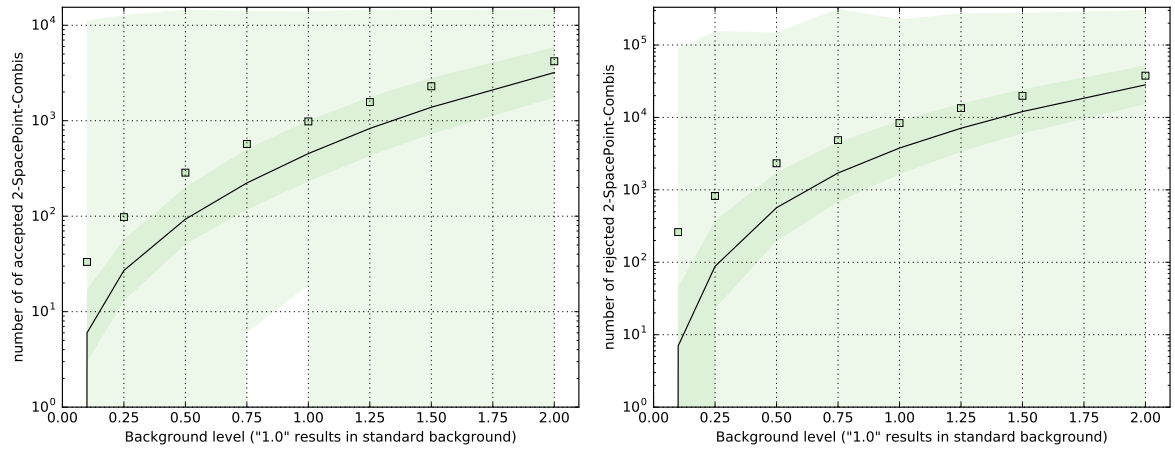
Figure 5.5: The number of Clusters (Figure 5.5a) and SpacePoints (Figure 5.5b) for different background levels without $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

The plots shown in Figure 5.5 could also fit in Chapter 3, but were placed here instead, since they use a plot type often used in this chapter. These plots represent six boxplots drawn as error bands for the range of $0\% \leq \text{background} \leq 200\%$. The black line represents the median, the square represent the means, the dark band represents the width between the 25% and an the 75% quartile, and the light band represents the range between the minimum and the maximum value. Such a plot is much more informative than just showing the mean and the standard deviation.

Figure 5.5a shows the number of clusters found in as a function of the background level. As expected, the number of clusters rises linearly with the background level, thereby confirming that the scaling tool for background in basf2 does its job. The number of clusters for the expected background level ($100\% \mathrel{\widehat{=}} 1.00$) is slightly below 300 clusters per event and varies between 100 and 700 clusters per event.

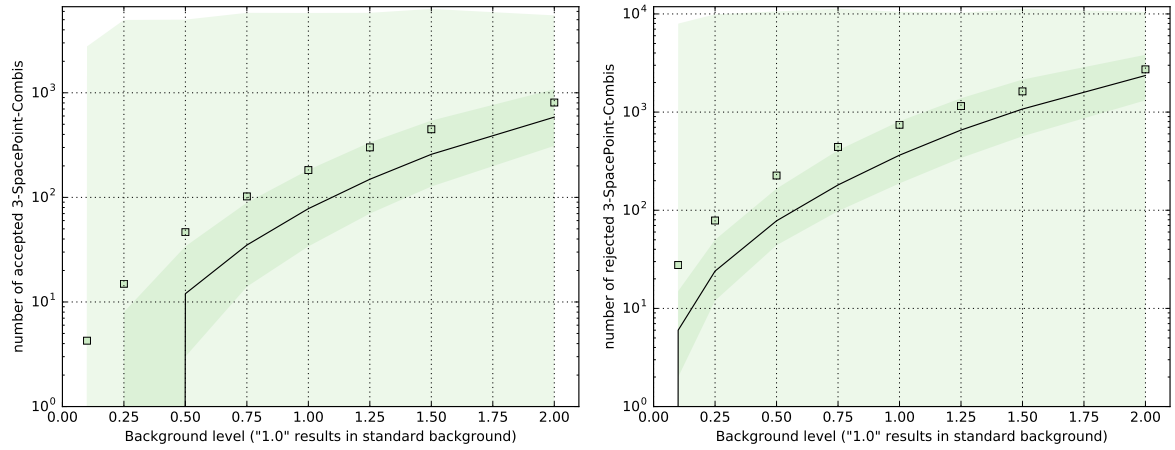
Figure 5.5b on the other hand clearly shows the combinatorial issues for the combination of SpacePoints, which is why a logarithmic scale was used to illustrate the behavior for increasing background levels. At the level of 100% background the number of SpacePoints rises to a mean of about 600 SpacePoints per event, while the maximum can be over 3,000 SpacePoints per event. The mean values are always higher than the median values due to the sensitivity of the mean regarding outliers. They therefore indicate that certain events contain jet-like structures. This results in a high number of clusters lying on a small number of sensors and therefore produce many ghost hits.

The next plots Figure 5.6 and Figure 5.7 are especially relevant when being compared with the values of Table 3.3 in Chapter 3, where a straightforward estimation of possible two and three-hit combinations is done, assuming that hits of neighboring are be combined.



(a) accepted two-hit combinations. (b) rejected two-hit combinations.

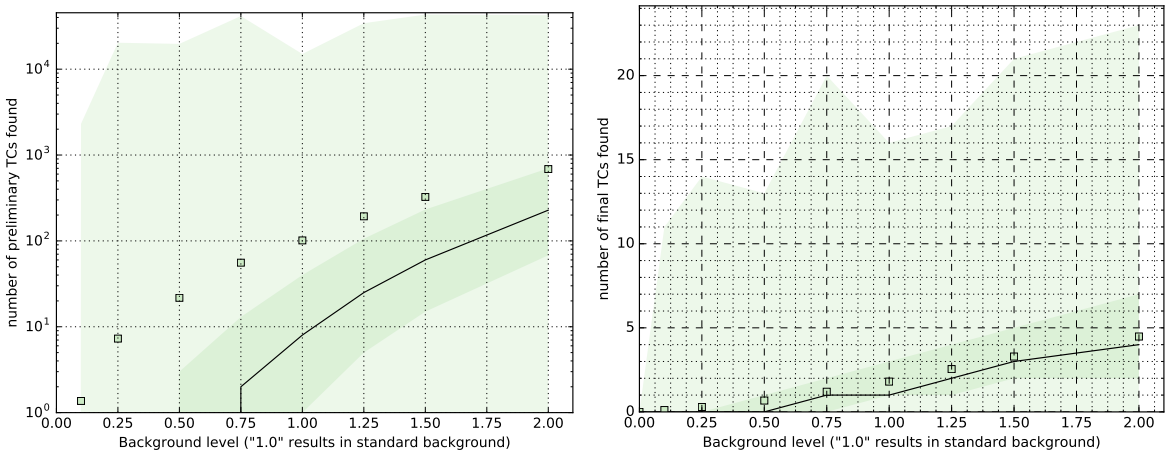
Figure 5.6: Total number of accepted two-hit combinations (Figure 5.6a) and the rejected ones (Figure 5.6b) for different background levels without $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.



(a) accepted three-hit combinations. (b) rejected three-hit combinations.

Figure 5.7: Total number of accepted three-hit combinations (Figure 5.7a) and the rejected ones (Figure 5.7b) for different background levels without $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Although numbers for the case of background combinations only are missing, it is clear that the number of occurring two-hit combinations (accepted+rejected) is below the calculated number of the table. This is due to the filtering done by sorting the hits into the SecMap, which automatically suppresses lots of irrelevant combinations. For the case of 100% background about 10.5% of all two-hit combinations allowed by the SecMap are accepted by the Filters activated. As a comparison, for 200% background the acceptance rate is 10% and therefore does not seem to get worse with the background levels to be expected. A similar check can be done for three-hit combinations, which results in an acceptance rate of 19.7% and 22.8% for the cases of 100% and 200% background, respectively. For the case of standard (100%) background, a mean of about 180 three-hit tracklets survive per event. Further filtering steps like the CA and four-hit filtering are then further reducing that number to get preliminary TCs by the TCC[66].



(a) Preliminary TCs per event.

(b) Final TCs per event.

Figure 5.8: The total number of TCs found after TCC (Figure 5.8a) and final TCs (Figure 5.8b) for different background levels without $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 5.8 shows the resulting preliminary (Figure 5.8a) and final (Figure 5.8b) TCs provided by the VXDTF for the case of background only. Up to the 50% background level, less than every second event results in preliminary TCs at all. But still the mean of 20 preliminary TCs for the 50% level indicate that a few events with bursts of background are dominating the output. The situation for 100% background is comparable, where 8 preliminary TCs are found in median, but 100 in mean, and over 10,000 preliminary TCs in the worst case. The number of final TCs per event is kept low by the clean-subset filter which leads to a mean of less than two final TCs per background event, even the maximum values occurring are staying in a low two-digit level.

[66] Track Candidate Collector of the VXDTF

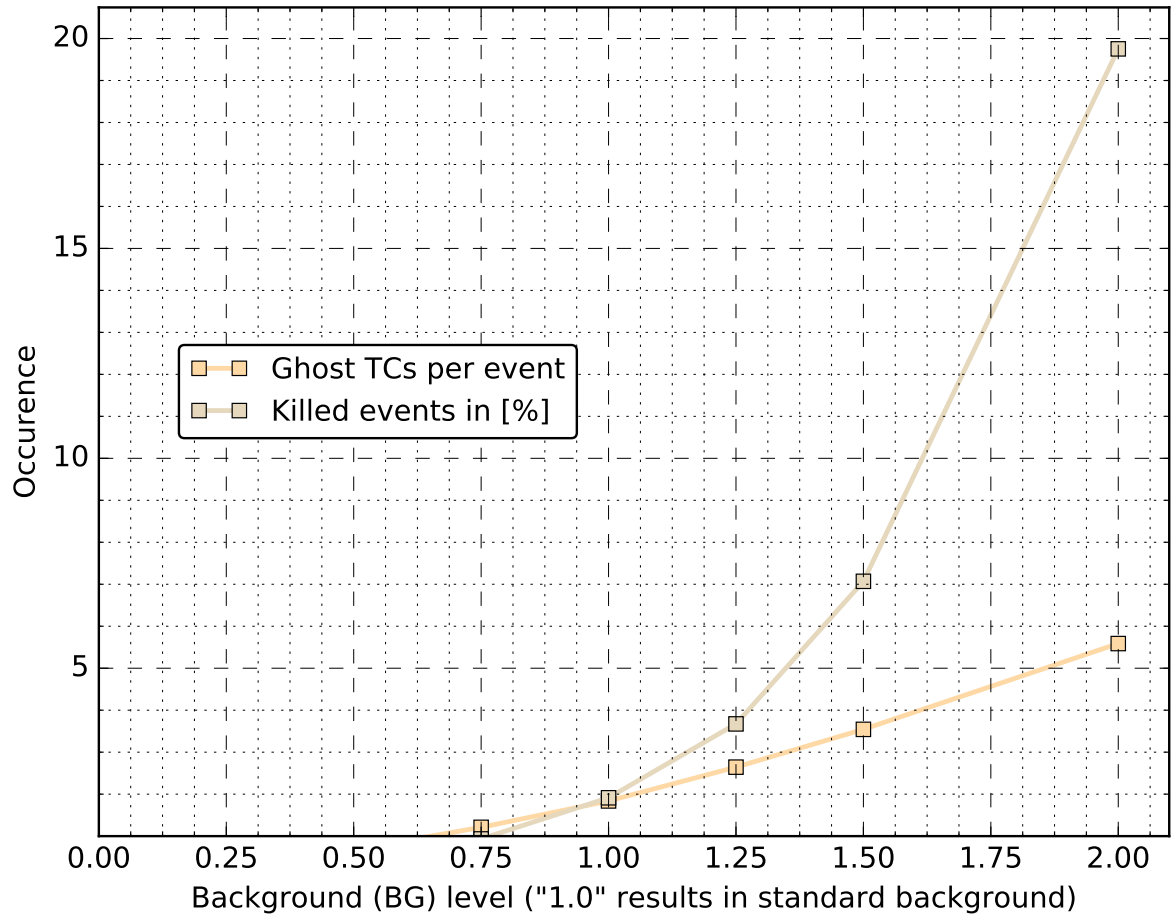


Figure 5.9: The plot summarizes the most relevant parameters over the background (BG) level - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 5.9 summarizes the output of the VXDTF in relation to the background level. The ghost TCs per event are the mean number of final tracks since none of the TCs are belonging to particles we actually want to track. This means that we have to expect about two tracks per event looking like realistic ones — at least from the VXDTFs' point of view — but are not relevant at all. These tracks then can only be suppressed by track fitting after P.Id.[67]

Another number found in Figure 5.9 is the percentage of killed events. Killed events are events where the number of accepted two-hit combinations surpasses a certain threshold — "killEventsForHighOccupancy" described in Section 11.2.3 — , which prevents the VXDTF from spending too much time a single event. In the long run that cut should be removed, but at the time being it is used to form a rough trade-off between time

[67] Particle Identification

consumption and efficiency. To be able to remove this cut, one has to optimize the performance of the Filters used in the VXDTF, which are relying on the cuts stored in the SecMap. This leads to a high number of parameters which need to be tuned, a task which can only be done with a solid machine learning approach within a reasonable amount of time. The same value for ”killEventsForHighOccupancy“ has been used in all standard setups and has only been changed for those setups specially marked as *tough*. For the case of having background only, the percentage of killed event stays below 2% for the expected (100%) background level and then increases to nearly 20% for 200% background, which is only used as a worst case scenario in this study.

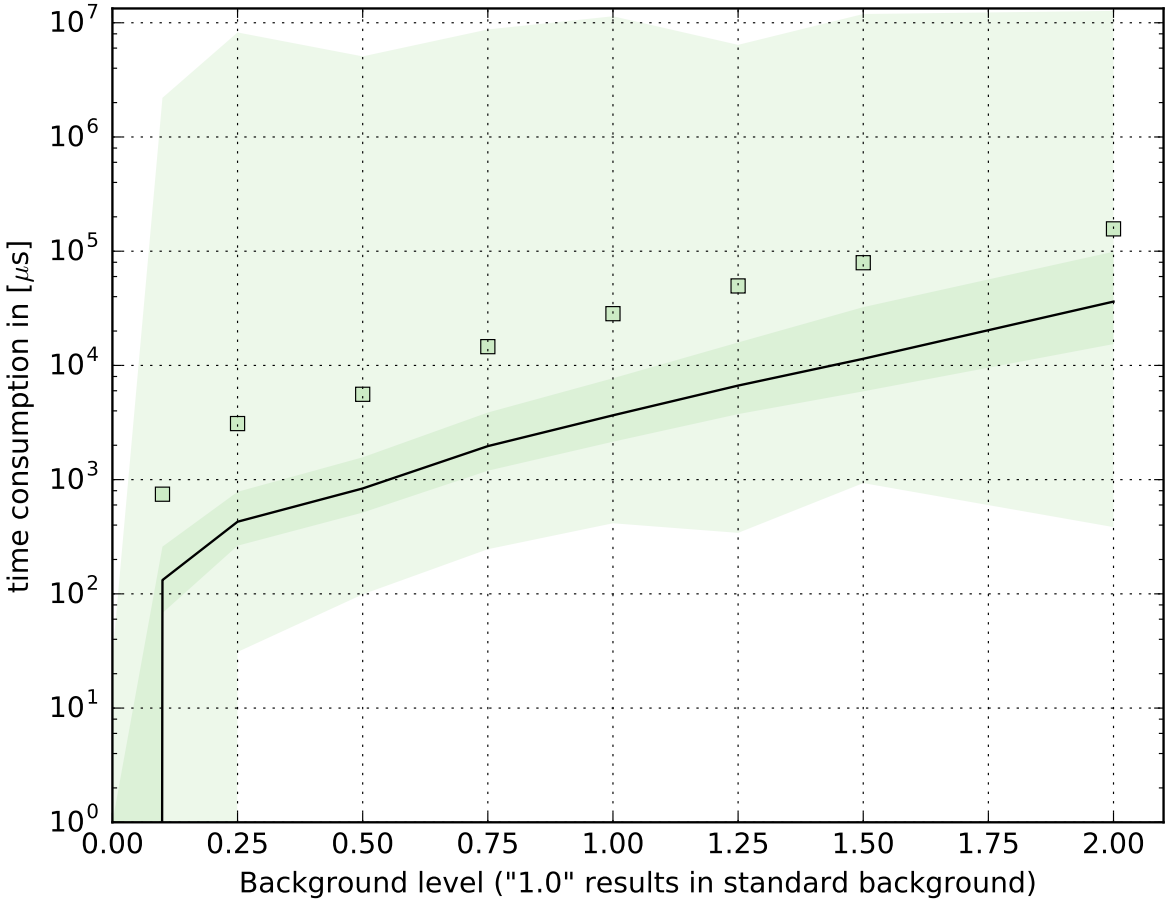


Figure 5.10: The dependency of the time consumption in relation to the background level - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1. The minor ticks at the logarithmic y axis are marking only the even numbers (2,4,6,8).

The time consumption of the TF n is plotted in Figure 5.10. The absolute values of that plot are of limited use since the machine on which these numbers were recorded (the machine setup is listed in Section 11.5) is not comparable to the expected HLT setup.

Nonetheless the numbers can be used as rough indicators of what to expect. The mean of about 15 ms per event in the case of standard background is much larger than the median of 3 ms per event due to a small number of events of unfortunate cluster constellations. This can be inferred from the value of about 10 s for the worst case event.

5.4 $\Upsilon(4S)$ events with background

The next step is the combination of $\Upsilon(4S)$ events with background. The situation presented here is therefore very close to the expected situation in the data taking phase.

5.4.1 Efficiencies

To evaluate the performance of the VXDTF the efficiencies as a function of several parameters is shown in this section.

Figure 5.11 shows the efficiency of the VXDTF versus the (transverse) momentum. The distribution looks like a copy of Figure 5.1, with the exception that the efficiency is significantly lower: 86.9% versus 90.8% efficiency over p and 87.1% versus 90.9% efficiency over $p_\top$. The fact that nearly 4% of efficiency has been lost compared to the clean $\Upsilon(4S)$ events, but no change in the overall distribution seems to be occurring, already indicates that this drop must be caused by some particular effect and not by the general TF layout. Indeed one can easily identify the "killEventsForHighOccupancy" parameter as the main cause for the situation, which will discussed below.

While Figure 5.11 does not show any distinctive dips, Figure 5.12a, which shows the efficiency versus θ, is more revealing. Most of the dips which could already be identified in Figure 5.2a are now worse, and therefore the arguments presented in Section 5.2 still hold: Coinciding gaps between the sensors on a ladder at several layers, and possibly a energy deposit threshold which might be too high, lead to missing hits accumulating in certain regions. This effect might be enhanced for added background since the probability is higher to have "good looking" background clusters in reach and therefore the probability increases that these hits deteriorate the results.

Figure 5.12b presents a very similar plot as Figure 5.2b, which indicates that the overlapping regions are not the cause of the distinctive dips seen in Figure 5.12a. But again the overall efficiency shows the same drop in the general efficiency level as the momentum plot.

The plots of efficiency versus the start vertex distance in Figure 5.13a and Figure 5.13b are practically identical to their counterparts without background added (Figure 5.3), with just a few differences to be spotted; Next to the typical general drop in efficiency — which can be seen in all plots in this section — only a few single bins seem to differ. In Figure 5.13a there are some bins with few entries in the range $0.4\,\text{cm} < r \leq 0.6\,\text{cm}$, where there is a visible drop, but that drop is only one sigma off the old mark and therefore not very severe. In Figure 5.13b the picture is only slightly more different

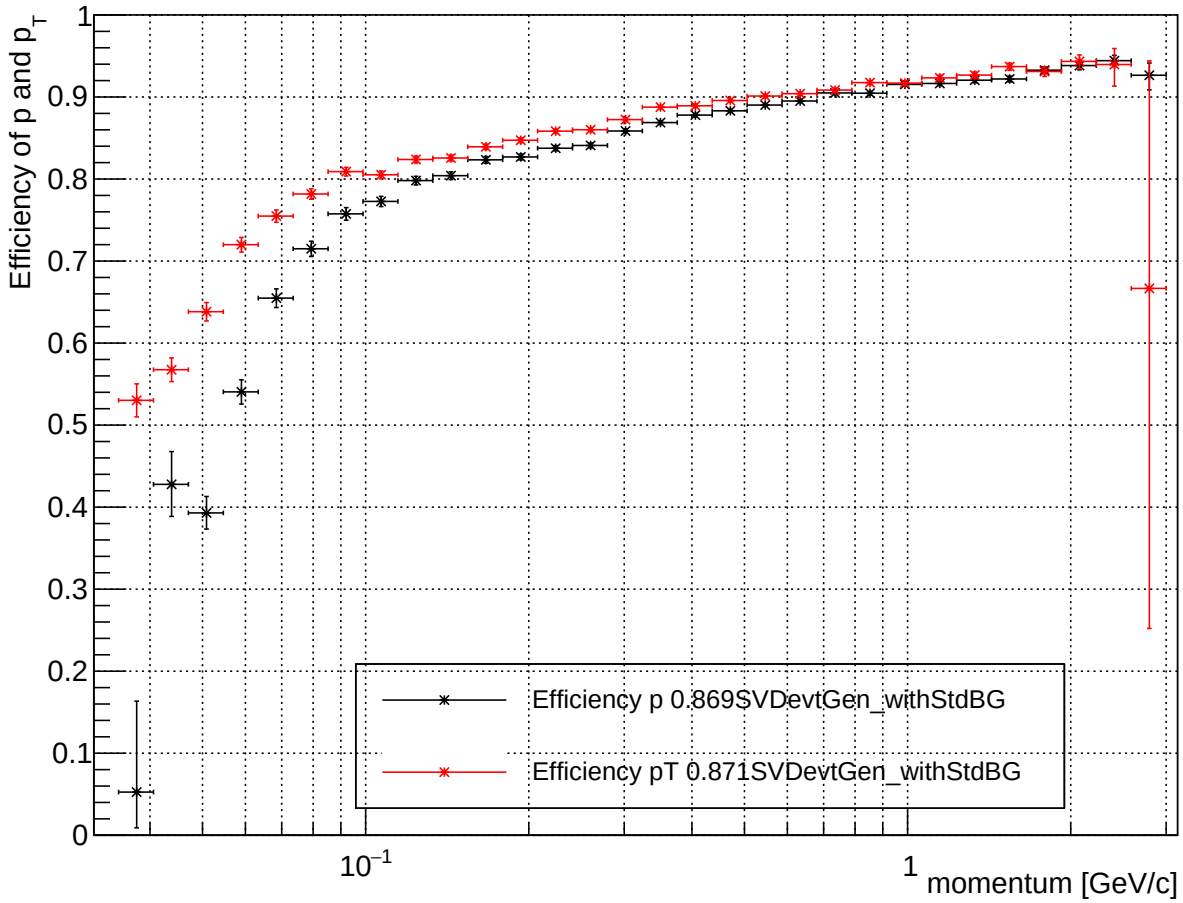
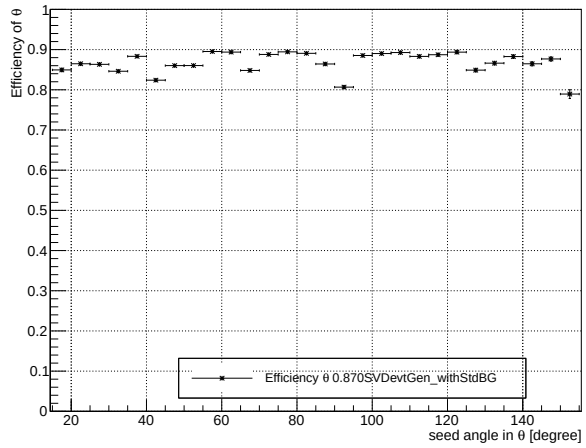


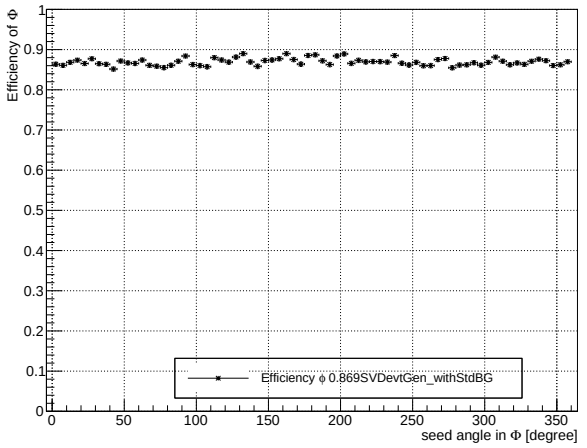
Figure 5.11: Efficiency versus momentum (black, total efficiency over the p range displayed = 86.9%) and transverse momentum (red, total efficiency over the $p_\top$ range displayed = 87.1%) for the case of expected full background level - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

from the corresponding plot without background. It seems, however, that the forward region suffers less from the impact of background than the backward region, and that the general difference between forward and backward region is larger.

Figure 5.14 shows the dependence of the efficiency on d_0. It repeats the same pattern: the additional background lowers the general efficiency by about 4%, but the overall structure stays basically the same. Two minor things should be noted, though: First, the plateau around 0 is somewhat broader than the one in Figure 5.4. Second, the bins far off 0 seem to have better efficiencies compared to the other plot. On the one hand the changes are within the error margins, but on the other hand the overall structure — meaning the relative positions of neighboring bins — is pretty similar to the plot without background and therefore that impact might be significant. Since the cuts stored in the

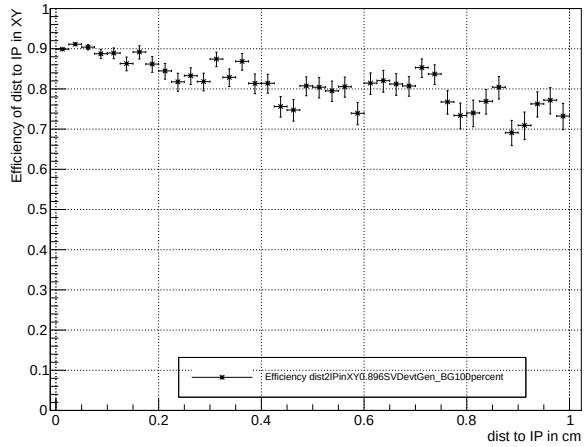


(a) Efficiency versus Theta, total efficiency over the θ range displayed = 87.0%.

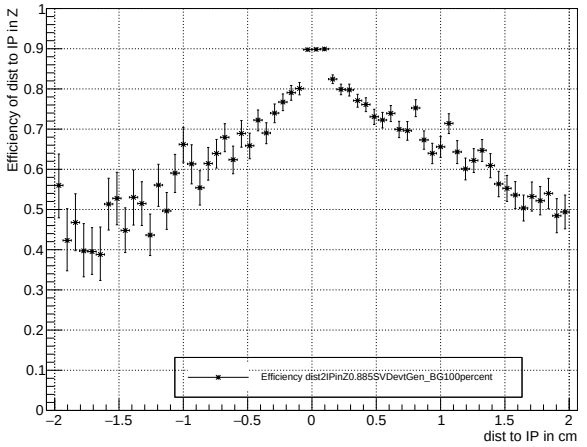


(b) Efficiency versus Phi, total efficiency over the Φ range displayed = 86.9%.

Figure 5.12: Efficiency versus Theta (Figure 5.12a) and Phi (Figure 5.12b) for the case of 100% background - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.



(a) Efficiency over vertex position in xy, total efficiency over the xy range displayed = 89.6%.



(b) Efficiency over vertex position in z, total efficiency over the z range displayed = 88.5%.

Figure 5.13: Efficiency versus distance of primary vertex from origin in xy (Figure 5.13a) and in z (Figure 5.13b) for the case of 100% background - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

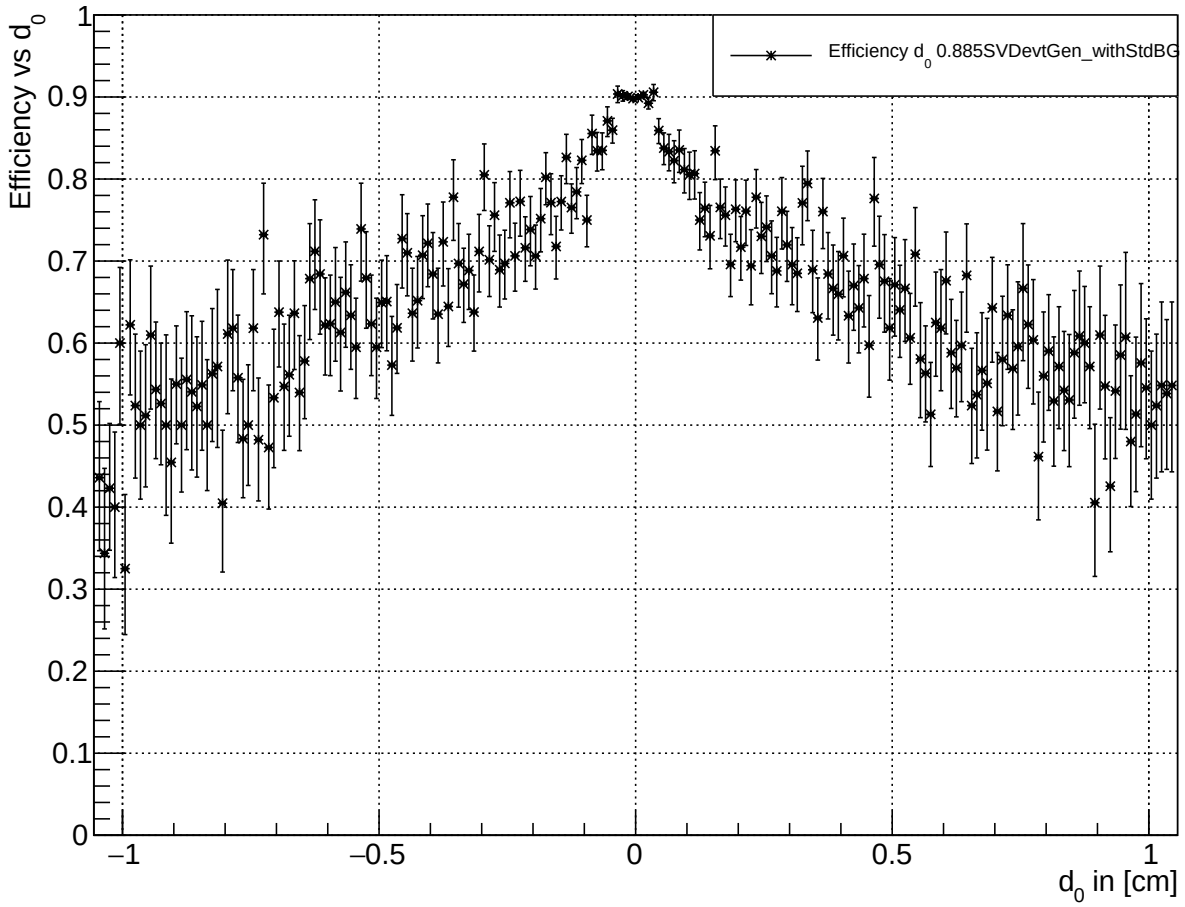
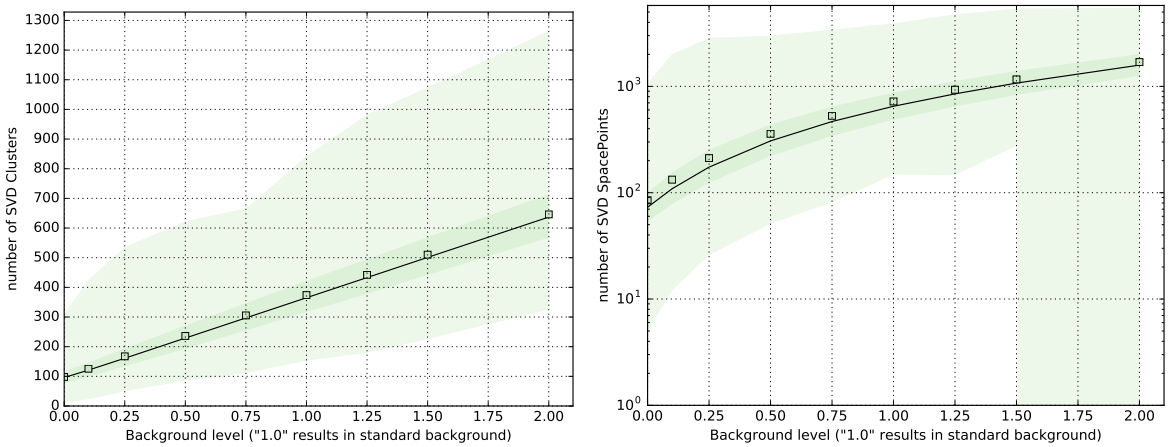


Figure 5.14: Efficiency versus d_0 in percent per bin for the case of expected full background level, total efficiency over the d_0 range displayed = 88.5% - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

SecMap are identical to the $\Upsilon(4S)$ setup without background, the impact can only be explained by the effects of adding the background. A solid cause for this, however can not be named, only the SecMap — as mentioned above — and the CA can be excluded from the list of possible causes. The CA could only have a negative effect (if any) on increased background, no positive one, since with an increasing number of hits the probability increases that sufficiently "good looking" bad hit lie near relevant ones.

5.4.2 From Clusters to Track Candidates

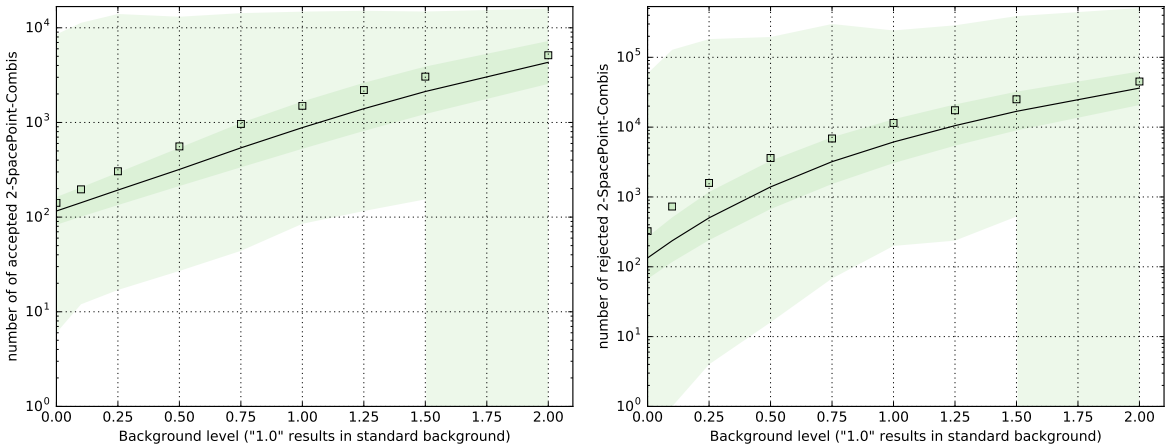
Figure 5.15 sketches the dependence of the number of clusters and SpacePoints on the background level. Clusters are scaling linearly from about 100 clusters per event with $\Upsilon(4S)$ events to only about 380 in the case of 100% expected background, while the



(a) Clusters. (b) SpacePoints.

Figure 5.15: Clusters (Figure 5.15a) and SpacePoints (Figure 5.15b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

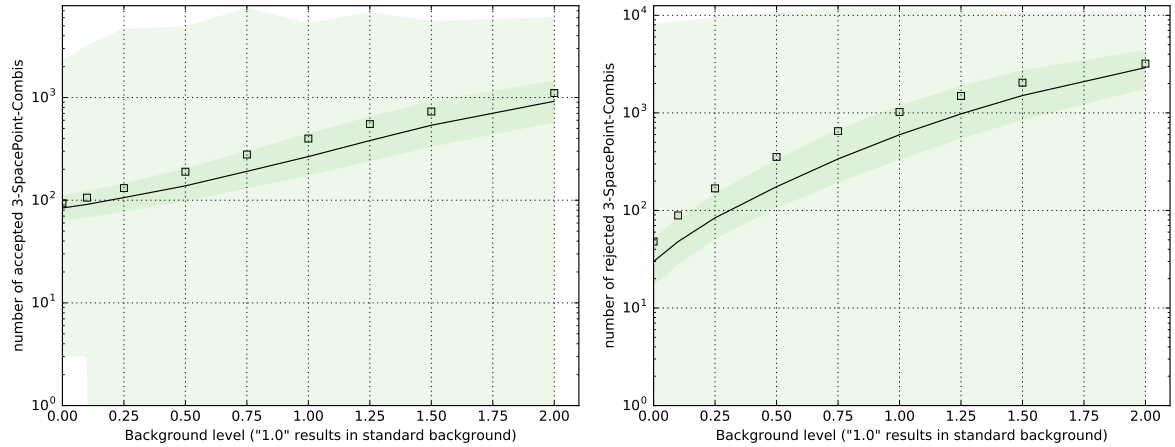
non-linear growth of SpacePoints goes up from a mean value of 85 SpacePoints in the case of no background to about 720 SpacePoints per event. The variation is rather large, and single events can therefore have up to 5,000 SpacePoints spread on 4 layers, where the majority of the SpacePoints will cover layer 3. These numbers once again emphasize the severity of the combinatorial issue to be faced. A TF therefore must focus on the worst case, not on the median of the events.



(a) accepted two-hit combinations. (b) rejected two-hit combinations.

Figure 5.16: The total number of accepted two-hit combinations (Figure 5.16a) and the rejected ones (Figure 5.16b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 5.16 shows the acceptance and rejection rates of two-hit combinations for different background levels added to the bare $\Upsilon(4S)$ events. The level of accepted two-hit combinations rises in absolute numbers from about 140 at 0% background — 30.5% acceptance rate — to about 1,500 at the 100% level — 11.5% acceptance rate — while the mean value for rejections increases from about 320 to 11,490 discarded two-hit combinations per event. The reason for the vast drop in acceptance rate is evident; "clean" $\Upsilon(4S)$ events do not allow a high number of bad combinations, while the added background is doing that, therefore that drop in the acceptance rate is expected. Focusing on higher background levels which allows to get an estimate on the robustness on the current VXDTF configuration one can see that the acceptance rate stays very low and even on 200% background level the rate is 10.2% which indicates that the TF does not reach its limits in filtering bad combinations yet.

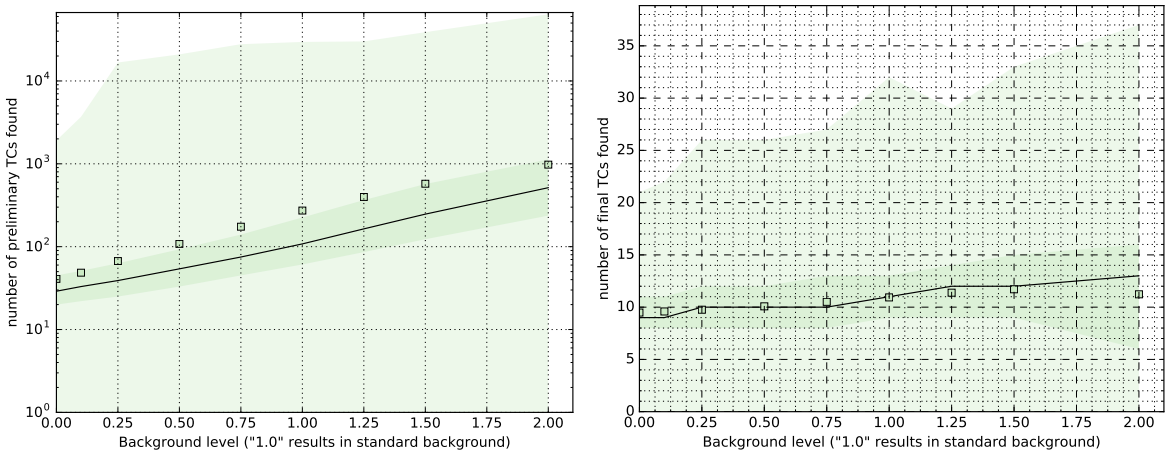


(a) accepted three-hit combinations.

(b) rejected three-hit combinations.

Figure 5.17: The total number of accepted three-hit combinations (Figure 5.17a) and the rejected ones (Figure 5.17b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

In Figure 5.17 one can see the performance of the three-hit Filters. The acceptance rate is much higher here, being 66% at 0% background, 28% at 100% background and 25.7% at 200% background level. The drop in effectiveness for filtering three-hit combinations versus the two-hit case may come from the fact that the two-hit Filters themselves are very effective already and the prefiltering done by the SecMap is less efficient in terms of filtering than two-hit filtering. But still three-hit Filters neglect about three of four combinations at full background level, which means that still many unwanted combinations do slip through the net of the two-hit Filters. The mean number of accepted three-hit combinations for 100% background then is 398.5 combinations per event, while the maximum value is near 5,000 accepted three-hit combinations per event. Stronger Filters might help getting these numbers down.



(a) Preliminary TCs per event.

(b) Final TCs per event.

Figure 5.18: The total number of TCs found after TCC (Figure 5.18a) and final TCs (Figure 5.18b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 5.18 shows how many preliminary and final TCs can be found per event is plotted. The number of preliminary TCs collected after the CA and four-hit filtering is not much lower than the number of accepted three-hit combinations; a mean of about 273 preliminary TCs versus about 400 three-hit combinations means that nearly all three-hit combinations were parts of valid TCs since four hit TCs are not uncommon, but the majority. What is interesting, though, is the fact that the worst case value of preliminary TCs is nearly 10 times more than the worst case for accepted three-hit combinations. This means that there are some jet-like structures causing an exploding number of TCs with a relatively small number of overlapping three-hit combinations. If the time consumption of the NN[68] were lower — a problem which will be discussed in the time consumption paragraph of this section — this would be a nice and effective way to filter these Trees of three-hit tracklets. The plot of the number of final TCs in Figure 5.18b again shows the power of the NN; from a mean of about 270 preliminary TCs for 100% background the number of final TCs drops down to a mean of about 11. So about 260 TCs per events are being removed after a fast circle fit identifies their quality and the NN keeps the best ones in the end with an efficiency of about 87%. What is also striking on this point is the fact that although the number of preliminary TCs is increasing much with the background level increasing, the mean value of final TCs stays nearly the same with about 9.5 TCs at 0% background to about 11.5 TCs at 200% background. Even in the worst case there are only about 35 TCs after finishing the VXDTF process. This indicates that one could test looser restrictions for the NN, such as counting single clusters shared between SpacePoints not as overlapping, to let through

[68] Neural Network

some more promising TCs and deal with them on the more precise track fitting level. The benefit would be a higher efficiency, while the ghost rate should stay on a level where it still does not cost too much for fitting.

5.4.3 Time consumption and overview

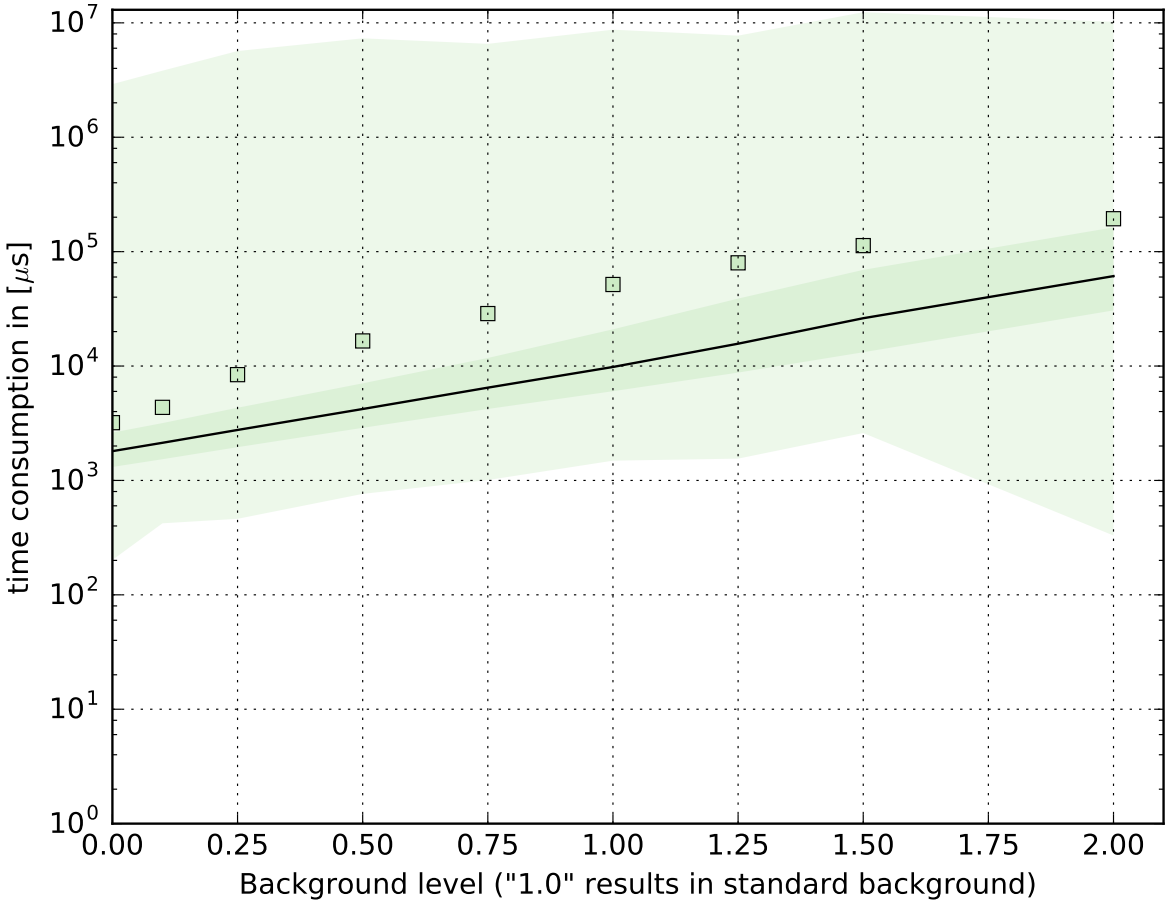


Figure 5.19: Dependency of the time consumption in relation to the background level including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1. The minor ticks at the logarithmic y axis are marking only the even numbers (2,4,6,8).

In Figure 5.19 the time consumption in relation to the added amount of background is plotted. The cut for "killEventsForHighOccupancy" is at 5,500, and therefore events with a high combinatorial issue are discarded. Nonetheless the time consumption rises significantly from a mean value of about 3 ms per event for $\Upsilon(4S)$ events without background to a mean of about 52 ms per event at 100% background level. This amounts to an increase by a factor of ≈ 17, while the number of SpacePoints increases by a factor

of ≈ 9 at the same time. The mean time consumption lies for each measured background level above the 75% quartile which means that the mean value is always dominated by a small number of events. The worst case values are about 10 s for a single event — over 200 times longer than the average event. This indicates that more efforts should be invested in fine tuning the parameters of the VXDTF to suppress these effects.

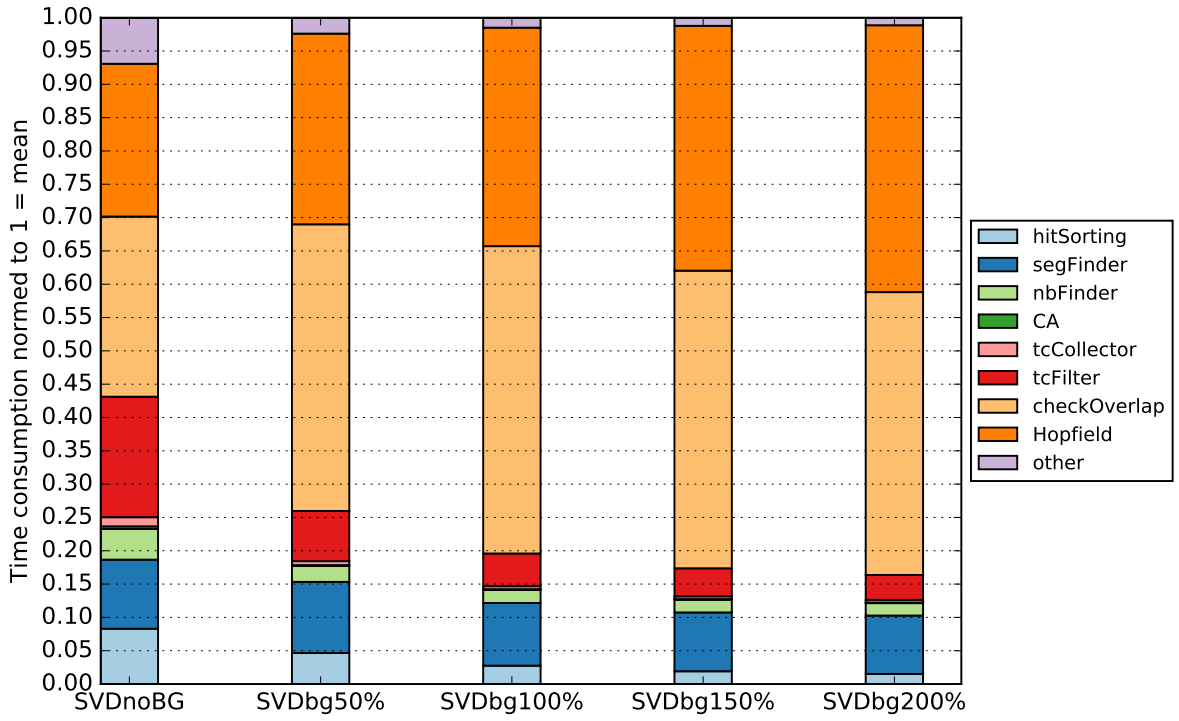


Figure 5.20: Time consumption of essential components for different background levels, computed from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

To allow a more detailed insight into where the time is actually spent, Figure 5.20 is very helpful. It plots the individual time consumption of several steps of the TF for different background levels, renormalized to 1 to allow a direct comparison. The dominating steps, i.e. the checking and the cleaning of the overlapping track candidates , already take about 50% of the time at $\Upsilon(4S)$ events without background, but then increases to about 75% of the total time consumption for 100% background level. This clearly shows that the code of these parts should be refactored and optimized. Additionally one should focus on a better training of the SecMap which should decrease the combinatorial issue and subsequently the time consumption of all parts of the VXDTF following after the n-hit Filters. The reason why the checking and cleaning of overlapping TCs takes so long is easy to identify: one has to compare each cluster or SpacePoint of a TC with the hits of all other TCs to find out if they are overlapping. The Hopfield NN is yet using a very

slow and non-vectorized library for doing its linear algebra computations, and therefore switching to a more modern library like the Eigen Library [43] would help a lot already.

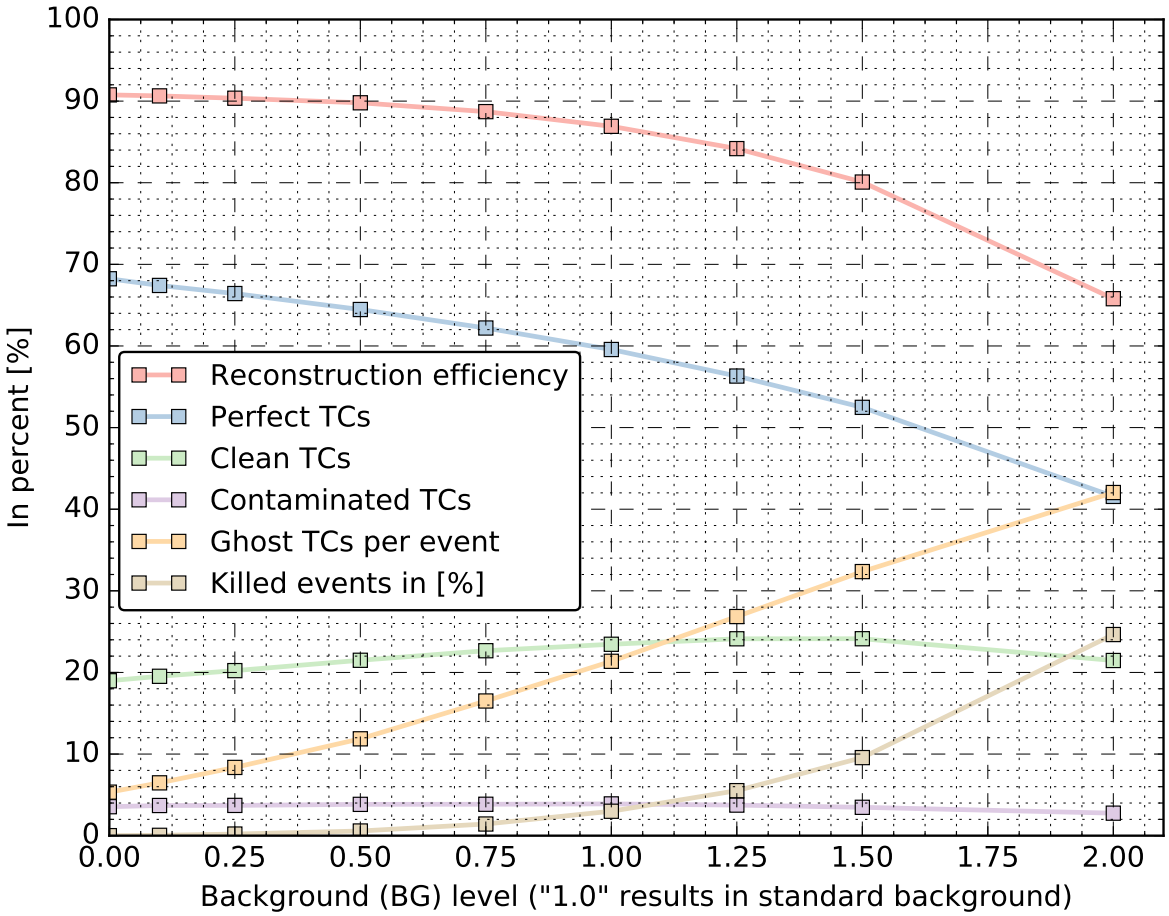


Figure 5.21: The most relevant parameters plotted over the background (BG) level including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

Figure 5.21 is the last plot of this section and gives an overview of many relevant parameters describing the performance of the VXDTF in relation to the background level, which is sketched up to the level of 200% background. The overall reconstruction efficiency — $\widehat{=}$ perfect + clean + contaminated TCs — drops from about 90.8% at 0% background to 86.9% for standard background level. The perfectly reconstructed tracks start at 68.2% at 0% background and drop to 59.6% at 100% background which means that this parameter is more sensitive to the background level than the overall value. TCs which were classified as clean increase from 19% to 23.5% at the full background level, while the level of contaminated TCs is barely affected and stays in the region of 3.5–4%. This means that those tracks which are not found as perfect any more are then either

clean — meaning missing hits, but no foreign hits added — or not found at all. The overall number of TCs increases, as already demonstrated in Figure 5.18b, which leads to a rising number of ghost TCs, where the purity was less than 70% and therefore the associated original tracks were lost. The fraction of ghost TCs in relation to the total number of TCs found increases from 5.3% at 0% background level to 21.4% at 100% background level; one out of five TCs must then be identified as bad by the track fitting modules to prevent them from degrading the final results. The last parameter plotted in Figure 5.21 is the percentage of killed events of the sample used. While the value is negligible at 0% background level, where only a single event was discarded, about 898 of the 30,000 events in the sample had to be aborted in the VXDTF, which is 3% of the whole sample. This number is rather high, especially for a mechanism which should be triggered only in rare cases, and is obviously the main cause for the efficiency drop observed in this plot, since killed events do not contain any final TCs at all. To investigate the impact of this parameter, another run of the sample has been done using a 10 times higher threshold. This will be discussed in the next section.

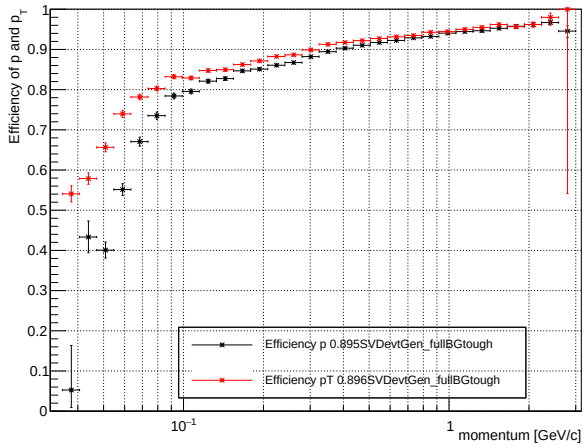
5.4.4 Behavior with higher thresholds

In Section 5.4 the performance of the VXDTF at different levels of background has been analyzed. It became clear that the parameter "killEventsForHighOccupancy" discards too many events and therefore is a bottleneck for the efficiency. To cross-check what a higher threshold leads to, the same sample was processed again, but with a 10 times higher threshold as in the previous section. This means that in this run events were discarded only if more than 50,000 two-hit combinations were accepted. To easily distinguish between these settings in this section, the setting using the lower threshold is called the *standard* setting, while the one using the higher threshold is called *tough* setting. While expecting to get higher efficiencies, an increase in time consumption has to be accepted too.

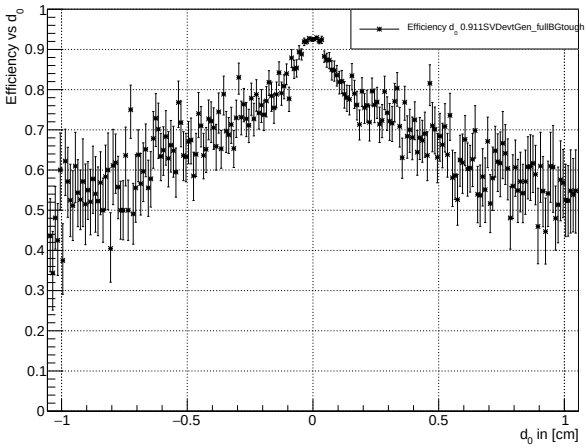
Figure 5.22 shows the plots of the efficiency versus the (transverse) momentum and d_0 at $\Upsilon(4S)$ events with 100% background level. Compared to Figure 5.11, where the standard setting the was used, the efficiency in Figure 5.22a is higher: 89.5% efficiency over p and 89.6% over $p_\top$, which is about 2.5% higher than the run with the standard setting. Beside that no specialties can be identified, which means that adding more high occupancy events does not have any relevant impact on momentum dependent efficiencies.

Figure 5.22b shows the efficiency in relation to d_0, plotted in the range $-1\,\text{cm} \leq d_0 \leq 1\,\text{cm}$. The structure is again practically identical with the corresponding plot in Figure 5.14, with the exception that the efficiency is now 2.6% higher. Again no visible impact of high occupancy events can be seen.

The efficiency versus the distance to the origin in x-y and z is plotted in Figure 5.23 and is again very similar to the version shown in Figure 5.13, with the exception of having higher efficiencies too. The difference in efficiency is about 2.6% for both plots compared to the version with the standard settings. Again only some bins differ in their relative

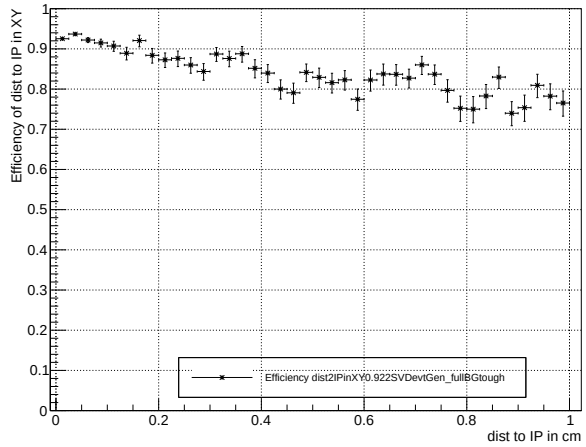


(a) Efficiency versus momentum (black, total efficiency over the p range displayed = 89.5%) and transverse momentum (red, total efficiency over the $p_\top$ range displayed = 89.6%).

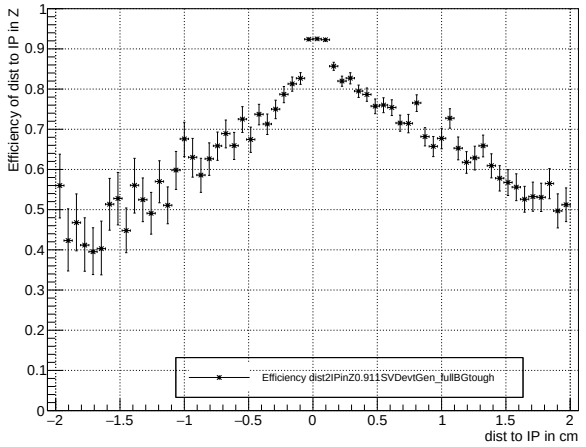


(b) Efficiency versus d_0, total efficiency over the d_0 range displayed = 91.1%.

Figure 5.22: Efficiency versus (transverse) momentum in (Figure 5.22a) and d_0 (Figure 5.22b) for the case of 100% background - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.



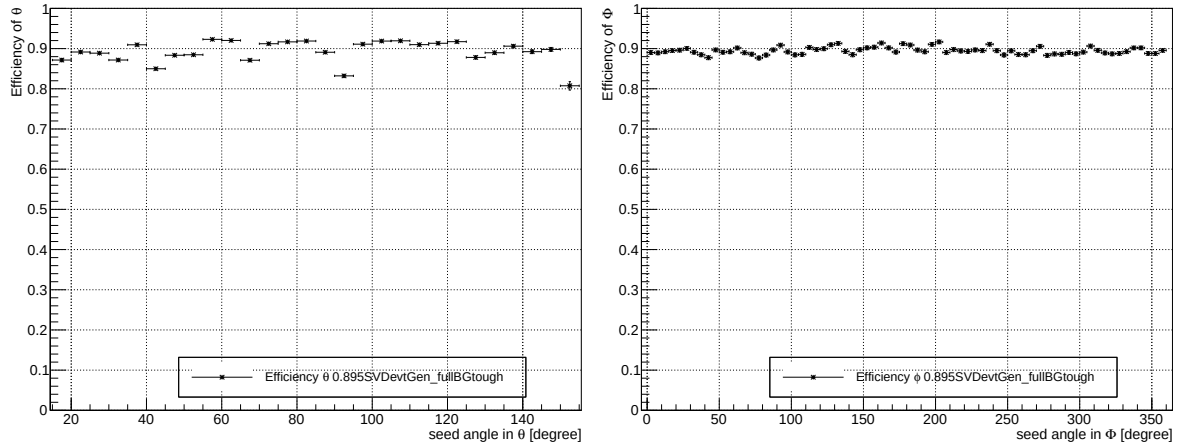
(a) Efficiency over vertex position in x-y, total efficiency over the x-y range displayed = 92.2%.



(b) Efficiency over vertex position in z, total efficiency over the z range displayed = 91.1%.

Figure 5.23: Efficiency versus distance of primary vertex from origin in xy (Figure 5.23a) and in z (Figure 5.23b) for the case of 100% background - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

values compared to the other version, and again no real impact of the high occupancy events can be observed.



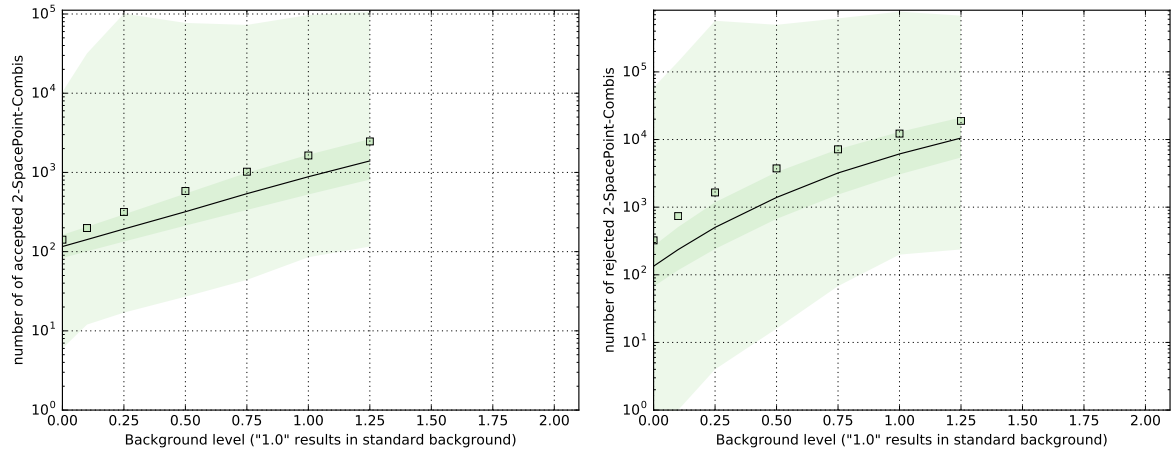
(a) Efficiency versus Theta, total efficiency over the θ range displayed = 89.5%.

(b) Efficiency versus Phi, total efficiency over the Φ range displayed = 89.5%.

Figure 5.24: Efficiency versus Theta (Figure 5.24a) and Phi (Figure 5.24b) for the case of 100% background - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

In Figure 5.24 the efficiency is plotted in relation to the θ angle and the Φ angle of the momentum seed of the particle tracks to be reconstructed. Again the plots look nearly identical to the versions with standard setting, with the exception of an overall higher efficiency, which is again about 2.5% higher. All bins over in Figure 5.24a look practically the same as in the other version, so no effects of the high occupancy events can be seen here either. For Φ in Figure 5.24b at least some bins differ visibly, but the overall structure stays roughly the same and therefore no real insights can be drawn from that plot.

Figure 5.25 shows a box plot band graph for accepted and rejected two-hit combinations, where the black line represents the median, the green squares are the mean values, the dark green band covers the values between the 25% quartile and the 75% quartile and the light green band covers the range between minimum and maximum value. The results are plotted up to the 125% background level, since higher runs take too long to be finished within a reasonable amount of time and therefore were not performed using the tough setting. The mean values stay near the ones plotted in Figure 5.16, but differ more and more compared to the old plot since with an increasing background level the number of accepted events increasingly differs between these two plots. This results in a mean value of 1,637 versus 1,500 accepted two-hit combinations at the 100% background level where the higher mean value is found in Figure 5.25. The maximum values differ even more, where the tough setting has 97,000 accepted two-hit combinations as maximum value while the standard setting stays at 14,814 accepted combinations. The mean acceptance rate for two-hit combinations is worse — which means higher —



(a) accepted two-hit combinations.

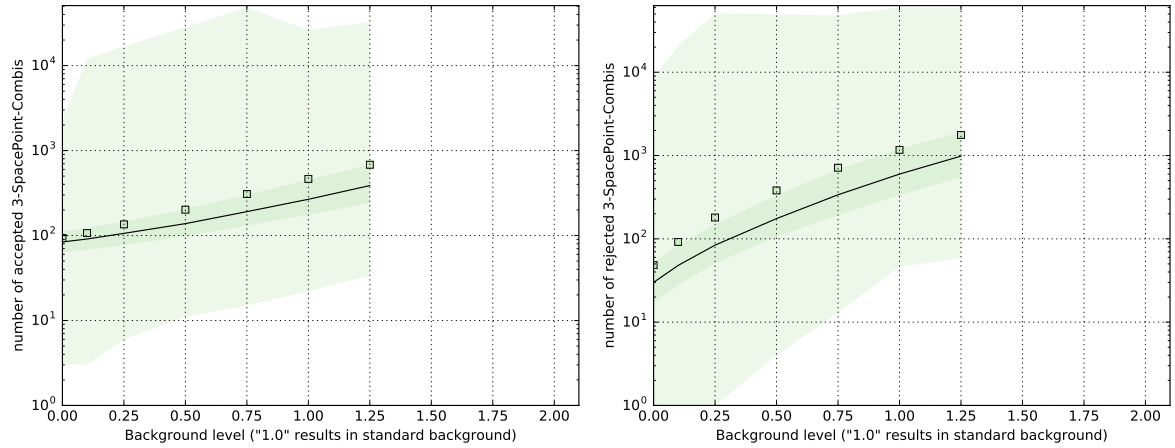
(b) rejected two-hit combinations.

Figure 5.25: The total number of accepted two-hit combinations (Figure 5.25a) and the rejected ones (Figure 5.25b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

in the tough setting, where 11.8% of all two-hit combinations allowed by the SecMap are accepted; a value which is slightly better at 11.5% for the standard setting. For rejected two-hit combinations the picture is comparable to the accepted combination, where the mean value for the tough setting was with 12,239 again a bit higher than with the standard setting having a mean of 11,489. The maximum values again differ visibly, where the tough setting has 777,512 rejected combinations as a worst case while the standard setting stays at 242,423 rejected combinations.

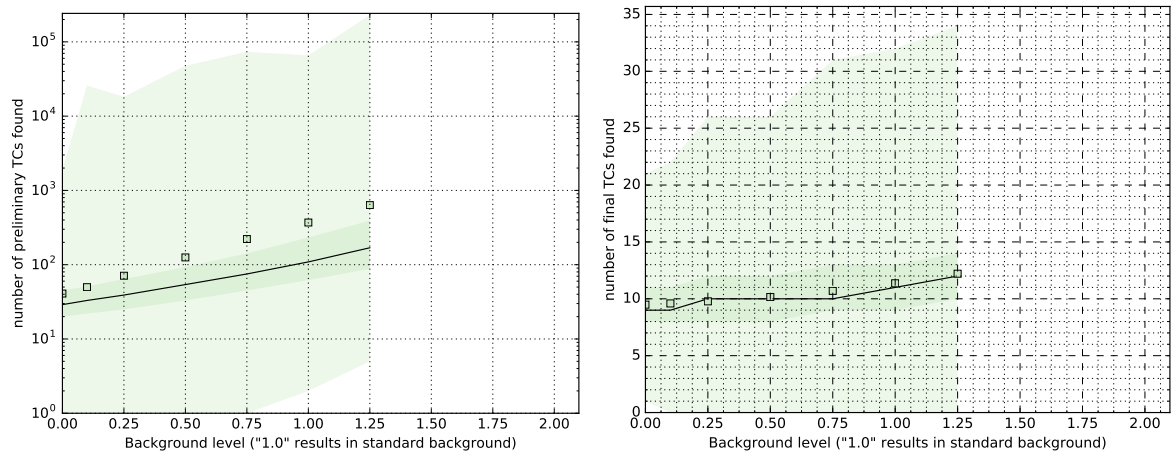
In Figure 5.26 the numbers for accepted and rejected three-hit combinations are again significantly higher than their standard setting counterpart of Figure 5.17. Noticeably different are the mean and maximum values again. The mean values now lie above the 75% quartile, which is not the case in the standard setting. The absolute value of the mean is about 15% higher for the case of 100% background, where the tough setting results in a mean of 464 accepted three-hit combinations, while the standard setting has only 398 accepted combinations. The maximum values differ more, where 26,158 accepted three-hit combinations are the worst case for the tough settings, while the standard setting has a maximum at 5,327 accepted combinations. For three-hit combinations the mean acceptance rate is also degrading in the tough setting, where 28.4% of total number of three-hit combinations are being accepted, while the acceptance rate for the standard setting is 28% and therefore a bit better. For rejected three-hit combinations the mean values are 1,170 in the tough setting versus 1,023 in the standard setting and the maximum values result in 60,094 versus 11,890.

Figure 5.27 sketches the number of preliminary and final TCs acquired using the tough setting again. In comparison with the standard setting plotted in Figure 5.18 for the



(a) accepted three-hit combinations. (b) rejected three-hit combinations.

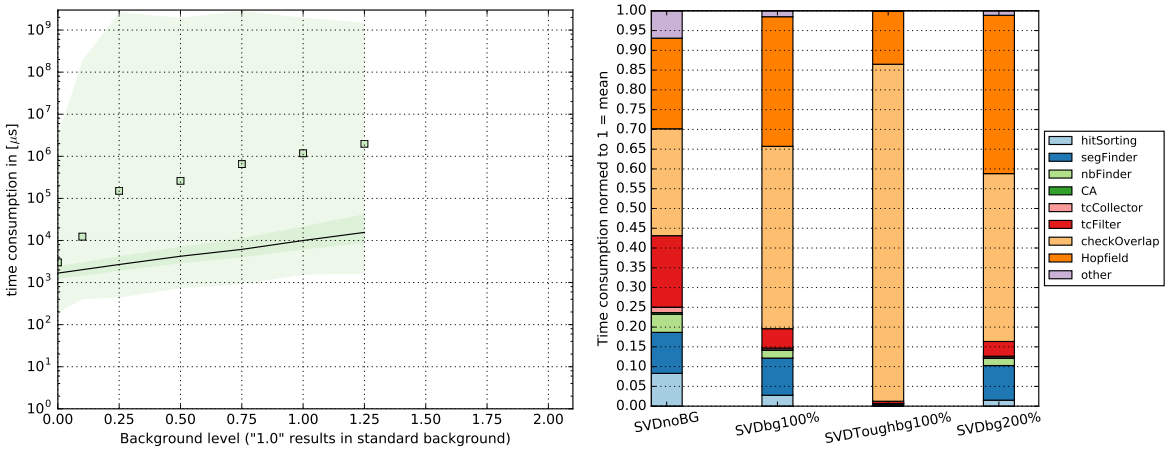
Figure 5.26: The total number of accepted three-hit combinations (Figure 5.26a) and the rejected ones (Figure 5.26b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.



(a) Preliminary TCs per event. (b) Final TCs per event.

Figure 5.27: The total number of TCs found after TCC (Figure 5.27a) and final TCs (Figure 5.27b) for different background levels including $\Upsilon(4S)$ events added - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

case of preliminary TCs, one again can see the impact of the high occupancy events, which pull the mean values upwards and especially the maximum values too. The mean value for the case of 100% background and using the tough setting is with 370 far higher than the standard setting, where the same background level results in a mean of 273 preliminary TCs. This value is impressing, since the increase of about 35% in the mean of preliminary TCs is caused by only a very small number of additional events processed compared to the standard setting. This increase can also be seen at the maximum number of preliminary TCs recorded for a single event, which is 65,708 in the tough setting versus 29,568 in the standard setting. After collecting the preliminary TCs, the VXDTF does simply find the overlaps between them and allows the Hopfield NN to find a clean subset. This effectively means that the number of preliminary TCs is the only factor influencing the time consumption of these two parts mentioned — a fact which will be discussed in one of the following paragraphs too.



(a) Total time consumption per event.

(b) Relative time consumption of the sub-parts of the VXDTF.

Figure 5.28: Dependency of the time consumption in relation to the background level (Figure 5.27b) and the relative time consumption of the individual subparts of the VXDTF - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

The number of final TCs is also affected, which can be seen when comparing Figure 5.27b using the tough setting with Figure 5.18b using the standard setting. The mean number of final TCs for the case of 100% background added increases from 10.9 TCs per event using the standard settings to 11.4 TCs per event in the tough settings. Compared to the preliminary TCs this increase is very moderate and even the maximum values stay at a reasonable level, where the tough settings produce the same maximum of 32 TCs as the standard setting again.

Figure 5.28a illustrates why the tough setting is not used as the standard setting, although the performance of the VXDTF is better when using the tough setting. The

reason is clear to see: 2.5% higher efficiency comes with the cost of over 23 times more time consumption on average. An average time consumption per event of about 1.2 s is definitively too slow by orders of magnitude to be accepted as the final result. The longest event recorded took over 1960 s to finish — a time span which is far beyond any acceptable threshold. Figure 5.28b already indicates a starting point where to improve the current code: For the case of 100% background and tough settings— which is presented in the third column of this plot —, about 85% of the time is consumed for finding overlapping TCs, and practically all of the rest is consumed in the Hopfield NN. This is where the

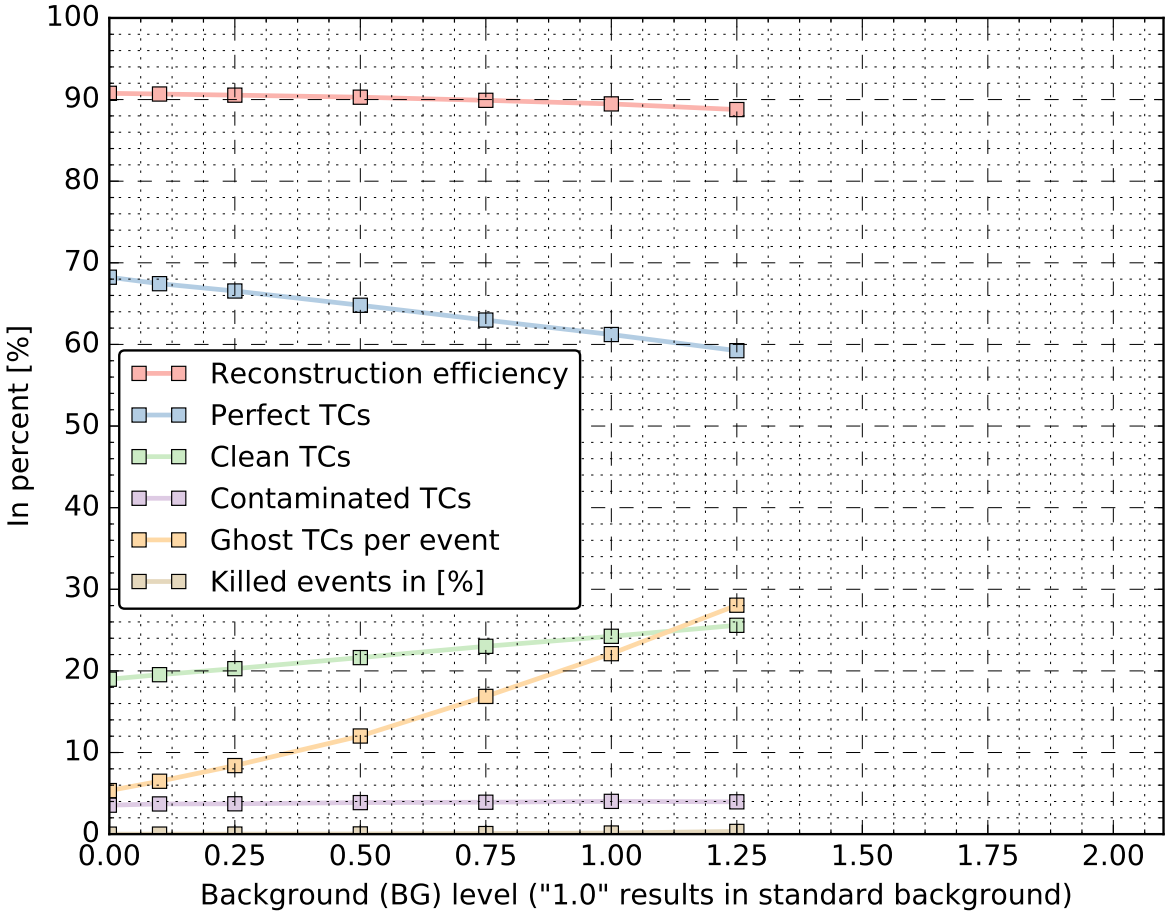


Figure 5.29: The plot summarizes the most relevant parameters over the background level - taken from a MC sample of 30,000 $\Upsilon(4S)$ events described in Section 11.1.

high increase of the mean number of preliminary TCs affect the performance and speed of the aforementioned components of the VXDTF. The combinatorial problem of finding overlaps clearly dominates here. But both parts of the code are not yet optimized for speed and this plot just makes this more clear. Of course refactoring that part will not

be the only way to solve the bad performance regarding time consumption; another essential part is to improve the pre-filtering done at two and three-hit level. The better the filtering works the less time consuming the latter parts of the code can be. Additional and independent of the work to be done in the VXDTF working on optimized SpacePoint combining will help a lot too. As already mentioned in Chapter 3, many parameters of clusters already stored indicate effective markers to filter background clusters.

Figure 5.29 serves as an overall performance plot, where many relevant markers are plotted. Compared to Figure 5.21 one can see that the efficiency does not drop at the same rate as in the standard setting any more. Additionally the number of killed events due to the tough "killEventsForHighOccupancy" threshold is now far lower: only 41 of 30,000 events had to be discarded, whereas the number for the standard setting is 898 of 30,000. But the number is not 0, which means that even the tough setting has not shown the worst case scenario — a statement which makes clear that the current performance at the current version of the basf2 framework of the VXDTF is not yet at a level to be used for the worst case scenarios to be expected. To solve the issue of time consumption, the ways discussed within the paragraphs of this sections should lead to a satisfying result in the end.

5.5 Conclusion

The insights won from analyzing the VXDTF performance for the cases of background only, $\Upsilon(4S)$ events only and for $\Upsilon(4S)$ events with gradually increased background levels are summarized here.

Background only events Section 5.3 has shown the performance of the VXDTF when facing realistic background, but no other tracks. For the case of 100% background a mean of about 500 SpacePoints per event lead to about 1,000 and 200 accepted two- and three-hit combinations, respectively. The mean of preliminary TCs then is 100 and the final TCs have a mean of only two TCs. These TCs are found within a mean of about 15 ms per event.

General insights can be retrieved from Section 5.2 and Section 5.4:

- The impact of the detector geometry itself is small, no major effects of overlapping or slanted sensors could be distinguished. Gaps between sensors seem to have visible impact, though.
- Momentum and its transverse component have a strong impact on the results. As expected, low momenta are harder to deal with and limited not only by the radial distance of the layers to the origin, but also by material effects.

- The particle charge does not seem to have a strong impact in the tracking performance. At higher d_0 values small deviations depending on the charge can be observed.
- The position of the vertex and the d_0 values of the tracks show the biggest impact on the efficiency, where tracks coming from the origin are clearly favored. The cause of this drop is not clearly identified at this point. A possible explanation is the choice of the cuts used for each Filter and Sector combination stored in the SecMap, which filter outliers independently and therefore increase the probability to filter good combinations.
- No bin in the efficiency plots versus θ, Φ, the start vertex position in xy and z and d_0 reaches the 100% mark, which indicates that a general bottleneck is keeping the VXDTF from performing better. A clear suspect can not be found, but further studies should contain runs with preselected reference tracks, where missing clusters or strong kinks are filtered out; such a set was not available for this study.
- A small difference of efficiency between the forward and the backward region is observed, which might be connected to the difference in sample size for the forward and the backward region and therefore might have an impact on the cut quality.

$\Upsilon(4S)$ events with background

- The background seems to magnify efficiency drops due to gaps between sensors.
- Background lowers the overall efficiency but is not extraordinarily weak at any of the parameters checked.
- Time consumption and the effects connected to the parameter ”killEventsForHighOccupancy“ are the most pressing issues revealed by this study.

Following the insights listed above, optimization of code parts is essential and a better training of the SecMap is needed. It also became clear that the filtering done by two and three-hit Filters is an essential step of the VXDTF. Being two years away from the actual start of the experiment and some more years before full instantaneous luminosity is reached, there is still enough time to find the optimal trade-off between time consumption and efficiency. An essential task which can be done independently of the VXDTF is the intelligent combination of clusters to SpacePoints, which should have a big impact on the time consumption of the TF. Many possibly fruitful approaches to deal with that are already mentioned in Chapter 3 and are not repeated here.

CHAPTER 6

COMBINED BEAM TEST 2014

6.1 Introduction

Big high-energy physics experiments take years, sometimes even decades of preparation time. In the case of Belle II hundreds of scientists and students have worked on the project since the publication of the "Letter of Intent" in April 2004 [44]. Many sub-groups are working on specific tasks in hardware and software, each one crucial for finally having an experiment ready to take data. To minimize possible show stoppers at the startup of the experiment, extensive tests and quality assurance are needed.

One of the big milestones in reaching a fully functional detector, including its readout, is a successful beam test, where many different groups are closely working together to get a "live" test of the sub-parts. This is not only essential for checking if the sub-parts are working individually, but also for ensuring that the parts fit and work together.

6.2 Experimental setup

In January 2014 the first combined VXD beam test took place at the DESY laboratory in Hamburg, Germany. This was the first time PXD and SVD sensors were used in the same beam at the same time. The task was to establish a raw version of the planned read-out for the Belle II experiment, including ROI-selection for the PXD using the HLT and the DATCON. The combined beam test was also the first time an on-site HLT (Section 6.3.2.1) was present and allowed on-line reconstruction to be performed. This was therefore the first time the VXDTF had to face reality instead of MC data.

Figure 6.1 illustrates the detector configuration of the beam test setup, which is described in Section 6.2.2. Figure 6.2, Figure 6.3 and Figure 6.4 show photographs made on-site by Michael Schnell and show the magnet, the PXD and the SVD. The read-out

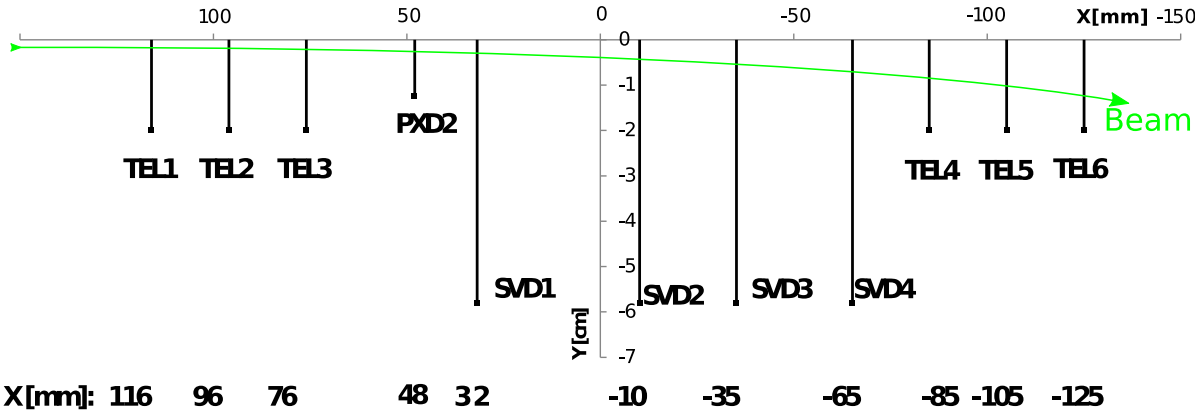


Figure 6.1: A schematic view of the beam test setup. The horizontal axis is the x axis, the vertical one is y and orthogonal to both is the z axis, which is not marked in the sketch. The magnetic field bends the beam — which enters from the left and points to the negative x direction — into the negative y direction. Photo courtesy of Tobias Schlüter.

chain for the DAQ is illustrated in Section 6.2.3, while the internal behavior of the HLT is sketched in Section 6.2.4. An analysis of the beam test focusing on the HLT can be found in Section 6.3.2; a more detailed off-line analysis is located in Section 6.3.3. More details about other parts of the HLT are described in [45].

6.2.1 Beam source

The site of DESY was chosen for the combined beam test because the aim was to test a close-to-final setup. Unlike CERN[69] — which can provide hadron beams for beam tests with an energy up to several hundred GeV — DESY can provide a electron beam with a few GeV and allows a trigger rate of a few kHz. In the combined beam test the beam energy was in the range from 2 to 6 GeV, with a rate of 800–1000 Hz.

6.2.2 Detector setup

The detectors — a PXD-sensor, 4 SVD-sensors and 6 Telescope[70]-sensors — were placed in a dry box inside the coil of a solenoid magnet. The magnet has a cylindrical shape and can provide a magnetic field of up to 1 T. During the beam test two magnetic field setups were used: No field and the maximum field strength of about 1 T (to be precise: 0.978 T) of magnetic field strength. Inside the dry box the detectors were placed as described in Figure 6.1. The specifications of the sensors used for the beam test (as listed in Table 6.1) differ from the sensors used in the final VXD design for the Belle II

[69] CERN: Conseil Européen pour la Recherche Nucléaire
[70] Sensor phalanx optimized for beam tests with extremely high sensor resolution.

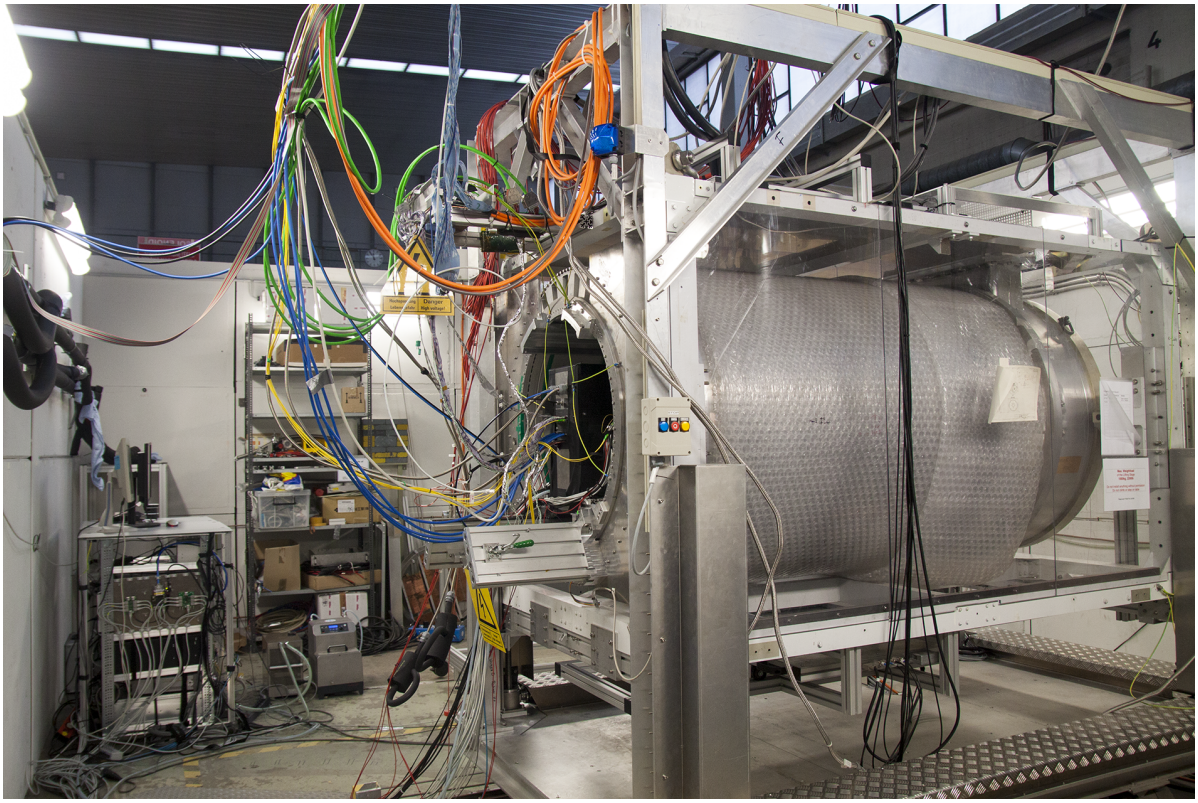


Figure 6.2: A photo taken from the magnet coil around the sensors. Photo courtesy of Michael Schnell [26].

experiment (listed in Table 2.2). This is due to the fact that the sensors used for beam test were pre-production test samples and did not yet have final production quality.

Contrary to initial plans, only one PXD sensor could be used since the other prototype was destroyed during beam test preparation. However, the most important tests like the ROI-chain and the common powering of the VXD detectors were carried out successfully. Four different detector types were used for the beam test, of which only two are actual parts of the VXD detector. They are listed here in geometrical order from upstream to downstream the beam:

- **Scintillators:** Four fast scintillators were used for initial triggering. They were mounted in two pairs, one in front of and one behind the detector setup. Their configuration and connection to the TLU[74] allowed reducing the recording of empty events (the so-called "dark rate").

[74] Trigger Logic Unit

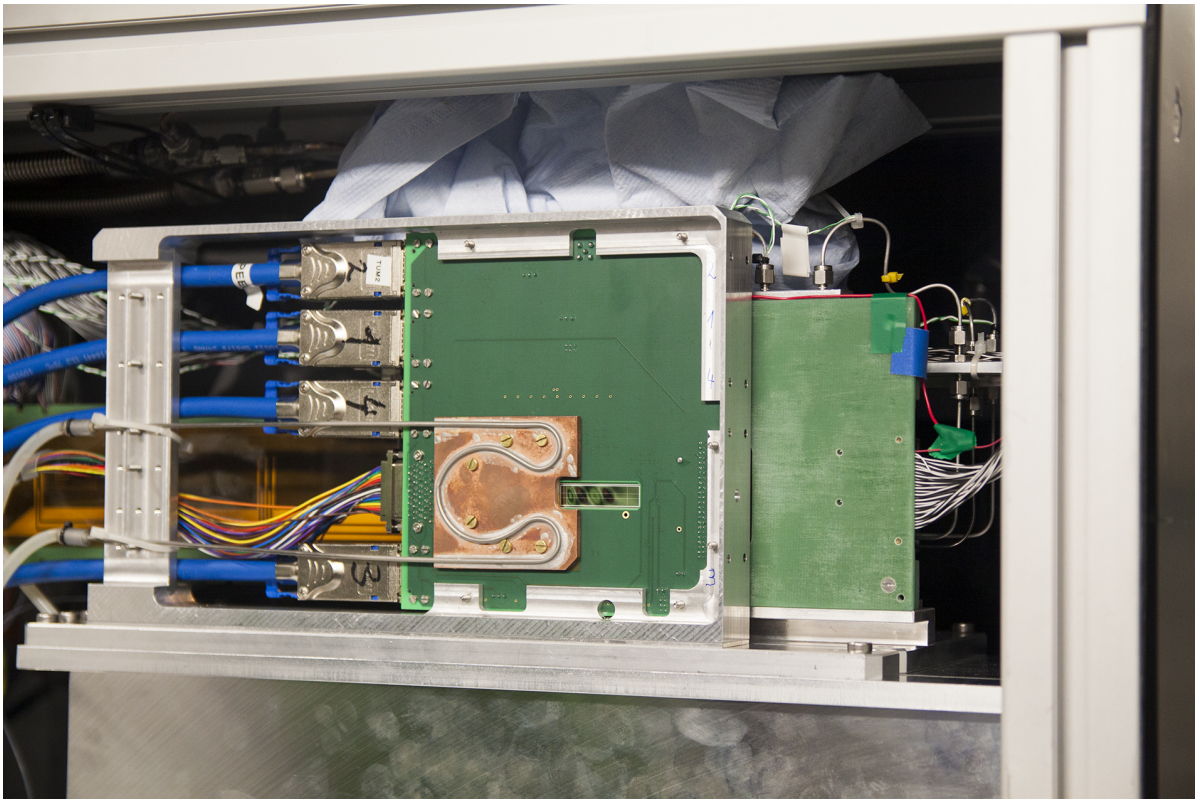


Figure 6.3: A photo taken from the PXD sensor mounted on a readout board. The actual sensor is the small green rectangle next to the copper cooling plate. The SVD sensors are mounted behind the PXD readout board and are not visible in this picture. Photo courtesy of Michael Schnell [26].

- **AIDA[75] Telescope:** six layers of pixel detectors of CMOS[76] type [46] provided by the AIDA collaboration were used. These sensors offer a read-out in kHz-speed, have a very high resolution (pitch size of only 2 µm) and are installed on two rails with 3 sensors each. Their read-out is independent of the VXD sensors and was merged off-line with the VXD data.
- **DEPFET PXD:** one PXD sensor of a test batch was provided and mounted on a read-out board as can be seen in Figure 6.3. Its specification differ in many parts from the final one, as this sensor is thinner (50 µm instead of 75 µm) and has a different size and resolution. The output was merged in the EVB2[77] with the SVD data.

[75] AIDA: The AIDA-2020 collaboration
[76] Complementary Metal-Oxide-Semiconductor
[77] EVent Builder II used as final event builder in the DAQ chain.

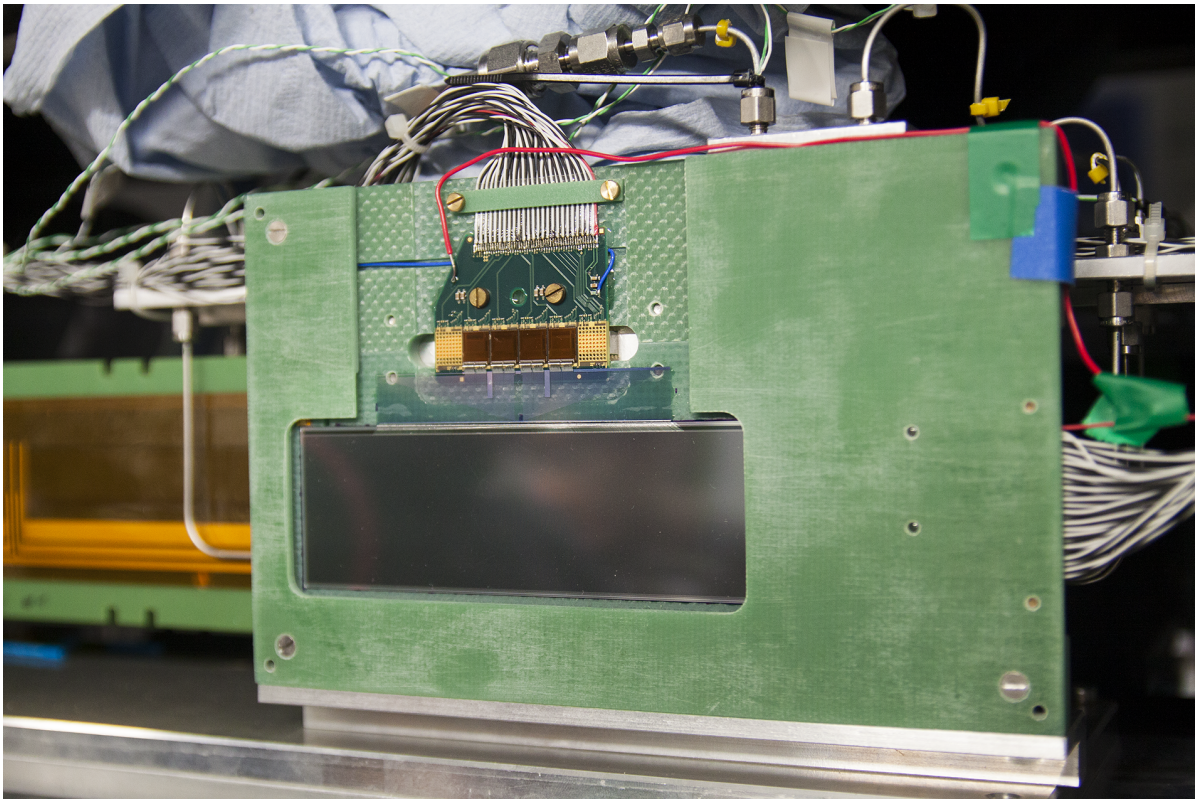


Figure 6.4: A photo taken from the SVD sensor mounted on a readout board. The sensor is the reflecting rectangular shape in the center of the picture. Photo courtesy of Michael Schnell [26].

- **SVD:** four sensors of the SVD were used, which were close-to-final in their overall specifications but had some other issues which had to be dealt with. Their noise level was much higher than in the final sensors, since some of them had already been used for other beam and radiation tests. Moreover, the sensor 5.1.5 had a broken APV 25 read out chip, which lead to the loss of about 9.6 mm of the active sensor area in direction of $r\phi$. To compensate for this, the sensor was re-positioned by about 5 mm to sit in the center of the beam.

6.2.2.1 Issues

During a beam test problems invariably turn up, but they are an important part of the process. Unexpected incidents make aware of possible issues which were not visible during isolated preparations. One of these unexpected problem was noise in the SVD after the PXD was installed. Since the SVD had worked fine for some days before the PXD was plugged in, the source of the noise could be narrowed down to some interactions between the SVD and the PXD hardware. The reason was the complexity of the installation,

Table 6.1: Table of specifications for the PXD (L2) the SVD (L3 — L6) and the Telescope sensors. The first number of the VxdID[71] represents the layer number used, while 'position' marks the position of the detector downstream the beamline, where the particles enter the setup at the highest positive value and leaves the detectors at the smallest value. The differing values of layer 5 are caused by a broken APV 25[72] chip which reduces the active area by $1/6$ in direction of $r\phi$.

VxdID	type	position [mm]	active area u×v [mm]	thickness [µm]	pixels/strips u×v	pitch u×v [µm]
7.2.1	EUDET[73]	116	10.6×21.2	50	576×1152	2×2
7.2.2	EUDET	96	10.6×21.2	50	576×1152	2×2
7.2.3	EUDET	76	10.6×21.2	50	576×1152	2×2
2.1.2	PXD	48	9.6×48	50	192×640	50×75
3.1.3	SVD	32	38.4×122.9	320	768×768	50×160
4.1.4	SVD	-10	57.6×122.9	320	768×512	75×240
5.1.5	SVD	-35	48×122.9	320	640×512	75×240
6.1.6	SVD	-65	57.6×122.9	320	768×512	75×240
7.3.4	EUDET	-85	10.6×21.2	50	576×1152	2×2
7.3.5	EUDET	-105	10.6×21.2	50	576×1152	2×2
7.3.6	EUDET	-125	10.6×21.2	50	576×1152	2×2

since one half-ladder, i.e. a single sensor and its read-out, of the PXD already needs several different voltages to run, which produce different alternate currents provided by the power supply. During the beam test the exact source of the problem could neither be eliminated nor located, since such severe noise resonances between the detectors were not anticipated beforehand. Therefore highly specialized equipment for tracking down such a problem was not at hand. This led to the development of a detailed grounding scheme by the SVD hardware group after the beam test to make sure that all devices coupled to each other use exactly the same point for ground. To achieve this, all devices and sub-parts, every chip and component which needs ground has now to be connected directly and without bypasses to the same ground. Detailed studies of that effect were done in Zaragossa, Spain, by R. Thalmeier, H. Yin, F. Arteche *et.al.* and allowed to develop proper measurement tools to be able to locate the exact source of the observed interferences during the next combined beam test currently scheduled for April 2016.

6.2.3 Data acquisition setup

As can be seen in Figure 6.5, the data acquisition for the beam test consisted of numerous components. To be recognized as an event, the particles of a beam have to trigger four scintillators placed in a crossed pair each before and after the detector setup (shown in

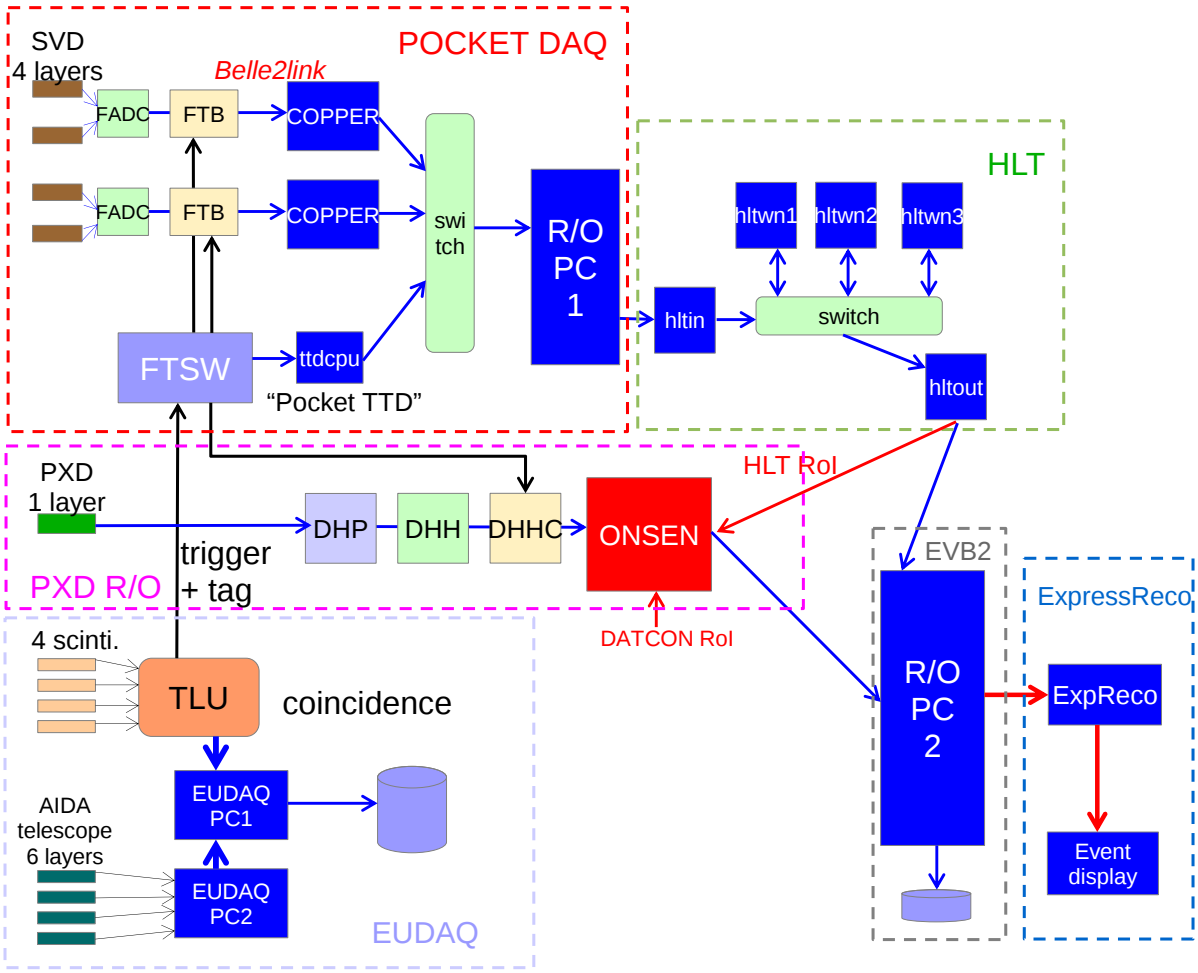


Figure 6.5: An overall diagram illustrating the readout of the combined VXD DESY test beam. A description can be found in section 6.2.3, Figure courtesy of Ryosuke Itoh [45].

the dashed box marked as EUDAQ[78] in Figure 6.5). Their signals are recorded by the TLU, a trigger logic unit, which informs the EUDAQ computers to record data from their AIDA Telescope and informs the POCKET DAQ[79] system which is responsible for the SVD readout. The first step of the POCKET DAQ after being informed by the TLU is to activate the FTSW[80], which is responsible for triggering the data recording of the different detectors of the VXD.

In the SVD system the FTSW triggers the FTB[81], which asks the FADC[82] chips to

[78] DAQ system used by the EUDET projects to read out their Telescope
[79] Data AcQuisition
[80] Frontend Timing Switch
[81] Finesse Transmitter Board
[82] Flash Analog-to-Digital-Converter

send their information temporarily stored on a ring-buffer to the COPPER[83] device. The COPPER then streams their information combined with the TTD[84]-output to flag the data with the proper timing information, which is then directed into the HLT (described in detail in 6.2.4).

In the PXD system the FTSW informs the DHHC[85] to load the data handled by the DHP[86] into the ONSEN[87], a device taking ROI information from the HLT and the DATCON. The ONSEN is the only device with live access to the PXD data and has only a small time window for rescuing pixel information from the PXD.

The output of the HLT and the ONSEN is then merged by the final event builder EVB2 to be stored on disc and to be used by the Express Reco running basf2 code with the full VXD information off-line available.

6.2.4 High level trigger setup

The HLT for the combined beam test was a simplified version of the HLT to be used in the Belle II experiment. It had to meet the following requirements which are also described in [45].

1. reading of detector data formats;
2. working with basf2 software, the same to be used as for the off-line Express Reco, but using different settings for the reconstruction modules;
3. efficient track finding and track fitting;
4. extrapolation of TCs to PXD planes;
5. definition of ROIs;
6. successful communication of ROIs to the ONSEN system.

Data collected by the SVD detector was delivered event-wise and had then to be distributed to the worker nodes running the basf2 software. The HLT architecture is designed to be highly scalable, and therefore only a small number of the final computing nodes was used for the beam test covering the essential tasks. The internal hardware structure of the HLT consisted of five computing nodes which were connected as follows:

- The nodes *hltin* and *hltout* were set up as twelve-core systems each and handled the event-wise data coming from the other DAQ devices and assign individual events to the worker nodes.

[83] COmmon Pipeline Platform for Electronics Readout
[84] Trigger Timing Distribution
[85] Data Handling Hybrid Controller
[86] Data Handling Processor
[87] ONline SElector Node

- The worker nodes *hltwn1*, *hltwn2* and *hltwn3* consisted of 24-core systems each and dealt with incoming events by applying the data flow as described in Figure 6.6. Parallel processing was established by assigning one event to each core.

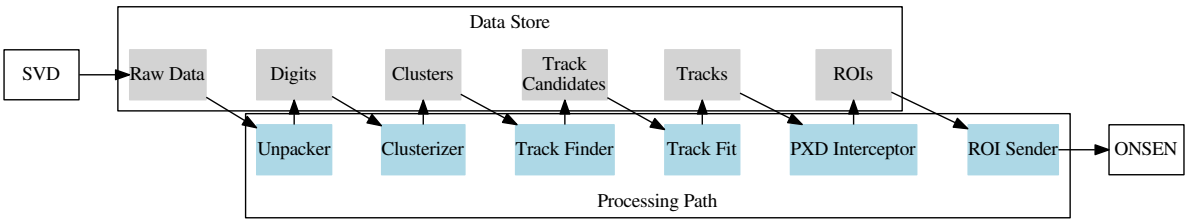


Figure 6.6: This Figure is a data flow diagram of a single HLT worker node, where the module 'Track Finder' is the VXDTF. The diagram is a modified version of the Figure presented in [45]

The data flow was executed as follows: The raw data is unpacked by the Unpacker, which converts it in Digits of the SVD. These Digits are then clustered by a Clusterizer into Clusters. The VXDTF is then using these Clusters for track finding and uses genfit::TrackCand as output. The TC is then fitted by a track fitter to get fitted tracks. These tracks are then used as a seed for the PXD Interceptor module to create ROIs by extrapolating the TC using its parameters to the PXD. These ROIs are then sent via the ROI Sender module to the ONSEN.

More details about the track finding approaches used in the combined beam test are described in Section 6.3.

6.3 Track finding at the combined beam test

The VXDTF is not the only TF implemented for SVD tracking. Along with the HLT, which uses the VXDTF, a FPGA-based method using a Hough transformation was implemented in the DATCON and used for ROI-finding, similar to the HLT-implementation of the VXDTF.

For the beam test analysis of the VXDTF three runs were selected to cover both on-line and off-line data processing. While run 104 was used as an example for an on-line run, the runs 470 and 510 were used for off-line analysis. Details about the run-specific settings can be found in Table 6.2. A short summary of the implementation of the FPGA-based TF can be found in Section 6.3.1, while the VXDTF and its differences from the standard version are described in Section 6.3.2.

Table 6.2: Settings of the runs used in the track finding studies.

run	field strength	particle beam	beam energy	events	analysis
104	0 T	e^-	5 GeV	5000	on-line/raw
470	0 T	e^-	3 GeV	4687	off-line/preselected
510	1 T	e^-	3 GeV	10282	off-line/preselected

6.3.1 FPGA based tracking in the DATCON

The FPGA based SVD only trackfinder implemented in the DATCON is discussed with more detail in [26]; a basic explanation of the Hough transform can be found in Section 2.7.1.2.

A different tracking approach compared to the VXDTF has been implemented in a FPGA, which uses a fast Hough transform in the $r\phi$ plane. The line parametrization written down in Eq. (6.1) is chosen to behave well in any situation:

$$d = x_i \cos\phi + y_i \sin\phi, \tag{6.1}$$

where d is the closest distance of the line to the origin and ϕ is the azimuthal angle of the vector pointing to the point of closest approach to the origin. Since charged particle tracks are bent in the $r\phi$ plane, a conformal mapping that maps circles through the origin to lines is carried out before the Hough transformation is applied. Then a fast Hough algorithm searches for intersections in the Hough parameter space, where hits lying on the same line correspond to lines intersecting at the same point in Hough space. To find the intersections faster, the Hough space is divided into two subspaces per parameter. Then for each subgroup it is checked if lines are entering the subspace; for each subspace with enough lines entering, the divider step is repeated and the line check repeated until the subspaces get smaller than the resolution of the measurements or not enough lines are intersecting. This is considerably faster than binning the whole Hough space without checks. In the end all areas are collected where enough intersections are detected. The corresponding hits are then collected as TCs.

6.3.2 Tracking in the the HLT

The differences to standard Belle II tracking are numerous: for the VXDTF itself a new SecMap had to be trained to match the geometry of the beam test. While the sensor positioning itself did not reveal any new issues, other aspects did. The design of the VXDTF used for the combined beam test was not decoupled from the detector type, and therefore the support for the Telescope-sensors had to be added. This meant not only adding the detector type itself but also solving the issue of the layer number. Since the carrier of the ladder-, layer- and sensor-ID — the VxdID — can only store the layer numbers of 0-7, the only layer-ID left which was not used up to that date

was the layer 7, while the layer number 0 was already reserved for the IP region. That peculiarity lead to the situation that the inner three Telescope sensors with layer 7 were upstream to the PXD layer 2 and the outer three layers of the Telescope sensors were placed downstream to the layer 6 of the SVD (see Figure 6.1 for more details). This was unfortunate for the VXDTF since it heavily relied on the assumption that smaller layer numbers mean proximity to the IP — which is actually the case for the Belle II VXD. The ordering of layers in distance to the IP is especially essential to the CA since this algorithm is an implementation of a *directed* graph without loops. Another related assumption valid for the Belle II-VXD turned out to be problematic for the beam test: the assumption that the IP would be at (0,0,0) and would lie next to (in case of the beam test: upstream to) the "innermost" layer, which would be the first Telescope sensor of the inner Telescope-mount. Since for the beam test the origin of the coordinate system was placed at the center of the magnetic coil (Figure 6.2) in the xy plane, the origin was placed between layer 3 and 4 of the SVD. The layer numbers and the origin of the coordinate system as indicators for which sensor is the inner one of any pair of sensors could therefore not be used anymore to define the direction of the graph for the CA. The design of the SecMap never foresaw a manual approach for defining the directed relations between sensors since the geometry of the Belle II VXD is far too complex to relate sensors by hand. Therefore the only feasible solution left was to introduce the possibility to set the IP coordinates at any point in the coordinate system and use the distance between the center of the sensor and the IP as a parameter for establishing the order of the sensors. Another thing which turned out to be important for the beam test was to implement another fast fitting algorithm — a straight-line fitter — since most of the runs recorded during the beam test had no magnetic field. As an early precaution for having at least some tracking capabilities for the beam test, a trivial track finder — the BaseLineTF[88] — was implemented. Its task was to do a fast reconstruction and to be a fall-back solution in case the VXDTF would have turned to be out incapable of dealing with the particular circumstances of a beam test. The BaseLineTF was activated only for cases where only one hit per sensor was recorded, and simply combined hits from neighboring layers to a TC.

All these peculiarities had to be considered and led to significant changes in the VXDTF implementation.

6.3.2.1 On-line tracking

Restrictions of the HLT had to be considered during beam test preparation. Former tests of the VXDTF had been done off-line and with MC data only, so that time consumption per event was not a priority. Therefore initial tests proved to be crucial for the beam test preparations. As a first step the time consumption had to be taken care of, which will be discussed in more detail in the next paragraph. Additionally an important part was to

[88] BaseLineTF: BaseLine Track Finder

establish some monitoring capability for on-line estimation of the tracking performance. This is described in the DQM[89] paragraph in this section. Unfortunately both problems had to be studied in closer detail off-line, since on-line information was not available during the beam test due to communication problems between the working groups.

Other aspects relevant to the on-line HLT restrictions are the missing PXD read-out information and the Telescope data which had to be merged off-line. At this early stage of development it is not yet decided whether alignment is to be performed on-line or off-line (e.g. between the runs) in the final experiment. Because of this, at the moment — and during the combined beam test — alignment is performed off-line. This was problematic for tracking since the actual geometry position differed considerably from the ideal case assumed by the framework. That situation had to be considered when tuning the VXDTF for the beam test. More details about the mis-alignment can be found in Table 6.3 and Section 6.3.3.1.

The time consumption threshold per event was calculated as follows. The DESY electron beam provided a maximum rate of about 1 kHz, which results in the same number of events per second. As described in Section 6.2.4 each core of a worker node processes a single event. Initial plans requested the HLT to be run on one worker node or 24 cores. These values were therefore used to calculate the maximum allowed value of the average time consumption per event. This results in $1000/24 \approx 40$ ms per event as a mean value to be achieved to preserve some time for the other modules of the HLT readout-chain, as shown in Figure 6.6. It was requested to limit the VXDTF to use no more than 10% of the total event limit, resulting in a maximum value of 4 ms for the VXDTF.

One problem regarding beam test preparations is to get a realistic estimation of the sensor performance, in particular the noise level, since it has an important impact on the time consumption. Unfortunately realistic estimations could not be determined or even estimated beforehand. This is due to the fact that the SVD sensors could only be provided shortly before the beam test — in the case of the PXD sensor, it was not available at all until the beam test already started. Using MC simulations for time consumption studies was not feasible since these sensors are not of final production quality and therefore their noise levels are much worse and sometimes readout chips are broken. This results in hot pixels or strips and in much more hits than actually produced by the beam particles. Therefore very conservative cuts had to be used for the VXDTF to reduce the severity of this issue.

The VXDTF provides mainly two methods to control the typical time consumption of the TF. One way is to narrow down the Filter cuts for selecting and combining Segments. This is a very powerful tweak which changes the time consumption within several orders of magnitude when facing full Belle II VXD with $\Upsilon(4S)$ events and background added. Unfortunately this method could not be used since the sensor positioning was up to several millimeters off the position of the ideal geometry stored in the framework. The

[89] Data Quality Monitoring

only approach left was therefore to limit the number of possible combinations allowed per event. This parameter is called "killEventsForHighOccupancy" and its impact on Belle II VXD tracking is discussed in Section 5.4.4. One might assume that having only four sensors and ideally one hit in each one does not present a combinatorial issue, but as mentioned above the sensors generally produced much more than one hit per sensor, especially for runs without magnetic field which suffered from secondary particles produced by the primary particle passing the magnetic coil before reaching the test setup. This led to a mean of 1100 possible (useful) hit combinations per event for runs without magnetic field, compared to a mean of 32 possible hit combinations in average to runs with a magnetic field [45]. Therefore a cut was set beforehand to 75 possible hit combinations, which kept the time consumption below 250 μs. More details about this topic can be found in Section 6.3.3, which reports studies done off-line.

6.3.2.2 On-line DQM

On-line DQM is very important for evaluating the performance of the tracking during a run. Since MC information is of course not available, other ways for measuring the performance had to be found. In the case of the combined beam test some basic tests were applied and the plots can be found in Figure 6.7:

1. Counting the number of TCs produced per event triggered by the scintillators described in Section 6.2.2. With a perfect event trigger, in each event at least one TC should be found. More than one TC per event can be found if for example secondary particles hit the sensors too or if noisy strips are triggered in a particular way. No TC can be found in an event if the occupancy is too high and therefore the event is dropped by the VXDTF, or if the trigger fired for other reasons than trackable beam particles. A plot with results for run 104 is shown in Figure 6.7*a*, where in about two out of three events a TC was found.

2. Comparing the seed position provided by the TF with an extrapolated point at the sensor created by using the seed momentum to extrapolate to the last sensor and back to the first sensor again. This is a test for determining the quality of the momentum seed and expects all hits to be clustered around the median line. Figure 6.7, subfigures *b)* and *c)* show the results for the u and the v direction, respectively. In run 104 about 10% of the tracks could not be fitted for reasons not determined, and of those that could be fitted, about 1% are significantly off the median line.

3. Comparing the seed momentum provided by the TF with the fitted momentum. This is another way to test the quality of the seed provided by the TF. The x component of the momentum vector points upstream, which is mostly parallel to the x axis due to the choice of the orientation of the coordinate system used for the beam test. Since the length of the momentum vector is set as a parameter by

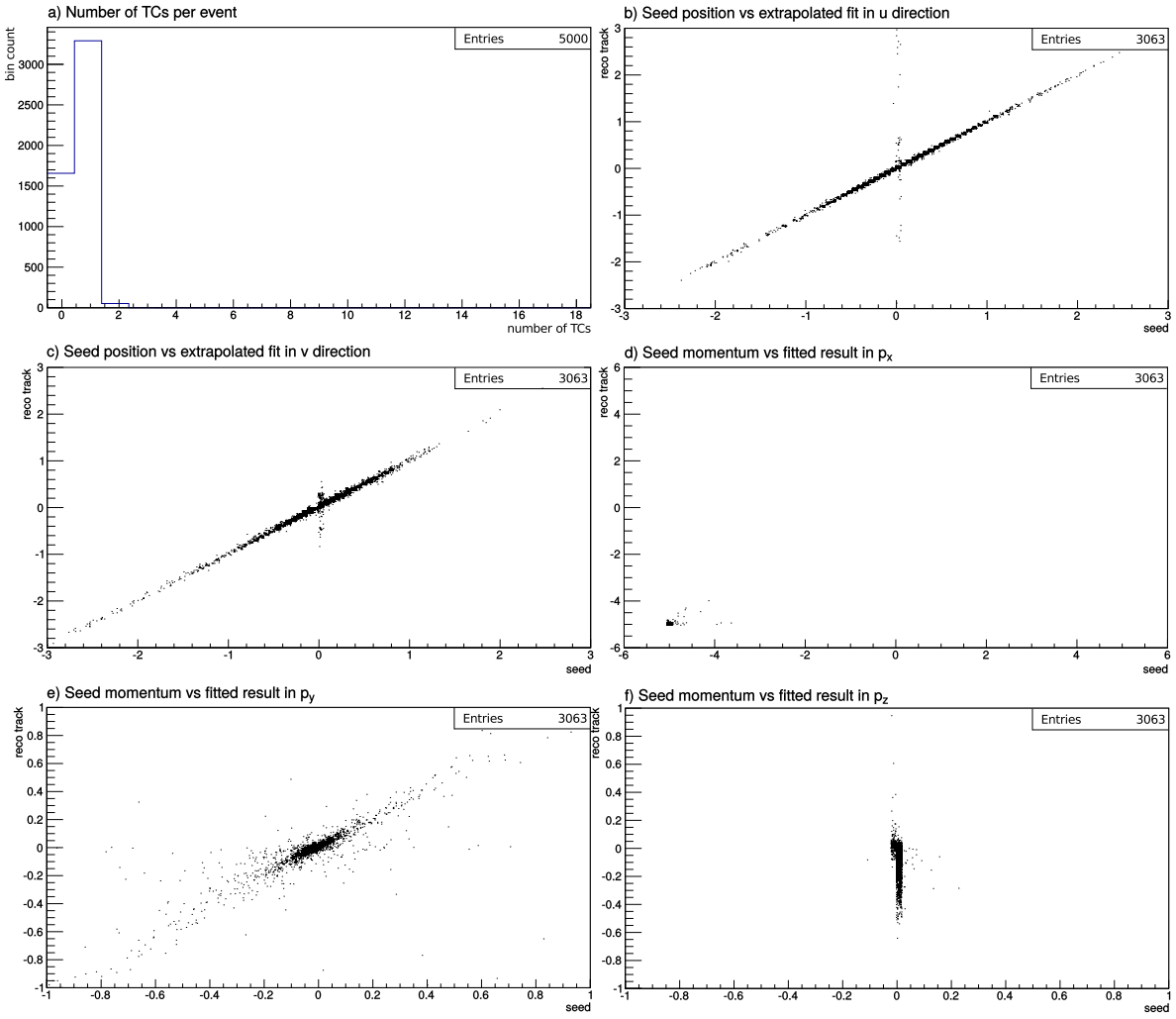


Figure 6.7: This collection of plots contains examples of the plots taken from run 104 in the combined beam test. a) shows the number of TCs detected per event, ideally it should have been one per event. b) and c) show the actual seed hit position versus the extraoplated one by the fitter for the case of u and v direction. Both should ideally form a diagonal line. d), e) and f) show the seed momentum estimation provided by the TF versus the momentum seed determined by the fitter. One should note the differing number of entries for the different plots: subfigure a) counts the total number of events, the other plots used only the TCs which could be fitted, which is less than the number of events. More details can be found in 6.3.2.1. Plots b-f: courtesy of Tobias Schlüter.

the VXDTF due to the fact that without magnetic field one can not estimate the momentum of the tracked particle, Subfigure *d)* shows the x value of the momentum seed and is dominated by the hardcoded momentum parameter defining the length

of the magnetic field vector as -5 for 5 GeV electrons, which had to be set since there was no magnetic field. Deviations in x direction on the plot means that the track was not orthogonal to the sensor planes. Therefore these cases are probably ghost tracks or outliers since the standard deviation of the scattering angle when passing through the magnetic coil at 5 GeV/c is rather small ($< 1°$). For the y component shown in *e)* again one expects the values to lie along the median, which is actually the case. Subfigure *f)* shows that there was a bug in the momentum seed creation in the VXDTF, which was fixed for later runs and could be made visible by the DQM plots. A correct plot would look like subfigure *e)*.

The DQM plots in Figure 6.7 show results of run 104, where no magnetic field was activated and therefore many secondary particles passed the sensors. The detailed settings are listed in Table 6.2. Figure 6.8 shows what typical events in a run without magnetic field look like. While simple cases as in Figure 6.8a were actually reconstructed using the BaseLineTF described in Section 6.3.2, for nontrivial cases as in Figure 6.8b, Figure 6.8c and Figure 6.8d the real TF was used. The BaseLineTF was not sensitive to mis-alignment and therefore turned out to be useful for alignment, as the VXDTF had difficulties with sensors displaced by more than 500 µm, since the fit then strongly disfavors the real hits because of mis-alignment. More details about alignment can be found in Section 6.3.3.2 and Table 6.3. The maximum sensor displacement for the Belle II VXD is expected to be less than 500 µm, and therefore the VXDTF is not in danger of running into a similar situation in the actual Belle II experiment.

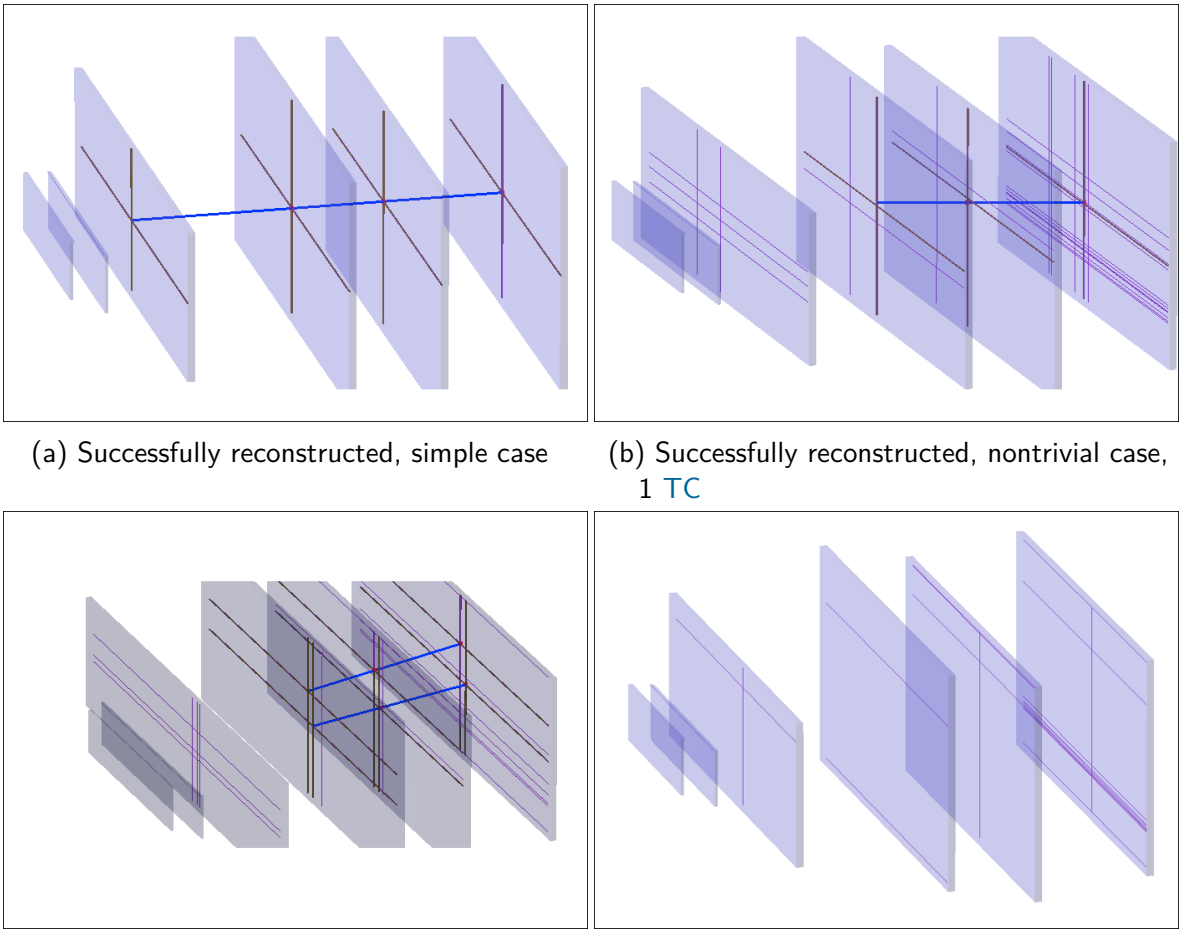
6.3.3 Off-line (PXD, Telescope and Alignment-support)

Along with the studies which were performed on-line, further studies were concentrating on off-line data. Although both studies rely on the VXDTF in the basf2 framework, the off-line analysis has more data available.

6.3.3.1 Differences to the HLT-setup

While the HLT is limited to information sent by the SVD, data from the PXD and the Telescope are available off-line only. Additionally alignment files were provided by Tadeas Bilka [45] and [47], which were extensively used in the study described in Section 6.3.3.2, while the most important alignment parameters are described in Table 6.3. The naming scheme used there is as follows:

- du: deviation in local u-direction of the sensor plane. In global coordinates this is a translation in z-coordinates.
- dv: deviation in local v-direction of the sensor plane. In global coordinates this is a translation in y-coordinates.



(a) Successfully reconstructed, simple case

(b) Successfully reconstructed, nontrivial case, 1 TC

(c) Successfully reconstructed, nontrivial case, 2 TCs

(d) Reconstruction failed

Figure 6.8: 4 different typical event types found in run 104, where the big violet boxes are the SVD sensors and the small ones are PXD sensors. The geometry used for that run yet assumed the usage of 2 PXD layers which was not decided to do otherwise yet at that time (see Section 6.2.2).

- $\mathrm{d}\gamma$: rotation around the local w-direction of the sensor plane (orthogonal to the sensor plane). In global coordinates this is a rotation around the x-axis.

The relative displacements between the PXD-layer 2 and the SVD-layer 3 of $\mathrm{d}u \approx 1300\,\mu\mathrm{m}$ and $\mathrm{d}v \approx 5500\,\mu\mathrm{m}$ have a significant impact on the reconstruction efficiency. This will also be discussed in Section 6.3.3.2.

The availability of recorded PXD- and Telescope-data allows for testing of tracking in more than the 4 layers of the SVD and more control over what is happening. While TCs covering hits of all detector types of the beam test are important for the alignment — which itself provides aligned geometry files — the possibility of re-running the TF on data of the same run allows to track down strange behavior and issues recognized

Table 6.3: Table of alignment parameters for the PXD (L2) the SVD (L3 - L6) and the Telescope sensors (L7). The meaning of the parameters is described in Section 6.3.3. $\mathrm{d}u$ and $\mathrm{d}v$ are given in units of µm, $\mathrm{d}\gamma$ in units of radians. Alignment data provided by Tadeas Bilka.

VxdID	type	$\mathrm{d}u$	$\mathrm{d}v$	$\mathrm{d}\gamma$
7.2.1	EUDET	0	0	0
7.2.2	EUDET	241	261	0.006
7.2.3	EUDET	407	281	0.003
2.1.2	PXD	290	-4836	-0.014
3.1.3	SVD	-1016	659	-0.010
4.1.4	SVD	-1021	-25	-0.017
5.1.5	SVD	-965	189	-0.015
6.1.6	SVD	-823	-257	-0.018
7.3.4	EUDET	-173	-307	-0.024
7.3.5	EUDET	-69	-428	-0.026
7.3.6	EUDET	0	0	0.019

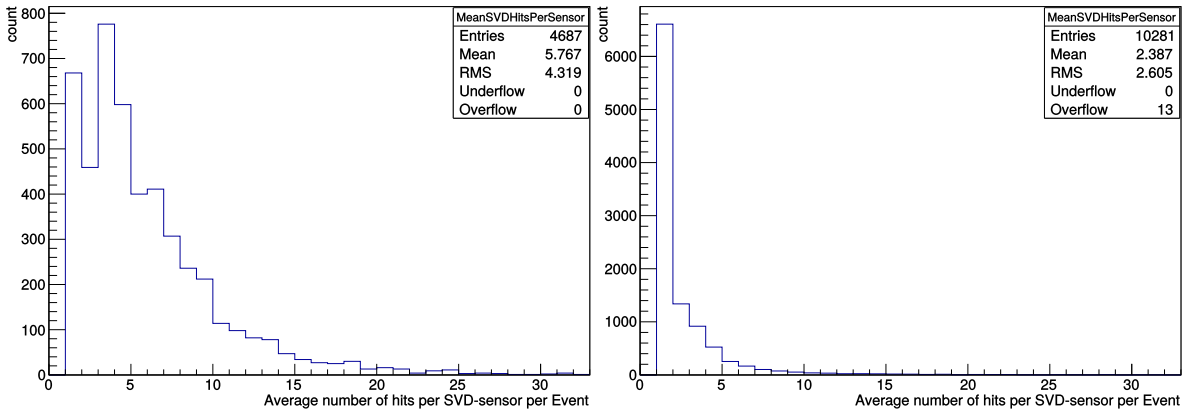
during on-line runs. To validate the efficiency of the VXDTF, the BaseLineTF was deactivated in the off-line studies, while it was activated in the on-line analysis to speed up processing of simple events. The colleagues from the university of Prague provided reference tracks using an independent method of finding track candidates, which allows for more detailed studies of the VXDTF-performance. More details about this approach are given in Section 6.3.3.2.

One complication for off-line tracking was the inconsistency in layer numbers which had to be introduced because of the restriction to 8 (0-7) layers in the VxdID used in the framework, which is described in more detail in Section 6.3.2. This led to the strange numbering of sensors listed in Table 6.1 and Table 6.3.

6.3.3.2 Off-line study of time consumption and reconstruction efficiency

A detailed study of the runs 470 and 510 was performed. They were chosen for their typical sensor behavior and because the presence of alignment files for these runs allowed for more checks. The run settings can be seen in Table 6.2. For run 470 the magnetic field was not activated, while a 1 T-field was present in run 510; consequently, in run 470, more low-momentum secondaries passed through the detector. This can be seen in Figure 6.9, where the number of cluster combinations per sensor is shown in a histogram for each of the two runs.

An important check that could not be done on-line is to evaluate the dependency of the time consumption of the VXDTF per event versus the maximum allowed number of accepted Segments for that event, which is called a Segment cap here. As mentioned



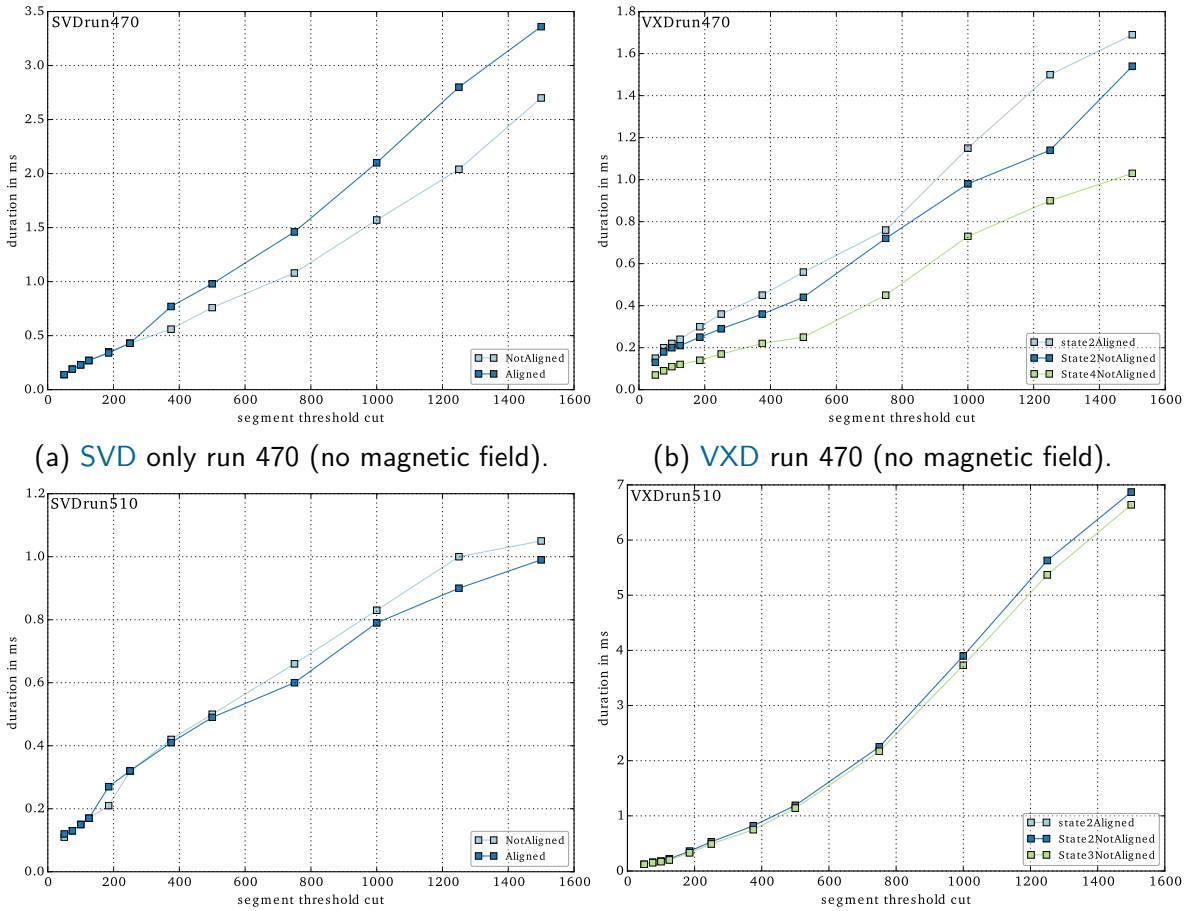
(a) Average number of hits per layer, run 470 (no magnetic field).

(b) Average number of hits per layer, run 510 (1 T)

Figure 6.9: These plots show the mean number of hits (≙ cluster-combinations) occurred per layer. The absence of the magnetic field in run 470 leads to more low momentum secondaries reaching the SVD-layers.

in Section 6.3.2.1, a conservative Segment cap of 75 was used for on-line reconstruction. Figure 6.10 shows the time consumption as a function of the Segment cap. For all the studies presented in this section, the values used for the Segment cap were: 50, 75, 100, 125, 185, 250, 375, 500, 750, 1000, 1250 and 1500. As can be seen in the Figure 6.10a and Figure 6.10c, the chosen Segment cap for on-line reconstruction of only 75 was far too conservative and even using a Segment cap of 1500 would not exceed the allowed 4 ms. When taking into account the number of TCs found per run, the plots of Figure 6.11 indicate that a Segment cap of about 750 Segments would lead to a good compromise between reconstruction speed and number of TCs. This would result in a mean time consumption per event of no more than 1.5 ms for the case of the HLT. Another interesting aspect of the values shown is that for SVD tracking — which resembles the situation at the HLT — the time consumption doesn't depend strongly on the presence of alignment. It also doesn't depend strongly on the number of TCs found, which can be seen especially in Figure 6.11b and Figure 6.11d, where runs with and without alignment provide practically the same number of TCs. But the number of TCs itself can mislead when interpreting the VXDTF performance, since TCs can have different numbers of hits. A closer look at the plots of Figure 6.12 reveals what could not be seen in Figure 6.11: that mis-aligned data has an impact on the TC length. To be able to interpret the plots correctly, one has to know that the maximum number of SVD Clusters in a single TC is 8 and only 1 for the PXD. One can distinguish between two typical situations, which will be discussed in the two following paragraphs.

In the case of a SVD-only reconstruction the VXDTF is able to collect TCs with more hits if an alignment file is used than for the case where no alignment file is used. In checks without alignment file, hits of layer 3 were barely ever added, which can be explained



(a) SVD only run 470 (no magnetic field).

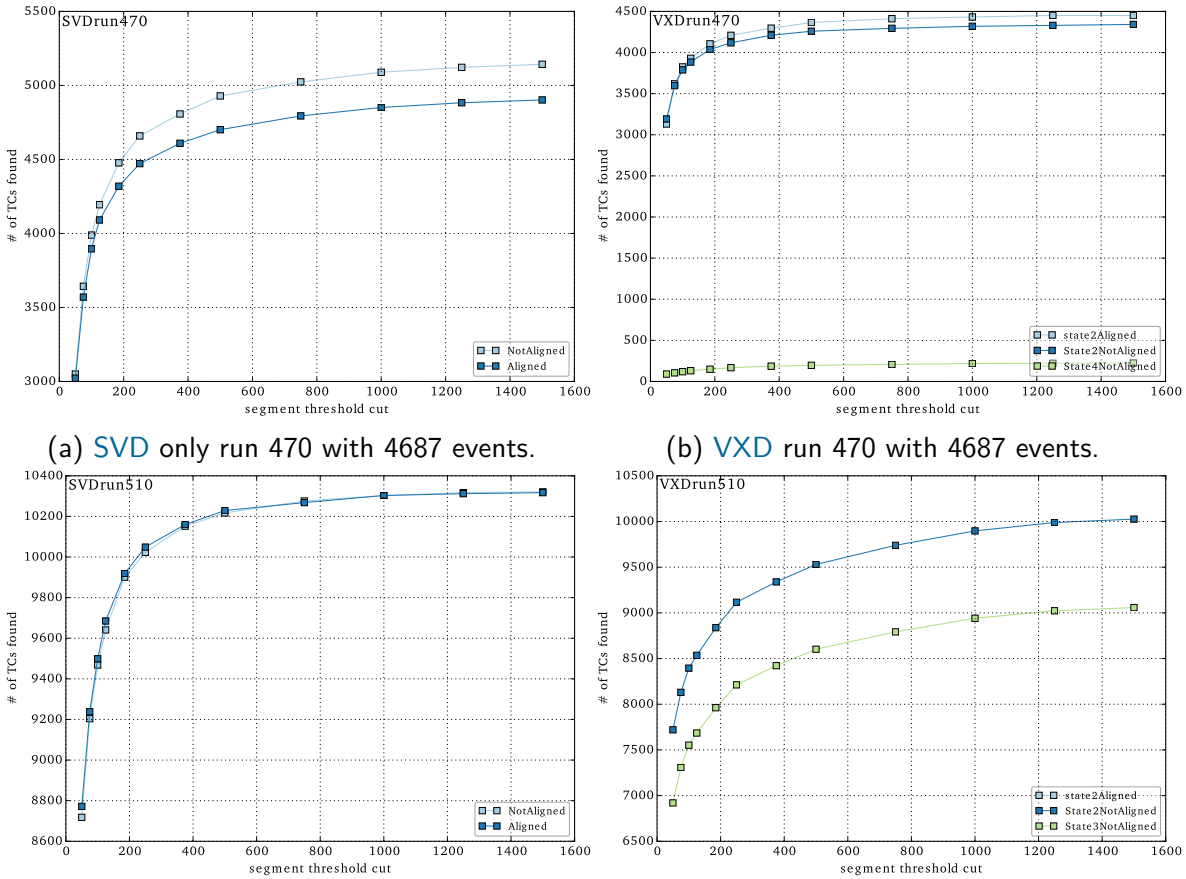
(b) VXD run 470 (no magnetic field).

(c) SVD only run 510 (1 T)

(d) VXD run 510 (1 T)

Figure 6.10: These plots display the behavior of the VXDTF regarding the duration per event versus the maximum Segment cap in different runs, with and without aligned files. For VXD-runs only there is a state value which indicates threshold used for Cell states, where higher numbers force longer TCs to be collected. State 2 requests 4 hits to be in a TC and state 4 requests 6 hits in a TC.

by the alignment results of Table 6.3. As can be seen in Figure 6.12a the runs with aligned input were better, but interestingly the Segmentcap had a negative impact on the TC length. Since increasing the Segment cap merely leads to adding more complex events, which were discarded by the lower Segment cap setting, this means that in more complex events the probability was very high to lose one of the hits for the final TCs since other hit combinations resulted in better QIs. For run 510 the difference between aligned and mis-aligned track reconstruction is negligible (Figure 6.12c). This indicates that the larger number of ghost hits in run 470 had a negative impact on TC length due to the bad QI of the fits in case of mis-aligned data. Therefore having mis-aligned data was less problematic in run 510, where a smaller number of background hits resulted in a



(a) SVD only run 470 with 4687 events.

(b) VXD run 470 with 4687 events.

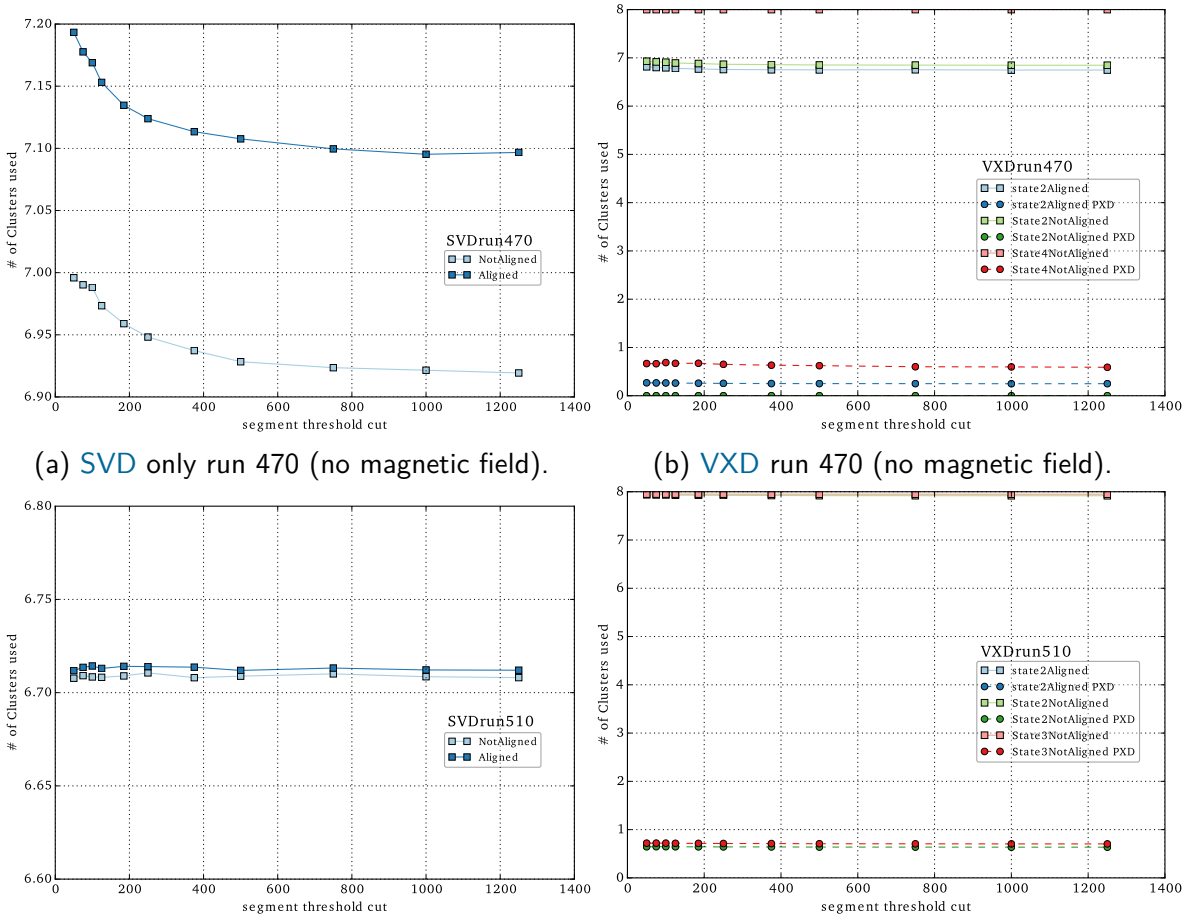
(c) SVD only run 510 with 10282 events.

(d) VXD run 510 with 10282 events.

Figure 6.11: These plots display the relation between the number of accepted Segments allowed per event and the total number of TCs found in the runs.

higher acceptance rate of such TCs with bad QIs, since they did not need to compete with other TCs so often.

A check on full VXD reconstruction in run 470 (Figure 6.12b) and therefore TCs with SVD and PXD hits, revealed that with mis-aligned input the VXDTF barely ever produced TCs with PXD hits added. In MC runs (with perfectly aligned data and no dead sensor strips) during the beam test preparation this issue never occurred. Therefore the mis-aligned input seems to have a strong impact. Although adding the alignment file to the reconstruction does improve the acceptance rate for PXD hits, still only about one out of four events resulted in a TC with PXD hits added. This indicates that the alignment file is still not perfect and the VXDTF is vulnerable to mis-aligned data since hits with an assumed high precision but wrong actual position lead to bad QIs for the TCs and therefore are discarded if a competing TC without that hit exists. This is the case for the chosen VXDTF setting, where a Cell state threshold of two is set, which means that tracks with a minimal number of hits of three plus virtual IP or four real



(a) SVD only run 470 (no magnetic field).

(b) VXD run 470 (no magnetic field).

(c) SVD only run 510 (1 T)

(d) VXD run 510 (1 T)

Figure 6.12: These plots display the behavior of the VXDTF regarding the number of Clusters used versus the maximum Segment cap in different runs, with and without aligned files. For VXD runs only there is a state value which indicates threshold used for Cell states, where higher numbers force longer TCs to be collected.

hits are collected. Although longer tracks are collected as well, they still need to have a good QI to survive as final TC in that process. Therefore it was tested whether the acceptance rate could be improved if higher Cell state thresholds were used. As can be seen in Figure 6.12b, a threshold state of four significantly increases the PXD hit acceptance rate. For run 510, as shown in Figure 6.12d, the acceptance rate for PXD hits was not influenced by the alignment; only increasing the Cell state threshold to three had a visible effect. State four for the Cell threshold resulted in no TCs at all, which indicates that for the case of a magnetic field with 1 T the probability was low to have hits in all sensors.

Another way of interpreting the quality of the VXDTF results was made possible by having a separate tracking technique implemented for the beam test, which will

Table 6.4: Contingency table for defining a reconstruction efficiency. Table also shown in [45].

	no reference TC	reference TC
no TC from VXDTF	n_{00}	n_{10}
TC from VXDTF	n_{01}	n_{11}

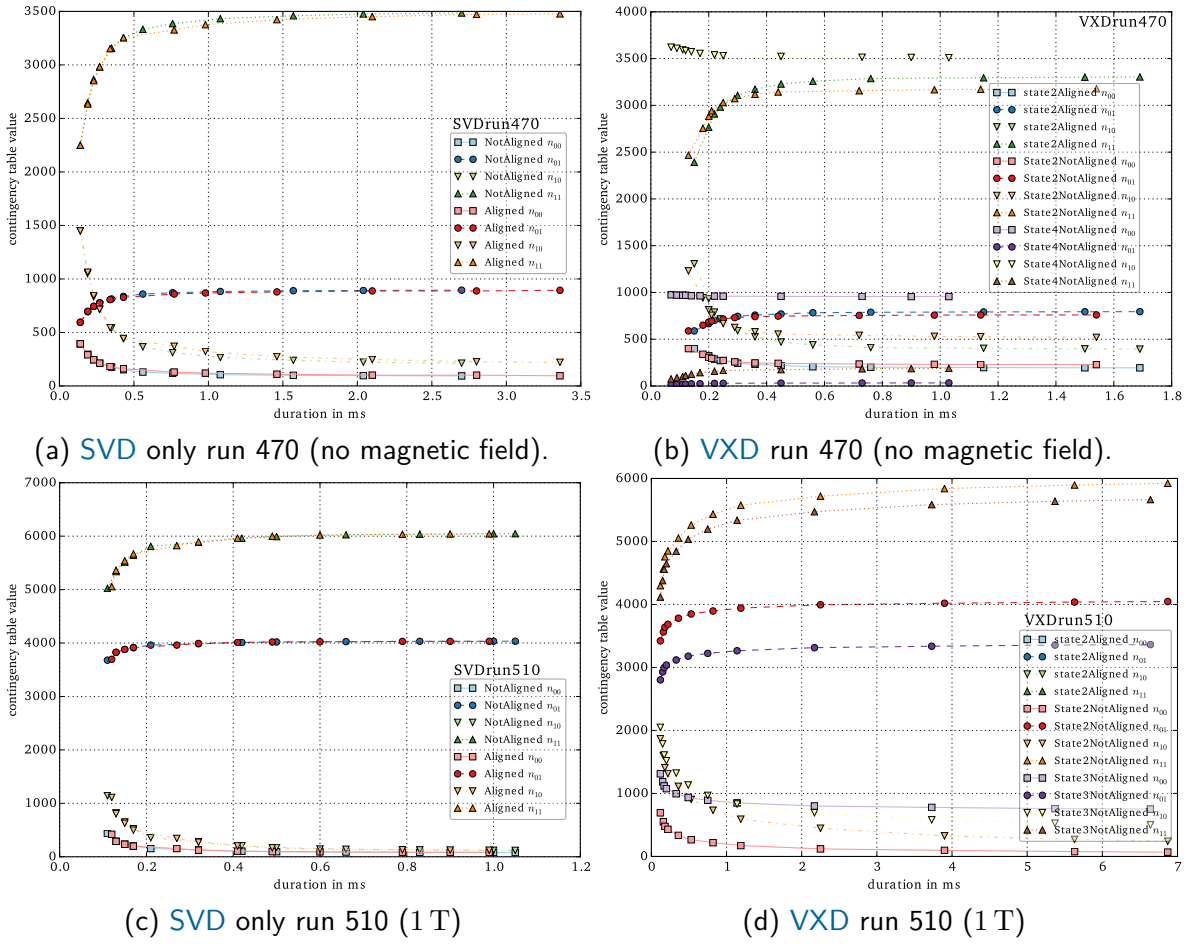
be used as a reference TF in the following tests. Since that procedure was very time consuming, a preselection of events was performed where loose correlations in several layers were demanded. The sample on which the tests were performed was the sample of the preselected events. To get the independent set of reference TCs, Peter Kodys searched for correlations in neighouring layers of the inner Telescope, the PXD and the SVD and combined these hits to reference TCs after a successful fit. Unfortunately the correlation check to finally obtain reference TCs was so time-consuming that still not all events of that sample could be analyzed, which leads to the situation that a realistic looking track in a high occupancy event was not stored as reference TC. Consequently, there were events in which the VXDTF was able to find a TC while no reference TC was produced. Despite this downside these reference TCs allowed for more realistic estimations of the performance of the VXDTF. To capture the performance of the VXDTF while considering the circumstances of the non-ideal reference TCs, a contingency table as shown in Table 6.4 was defined. n_{00} counts the events where neither the VXDTF nor the reference TF did find tracks; n_{11} counted the events where both TFs found a track; n_{01} counted the events where a VXDTF TC was found, but no reference TC and n_{10} vice versa. Figure 6.13 shows the values determined for the contingency table for all cases studied in relation to the time consumption of the reconstruction. The n_{11}-results should increase up to the total number of events with existing reference TCs, n_{01} for ideal reference TCs should stay low, which would be valid for n_{00} as well. The raw numbers of the contingency table are not suitable to capture the performance adequately, which is the reason why several efficiencies were defined. All the efficiencies defined rely on the contingency table and try to emphasize different aspects of the situation. They are defined as follows:

1. the number of events where both a reference TC and a VXDTF TC was found, divided by the number of events with a reference TC:

$$\epsilon_1 = \frac{n_{11}}{n_{10} + n_{11}}, \tag{6.2}$$

2. the number of events where the VXDTF found a TC, divided by the total number of events ($\hat{=} n_{\text{tot}}$) where a TC was found by either approach:

$$\epsilon_2 = \frac{n_{01} + n_{11}}{n_{\text{tot}} - n_{00}}, \tag{6.3}$$



(a) SVD only run 470 (no magnetic field).

(b) VXD run 470 (no magnetic field).

(c) SVD only run 510 (1 T)

(d) VXD run 510 (1 T)

Figure 6.13: These plots display the behavior of the VXDTF regarding the contingency table values defined in Table 6.4 versus the duration per event in different runs, with and without aligned files. For VXD runs only there is a state value which indicates threshold used for Cell states, where higher numbers force longer TCs to be collected.

3. the number of events where both tracking approaches found TCs or both approaches found nothing, divided by the total number of events:

$$\epsilon_3 = \frac{n_{00} + n_{11}}{n_{\text{tot}}}. \tag{6.4}$$

As mentioned before, not all events of the test sample had reference TCs. Therefore none of the listed types of efficiency fulfill the task of giving a fully satisfying estimate of the VXDTF performance. Each of these definitions have their individual weaknesses when considering the main flaw of the reference tracks, the non-existence of tracks for high occupancy events. ϵ_1 (6.2) does not take into account events where no reference TC was found, while ϵ_2 (6.3) uses the reference TC mainly for excluding events, where no

tracking method found any TC. ϵ_3 (6.4) is a typical definition to be used if the reference system is close to an ideal one, therefore this definition will mostly check the similarity of the tracking methods in terms of behavior in high occupancy cases. However, while all three definitions have their flaws, they provide a more reliable picture when evaluated together.

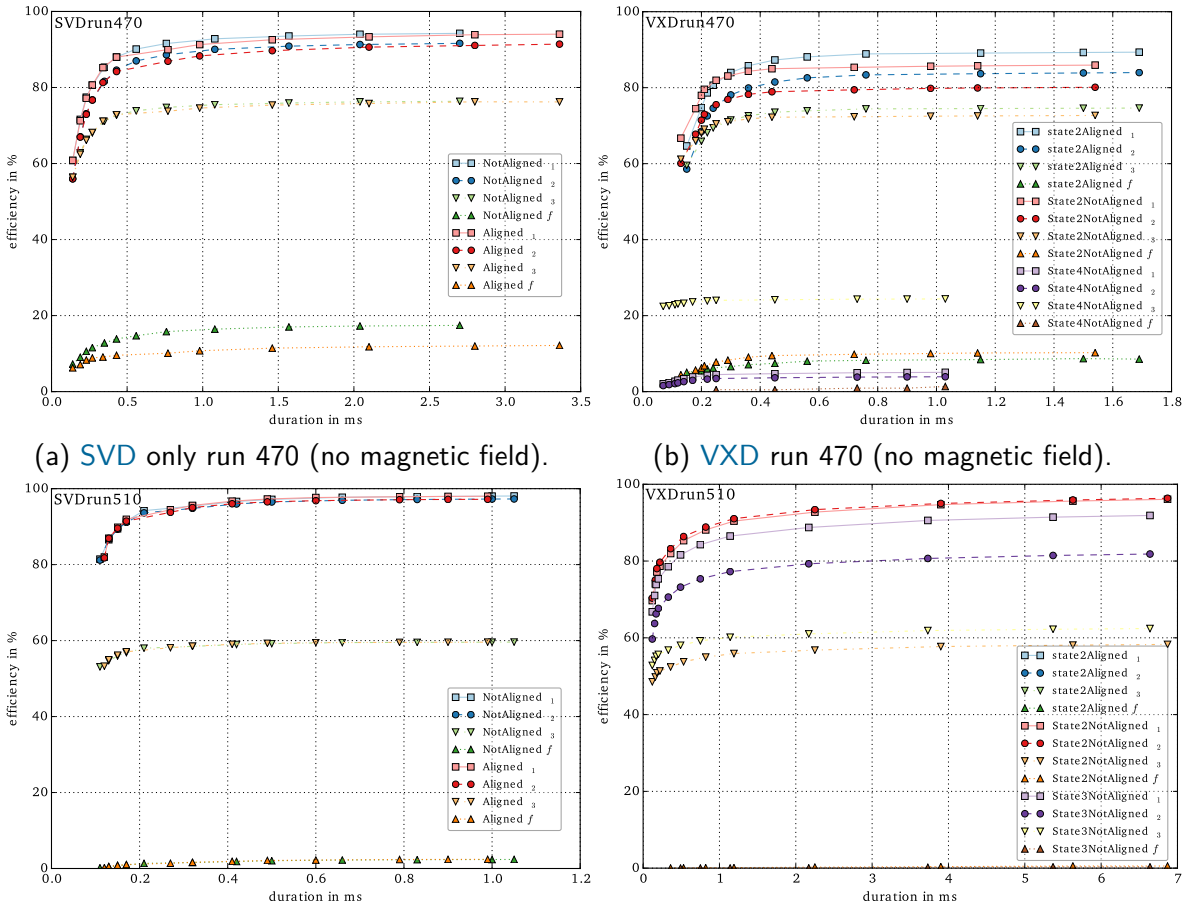
Additionally another parameter was defined, the ghost rate f. The assumption of having only one track per event allows to define the ghost rate as follows:

$$f = \frac{100}{n_{01} + n_{11}} \cdot (n_{\text{CA-TC}} - (n_{01} + n_{11})) \tag{6.5}$$

where for the total number of TCs found by the VXDTF was named $n_{\text{CA-TC}}$. Especially for runs without magnetic field this definition is conservative since the particle beam produces low momentum electrons when passing the magnetic coil (see Figure 6.2) which can hit the sensors as well. This is why runs with activated field have much fewer hits and much lower ghost rates. Alternative explanations like noisy strips are therefore neglected to be the source for ghost TCs.

The plots in Figure 6.14 indicate a saturation of the efficiencies for Segment caps corresponding to 1–1.5 ms in run 470 (Figure 6.14a), which would be a Segment cap of 500–750. While the efficiencies ϵ_1 and ϵ_2 reach more than 90%, ϵ_3 stays at a level of about 75%, since for higher Segment caps, the VXDTF is becoming more and more successful in reconstructing events which had no TCs. The ghost rate stays near 20% for mis-aligned input, while it remains near 10% for aligned input. Therefore the ghost rate depends on whether alignment has been performed as opposed to the efficiencies which show little to no dependency on alignment.

For run 510 (Figure 6.14c) the efficiencies ϵ_1 and ϵ_2 are almost 100%, while ϵ_3 does not exceed 60%. This is due to the fact that it was more difficult to deliver reference TCs for runs with an activated magnetic field. The difficulties come from the issue that for curved particle tracks, the allowed cuts between two neighbouring layers had to be extended, which led to a higher number of skipped events due to the increasing combinatorial problem. The VXDTF is less sensitive to that fact. Runs with an activated magnetic field had a lower number of hits in total, which resulted in a lower ghost rate for the VXDTF. For the ghost rate, no difference between aligned and mis-aligned input was found in run 510. For the VXD runs (Figure 6.14b and Figure 6.14d) the sensitivity to aligned input was more distinctive. This is caused by the large displacement of the PXD sensor and results in a small probability of having PXD hits in the final TCs, which is especially visible in Figure 6.12b, where no PXD hit at all was found in final tracks for mis-aligned input and a Cell state threshold of two. Therefore runs with higher Cell state thresholds were performed using state a state of four for run 470 and three for run 510. A threshold of state four in run 510 resulted in no TCs at all, so the next lower value was used. Since these special runs resulted in much smaller numbers of events with TCs found, their efficiencies are much lower than the runs with state two; at the same time,



(a) SVD only run 470 (no magnetic field).

(b) VXD run 470 (no magnetic field).

(c) SVD only run 510 (1 T)

(d) VXD run 510 (1 T)

Figure 6.14: These plots display the behavior of the VXDTF regarding the efficiencies and ghost rate defined in equations (6.2), (6.3), (6.4) and (6.5) versus the duration per event in different runs, with and without aligned files. For VXD runs only there is a state value which indicates threshold used for Cell states, where higher numbers force longer TCs to be collected.

their tracks had more hits.

Another relevant topic is the time consumption of the individual parts of the VXDTF when working with different Segment caps. Figure 6.15 shows the results in one plot, where SVD only runs are shown with and without magnetic field, and with and without aligned input. One can see that for higher Segment caps the Hopfield part and the checking for overlapping TCs clearly dominate the time consumption. No matter if the input is aligned or not, or if there is a magnetic field activated, these two steps consume about 75% of the total time needed. For a Segment cap of 75 these two parts consumed only about 20% of the total time. For the case of the full Belle II SVD the time consumption of the sub-parts differs considerably, as can be seen in Figure 5.28b. There HNN and clean overlaps dominate the time consumption for all cases with background added. Only

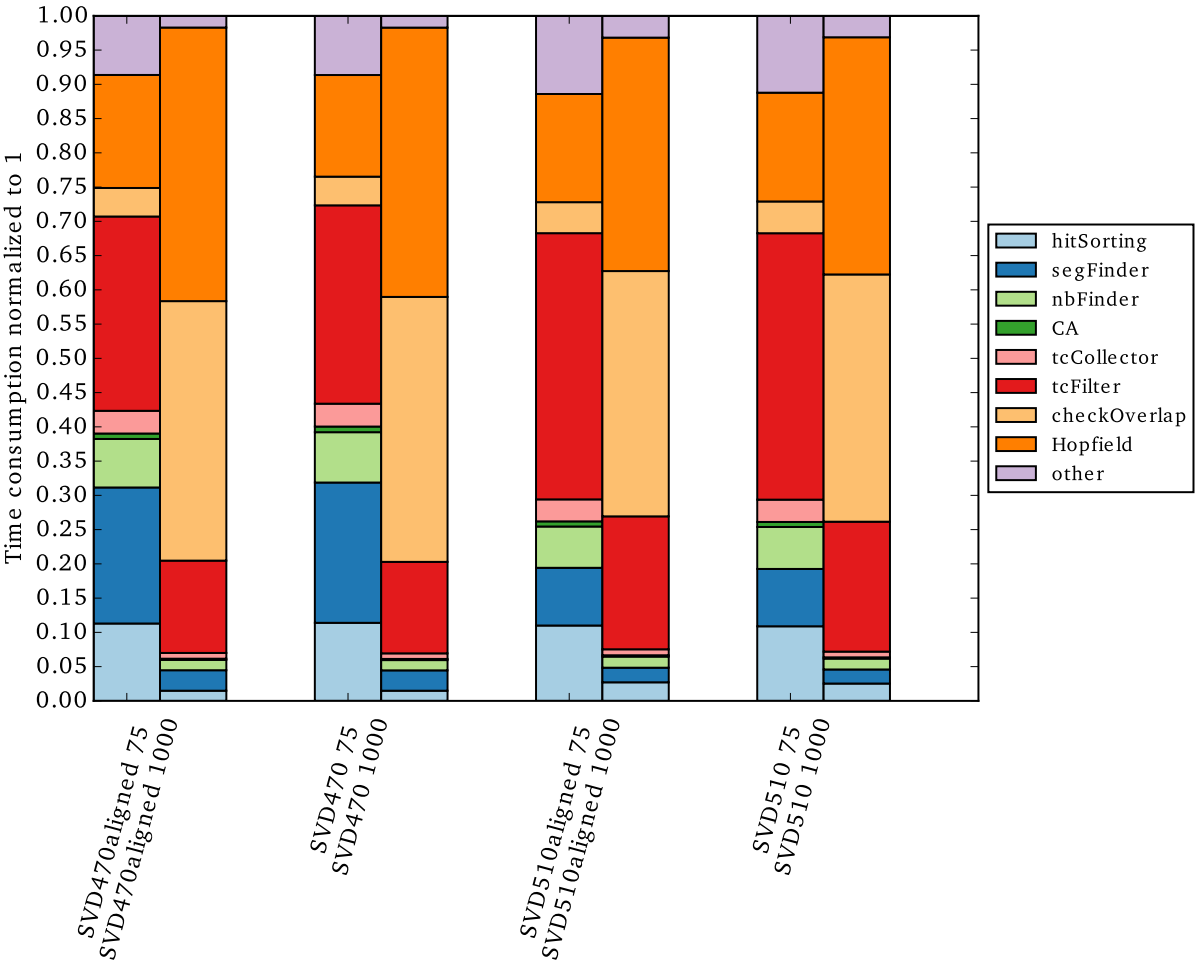


Figure 6.15: Comparison of time consumption regarding sub-parts of the VXDTF normalized to 1 for better comparability. '75' stands for runs with a Segment cap of 75 and '1000' for a Segment cap of 1000.

for $\Upsilon(4S)$ events without background added the results seem to be comparable. Since for the beam test the VXDTF had very loose cuts to compensate the bad pre-alignment and therefore the filtering effect of the SecMap was practically nullified, the significance of the time consumption is questionable in any case.

A detail only visible at a second glance is that the relative time consumption of the individual parts are practically identical between aligned runs and the same runs with mis-alignment. This is especially puzzling since their absolute time consumption actually differ significantly (see Figure 6.10a versus Figure 6.10c and Figure 6.10b versus Figure 6.10d). Since other parameters like the number of TCs and the number of clusters added are different as well, the only aspect left is the number of clusters per sensor, which is of course the same for aligned and mis-aligned runs of the same data set. A simple

explanation therefore could not be found.

One aspect that has not been discussed so far is the performance of the VXDTF when allowing tracking over all layers of the SVD, PXD and the Telescope. Although not added to the plots shown previously, it is possible to do tracking that way. In addition to the issues mentioned in Section 6.3.3, the (mis-)alignment — which already had an influence at VXD tracking only — is more severe and therefore TCs spanning all sensors at once are rarely seen. In fact, when forcing a Cell state of 10, which is the highest value possible, only 0.5% of all tracks contain a PXD hit at all. On the other hand, Telescope Clusters were added much more frequently and as a mean value, 4.74 Telescope Clusters were added per track. The time consumption remains below 500 µs for any Segment cap chosen and does not behave as expected, since more sensors mean more hits, and therefore greater combinatorics. The difference in the treatment of PXD and Telescope clusters indicate that the substantial mis-alignment of the PXD sensor compared to the comparatively well-aligned Telescope sensors could not fully be removed by the alignment files. This issue remains to be examined in more detail using the data of the next beam test.

6.4 Conclusion

Combined beam tests are essential for the preparation of large-scale experiments such as Belle II. This test made clear that beam tests of single sensors or detector types can not replace combined tests, since communication between the working groups and interfaces of the hardware and software parts have to be tested as well.

For the case of the VXDTF this beam test was essential, especially to prove that the concept of the TF is functional in real-live conditions. Overall there were many lessons learned:

- This was the first time the VXDTF had to deal with sensor inefficiency because of bad strips and broken readout chips. For the long run there have to be ways to test this with MC data as well, since the VXDTF struggled with missing hits.
- For the VXDTF it was the first time to be tested with mis-aligned input, which resulted in a bad efficiency because of bad QIs for the TCs. Although the final geometry will be implemented with smaller error margins, the issue will remain. A way to increase hit errors for possibly mis-aligned data or skipping the clean subset step with the HNN could help to decrease the impact on the efficiency.
- The Segment cap of 75 chosen for on-line runs was definitively too low and can be safely increased to 500–750 without major impact on the time budget. This will increase the number of TCs significantly.
- One detail to be tested in subsequent runs is to deactivate overlap checks and HNN filtering. Since the issue of ghost hits in the SVD is severe for beam tests

— especially for runs without magnetic field —, the probability is high to discard correct TCs while keeping the ghost TCs.

- Another relevant reminder for the next beam test is to enforce better communication between the working groups of the hardware and software parts since this has a significant influence on the output of such a test. As one example to be mentioned was the deactivation of the DQM system of the HLT since some tests turned out to be too slow for on-line tests. This was not communicated to those writing the DQM modules until after the beam test, when any further changes were futile.

- The relevance of the DQM output should be increased for the next time. Tests like the number of clusters per TC and which sensors were used for final TCs could deliver relevant hints for mis-alignment and other tuning steps.

With this beam test, an essential milestone was passed successfully, since the full chain of reconstruction worked and did not exceed any critical limits. In the end one can say that many lessons were learned and weak spots identified, nonetheless it was a huge success proving the functionality of the VXDTF years before the actual experiment starts.

CHAPTER 7

VXDTF 2 AND PRELIMINARY RESULTS

7.1 Motivation

The VXDTF and its performance has been thoroughly discussed in the previous chapters. As already identified in Chapter 5, the implementation of the VXDTF does not deliver perfect results yet. The efficiency level for realistic conditions with background is at about 90% of its theoretically possible maximum, and the time consumption is too high. The main task of the VXDTF has been implemented in a single module, which is now difficult to maintain because of its complexity. The code base of the VXDTF has now more than 44,000 lines of code, which is too much to maintain and improve for a single person. Therefore a complete restructuring of the VXDTF — the VXDTF 2 — has been started in 2014. Its goal is a higher coverage of unit and integration tests, more flexibility achieved by using a multi-module design, and improved ways to probe into the intermediate steps of the algorithms. The refactored code will repair some bad design decisions made in the first version, will add several minor features, and will remove bottlenecks introduced in the current version. Packing the individual steps into separate modules with standardized interfaces additionally allows to share the maintenance and development workload between several people or groups.

The redesign will feature an improved way to train the SecMaps, the possibility to store three- and four-Sector combinations and their specific cuts for the corresponding Filters, which should improve the efficiency for these Filter types. Related to the redesign is the necessity to develop better debugging tools, which become possible because of the improved logging system implemented during the refactoring process.

However, the redesign could not be finished within the time frame reserved for this thesis, which is why the major design ideas and some aspects of the VXDTF 2 are discussed in this chapter for documentation purposes. Since the code has not been finished yet, there are no final results to be shown here. Preliminary results, however,

can be found in Section 7.9 and should be handled with care because of the early stage in development of the VXDTF 2.

7.2 Overview

The structure of the VXDTF 2 is no longer monolithic, but has been split up into several smaller modules. Standardized interfaces allow independent code development of the individual parts. Figure 7.1 shows the main modules of the VXDTF 2 that are executed for each event. The entire chain can be repeated several times, using different settings

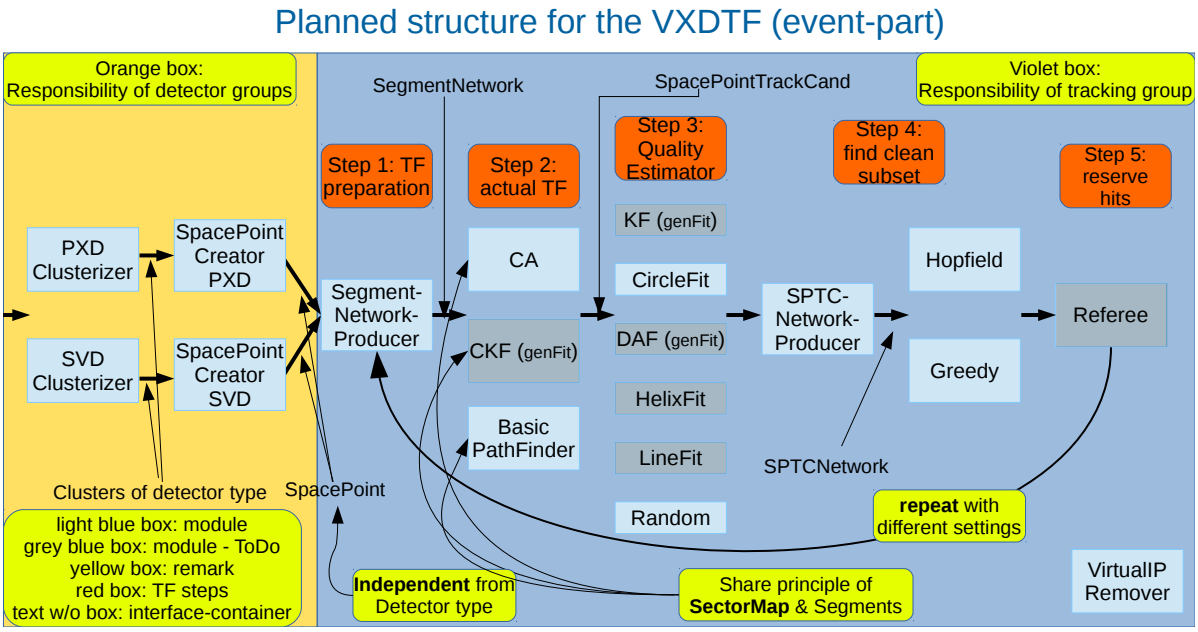


Figure 7.1: The structure of the VXDTF 2. At the time this thesis was written, the modules marked in gray were not implemented yet.

stored in the SecMap. This allows to perform e.g. momentum dependent passes, or passes using different TF algorithms for different momentum ranges, or special tasks like finding curlers. The modules responsible for the training and the loading of the SecMap are not shown in this overview, since they are not applied on-line.

This section summarizes the steps performed and the modules performing them, along with the interface classes and their intended purpose.

Before starting the VXDTF 2 modules, the detector hits are converted to SpacePoints, the detector-independent format. This task is not part of the VXDTF 2 any more and will be maintained by the software groups of the tracking detectors. This has the additional benefit that some of the clusters can be discarded already at this point, which reduces

the combinatorial problem for the TF. More details on the specifications and specialties of this step are discussed in Section 7.3.

The SpacePoints are the input for the first event-wise applied VXDTF 2 module, the SegNetModule[90], which will be discussed in more detail in Section 7.4. The SegNetModule applies the n-hit filters to the SpacePoints. Its task is to reduce the possible combinations of SpacePoints and map these combinations into the SegNet[91]. The output is then stored in the SegNet, a container to store graphs, which is used to store directed graphs of Sectors, SpacePoints or Segments. It additionally allows to store intermediate steps, which can be used to investigate lost hits or track Segments.

The SegNet is then used by the actual TFs, which will produce preliminary tracks using the SPTC[92] as output format. More details about the planned TF algorithms can be found in Section 7.5. After the TCs are created one still needs to remove the virtual IP from the TCs. This can be done before or after the quality estimation step.

All following steps are then not mandatory, since these preliminary tracks contain all the information to be transformed into the official TC format of the framework. Comparing the number of collected preliminary TCs and a filtered number of final TCs, as plotted in Figure 5.27, nontrivial events typically result in a very high number of preliminary TCs, which have to be filtered before more time consuming modules of other groups can use the output of the VXDTF 2.

Preliminary TCs often share one or several hits, if several passes are performed, and the same TC can be found more than once. To find optimal clean subsets of TCs which do not share SpacePoints or clusters, the quality of the TCs has to be measured, which is done using the fitters applied in the quality estimator step. The input and output is in this case identical, namely the SPTCs, but some of them could be deactivated, if the quality estimator modules apply cuts for filtering bad TCs. All modules of this step must provide a QI expressing the quality of the TC tested. The various modules to be used in this step are described in Section 7.6.

The final step is finding which TCs overlap and selecting a non-overlapping subset with the best possible quality. This is done in two sub-steps:

- The SPTC Network Producer Module, which finds overlapping TCs and forms a network, which is stored in the container "SPTCNetwork", where overlapping TCs are nodes and linked with each other, if they are overlapping. In the standard setting two TCs are overlapping if they share a cluster. But other ways to define an overlap are implemented as well, e.g. one can define that two TCs are overlapping if they share a SpacePoint and not only clusters. More about this extended definitions of the overlap can be found in Section 7.7.1.

- The SPTCNetwork is then used as an input for the filtering step, where a non-overlapping — clean – subset of TCs is found.

[90] SegNetModule: SegmentNetworkProducerModule
[91] SegNet: Segment Network
[92] SPTC: The Space Point Track Candidate.

All the modules relevant for this step are described in Section 7.7.

The best TCs of the final clean subset are allowed to reserve their hits. Reserved hits are then blocked for further use. Specifics and possible pitfalls of this step are discussed in Section 7.8.

7.3 Step 0: SpacePoints

In the zeroth step SpacePoints — the hit class used in VXDTF 2 — are created.

The SpacePoint class masks the detector specific aspects of the hits from the TF, which does not need to know to which detector it is applied. From the TFs point of view hits are points in 3D space with corresponding errors. Therefore all clusters — which store their hit information locally connected to their respective sensor — are converted to global coordinates. In the case of the SVD two clusters, one of u type and one of v type, where u and v are the local coordinates, are combined to one SpacePoint. For pixel sensors, this is not needed. Next to the coordinates extra information regarding the hit quality, based on available information like energy deposit, cluster shape and hit time, will be stored and used by the VXDTF 2. For example, hits with high energy deposit could be considered only in low-momentum passes. Also, the optimal time resolution of about 2 ns for the case of SVD hits allows to recognize hits of (back-)curling particles. For the PXD, the cluster shape can be used to determine possible background particles and allows to add rough guesses of the possible direction of flight, which then could be used by two- or three-hit Filters in the VXDTF 2. As in the case of the SVD, the energy deposit of PXD clusters can be used to determine rough momentum or $\beta\gamma$ estimates. At the moment, however, none of these tasks is implemented yet. Further investigations into these topics will be performed by the SVD software group.

7.4 Step 1: SegmentNetworkProducer

The SegNetModule is a cornerstone of the redesigned architecture of the VXDTF 2. Its task is to provide a versatile container which can be used by different track finding algorithms such as a Cellular Automaton (CA), a Deterministic Annealing Filter (DAF[93]) or a Combinatorial Kalman Filter (CKF). The container is meant to store networks of directed graphs. The creation of the container should keep the overhead low and effectively reduce the combinatorial problem omnipresent in local and semi-local tracking algorithms. The SegNetModule applies Filters of different complexity levels, starting with a one-hit Filter. The different Filter levels are similar to those discussed in the VXDTF chapter (see Section 4.3.3 for more details). The access to the Filters is provided by interface classes of the SecMap, enabling the application both of purely geometrical

[93] Deterministic Annealing Filter

Filters like Distance3D and more sophisticated ones like fully trained neural networks [36].

The basic procedure of the SegNetModule works as follows: Applying the one-hit Filter results in a network where the nodes are active Sectors, which will be explained in detail in the following sections. The step which applies two-hit Filters produces a network of hits, and the step applying three-hit Filters results in a network of Segments. The final network contains several directed graphs linking Segments which can be used by the tracking algorithms mentioned in Section 7.5. The implementation of the network object is very versatile and allows to temporarily add features to the objects stored in the networks. Such a feature, which is attached to the objects, is e.g. the Cell functionality needed for the CA. All of this will be explained with more details in the following sections.

7.4.1 Associated Classes

The design of the classes relevant for the module is summarized here, to allow a deeper understanding of what the code does.

The heart of the SegNet is a container which stores the network in an easy-to-use and efficient way. The class name of the network is `DirectedNodeNetwork`. The elements of the network are called `DirectedNode` which are linked to other `DirectedNode`s.

Next to them the `ActiveSectors` and `TrackNodes` will be discussed in the following sections.

7.4.1.1 DirectedNode

The class `DirectedNode` handles the storing and access of the carried object — entry — and keeps track of the links to its inner and outer neighbouring nodes. These nodes are templated and have only some very loose restrictions to the objects stored in them:

1. The stored class of template `EntryType` has to have an `== operator` defined.
2. All nodes of the network store an object of `EntryType`.

Next to the entry object a meta-info object can be attached to a node. Since it is implemented as a template in the `DirectedNode` class as well, the sole prerequisite for the meta-info class is that there has to be a public constructor with no arguments being passed. `DirectedNodes` are created by the `DirectedNodeNetwork` and cannot be created by the user. The public functions only allow access to the entries of the `DirectedNode`s, to the inner and outer connected `DirectedNode`s and to the meta-info object attached (if any). Additionally one can ask for the index number of the `DirectedNode` in the `DirectedNodeNetwork`.

7.4.1.2 DirectedNodeNetwork

The class `DirectedNodeNetwork` adds, stores and connects the `DirectedNode`s and keeps track of the inner and outer ends of the network. The prerequisites for the `DirectedNodeNetwork` are the following:

1. All nodes store the same `EntryType` and the same `MetaInfoType`. Links between the nodes cannot carry any extra information, with the exception of the direction: inner and outer).
2. `DirectedNode`s cannot be deactivated after being stored in the network and links not deleted (changes after the initial creation of the network are not needed for the SegNet and would only complicate things).
3. Can be walked through in a directed way (each `DirectedNode` knows its inner and outer partners) or in an undirected way.
4. Inner and outer end points of the networks are always automatically updated when filling the network.
5. There are three different ways to add nodes to the network:
   a) Add a pair of nodes, where one is the outer and one is the inner node. Links are established in both directions.
   b) Add another inner node to the last outer node passed.
   c) Add another outer node to the last inner node passed.
6. There cannot be any nodes in the network which have no links at all.
7. Adding (and linking) the same pair of entries twice results in a warning.
8. The user has to take care of the lifetime of objects to be linked in the network since it only stores a reference of the objects passed.

Of the three ways to add `DirectedNode`s, the latter two are there only to speed up the interface, since adding pairs of nodes all the time would lead to the situation that nodes will be temporarily re-added several times.

Although the basic design is settled, there is still room for speed optimization. This is then a matter for forthcoming refactoring and optimization steps (to mention one: switch from classic pointers of `C++03` to `uniqueptr` of `C++11`).

The design of the SegNet creates networks in a directed way by adding new nodes pair-wise, which are then connected to existing nodes, if needed. That approach leads to some unnecessary temporary recreations of `DirectedNode`s when adding pairs of nodes in random order. This is a possible downside which should be worked on, if upcoming studies show a relevant fraction of double- and multi-creations of the same nodes. Another

issue is that the for each node added it must me checked if that node was already added before. This should also be addressed if that design turns out to be a bottleneck in the reconstruction process.

7.4.1.3 Active and Static Sectors in the SectorMap

In the new design the dynamic and the static part of the SecMap are decoupled. The SecMap will be trained once and then stored as static object, in which all the Sectors, their combinations and the Filters with their cuts are stored. The `StaticSector` is meant for storing static info like the geometrical position it is attached to and filters for Sector combinations and their cuts. Additionally for each Filter step there is a separate SecMap stored, which means that the two-hit Filter cuts are in a different SecMap than the three-hit Filters. This leads to a simpler structure and easily allows to use cuts uniquely for each three-Sector combination. Three-Sector combinations were not possible in the old version, where only two-Sector combinations could be stored. Limitations of the old VXDTF design were also discussed in Section 4.3.3.3.

To be able to attach SpacePoints event-wise, a simple interface class to the static part of the SecMap — called `ActiveSector` — is created event- or pass-wise for all Sectors where compatible SpacePoints could be found. The task of the `ActiveSector` is to allow access to the `StaticSector` of the SecMap and to know all SpacePoints attached to it. While the SecMap is a static object, which is trained for long time periods and can not be modified event-wise, the `ActiveSectors` live only for the current event. The `ActiveSector` collects the hits found for it and allows accessing them too.

7.4.1.4 TrackNode

The old VXDTF used a hit class called `VXDTFHit`, which served the same purpose as the SpacePoints do now on the detail level discussed in this thesis. Although they can be treated as identical objects in many discussions, for the case of the SegNet this approach does not apply, since some tasks were performed by the `VXDTFHit`, which can not be taken over by the SpacePoint.

In the SegNet therefore no actual SpacePoints are stored, but an intermediate class called `TrackNode`. No perfect solution could be found, but `TrackNodes` have been identified as easiest solution.

Unlike SpacePoints `TrackNodes` are pass dependent — like `ActiveSectors` — and will be created for each SpacePoint to be used in the current pass. They are tied to their SpacePoint and will be used in their stead as an interface class between `AcitveSectors` and SpacePoints. More details about the discussion of `TrackNodes` versus SpacePoints can be found in Section 10.3.

7.4.1.5 Segment

Another class relevant for the SegNetModule is the `Segment`. Its task is to know two compatible `TrackNodes` and the sectors they are lying on and is therefore a very simple class. They are the objects produced in the final round of the SegNetModule, which will be discussed in the next section.

7.4.2 Deployment of the module

In this section, we describe the deployment of the actual SegNetModule and point out the differences to the basic idea of the old VXDTF. The module performs the following steps:

- `matchSpacePointToSectors(...)` - for each SpacePoint given, find according `AcitveSector`, which will be created if not found in existing list of `AcitveSectors`, and store them in a fast and intermediate way.
- `buildActiveSectorNetwork(...)` - build a `DirectedNodeNetwork<ActiveSector>`, where all `ActiveSectors` are stored which have `TrackNodes` *and* compatible inner- or outer neighbours.
- `segFinder/buildTrackNodeNetwork(...)` - use `TrackNodes` stored in `ActiveSectors` to build `TrackNodes` which will stored and linked in a `DirectedNodeNetwork<TrackNode>`.
- `nbFinder/buildSegmentNetwork(...)` - use connected `TrackNodes` to form Segments which will stored and linked in a `DirectedNodeNetwork<Segment>`.
- `storeToStoreObjPtr(...)` - fill output format.

The following paragraphs describe more details about the steps introduced above.

After executing the function `matchSpacePointToSectors(...)`, we have got all sectorIDs, which are relevant for this SecMap and event and for each of them, their SpacePoints are collected and wrapped into `TrackNodes`. For SpacePoints where no valid Sector was found (relevant e.g. for low momentum passes which do not use all layers), are neglected during this step.

After executing the function `buildActiveSectorNetwork(...)`, all `ActiveSectors` which had `TrackNodess` but no compatible other `ActiveSectors` with `TrackNodess` this event, are *stored*. This consequently means that `TrackNodess` without any valid partners to be combined later on are neglected during this step. This is an one-hit Filter. To be more correct, the `ActiveSector` is actually a `ActiveSector<StaticSector, TrackNode>` which was omitted to improve readability.

After executing the function `buildTrackNodeNetwork(...)`, we got a network, where each `DirectedNode` carries a single `TrackNode` each and the *links* between the `DirectedNodes` are implicitly the Segments, but not the class `Segment` yet. Therefore the Segments are

there, but not visible (no extra class for them), since their only job at the moment is to link `TrackNodes`. This is a reason why the old name `segFinder` (although the same Filters - translated into the new design - are used) is maybe a bit misleading, which is why the name was changed to: `buildTrackNodeNetwork`.

After executing the function `buildSegmentNetwork(...)`, we got a network, where `Segments` form the nodes and the links between them are implicitly that what we called Neighbours so far. If we had more time, one could use the last section as a blueprint for easily writing a recursive network-algorithm, which simply continues sticking pairs of nodes of the input network into a single node of the output network (e.g. the next step would take neighbouring Segments and would combine them to `three-hit segments`, which would then form the nodes of the new `DirectedNodeNetwork`). For four-hit Filters the current implementations already have some useful Filters, but for five- and more-hit Filters, this approach would simply accept all combinations, but would still deliver automatically the longest chains. In that case one only has to search for multi-hit Segments which are inner and outer end of the network at the same time and collect them before starting the next iteration, which then results in another tracking algorithm implemented. We had to neglect the implementation of that approach due to lack of time.

The output of the SegNet is as follows:

- A StoreArray[94] container `StoreObjPtr<DirectedNodeNetworkContainer>`, which contains three networks, the `DirectedNodeNetwork<ActiveSector>`, `DirectedNodeNetwork<TrackNode>` and the `DirectedNodeNetwork<Segment>`.

Which then serves as the input for the track finder algorithms described in the following section.

7.5 Step 2: Track Finder Algorithm

All track finder modules implemented for the VXDTF 2 share the same interface, which reduces the amount of repeated lines of code. The unified interface is additionally predestined for fast prototyping of new ideas and simple comparative run settings.

All modules found in this section load the SegNet as input and produce SPTCs as output.

Shared Classes among modules of this section:

- PathCollectorRecursive:

  `findPaths(SegNet)` is the main function of the PathCollectorRecursive and takes all seeds and collects all possible paths starting from that seed.

- SpacePointTrackCandCreator:

[94] A storage container used for exchanging data between modules in basf2.

`createSPTCs(tcContainer, paths)` is the main function of the SpacePointTrackCandCreator and creates a SPTC for each path given including the momentum seed (if needed).

7.5.1 Basic Path finder

The Basic Path Finder module uses the same infrastructure as the CA and shares code with the CA. Its task is to process a naive path-following approach to compare its performance with the CA and is therefore primarily intended for testing purposes.

The module searches for all outer ends in the SegNet and then collects all possible paths leading inwards. All collected paths with sufficient number of hits are then stored as SPTCs to be processed by the succeeding steps. This implies creating a momentum seed for following modules, which is a momentum vector estimation for the innermost hit of the TC created. Compared to the CA, there is no Cell state, which increases the possible number of paths collected, but also increases the number of ghost TCs collected.

First tests showed higher track efficiencies than the CA for very high occupancies and badly tuned Filters, but this additionally results in much higher numbers of preliminary TCs collected. The speed of the CA is comparable with the speed of the basic path finder in all realistic scenarios.

7.5.2 Cellular Automaton

The Cellular automaton used for the VXDTF 2 is a ported version of the CA described in Section 4.4.1, and has been adapted to the changed interface. No algorithmic changes were made.

Like the Basic Path Finder of the previous section it uses the "PathCollectorRecursive" and the "SpacePointTrackCandCreator".

Since the Basic Path Finder is not faster, but produces higher numbers of ghost TCs, the CA is used as standard setting for this step.

7.5.3 Combinatorial Kalman Filter

The CKF module could not be implemented in the VXDTF 2 within the time given for this thesis and therefore can only be sketched here as a starting point for possible future implementations.

Section 2.7.1.4 describes briefly how the algorithm works. The key points are the definition of a seed by using three preselected hits and extrapolation of the estimated track parameters for that seed. The extrapolation leads to hits on sensors along the extrapolated path, where the n ($x \geq 1$) best next inner hits for each sensor passed are collected. For each hit collected this way the path is continued independently until the end of the detector is reached or the χ^2/QI-threshold is exceeded.

The SegNet can serve as a starting point and can help to reduce the combinatorial issue, since it already has preselected chains of hits stored in its network. Additionally it is very easy to find useful seeds for the algorithm since one simply needs to loop over all outer ends of the network to get these seeds. One specialty has to be considered, though. The SVD has only four layers, and the median length of TCs is therefore only four hits, which means that a typical seed of three hits can only perform a single hit-collecting step, rendering this approach effectively useless, Therefore two alternatives could be performed instead:

- Use a basic KF instead, since it will not be slower for the case of four hits and three hits serving as the input.
- Use a seed with only two hits and the virtual IP as a helper hit to be able to create a sufficiently valid momentum seed. This would lead to the situation that at least two hit collecting steps can be performed for the median TC length.

The CKF provides its own QI and therefore step 3 of the following section can be skipped for it.

7.6 Step 3: Quality Estimator

The output of the modules of this step is a Quality Indicator (QI) which is a real number in the range [0;1], where a value close to 0 indicates a bad TC, while a value close to 1 indicates a good one. QIs are frequently based on χ^2 probabilities, but other approaches are allowed nonetheless. The only essential requirement is to not mix QIs of different estimators since they need to be comparable in the following steps. Many of the following estimators are not yet implemented since the VXDTF 2 is not yet ready to be used. Estimators that are not finished or not yet implemented will be marked in their respective sections.

7.6.1 Kalman Filter

The Kalman filter is, like in the implementation used for the VXDTF (more details on the old version can be found in Section 4.5.5), a simple interface to the GENFIT package [33], where many sophisticated fitters like the Kalman filter are already implemented. This module is not yet implemented.

7.6.2 Circle Fit

The circle fit is, like in the VXDTF version, based on the Karimäki paper [39]. Since the CDC TFs use their own implementation of the same paper, it is planned to merge the code to use one unified fitter. However, this has not yet been done and therefore the current implementation uses the one presented in Section 4.5.3.

7.6.3 Deterministic Annealing Filter

The deterministic annealing filter is a very sophisticated technique introduced by Strandlie and Frühwirth [48], where hits compete with each other to find outliers. Like for the Kalman filter this algorithm is already implemented in GENFIT [49] and therefore this module can interface directly to the GENFIT version.

Like the interface to the Kalman filter, this module is not yet implemented.

7.6.4 Helix Fit

A relatively fast helix fit like the one described in Section 4.5.4 is planned to be implemented too. Like for the circle fit the same paper was used by the CDC TF and the VXDTF and therefore a merged implementation is planned to be installed.

This module is not yet implemented.

7.6.5 Line Fit

Although the circle fit and the helix fit algorithms chosen perform safely in the straight line case, for runs without magnetic field a line fit is relevant. The reason for this lies in the small number of layers, where a line fit could perform an easier fit than a circle or helix fit, since the number of parameters to be estimated is smaller and therefore a smaller number of hits are sufficient to get a useful result. For the most typical case of four hits in the SVD, the number of degrees of freedom is then significantly higher for a line fit than for a helix fit. The line fit implemented in VXDTF will not be ported, though, as a different parametrization is required.

This module is therefore not yet implemented.

7.6.6 Four-hit Filters

In the redesign the four-hit Filters do not have their proper place yet. Since they are very fast, they could be implemented as an intermediate step between step two and three. Additionally that sub-step does only make sense if the SPTCs were created by the Basic Path Finder or the CA, since the CKF filters TCs which would be filtered by the four-hit filters in any case.

7.7 Step 4: Find clean subsets

This step is actually consisting of two sub-parts. In the first sub-part, the overlaps of TCs are detected and in the second one these overlaps are removed by deactivating preliminary TCs until the final TCs are cleaned. In Section 7.7.1 the module which finds the overlaps is defined, while the subset cleaning modules are discussed in Section 7.7.2 and in Section 7.7.3.

7.7.1 SPTC Network Producer

This module was implemented as a first working overlap finding module. Its design is overly complicated and should be replaced by a more readable and faster version. As already identified in Section 5.4.4 and Figure 5.27a in particular, the number of preliminary TCs can become very high for $\Upsilon(4S)$ events with full background. Therefore the implementation must be very fast. Additionally the module defines what actually constitutes an overlap. Some possible definitions are listed here:

- Two TCs are overlapping if they share a single cluster. This is the standard way used for the old VXDTF.
- Two TCs are overlapping if they share a SpacePoint, which is a combination of two clusters for the case of the SVD.
- Two TCs are overlapping, if they share a pair of SpacePoints, which is a Segment of the TC, as defined in Section 7.4.1.5.

The algorithms for finding a clean subset of TCs only need to know which TCs are overlapping, independent the actual definition of an overlap. The best trade-off between a high track finding efficiency and a low ghost rate depends on the cleaning algorithm and the correct choice of definition for overlaps. Therefore it is suggested here to perform such studies using the different overlap definitions from above.

7.7.2 Hopfield Neural Network

The SPTC network is the input to the HNN that finds subsets of non-overlapping TCs. The module uses a code ported from VXDTF (see Section 4.6.2 for more details) and gives the same results. In the long run some code optimizations should be applied since the current version still uses very slow classes for vectors and matrices.

7.7.3 Greedy Algorithm

Like for the HNN the greedy algorithm was ported from the VXDTF version described in Section 4.6.1. The module uses the same input as the HNN module.

7.8 Step 5: Reserve Hits

Track finding can be performed in several passes per event, using different SecMaps for different momentum ranges. In order to reduce the combinatorics in the subsequent passes, the best TCs of a pass can block further use of their hits, i.e. clusters or space points. In the VXDTF, the best — in terms of the QI — x% of the TCs found reserved

their hits. In this step, other ways to find these "best" TCs can be explored and compared to the QI approach.

The module is not yet implemented, but is needed for any test performing more than one pass.

7.9 Preliminary results

The VXDTF 2 is still under construction and many essential parts are still missing. For example it is currently not possible to store a SecMap, which therefore needs to be recreated for each test run. Additionally using several SecMaps to cover different momentum ranges cannot yet be tested. A valid clean subset step is still missing as well, which means that the final results contain overlapping TCs and a non-zero number of clone TCs. More details about important markers for estimating the reconstruction efficiency can be found in Section 5.1.1. Another issue is the quality of the analyzer module, which had to be rewritten to fix some design flaws in the old version and to support the new interface. This leads to the situation that even the analyzer itself can not be fully trusted yet, since there are still some bugs hidden in its code. Therefore all results listed in this section are preliminary and shall solely be regarded as a snapshot of the current development and as a sneak preview of what is yet to come.

To examine the behavior of the VXDTF 2, a training sample was created with 100,000 $\Upsilon(4S)$ events and an additional particle gun creating low momentum muons. This training sample was then used to train a SecMap for the VXDTF 2. This SecMap covers the full momentum range and the full geometry of the detector.

The test sample contained 3,000 $\Upsilon(4S)$ events with 100% background added. The training and the test sample do not share events.

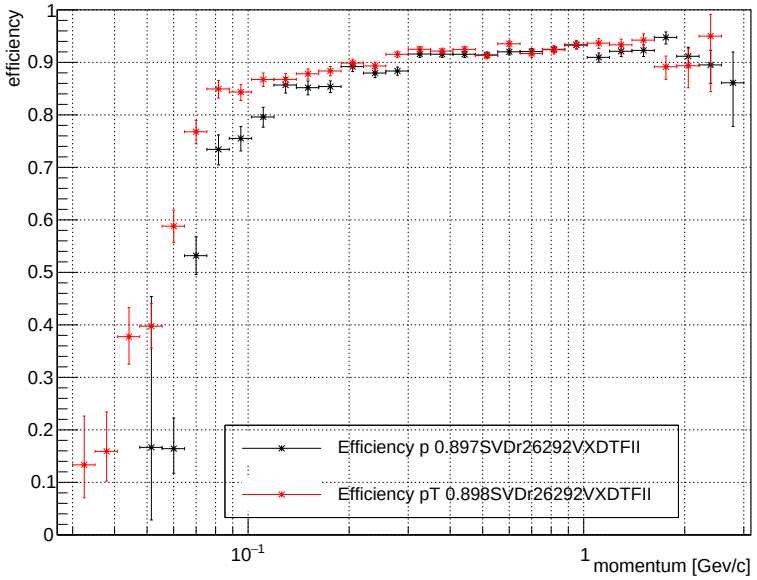
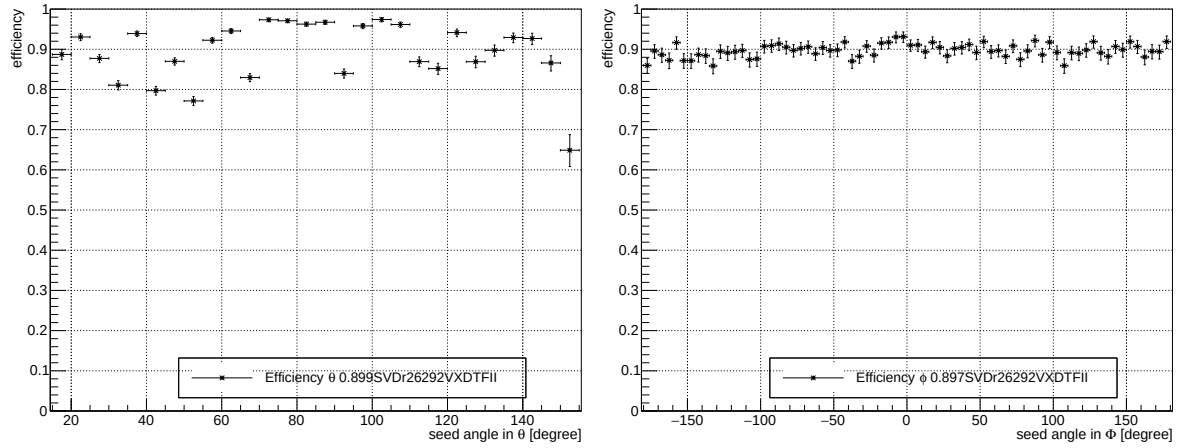


Figure 7.2: Efficiency versus momentum (black, total efficiency over the p range displayed = 89.7%) and transverse momentum (red, total efficiency over the p_T range displayed = 89.8%) for the case of 100% background.

The reconstruction chain contained the SegNetModule, the CA and the analyzer module. Reference TCs were produced by a MC TF and were filtered to have at least three SpacePoints per track. This setting differs from the settings used for the old VXDTF,

which did not have a feature for filtering by SpacePoints. This should therefore lead to slightly better efficiencies if no bugs or other problems of the VXDTF 2 would exist.

In Figure 7.2 the efficiency versus the (transverse) momentum is plotted. The overall picture is comparable to Figure 5.11, where the VXDTF was run with 100% background as well. Differences are visible in the low momentum range below about 70 MeV/c, where the VXDTF 2 has a worse efficiency as the VXDTF. For higher momenta above about 200 MeV/c the efficiency level is practically stable above 90% efficiency.



(a) Efficiency versus Theta, total efficiency over the θ range displayed = 89.9%.

(b) Efficiency versus Phi, total efficiency over the Φ range displayed = 89.7%.

Figure 7.3: Efficiency versus Theta (Figure 7.3a) and Phi (Figure 7.3b) for the case of 100% background.

Figure 7.3 shows the efficiency versus θ (Figure 7.3a) and Φ (Figure 7.3b). Especially the efficiency versus θ shows some interesting structures. There are several dips in the efficiency and some of them coincide with smaller dips that can already be seen in the results of the VXDTF (Figure 5.12a). This indicates that it can be a geometrical issue like bad constellations of hits, or the Filters used for both TFs might have some issues with certain values. Since the sample of reference tracks is not identical for both TFs, which could have been another possible source for similar patterns in both TFs, a problem related to the sensor geometry seems to be the most likely culprit. For the efficiency versus Φ as shown in Figure 7.3b there seems to be no relevant dip to be seen.

VXDTF and VXDTF 2 differ in their capability to store Filter cuts in the SecMap. While the SecMap of VXDTF can only store two-Sector combinations and therefore all Filter cuts are limited to that constraint as well, the SecMap of VXDTF 2 allows to store three-Sector combinations as well, which is a considerable advantage for the effectiveness of three-hit Filters. This can be verified by looking at the acceptance rates for two and three-hit Filters: The acceptance rate for two-hit Filters of the VXDTF 2 (33%) is worse than for the VXDTF in case of $\Upsilon(4S)$ events with 100% background (11.5%). The reason why the VXDTF 2 has a worse acceptance rate than its predecessor

is simple: Although both use the same set of Filters, their cuts differ. The VXDTF can work with three different passes and therefore applies far more precise cuts for the different momentum ranges, while the VXDTF 2 can only apply a single SecMap for the whole momentum range at the moment and therefore has to work with less precise cuts. For three-hit Filters, however, the situation changes. There the VXDTF 2 has a better acceptance rate of 20.1%, while the VXDTF has a worse one of 28%. There are two effects responsible for that. First, as mentioned above, VXDTF 2 can now store independent cuts for a three-Sector combination, which results in more precise cuts. Second, the higher number of accepted two-hit combinations means that more of the "bad" combinations are accepted than is the case for the VXDTF. Therefore the three-hit Filters can filter them as well, since they are more powerful than two-hit Filters, having more information available because of the third hit. Both effects together seem to outweigh the fact that still only a single SecMap for the full momentum range was used.

As mentioned at the beginning of this section, the final TCs of the VXDTF 2 are not yet filtered to get a clean subset of non-overlapping TCs. Therefore the ghost and clone rates are high. Of about 290 TCs reconstructed per event, 73% are clones, 23.9% are ghosts and only about 3% are connected to a reconstructed track. The mean number of preliminary TCs created by the VXDTF in its run was about 273 TCs, which means that the VXDTF 2 is not much less efficient in its run. Since the overall efficiencies are comparable as well, the results shown here for the VXDTF 2 look promising when considering its early stage of development.

CHAPTER 8

OUTLOOK

The basic concept of the VXDTF has been developed by Rudolf Frühwirth and since October 2010 a working concept was built and then the full version was implemented, tested and expanded by the author of this thesis. Several students did provide input as well; Thomas Fabian (testing of Filters), Stefan Ferstl (interface to the event display) and Thomas Madlener (tests of machine learning Filters) helped to improve the VXDTF. VXDTF 2 was co-developed with Eugenio Paoloni, which implemented the new container of the SecMap and defined the way to use Filters, while the author did the rest: implement the overall structure and the separate modules, outsource the SpacePoint and write tests for cross-checks.

In March 2016 the author of this thesis mostly stopped working on the VXDTF 2 implementation to be able to write this thesis and several other groups took over, who then will complete the work. There is still much work to do and several milestones need to be achieved yet. First the VXDTF 2 needs to surpass its predecessor, the VXDTF. At the moment the VXDTF is still used as the standard TF for SVD and VXD tracking, but shall be replaced by VXDTF 2 as soon as possible. By the end of 2017 the VXDTF 2 then needs to be able to reconstruct first events of real particle decays in a test setup of one single ladder of each layer of the VXD, which will be installed in the Belle II detector to test if the reconstruction of tracks will work and if the readout chain will work. This is more than a simple beam test, since it will be deployed at the SuperKEKB and not a beam test facility and the events recorded will be real particle decays coming from e^+e^- collisions. This simplified but still sophisticated setup is called BEAST[95] Phase II and will run for a few months with the SuperKEKB set to the resonance energies for $\Upsilon(nS)$ and record first physics events then. After a few months of that test run the machine will be shut down again and the BEAST Phase II installation will be replaced by the actual VXD. The installation of the VXD will be completed in April 2018 and then used

[95] Beam Exorcism for A STable experiment

for cosmic runs until the final preparation for the reactivation of the SuperKEKB are done. The VXDTF and its successor VXDTF 2 are not suitable for cosmic runs and therefore will probably not be used for that. First physics runs with the full Belle II including the full VXD will then start in November 2018, where the official main data taking cycle for Belle II and the VXDTF 2 will begin and last for several years.

For the VXDTF and its successor VXDTF 2 then the time has come to finally live up to what was started nearly a decade before. With the tracks it will reconstruct the TF will be able to help find new insights into how the universe is working and will help to find what might lie beyond the Standard Model of High Energy Physics. For several years it will be used side by side with other TFs implemented and will reconstruct several billion events until then. It therefore will have a relevant impact on the outcome of the Belle II experiment.

Vienna, October 2016

Part II

Technical details.

CHAPTER 9

FORMULAS OF N-HIT FILTERS

The formulas are kept short by using the results of other Filters as parameters. x_1, y_1, z_1 represent the global x, y and z coordinates of the first/outer hit ($\widehat{=}\,\overrightarrow{h_1}$). $\Delta x_{12} \widehat{=} x_1 - x_2$ is the signed x-distance between $\overrightarrow{h_1}$ and the next inner hit $\overrightarrow{h_2}$. f_a represents the result of the formula for a Filter with the shortcut a.

The dimension of the result is written in square brackets.

9.1 Two-hit filters

The formula for *distXY* has the unit of [cm^2] and is:

$$\Delta x_{12}^2 + \Delta y_{12}^2 = f_{XY} \tag{9.1}$$

The formula for *dist3D* has the unit of [cm^2] and is:

$$f_{XY} + \Delta z_{12}^2 = f_{3D} \tag{9.2}$$

The formula for *distZ* has the unit of [cm] and is:

$$\Delta z_{12} = f_Z \tag{9.3}$$

The formula for *slopeRZ* has the unit of [rad] and is:

$$\arctan \frac{\sqrt{f_{3D}}}{\Delta z_{12}} = f_{sRZ} \tag{9.4}$$

The formula for *nDist3D* has no unit and is:

$$\frac{f_{XY}}{f_{3D}} = f_{n3D} \tag{9.5}$$

9.2 Three-hit filters

The formula for *angleXY* has the unit of [cm^{-2}] and is:

$$\frac{\Delta x_{12}\Delta x_{23} + \Delta y_{12}\Delta y_{23}}{(\Delta x_{12}^2 + \Delta y_{12}^2)(\Delta x_{23}^2 + \Delta y_{23}^2)} = f_{aXY} \tag{9.6}$$

The formula for *angle3D* has the unit of [cm^{-2}] and is:

$$\frac{\Delta x_{12}\Delta x_{23} + \Delta y_{12}\Delta y_{23} + \Delta z_{12}\Delta z_{23}}{\left|\left|(\overrightarrow{h_1} - \overrightarrow{h_2})\right|\right|^2 \left|\left|(\overrightarrow{h_2} - \overrightarrow{h_3})\right|\right|^2} = f_{a3D} \tag{9.7}$$

The formula for *angleRZ* has no unit and is:

$$\text{with } r_{12} = \sqrt{\Delta x_{12}^2 + \Delta y_{12}^2} \quad \Rightarrow \quad \frac{r_{12}r_{23} + \Delta z_{12}\Delta z_{23}}{\sqrt{(r_{12}^2 + \Delta z_{12}^2)(r_{23}^2 + \Delta z_{23}^2)}} = f_{aRZ} \tag{9.8}$$

The formula for *circleDist2IP* has the unit of [cm] and is:

$$\text{with } z_i = 0: \quad \Rightarrow \quad \left|\left|\left|\overrightarrow{origin} - \overrightarrow{f_{CC}}\right|\right| - f_r\right| = f_{CCd2IP} \tag{9.9}$$

The formula for pT has the unit of [GeV/c] and is:

$$f_r \left|\left|\vec{B}\right|\right| c10^{-11} = f_{p_\top} \tag{9.10}$$

The formula for *deltaSlopeRZ* has the unit of [rad] and is:

$$f_{sRZ23} - f_{sRZ12} = f_{dsRZ} \tag{9.11}$$

The circle center using Cramer's rule defined in eq. (9.12) is used for increased readability in some of the following formulas.

$$\begin{aligned} \boldsymbol{A}x = \boldsymbol{b} \;\leftrightarrow\; \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} x = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \;\leftrightarrow \\ \begin{pmatrix} y_2 - y_3 & x_3 - x_2 \\ y_2 - y_1 & x_1 - x_2 \end{pmatrix} x = \begin{pmatrix} \frac{x_1+x_2}{2} - \frac{x_2+x_3}{2} \\ \frac{y_1+y_2}{2} - \frac{y_2+y_3}{2} \end{pmatrix} \end{aligned}$$

$$\text{determinant: } s = \frac{b_1 a_{22} - b_2 a_{21}}{a_{11} a_{22} - a_{12} a_{21}} \tag{9.12}$$

$$\begin{pmatrix} \frac{x_2+x_3}{2} + s(y_2 - y_3) \\ \frac{y_2+y_3}{2} - s(x_2 - x_3) \\ 0 \end{pmatrix} = \boldsymbol{f_{CC}} = \overrightarrow{f_{CC}}$$

Like f_{CC} of eq. (9.12) the radius of the circle — as calculated in eq. (9.13) — is used to increase the readability of the following formulas.

$$\text{with } z_i = 0: \quad \Rightarrow \quad \frac{\left|\left|\overrightarrow{f_{CC}} - \vec{h_1}\right|\right| + \left|\left|\overrightarrow{f_{CC}} - \vec{h_2}\right|\right| + \left|\left|\overrightarrow{f_{CC}} - \vec{h_3}\right|\right|}{3} = f_r \tag{9.13}$$

The formula for *deltaSoverZ* has the unit of [rad . cm], uses helpers $\alpha_{...}$ and the result taken from eq. (9.12) to increase readability:

$$\begin{aligned} \alpha_{12} &= \frac{(\vec{h_1} - \overrightarrow{f_{CC}}).(\vec{h_2} - \overrightarrow{f_{CC}})}{\left|\left|\vec{h_1} - \overrightarrow{f_{CC}}\right|\right| \left|\left|\vec{h_2} - \overrightarrow{f_{CC}}\right|\right|} \\ \alpha_{23} &= \frac{(\vec{h_2} - \overrightarrow{f_{CC}}).(\vec{h_3} - \overrightarrow{f_{CC}})}{\left|\left|\vec{h_2} - \overrightarrow{f_{CC}}\right|\right| \left|\left|\vec{h_3} - \overrightarrow{f_{CC}}\right|\right|} \\ \Rightarrow \quad & \arccos(\alpha_{12})\Delta z_{23} - \arccos(\alpha_{23})\Delta z_{12} = f_{dSZ} \end{aligned} \tag{9.14}$$

The formula for *deltaSlopeSoverZ* has the unit of [rad . cm], uses helpers $\alpha_{...}$ and the result taken from eq. (9.13) to increase readability:

$$\arcsin \frac{\Delta z_{12}}{\arccos(\alpha_{12}) f_r} - \arcsin \frac{\Delta z_{23}}{\arccos(\alpha_{23}) f_r} = f_{dsSZ} \tag{9.15}$$

The formula for *helixParameterFit* has no unit and uses helpers $\alpha_{...}$

$$\frac{\alpha_{12}\Delta z_{23}}{\alpha_{23}\Delta z_{12}} = f_{hp} \tag{9.16}$$

9.3 Four-hit filters

The formula for *deltaPt* in [GeV/c] is:

$$(f_{r123} - f_{r234}) \left|\left|\vec{B}\right|\right| c10^{-11} = f_{dp_\top} \tag{9.17}$$

The formula for *deltaDistCircleCenter* in [cm] is:

$$z_i = 0 : \left|\left|\left|\overrightarrow{f_{CC123}}\right|\right| - \left|\left|\overrightarrow{f_{CC234}}\right|\right|\right| = f_{ddCC} \tag{9.18}$$

CHAPTER 10

DISCUSSION ABOUT DESIGN DECISIONS

During the redesign process for the VXDTF 2 there were some situations where decisions had to be made, where other people might have decided otherwise. Those decisions, which are likely to be worthy for discussions, are presented in this section and the arguments, why the taken paths were chosen, are listed in the following sections.

10.1 Why one should use multi-hit segments

In the last few years the CA as a main driver behind the Segment-idea evolved in manifold directions. One of the major developments was to introduce Segments containing more than the classical "two-hit Segments" used until then. This is an important step to allow more fine-grained filtering of bad combinations of Segments. The reason for that is that two-hit Segments in the CA-concept form neighborhoods with other two-hit Segments, which are then filtered by Filters using the three hits of two neighboring Segments (two neighboring Segments share one hit). This means that the most complex Filters one can apply can not use more information provided as of those three hits. A typical example for such a Filter is Angle3D. Filters using four or more hits can be applied on three connected Segments as well. The information, however, can not be stored when staying at the two-hit Segment level since two pairs of connected Segments — which share the middle Segment — can both be valid Neighbours but the overall combination could be invalid when applying a four-hit Filter like ziggzaggXY. This information of the "Neighbour of my Neighbour" can not preserved this way. Of course one could start collecting all valid paths through that network and then apply these more sophisticated Filters later-on, but in that case paths would be collected which would not survive that step. Depending on the occupancy of the events and therefore the combinatorial issue, this would mean a lot

of needless work. Therefore using three-hit Segments would allow to map the results of four-hit Filters and longer Segments could accept even better Filters (starting with about four hits (three hits are sufficient, but allow only one degree of freedom), the KF would be one of these Filters. For cases when there are no full hits (consisting of a 2D-pixel or two 1D-strips combined) available, this problem is even more severe, since then even comparably simple Filters run into that problem.

10.2 Why we didn't

Although the internal design of the SegNetModule supports multi-hit Segments. They are not used in the end. Several reasons contributed to this decision:

- The longer the Segments the higher the overhead, since three-hit Segments need to share two hits with neighboring Segments or filtering information is lost. Therefore introducing longer events can lead to slower reconstruction speed.
- The more hits used per Segment, the more layers needed for that step. It is clear that one can not build 10-hit Segments when having only three layers and therefore simply not enough hits for such Segments. Belle II can provide on-line information of the SVD, which has four layers of double-sided strip sensors.
- The longer the Segments, the more difficult it gets to deal with non-standard cases. Of course having four layers and expecting to have four-hit TCs strongly support using at least three, or even better four-hit Segments, since that would provide completely drafted paths which are easily collected as TCs without further steps. But having the possibility of missing hits, broken sensors or simply two hits in one layer due to the overlapping parts means that one can have tracks with 3–8 hits in our four-layer detector. Requesting longer Segments would automatically filter those shorter stubs. Although one could work with bypasses, this would increase again the amount of work and would minimize the boosting effect of multi-hit Segments.
- Next to these theoretical issues mentioned above, the hardest support for our decision can be seen in Figure 5.28, which clearly states that the highest amount of lost time is coming from finding overlapping TCs, which already had a full fit applied and therefore only good TCs are existing at that point. This effectively means that the time gain of higher order Filters comparatively small. Therefore one should invest the effort needed for implementing multi-hit cells into improved pre-selection via one and two-hit Filters.

10.3 TrackNodes versus SpacePoints

One of the issues occurred when redesigning VXDTF, was the proper replacement of the hit class used in the old design: `VXDTFHit`. The problem was that the chosen successor of that class, the SpacePoint, could not take over all tasks of the old hit class due to restrictions laid on objects to be stored on the official interface for modules in basf2, the `StoreArray`. This section does discuss the issue in more detail to allow retracing the decision made.

The issue about the necessity of `TrackNode`s as an intermediate class to SpacePoints has been discussed at several meetings. One mail of the discussion is quoted here as a reminder:

> *[...]*
> *As a consequence, there is no 1–1 correspondence between SpacePoints and VXDTFHits:*
>
> - *there are cases when SpacePoints are created, but no VXDTFHit: e.g. the SpacePoint does not lie in a valid Sector (e.g. low momentum pass), or when former passes have reserved a certain cluster or cluster-combination.*
> - *there are cases when VXDTFHits are created, but no SpacePoints: e.g. virtual interaction point, for each valid cluster-combi, nSpacePoints == 1, but nVXDTFHits leq nPasses. Relations between StoreArray objects (which are an approach proposed by Martin Heck last autumn in Pisa) can not completely cover tasks formerly fulfilled by the VXDTFHit, since that interface is not designed for efficiently switching on and off relations.*
>
> *[...]*

The implemented design of the SegNet reduces the need of a special `TrackNode`-class since old essential things like the connection to other hits are now stored by the `DirectedNodeNetwork` and the definition of overlapping TCs is now handled in a different way and independently implemented in the `SPTCNetworkProducerModule`. Still the issue needs to be resolved since a SpacePoint can not know which active-sector it is attached to. The other main issue is the treatment of the virtual interaction point (VIP). This means that there are two options with their individual advantages and downsides:

1. Do not introduce an extra class for the `TrackNode`s, but use SpacePoints instead.

   This has the advantage, that the code can be kept shorter and no extra implementations are needed. The downside on the oder hand is that the hit for the IP does not fit to the other SpacePoints in various aspects:

   - It is not associated to a detector (`SpacePoint::m_sensorType = ?`, maybe `Const::IR`?).

- It does not lie on a sensor (`SpacePoint::m_vxdID = ?`, maybe 0?).
- It has therefore got no local coordinates (`SpacePoint::m_normalizedLocal = ?`, maybe `{0, 0}`?).
- There is no constructor allowing to set all variables needed for a virtual interaction point (e.g. `SpacePoint::m_position` and `SpacePoint::m_clustersAssigned`) when there is no `XYZCluster` given. When adding such a constructor, its only purpose would be to construct a SpacePoint to be used as virtual IP, which seems a bit ill-designed.
- How to store it? Since it is not of the same detector type as the others, a `StoreArray` just for the hit of the virtual IP would have to be used so the modules can access it like the other SpacePoints.

The second (and more severe) downside is the storing of the Sector-link to a SpacePoint. While this is no problem for building SpacePointNetworks, since ActiveSectorNetworks are used as a starting point, the issue becomes apparent when building SegmentNetworks: the SpacePoint have no Active- or StaticSector-Information and therefore have lost their information. This could be solved using relations between StaticSectors and SpacePoints, but that could not be tested so far.

2. Introduce an extra class `TrackNode`.

   Advantage is that many of these issues mentioned above can be encapsulated in the `TrackNode`-class, but it would mean that another layer of abstraction has to be introduced just for virtual IPs (and the Sector-information) again. Storing would be solved by keeping the information directly in the output of the SegNet.

As already indicated in Section 7.4.1.4, the current implementation does use `TrackNodes` as an intermediate class.

10.4 Building TrackNode and Segment Networks

Concerning the building of our second (`DirectedNodeNetwork<TrackNode>`) and third (`DirectedNodeNetwork<Segment>`) network, there are two alternative ways:

- In a *directed* manner where `sectors = activeSectorNetwork.getOuterEnds()` or in an
- *undirected* one, where `sectors = activeSectorNetwork.getNodes()` will be used.

Going through the `DirectedNodeNetwork` in an undirected way is easier because of the fact that one does not have to care whether or not all `DirectedNodes` have really been passed as intended, since all have been visited anyway. The downside is that there could

be cases that the same `TrackNode` will be added more than once. This is captured by the design of the `DirectedNodeNetwork` and therefore will not lead to unintended behavior, but the issue of the possible overhead remains and has to be studied in more detail, before deciding if refactoring that part is needed. Since that implementation is a bit tricky, the undirected approach has been used for now, since the code for that one is easier to read.

CHAPTER 11

VXDTF - SETTINGS AND SETUP

The VXDTF including all other modules used, are implemented in the basf2 framework, which is described in close detail in [20] and [50].

11.1 Setup of the simulation for evtGen-runs

For the studies presented in Chapter 3 and Chapter 5, the `evtGenInput`-Module in its standard-settings has been used. The `evtGenInput` module creates $\Upsilon(4S)$-events using a decay file with branching ratios taken from the PDG2010-version [22]. Uncertainties coming from the spacial expansion of the IP itself are taken into account, but because of the nano beam scheme [4], these uncertainties are very small and therefore the primary vertices of the trackable particles lie very near to the origin (more details on that in Chapter 3). Additionally a perfect alignment is assumed and the geometry is reduced to contain only the beampipe, the PXD and the SVD. It is also assumed that there are no dead pixels or strips. The magnetic field is set to a constant value of 1.5 T, which has been cut of at the outside of the SVD, which prevents curlers from the CDC coming back but not curlers from staying in the SVD-region. As a trade-off between a big sample-size and time consumption for studying effects, the evtGen-sample created for the studies mentioned above has 30,000 events, which is the expected amount of physically relevant events produced in one second of a Belle II-run at target luminosity. The background-files taken for studies of the VXDTF-behavior are created using the same settings as for the 12th-background campaign, but contains accumulated background-events over 20,000 µs instead of only 2,000 µs.

11.2 VXDTF standard setting

The standard setting of the VXDTF uses three momentum passes where each pass uses its own SecMap. The first pass applied is the high momentum pass, the second one the medium momentum pass and the last one is the low momentum pass. All SecMaps were trained on a training sample containing 150,000 $\Upsilon(4S)$ events, where each event additionally contained 10 muon tracks created by a particle gun uniformly distributed over the full θ and Φ range of the detector. These muon tracks were uniformly distributed over a transverse momentum range of $30\,\mathrm{MeV}/c - 3.5\,\mathrm{GeV}/c$ and had a smeared start vertex in range $x \in [-0.1\,\mathrm{cm}; 0.1\,\mathrm{cm}]$, $y \in [-0.1\,\mathrm{cm}; 0.1\,\mathrm{cm}]$ and $z \in [-0.5\,\mathrm{cm}; 0.5\,\mathrm{cm}]$.

11.2.1 SecMap dependent settings

- The high momentum pass uses reference tracks for training of the momentum range $p_{high} \geq 400\,\mathrm{MeV}/c$. It does *not* use the "ziggZaggXY" filter during reconstruction. The SecMap `shiftedL3IssueTestSVDStd-moreThan400MeV_SVD` is used for this pass.

- The medium momentum pass uses reference tracks for training of the momentum range $100\,\mathrm{MeV}/c \leq p_{medium} \leq 400\,\mathrm{MeV}/c$. It does use the "ziggZaggXY" filter during reconstruction. The SecMap `shiftedL3IssueTestSVDStd-100to400MeV_SVD` is used for this pass.

- The low momentum pass uses reference tracks for training of the momentum range $25\,\mathrm{MeV}/c \leq p_{low} \leq 100\,\mathrm{MeV}/c$. It does use the "ziggZaggXY" filter during reconstruction. The SecMap `shiftedL3IssueTestSVDStd-25to100MeV_SVD` is used for this pass.

11.2.2 General SecMap settings

All SecMaps for the VXDTF in this thesis cover all sensors of the SVD and use the same Sector settings for all sensors as well. For each sensor the relative coordinates in u and v are defined, where for each local dimension '0' marks the lower edge in the local coordinate system and and '1' marks the upper edge. The Sector cuts for u and v are 0., 0.33, 0.67, 1.0 and therefore form a 3×3 division of each sensor resulting in 9 Sectors per sensor. To get rid of outliers in terms of start vertex position of tracks, tracks from the training sample which have a start vertex distance of more than 1 cm in the xy plane from the origin or more than 2 cm in the z direction from the origin are neglected for training. Only the first outgoing arms of the tracks are considered and therefore no back-curling track segments are used. Additionally only tracks with at least 3 SpacePoints are accepted.

All SecMaps have the following Filters activated:

- two-hit Filters: “Distance3D”, “DistanceXY”, “SlopeRZ”
- two-hit + virtual IP Filters: “Angles3DHioC”, “AnglesXYHioC”
- three-hit Filters: “Angles3D”, “DeltaSlopeRZ”
- three-hit + virtual IP Filters: “DeltaPtHioC”, “DeltaDistance2IPHioC”

For Sector combinations occurring during training more than 1,000 times the min and max quantiles of 0.001 and 0.999 are used for the Filter cuts. For smaller sample sizes the smallest and the highest entries are used for those cuts. For each Sector having more than one Friend stored, all Sector-Friend combinations are checked for their relative frequency in the training sample. Sector-Friend combinations with a relative frequency of less than 0.5% are discarded.

11.2.3 Settings independent from the SecMap

The high occupancy Filters of the previous section (x-hit + virtual IP) are activated in an event if there is at least one sensor with 30 SpacePoints on it. The reconstruction of an event is aborted if more than 5,500 accepted two-hit combinations are detected (=“killEventForHighOccupancyThreshold”) or if more than 500 preliminary TCs do overlap (=“killBecauseOfOverlappsThreshold”).

11.3 VXDTF tough setting

The tough setting for the VXDTF is identical to the standard settings described in Section 11.2, with one exception, though: The “killEventForHighOccupancyThreshold” is increased to 50,000 accepted two-hit combinations and the “killBecauseOfOverlappsThreshold” is set to 5,000.

11.4 VXDTF II setting

The training sample for the SecMap of the VXDTF 2 contains 100,000 $\Upsilon(4S)$ events with the same settings as described in Section 11.1. Here a particle gun is also used to create 5 additional muon tracks per event with the transverse momentum range of 50–200 MeV/c. The minimum number of SpacePoints per track of the training sample is three and only tracks with a transverse momentum in the range [30 MeV/c ; 3.15 GeV/c] are considered for SecMap training. For the VXDTF 2 the following filters are activated:

- two-hit Filters: “Distance3D”, “DistanceXY”, “SlopeRZ”
- three-hit Filters: “Angles3D”, “DeltaSlopeRZ”

Only one SecMap is used covering all sensors of the SVD and a the local Sector cuts for u and v 0., 0.3, 0.7, 1. and therefore result in $3 \times 3 = 9$ Sectors per sensor. For all Sector-Friend combinations occurring during training, the min and max values of the samples are used for the Filter cuts.

11.5 Computer setup

The computer used for the studies is a laptop computer running on ubuntu 14.04 LTS using 8 GByte of RAM and a Intel(R) Core(TM) i7-4710HQ CPU at 2.5 GHz with 4 physical cores and Hyper Threading. All tests of the VXDTF and the VXDTF 2 were performed on the revision 26292 from March 12th, 2016.

CHAPTER 12

ERROR-BAR HANDLING IN THE PLOTS SHOWN

This thesis uses ROOT [51] for many plots. All efficiency plots of Chapter 5 and Section 7.9 are created with it. In histograms error bars for efficiencies are not straight forward since the creation of such an efficiency histogram h_{eff} is performed using a division of $h_{eff} = \frac{h_{reco}}{h_{ref}}$, where h_{reco} stands for the histogram of reconstructed tracks and h_{ref} stands for the histogram of reference tracks. h_{reco} is a subset of h_{ref} because of the fact that only successfully reconstructed (means successfully matched with a reference track) tracks are to be found in h_{reco}. Therefore not a simple histogram division is used, but the `ROOT::TGraphAsymmErrors` class, which contains a proper error propagation for such cases and allows to create error bars correctly. In fact they calculate a 68.3% confidence interval using Clopper Pearson. It can not exceed an efficiency fraction of 1 ($\hat{=}$ 100% efficiency).

List of Figures

List of Tables

GLOSSARY

ab^{-1} measurement unit for high integrated luminosities.. 12, 191

AIDA The goal of this europan collaboration is to unify software approaches for data reconstruction in HEP experiments and analyzes technologies for tracking detectors to be used for the ILD experiment, http://aida2020.web.cern.ch/. 116, 191

B factory There are have been built three B factories so far: KEKB PEP-II and SuperKEKB.. 191, VII

BaBar The BaBar experiment, situated in the U.S.A.. Next to Belle II one of the first generation b-factory-experiments collecting data from 1999-2008 detecting collisions produced by the PEP-II collider.. 12, 191

BaseLineTF A pseudo-TF used as a fall-back-solution for the combined beam test.. 123, 191

Belle The Belle experiment, situated in Japan. Next to BaBar it was a first generation b-factory experiment collecting data from 1999-2010 detecting collisions produced by the KEKB collider.. 191, VII

Belle II The Belle 2 experiment, situated in Japan. Is the successor of Belle and currently the only b-factory experiment of the second generation.. 191, VII

CDC The third tracking detector is a wire-based gas-chamber detector in Belle II and envelopes the silicon based VXD-detectors. Used for tracking, momentum estimation and triggering in the HLT.. 16, 191

Cell Within the CA Cells represent the elements of the directed graphs. Cells have a integer state which can be increased if they have a Neighbour. In the VXDTF-package Sectors, hits and Segments are treated as Cells.. 5, 191

CERN The worlds biggest collaboration for particle physics. Situated in Geneva, Switzerland it has found essential particles of the SMHEP like the $W^{\pm}$-, Z^0 or the Higgs-Boson and is currently operating the LHC.. 114, 191

CESR Started collisions in 1979 and was run at about 10 GeV for nearly 20 years.. 11, 191
CKM The Cabbibo-Kobayashi-Masukawa-mechanism describes the ***CP*** violation as a complex phase in quark mixing between the three quark generations.. 11, 192
CMS One of the four big experiments installed at the LHC.. 59, 192
***CP* violation** A violation of the charge-parity-symmetry. Incorporated in the CKM-mechanism as part of the SMHEP and is at the moment the only experimentally supported source of the matter-antimatter-asymmetry in the universe.. 3, 192

DATCON An FPGA-based TF relying on Hough transformation used as an independent parallel tracker and ROI-finder to the HLT in on-line reconstruction, more details can be found in 6.3.1.. 4, 192
DESY Situated in Hamburg, Germany, it is currently one of the few sites capable of providing test beams needed for the VXD-development.. 192, V
DORIS-II Used for various purposes and used in the 10 GeV-region for first B-meson studies in the years of 1978-1992.. 11, 192

fb^{-1} measurement unit for high integrated luminosities. In b-factories 1 fb^{-1} can be translated in $\sim 1,000,000$ $\Upsilon(4S)$ events, which are therefore a bit less than 500,000 $\mathrm{B}^0\,\overline{\mathrm{B}}^0$-events.. 12, 192
Filter Filters are used in the VXDTF and the SegNetModule to narrow down the possible combinations of hits. They can be as simple as a distance measurement in 3D compared with a given range of cuts or as complex as a NN or a BDT[96].. 192, VIII
Friend Friends are the links or edges in the network of Sectors, they store which Sectors are compatible.. 58, 192

HER The ring containing the higher energy beam, the term is valid for all asymmetric colliders, especially for the B factories. In the case of SuperKEKB the HER does contain electrons with 7 GeV/c per particle.. 192
HLT A CPU-based farm running the basf2-framework and with it the VXDTF on SVD-data to do ROI-finding for the PXD.. 4, 192
HNN A feedback NN without hidden layers — and therefore the input being the output layer – and symmetric weights between the neurons.. 75, 192

ILC A planned linear collider experiment using leptons with higher energies LEP[97] was able to produce, it is planned to be used as a Higgs-factory. More informations at `http://arxiv.org/abs/hep-ph/0508181`. 188, 193
ILD A planned detector to be used at the ILC[98] experiment. More informations at `http://www.ilcild.org/`. 59, 193

[96] Boosted Decision Tree
[97] LEP: Large Electron–Positron Collider
[98] ILC: International linear collider

KEKB Operated from 1999 to 2010 it produced the events tracked and stored by the Belle-experiment.. 193, VII

LEP The biggest lepton accelerator ever build was situated at CERN, Switzerland/France and collected data from 1989-2000.. 188, 193
LER The ring containing the lower energy beam, the term is valid for all asymmetric colliders, especially for the B factories. In the case of SuperKEKB the LER does contain positrons with 4 GeV/c per particle.. 193
LHC The biggest hadron collider ever built, currently up to 13 TeV of collision energy are possible.. 193

Neighbour A Neighbour is a compatible inner Cell for given Cell. Cells are connected if the Filters applied have accepted that combination. Cells are compatible if they are connected and their states match.. 57, 193

PEP-II Operated from 1999 to 2008 it produced the events tracked and stored by the BaBar-experiment.. 11, 193
PXD The first tracking detector is a silicon based DEPFET pixel detector characterized by its high resolution of about 8 mega pixels, its thinness of only 75 µm and its relatively high readout rate of 50 kHz.. 193, III

SecMap A static map storing which Sectors are to be combined for tracking. Each Sector-combination has its own cuts for the Filters used.. 193, IV
Sector Within the VXDTF-package there exist two types of sectors: StaticSectors, which are subdivisions of sensors and are stored in the SecMap and contain Sector-combinations and cuts, and ActiveSectors which are created for each event and linked to their StaticSector. They store hits found in that event and are woven into a network for further steps.. 57, 193
Segment Segments in the VXDTF-package are two-hit-combinations of hit which were accepted by the Filters applied.. 57, 193
SegNet A SegmentNetwork is a directed network of accepted Segment-combinations. It is the base for all tracking algorithms used in the VXDTF-package, especially the CA.. 143, 193
SegNetModule Central module of the VXDTF-package building a SegNet in the process.. 143, 194
SMHEP The Standard Model of High energy physics describes what matter and anti-matter is built of and can not only describe commonly known Hadrons like protons and neutrons but also particles which are not observed in every-day-physics like Higgs-Bosons and Tauons.. 7, 194
SpacePoint SpacePoints are generalized hits in the VXD. In the case of the PXD SpacePoints are simply the global positions and -errors of the clusters found. In the case of SVD, the SpacePoints are the global positions and -errors for single Clusters or a combination of u- and v-clusters.. 5, 194

SPTC the SPTC is a container storing TCs consisting of SpacePoints and is used in the VXDTF 2 environment.. 143, 194
SuperKEKB The SuperKEKB-collider, situated in the Japan is currently the only b-factory in construction and will start in 2017 with commissioning phase. It uses e^+e^- for creating b-mesons at the $\Upsilon(4S)$-resonance.. 194, VII
SVD The second tracking detector is a silicon-based DSSD and used in the HLT for tracking and ROI-finding, its most outstanding feature is the time resolution of only 3 ns, allowing curler detection on-line.. 194, III

VXD A terminus combining the silicon based tracking detectors in Belle II, the PXD and the SVD due to their similarity in many aspects.. 194
VxdID Used in the basf2-framework to uniquely identify a sensor in the VXD-detector. It has the following naming scheme: LayerID.LadderID.SensorID. 194

X_0 The definition is depending on the particle type. For electrons/positrons it is the mean distance when about 62% of is energy is lost by bremsstrahlung. For photons it is $^7/_9$ of the mean free path for a pair production. X_0 is dependent on density and type of the material. 20, 194

ACRONYMS

ab^{-1} ab^{-1}: inverse atto-barn $\widehat{=}$ 1000 fb^{-1}. 12, 13
AIDA AIDA: The AIDA-2020 collaboration. 116, 119
APV 25 sensor readout chip originally developed for the CMS experiment, used by the SVD.. 117, 118, 186

B factories more details to be found in B factory. 3, 7, 10–13, 26
B factory B factory: a collider experiment optimized for producing and reconstructing B mesons. 3, 8, 11–13, 27, 191, VII
B meson charged: $B^+=u\bar{b}$, $B^-=b\bar{u}$, uncharged: $B^0=d\bar{b}$, $\overline{B}^0=b\bar{d}$. 3, 8, 11, 12, 35, 36, 191, VII
BaBar BaBar: experiment of the first generation B factories. 12, 16, 17, 32, 187, 189
BaseLineTF BaseLineTF: BaseLine Track Finder. 123, 127, 129
basf2 The Belle Analysis Software Framework 2, more details in [20]. 17, 56, 60, 74, 88, 110, 120, 127, 149, 167, 171, 188, 190, 194
BDT Boosted Decision Tree. 188
BEAST Beam Exorcism for A STable experiment. 157
Belle Belle: experiment of the first generation B factories. 12, 14, 16, 33, 83, 187, 189, VII
Belle II Belle II: experiment of the second generation B factories. 3, 4, 13–18, 20, 21, 24, 29–33, 35, 37, 38, 43, 44, 54, 55, 60, 72, 73, 77–79, 113, 114, 120, 122–125, 127, 137, 139, 157, 158, 166, 171, 177, 187, 190, 199, VII, VIII

CA Cellular Automaton. 4, 5, 31, 32, 57, 59, 62, 64, 65, 67–70, 77, 84, 90, 96, 99, 123, 144, 145, 150, 152, 154, 165, 180, 187, 189, 191, VIII
CDC CDC: Central Drift Chamber. 16, 28, 29, 44, 51, 54–56, 61, 75, 79, 151, 152, 171
Cell Cell: CA-Cell.. 5, 31, 57, 65, 68–71, 131–133, 135–137, 139, 145, 150, 184, 189, 193
CERN CERN: Conseil Européen pour la Recherche Nucléaire. 114, 189
CESR CESR: Cornell Electron Storage Ring. 11

CKF Combinatorial Kalman Filter. 32, 55, 77, 144, 150–152
CKM CKM: CKM mechanism. 11
CMOS Complementary Metal-Oxide-Semiconductor. 116
CMS CMS: Compact muon solenoid at the LHC.. 59
COPPER COmmon Pipeline Platform for Electronics Readout. 120
CP* violation** ***CP violation: charge-parity violation. 3, 8, 10–13, 188

DAF Deterministic Annealing Filter. 144
DAQ Data AcQuisition. 4, 114, 116, 119, 120, 192, 193
DATCON DATCON: DATa CONcentrator. 4, 113, 120–122
DEPFET DEPleted p-channel FieldEffect Transistor. 16, 116, 189, III, VII
DESY DESY: Deutsches Elektron Synchrotron. 4, 11, 72, 113, 114, 119, 124, 183, V, VIII
DHHC Data Handling Hybrid Controller. 120
DHP Data Handling Processor. 120
DOF Degrees Of Freedom. 72
DORIS-II DORIS-II: DOppel-RIng-Speicher. 11
DQM Data Quality Monitoring. 124, 125, 127, 140
DSSD Double Sided silicon Strip Detector. 15, 16, 190

EUDAQ DAQ system used by the EUDET projects to read out their Telescope. 119
EUDET Project for preparing detector readout and data aquisition towards the International linear collider.. 118, 119, 129, 192
EVB2 EVent Builder II used as final event builder in the DAQ chain.. 116, 120
Express Reco Express Reconstruction system. 55, 56, 120

FADC Flash Analog-to-Digital-Converter. 119
fb^{-1} fb^{-1}: inverse femto-barn, $\widehat{=}$ 500,000 $\mathrm{B}^0\overline{\mathrm{B}}^0$-events.. 12, 188, 191
Filter Filter: filters hits combinations by calculating a value, which is then compared with predefined cuts.. 57–71, 85, 90, 92, 98, 101, 111, 124, 141, 144, 145, 147–150, 152, 155–157, 161, 165, 166, 172–174, 180, 185, 186, 189, 193, VIII
flavor there are six flavors for quarks and six for leptons.. 9
FPGA Field Programmable Gate Array.. 4, 55, 121, 122, 188
Friend Friend: A compatible inner Sector connected to an outer one.. 58–61, 63, 64, 173, 174, 179, 180
FTB Finesse Transmitter Board. 119
FTSW Frontend Timing Switch. 119, 120

HEP High Energy Physics. 7, 187, 194
HER HER: High Energy Ring. 17, 18, 26, 27, 177
HLT HLT: High Level Trigger. 4, 55, 92, 113, 114, 120, 121, 123, 124, 127, 130, 140, 183, 187, 188, 190
HNN HNN: Hopfield Neural Network. 75, 76, 137, 139, 153

ILC ILC: International linear collider. 188
ILD ILD: International linear detector. 59, 187
IP Interaction Point. 3, 13, 14, 26, 27, 29, 31, 35, 37, 38, 40, 42, 46, 53, 54, 59–62, 67–71, 82, 86, 87, 123, 132, 143, 151, 168, 171, 173, 193

KEKB KEKB: first generation B factory in KEK. 11–14, 27, 185, 187, VII
KF Kalman Filter. 32, 49, 51, 74, 151, 166

LEP LEP: Large Electron–Positron Collider. 188
LER LER: Low Energy Ring. 17, 18, 26, 27, 177
LHC LHC: Large Hadron Collider. 59, 187, 188, 192

Machine Learning Automated training of parameters for algorithms using multivariate analysis.. 62
MC Monte Carlo. 22, 23, 25, 29, 32, 33, 37, 39–41, 44–47, 49–53, 60, 77, 79–81, 83, 84, 86–92, 94–102, 104–109, 113, 123–125, 132, 139, 154, 178–182, 185, VIII

Neighbour Neighbour: Neighbour-Cell. 57, 65, 70, 71, 149, 165, 187
NN Neural Network. 99, 101, 108, 109, 188

ONSEN ONline SElector Node. 120, 121

PEP-II PEP-II: first generation B factory in SLAC. 11–14, 27, 185, 187
P.Id. Particle Identification. 91
P.O.C.A. Point Of Closest Approach. 37, 38, 66, 73
POCKET DAQ Miniaturized DAQ system used for beam tests.. 119
PXD PXD: The PiXel Detector. 5, 16, 18–22, 24, 27–29, 36, 39, 43, 48, 55, 56, 60, 71, 80, 113–118, 120, 121, 123, 124, 127–130, 132–134, 136, 139, 144, 171, 177, 183, 185, 186, 188–190, III, IV, VII, VIII

QCS Final beam focussing magnets before the IP.. 26
QED Quantum Electro Dynamics. 27, 28
QI Quality Indicator. 32, 71, 72, 74–77, 131–133, 139, 143, 150, 151, 153, 154, VIII

RBB Radiative Bhabha scattering. 27, 28
ROF ReadOut Frame. 27, 55, 80
ROI Region Of Interest. 30, 55, 71, 113, 115, 120, 121, 188, 190, VIII

SecMap SecMap: SectorMap. 4, 56–65, 67–69, 85, 86, 90, 92, 96, 98, 101, 106, 111, 122, 123, 138, 141, 142, 144, 147, 148, 153–157, 172–174, 180, 189, IV, VIII
Sector Sector: Subdivision of a VXD-sensor which is able to store Filter cuts.. 57–67, 77, 111, 141, 143, 145, 147, 148, 155, 156, 167, 168, 172–174, 179, 180, 187–189, 192
Segment Segment: Two compatible hits form a Segment.. 57, 64–66, 68, 70, 124, 129–133, 136–139, 143, 145, 148, 149, 153, 165, 166, 180, 184, 187, 189
SegNet SegNet: Segment Network. 143–147, 149–151, 167, 168, 189

SegNetModule SegNetModule: SegmentNetworkProducerModule. 143, 148, 154, 166, 188
SMHEP SMHEP: Standard Model of HEP. 7–11, 33, 177, 187, 188
SpacePoint SpacePoint: Global position of a hit in the VXD.. 5, 21, 43, 45–51, 55, 56, 58–61, 64, 66, 70, 72, 80–84, 88, 96, 97, 99–101, 110, 111, 142–144, 147, 148, 153–155, 157, 167, 168, 172, 173, 179–181, 185, 190
SPTC SPTC: The Space Point Track Candidate.. 143, 149, 150, 152, 153, 190
StoreArray A storage container used for exchanging data between modules in basf2.. 149, 167
SuperKEKB SuperKEKB: second generation B factory in KEK. 3, 11–15, 17, 25–27, 35, 53, 54, 77, 157, 158, 177, 185, 187–189, VII, VIII
SVD SVD: The Silicon Vertex Detector. 5, 15–25, 28–30, 33, 36, 37, 39, 43–46, 48, 53, 55, 56, 60, 70, 71, 80, 82–84, 113, 114, 116–124, 127–135, 137, 139, 144, 151–153, 157, 166, 171, 172, 174, 177, 178, 183–186, 188–190, 199, III, IV, VII, VIII

TC Track Candidate. 4, 5, 32, 48, 55–57, 63, 64, 67, 68, 70–72, 74–77, 80–82, 90, 91, 99–103, 106–110, 120–123, 125, 126, 128, 130–140, 143, 144, 150–154, 156, 166, 173, 181–184, 190, IV, VIII
TCC Track Candidate Collector of the VXDTF. 90, 99, 107, 181, 182
Telescope Sensor phalanx optimized for beam tests with extremely high sensor resolution.. 114, 116, 118, 119, 122–124, 127–129, 134, 139, 186, 192
TF Track Finder. 4, 5, 21, 28, 29, 35, 36, 43, 48, 51, 54, 55, 60, 62, 80, 81, 84, 92, 93, 97, 98, 101, 111, 121, 124–128, 134, 139, 142–144, 151, 152, 154, 155, 157, 158, 183, 187, 188
TLU Trigger Logic Unit. 115, 119
Tree Directed graph wihtout loops. 58, 59, 61, 68, 99, VIII
TTD Trigger Timing Distribution. 120

VXD VXD: VerteX Detector. 3, 5, 15–18, 20, 21, 25–30, 32, 35, 38–44, 48, 51, 54, 55, 57–62, 72, 75, 77, 79, 113–116, 119, 120, 123–125, 127, 131–133, 135–137, 139, 157, 158, 177, 183, 184, 187–190, 193, 194, III, V, VII
VxdID VxdID: Vertex detector IDentifier. 118, 122, 129, 186
VXDTF The VXD Track Finder package. 3–5, 21, 33, 35, 38, 55–57, 59, 61, 62, 64–67, 69–82, 84, 85, 87, 90–93, 98, 99, 101–103, 108–111, 113, 121–127, 129–141, 144, 147, 148, 151–158, 167, 171–174, 179, 180, 182–189, 194, III–V, VII, VIII
VXDTF 2 The refactored version of the VXD Track Finder package. 5, 78, 141–144, 149–151, 154–158, 165, 173, 174, 184, 190, V, VIII

X_0 X_0: Radiation length. 20, 21, 24

BIBLIOGRAPHY

[1] Jakob Lettenbichler. Pattern recognition in the Silicon Vertex Detector of the Belle II experiment. Master's thesis, University of Vienna, May 2012.

[2] A. J. Bevan et al. *The Physics of the B Factories*, volume C74. 2014.

[3] P.F. Harrison and H.R. Quinn, Editors. *The BABAR Physics Book*. SLAC Report 504,the BaBar Collaboration , October 1998.

[4] Z. Doležal and S. Uno (editors). *Belle II Technical Design Report*. Belle II collaboration, KEK, Oct, 2010. http://arXiv.org/abs/1011.0352v1.

[5] A. Abashian *et al.* The Belle Detector. *Nuclear Instruments and Methods in Physics Research A 479 (2002) 117*, 2002.

[6] D. Boutigny et al. *BaBar technical design report*, 1995.

[7] Robin Glattauer. *Measurement of the decay $B \to Dl\nu_l$ in fully reconstructed events and determination of the CKM element $|V_{cb}|$*. PhD thesis, Technical University of Vienna, 2016.

[8] online. http://www.en.wikipedia.org, 2016.

[9] A. D. Sakharov. Violation of CP Invariance, c Asymmetry, and Baryon Asymmetry of the Universe. *Pisma Zh. Eksp. Teor. Fiz.*, 5:32–35, 1967. [Usp. Fiz. Nauk161,61(1991)].

[10] J.H. Christenson, J.W. Cronin, V.L. Fitch, and R. Turlay. Evidence for the 2 pi Decay of the k(2)0 Meson. *Phys.Rev.Lett.*, 13:138–140, 1964.

[11] M. Kobayashi and T. Maskawa. CP Violation In The Renormalizable Theory Of Weak Interaction. *Prog. Theor. Phys. 49, 652 (1973)*, 1973.

[12] N. Cabibbo. Unitary Symmetry And Leptonic Decays. *Phys. Rev. Lett. 10, 531 (1963)*, 1963.

[13] J. J. Aubert et al. Experimental Observation of a Heavy Particle J. *Phys. Rev. Lett.*, 33:1404–1406, 1974.

[14] SW Herb, DC Hom, LM Lederman, JC Sens, HD Snyder, JK Yoh, JA Appel, BC Brown, CN Brown, Walter R Innes, et al. Observation of a dimuon resonance at 9.5 gev in 400-gev proton-nucleus collisions. *Physical Review Letters*, 39(5):252, 1977.

[15] G. Taylor. The Belle silicon vertex detector: present performance and upgrade plans. *Nuclear Instruments and Methods in Physics Research A 501 (2003) 22–31*, 2003.

[16] Hans-Günther Moser. The Belle II DEPFET pixel detector. *Nucl. Instrum. Meth.*, A831:85–87, 2016.

[17] Gagan B. Mohanty. Belle II Silicon Vertex Detector. *Nucl. Instrum. Meth.*, A831:80–84, 2016.

[18] Andreas Moll. *Comprehensive study of the background for the Pixel Vertex Detector at Belle II*. PhD thesis, Ludwig Maximilian Universität München, 2015.

[19] Martin Ritter. *Measurement of the Branching Fraction and Time Dependent* CP *Asymmetry in* $B^0 \to D^{*-}D^{*+}K^0_S$ *Decays at the Belle Experiment*. PhD thesis, Ludwig Maximilian Universität München, 2013.

[20] DorisYangsoo Kim. The software library of the Belle II experiment. *Nucl. Part. Phys. Proc.*, 273-275:957–962, 2016.

[21] Immanuel Gfall. Mechanical Design of the Belle II Silicon Vertex Detector. Master's thesis, University of Vienna, 2012.

[22] K. A. Olive et al. Review of Particle Physics. *Chin. Phys.*, C38:090001, 2014.

[23] Fabjan, C. W., Schopper, H. (editors). *Landolt-Börnstein New Series, Group I: Detectors for Particles and Radiation. Part 1: Principles and Methods*, volume Volume 21B1. Springer-Verlag Berlin Heidelberg, 2011.

[24] Frühwirth, Regler, Bock, Grote, Notz. *Data Analysis Techniques for High-Energy Physics*. Cambridge unversity press, Second edition, 2000.

[25] Strandlie, Are and Frühwirth, Rudolf. Track and vertex reconstruction: From classical to adaptive methods. *Rev.Mod.Phys.*, 82:1419–1458, 2010.

[26] Michael Schnell. *Development of an FPGA-based Data Reduction System for the Belle II DEPFET Pixel Detector*. PhD thesis, Rheinischen Friedrich-Wilhelms-Universität Bonn, May 2015.

[27] A. A. Glazov, I. V. Kisel, E. V. Konotopskaya, and G. A. Ososkov. Track filter on the basis of a cellular automaton. 1991.

[28] R. Frühwirth. Application of Kalman filtering to track and vertex fitting. *Nucl.Instrum.Meth.*, A262:444–450, 1987.

[29] R. Mankel. A Concurrent track evolution algorithm for pattern recognition in the HERA-B main tracking system. *Nucl. Instrum. Meth.*, A395:169–184, 1997.

[30] Y. Sato et al. Measurement of the branching ratio of $\bar{B}^0 \to D^{*+}\tau^-\bar{\nu}_\tau$ relative to $\bar{B}^0 \to D^{*+}\ell^-\bar{\nu}_\ell$ decays with a semileptonic tagging method. 2016.

[31] Robin Glattauer. *Measurement of the decay $B \to Dl\nu_l$ in fully reconstructed events and determination of the CKM element $|V_{cb}|$*. PhD thesis, Technical University of Vienna, 2016.

[32] Wolfgang Waltenberger. RAVE: A detector-independent toolkit to reconstruct vertices. *IEEE Trans. Nucl. Sci.*, 58:434–444, 2011.

[33] Johannes Rauch and Tobias Schlüter. GENFIT — a Generic Track-Fitting Toolkit. *J. Phys. Conf. Ser.*, 608(1):012042, 2015.

[34] Christian Pulvermacher. dE/dx particle identification and pixel detector data reduction for the Belle II experiment. Master's thesis, KIT, 2012.

[35] I. Abt, I. Kisel, S. Masciocchi, and D. Emelyanov. CATS: A cellular automaton for tracking in silicon for the HERA-B vertex detector. *Nucl.Instrum.Meth.*, A489:389–405, 2002.

[36] Thomas Madlener. Machine-learning assisted track finding in the Silicon Vertex Detector of the Belle II experiment. Master's thesis, Technical University of Vienna, 2015.

[37] Robin Glattauer, Rudolf Frühwirth, Jakob Lettenbichler, and Winfried Mitaroff. Forward Tracking in the ILD Detector. 2012. Proc. 13th Int. Linear Collider Workshop (LCWS '11), Granada, Spain, 26-30 Sept. 2011.

[38] Robin Glattauer. rack Reconstruction in the Forward Region of the Detector ILD at the Electron-Positron Linear Collider ILC. Master's thesis, Technical University of Vienna, 2012.

[39] Veikko Karimaki. Effective circle fitting for particle trajectories. *Nucl. Instrum. Meth.*, A305:187–191, 1991.

[40] R. Frühwirth, A. Strandlie, and W. Waltenberger. Helix fitting by an extended Riemann fit. *Nucl. Instrum. Meth.*, A490(1-2):366–378, 2002.

[41] R. Frühwirth. Selection of optimal subsets of tracks with a feedback neural network. *Comput.Phys.Commun.*, 78:23–28, 1993.

[42] Y. Shrivastava, S. Dasgupta, and S. M. Reddy. Guaranteed convergence in a class of hopfield networks. *IEEE Transactions on Neural Networks*, 3(6):951–961, Nov 1992.

[43] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[44] S. Hashimoto, M. Hazumi, J. Haba, J. W. Flanagan, Y. Ohnishi, K. Abe, K. Abe, T. Abe, I. Adachi, T. Agoh, et al. Letter of intent for KEK Super B Factory. 2004.

[45] T. Bilka et al. Demonstrator of the Belle II online tracking and pixel data reduction on the High Level Trigger system. *IEEE Trans. Nucl. Sci.*, 62(3):1155–1161, 2015.

[46] A. Bulgheroni. First test beam results from the EUDET pixel telescope. In *Proceedings, 2007 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC 2007): Honolulu, Hawaii, October 28-November 3, 2007*, volume 3, pages 1878–1884, 2007.

[47] Tadeáš Bilka. Alignment of the Belle II Vertex Detector. *PoS*, Vertex2014:048, 2015.

[48] A. Strandlie and R. Frühwirth. Adaptive multitrack fitting. *Comput. Phys. Commun.*, 133:34–42, 2000.

[49] Moritz Nadler and Rudolf Frühwirth. Robust track fitting in the Belle II inner tracking detector. *J. Phys. Conf. Ser.*, 396:022037, 2012.

[50] Ryosuke Itoh, Soohyung Lee, N. Katayama, S. Mineo, A. Moll, T. Kuhr, and M. Heck. Implementation of parallel processing in the basf2 framework for Belle II. *J. Phys. Conf. Ser.*, 396:022026, 2012.

[51] R. Brun and F. Rademakers. ROOT: An object oriented data analysis framework. *Nucl. Instrum. Meth.*, A389:81–86, 1997.

ACKNOWLEDGEMENTS

I worked on this project for several years, but I could not have succeeded without the help of my family, my friends, my colleagues and my supervisors.

First of all I personally want to thank *Rudolf Frühwirth*, my supervisor, for all his patience and his guidance. He helped me to master every pitfall on the way and allowed me to improve myself with the exact right amount of support I needed for it. This project would simply not exist if he wouldn't have started it all.

I also want to thank the director of our institute *Jochen Schieck* and its deputy director and group leader of the SVD *Christoph Schwanda* who both have supported this project and helped me out whenever I needed it. Together with the relentless support of the *Austrian Science Fund (FWF)* and their financing of project P24182-N16, they helped to make this project possible as well.

For their outstanding support I also want to thank my colleagues in the office, *Robin Glattauer*, *Felicitas Breibeck*, *Moritz Nadler* and the students I helped to supervise: *Thomas Madlener*, *Thomas Fabian* and *Stefan Ferstl*.

Additionally I would like to express my thanks to my former colleagues of the Belle II collaboration, especially (in alphabetical, but incomplete, order): *Tadeas Bilka*, *Giulia Casarosa*, *Oliver Frost*, *Thomas Hauth*, *Martin Heck*, *Christian Kiesling*, *Peter Kodys*, *Thomas Kuhr*, *Peter Kvasnicka*, *Andreas Moll*, *Eugenio Paoloni*, *Christian Pulvermacher*, *Martin Ritter*, *Tobias Schlüter* and *Manfred Valentan*.

Next I thank *Christian Thomay* for his help on proofreading my thesis and his encouraging support during my whole time as a student of physics.

I also want to thank all my friends and my family, especially my father *Anton Lettenbichler*, who is also a livelong friend of mine.

Last but not at least I want to thank my girlfriend *Carola Blazsovsky* for her patience during the stressful times we walked together.

Jakob Lettenbichler

Education

2012 – 2016 **PhD thesis**, *Institute for high energy physics*, Vienna.
2012 – 2016 **Doctoral study of Mathematics**, *Technical University of Vienna*.
2010 – 2012 **Diploma thesis**, *Institute for high energy physics*, Vienna.
2002 – 2012 **Diploma study of Physics**, *University of Vienna*.
1994 – 2002 **Matura**, *Bundesrealgymnasium Schloss Wagrain*, Vöcklabruck.
1990 – 1994 **Volksschule**, *Vöcklamarkt*.

PhD thesis

title *Silicon tracking approaches for the Belle II experiment*
supervisor Rudolf Frühwirth
description Finding tracks in realtime of low energy particle tracks in a detector of a electron-positron collider can be a challenging task. The thesis describes a fully working implementation of such a Track Finder to be used in the Silicon vertex detector of the Belle II experiment.

Diploma thesis

title *Pattern Recognition at the Silicon Vertex Detector of the Belle II experiment*
supervisor Rudolf Frühwirth
description Low momentum tracking with a small number of hits per track is challenging. The Thesis describes a working draft of a track finder capable of reconstructing low momentum tracks.

Summer schools and workshops

CSC 2013 CERN Summer School 2013 - Base Technologies (Many core software design, Networking), Data Technologies (storage, cryptography, security)& Physics computing (Data and multivariate analysis
Workshop 2013 on Parallel Programming Technologies at the IT-Center in Hagenberg
tCSC 2014 thematic CERN Summer School 2014 - Programming for Concurrency (modern C++, parallelism and thread safety), Data oriented design (vectorization), Memory Programming (memory awareness), Efficient Computing (CPU architecture, performance monitoring and compilers), Acceleration (programing for accelerators)

Claretinergasse 3/3/5 – 1220 Wien
+43 (0)6508520509 • jkl@jodoschka.com, jkl@geosono.com

Nonscientific experience

Vocational

Nov 2006–Jan 2010 **part time shop assistant**, *Saturn Electro-Handelsges.m.b.H Wien Gerngross*, Vienna.
Consulting and selling of TV's, beamers and satellite recievers

Repeatedly in Dec 2004–Feb 2012 **part time shop assistant**, *Ski Sport Salvista*, Itter.
Consulting, preparing and selling skis, boots and snowboards.

Languages

German **first language**

English **fluent**

French **basic knowledge** *6 years of active practicing at school, idle since then*

Computer skills

Programming C, C++03, C++1x, Python & Octave (Matlab)

Markup LaTeX & HTML

Office Tools MS Word, Excel or similar

OS Windows, Linux

Multimedia Gnuplot, Gimp, Inkscape, Adobe Photoshop & Premiere

Interests

stories includes reading, watching movies, storytelling, writing (short and long stories, screenplays, ...) and movie making

maker and DIY tinkering with Arduino, Raspberry Pi and 3D printing. Designing my own furniture

sports Basketball, climbing, jogging, biking ...

food eating and cooking good meals

Claretinergasse 3/3/5 – 1220 Wien
+43 (0)6508520509 • jkl@jodoschka.com, jkl@geosono.com