Karlsruhe Institute of Technology

ETP-KA/2020-25

Trigger Studies for $e^+e^- \to \pi^+\pi^-\gamma$ and $e^+e^- \to \mu^+\mu^-$ and Implementation of a New DAQ Logging and Monitoring System at Belle II

Anselm Baur

Master's Thesis

30.10.2020

Institute of Experimental Particle Physics (ETP)

Advisor: Prof. Dr. Günter Quast
Coadvisor: Prof. Dr. Florian Bernlochner

Editing Time: 01.11.2019 – 30.10.2020

Karlsruhe Institute of Technology

ETP-KA/2020-25

Trigger-Studien für $e^+e^- \to \pi^+\pi^-\gamma$ und $e^+e^- \to \mu^+\mu^-$ und Implementierung eines neuen DAQ-Logging- und Monitoringsystems bei Belle II

Anselm Baur

Masterarbeit

30.10.2020

Institut für Experimentelle Teilchenphysik (ETP)

Referent:	Prof. Dr. Günter Quast
Korreferent:	Prof. Dr. Florian Bernlochner

Bearbeitungszeit: 01.11.2019 – 30.10.2020

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, 30.10.2020

. . . . . .

(**Anselm Baur**)

Contents

1. Introduction

In the 17th century, at the time of Isaac Newton, the father of classical mechanics, no one would have believed that mankind would one day build huge machines in order to solve the greatest riddles that nature has imposed on us. The purpose of these machines is to measure the properties of the smallest particles in the universe, particles with no observed spatial expansion.

One of the greatest mysteries of nature today is the fact that matter and antimatter were not completely annihilated after the Big Bang, although produced in equal amounts. The consequence is the baryon asymmetry and universe still exists. One mechanism found to explain the baryon asymmetry is CP violation. However, the Standard Model of particle physics provides no sufficient strength for the extent of the observed baryon asymmetry. This leads to physics beyond the Standard Model.

The Standard Model is far from being complete. Dark matter, for example, can be observed, but is still not considered in this model. The anomalous magnetic moment of the muon, one of the most precise measurements in physics, also gives a possible hint of physics beyond the Standard Model because of its significant deviation from the theoretically predicted value.

Since 2018, the particle accelerator SuperKEKB has been producing particles by collisions of electrons and positrons at the energy of the $\Upsilon(4S)$ resonance, which allows to create conditions as they existed in the early universe. The Belle II experiment is based on the huge Belle II particle detector located around the interaction point of SuperKEKB. Taking advantage of the clean environment in electron-positron colliders, high-precision measurements are performed to investigate the riddles of nature.

For high-precision measurements, all influences, however small, must be taken into account when evaluating the measurement data. Sophisticated analyses must therefore be performed to estimate the measurement uncertainties and determine possible correction factors. In this thesis two analyses were conducted to estimate efficiencies of the Belle II trigger system. The first one is particularly relevant for a measurement which is an important input for the calculation of the theoretical anomalous magnetic moment of the muon. The second is essential for the search for Dark Matter.

The Belle II experiment is expected to produce a huge data set of $50\,\mathrm{ab}^{-1}$ within the next years. It will have to be evaluated in years of analyses. In order to record this huge amount of data, it is essential that both the SuperKEKB accelerator and the Belle II detector

run smoothly throughout. To prepare the Belle II detector for this task in the coming years, a new unified and centralized logging and monitoring architecture was developed and implemented in this thesis.

2. Experimental Setup

This chapter provides a short overview of the Belle II detector and its subdetectors in Section 2.1. The data acquisition and slow control system is introduced in Section 2.2. In Section 2.3 the two Belle II trigger systems are described. The three phases of data taking and the dataset naming schema is introduced in Section 2.4.

2.1. Detector

The Belle II detector is located around the interaction point (IP) of the electron and positron beam of the *SuperKEKB particle accelerator*. The detector itself consists of several layers of subdetectors, as illustrated in the schematic cross section in Fig. 2.1.

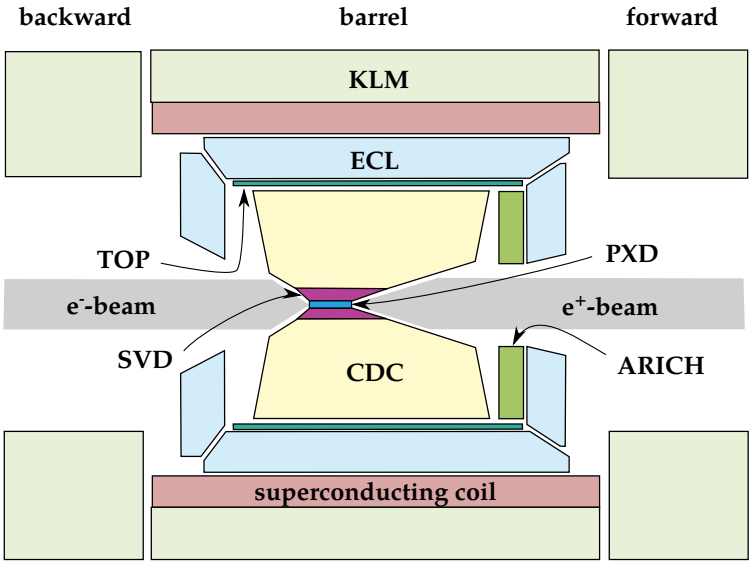


Fig. 2.1.: Overview of the Belle II detector and its subdetector components. The electrons and positrons are collide at the center of the detector (interaction point).

From inside out, the subdetectors are shortly introduced. The inner most DEPFET[1] based *pixel detector (PXD)* and the double-sided-strip *silicon vertex detector (SVD)* compose

[1] depleted p-channel field-effect transistor (DEPFET)

Belle II's *vertex detector (VXD)*, used for precise reconstruction of vertices of particle decays. This is followed by the *central drift chamber (CDC)*, which is the core instrument for tracking. The *time of propagation (TOP) counter* in the barrel region and the *aerogel ring imaging Cherenkov (ARICH)* detector in the forward end-cap are used for identification of charged particles (PID). It is tuned to discriminate pions and kaons by reconstructing their angles of Cherenkov radiation emission. The *electromagnetic calorimeter (ECL)* is used for energy measurements, mostly for electrons and photons. The ECL is surrounded by the *superconducting solenoid magnet* providing a homogeneous magnetic field in the inside of the detector. The K_L^0- *muon detector (KLM)* is the outer most detector used to identify muons and K_L^0 mesons. A detailed explanation of the detector components can be found in [1, 2].

2.2. Data Acquisition System

The Belle II *data acquisition (DAQ) system* consists of several subsystems. They are responsible for the trigger and timing distribution, the data links, the event building, and the high level event triggering [3]. Figure 2.2 shows a schematic illustration of the detector readout performed by the DAQ system. Due to considerations of readout, bandwidth, and latency, the PXD data readout is treated separately from the other subdetectors until the final event building and storage.

2.2.1. Detector Readout and Online Processing

The *subdetectors front-end readout system* sends via the *Belle2Link* interface the subdetector data selected by the Level 1 trigger to the *common pipeline platform for electronics readout (COPPER) modules*. *Readout PCs* aggregate the data from $3-5$ connected COPPER modules each and forward them to the *event builder 1*. Here, the data of the subdetectors are merged to an event. For further filtering the *High Level Trigger (HLT)* performs a full event reconstruction on each event, using the same software as for offline reconstruction. Within that reconstruction without PXD data, particle tracks are extrapolated into the PXD volume and so-called *regions-of-interest (ROIs)* are defined. The HLT sends the ROI information of the selected events to the PXD *online selection nodes (ONSEN)* and forwards these events to the *event builder 2*. Here, the PXD data is received via the PXD ONSEN and added to the events. Finally, the events are sent to the *online* storage system from where they are copied to the *offline* storage system for data processing and offline analysis. A detailed description of the readout and online selection can be found in [2, 3].

2.2.2. Slow Control System

The *DAQ slow control system* (`daq_slc`) is a network-based software system to control the detector and the DAQ system. It synchronizes all the subsystems using a network distributed memory protocol described further down. A relational unified database system based on *PostgreSQL* [4] is used to store and load configuration and logging information. The slow control system can be controlled via a graphical user interface (GUI). In addition, it also provides the status monitor of the detector operation. [5]

`daq_slc` is designed to manage mainly the following three tasks [6].

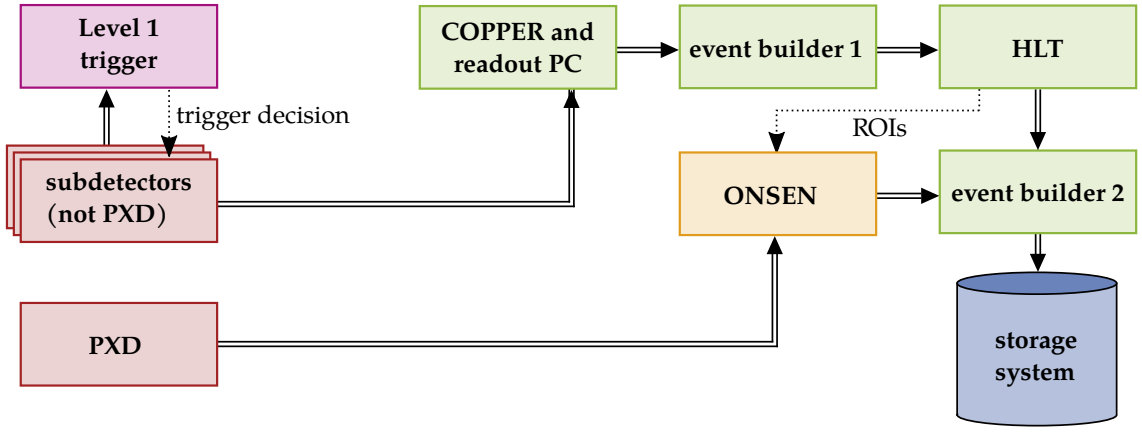


Fig. 2.2.: Schematic overview of the detector readout.

- *Run control*, configuring the COPPERs and the front-end system for the detector readout (except PXD) using a configuration database, as well as starting processes in the back-end, e.g. starting/stopping the full reconstruction on the HLT.
- *High voltage (HV) power supply control* to control the power supply for the subdetectors regarding to the status of SuperKEKB.
- *Monitoring* of sensors and SuperKEKB *EPICS* [7] processed variables (PVs) and *archiving* them with the *EPICS archiver appliances* [8].

The underlying *network shared memory (NSM)*[2] protocol is used by `daq_slc` to distribute control messages and to synchronize shared memory between hosts in the DAQ slow control network. Each host has to run a NSM daemon (`nsmd2`) to connect to a NSM network. An arbitrary number of so-called NSM nodes can be set up on a host. Each `daq_slc` application starts such a NSM node for run control and inter process communication (IPC). Fig. 2.3 shows schematically two hosts connected via a NSM network.

NSM provides two types of data distribution mechanisms between the NSM nodes. First, the *NSM distributed shared memory*: The data in the shared memory is packed in UDP messages and distributed to all `daq_slc` hosts in the NSM network sharing the memory. Second, the *NSM messages* sent via a TCP connection to corresponding nodes in the NSM network. The NSM messages consist of a text field for a *NSM request* and a buffer for parameters. The requests are defined execution commands for the remote node. For each NSM request, a callback function has to be implemented on the dedicated nodes receiving it.

The management of the DAQ system via parameter and status distribution is performed by the `daq_slc` *run control system*. All applications of that system form, via their NSM nodes, the *NSM run control network*. It configures the front-end readout system for the detector readout and is responsible for the synchronization and monitoring of the *run control states* of all subsystems (e.g. `READY`, `RUNNING`, `ERROR`, etc.). The run control system consists of a multi-level tree structure. It manages the entire data taking. Here, the control of PXD

[2] NSM is currently in version 2, often referred to as NSM2 in the literature. Here, NSM refers to NSM2.

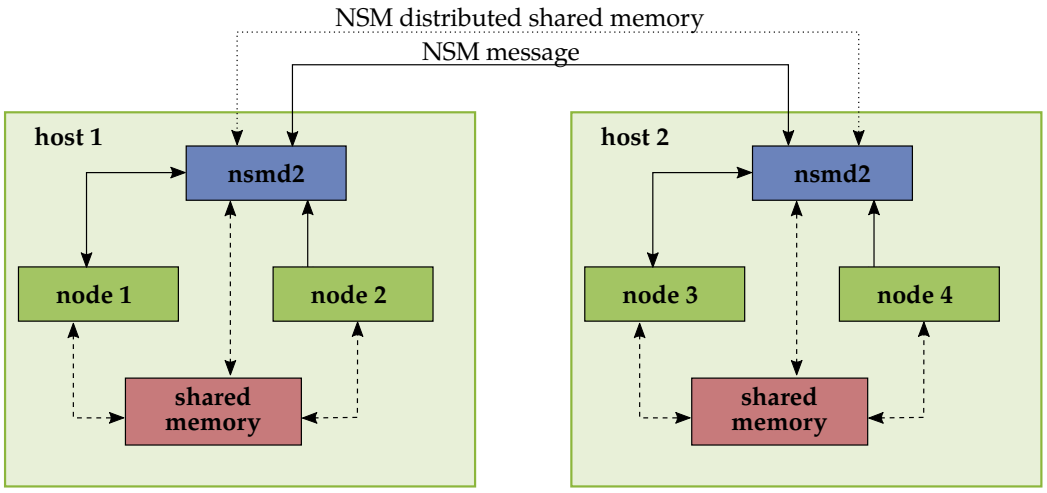


Fig. 2.3.: Schematic illustration of two hosts in the DAQ system running slow control processes. Each process creates its own NSM node. The NSM messages and NSM distributed shared memory are sent between the two NSM daemons (`nsmd2`). The nodes and the NSM daemons accessing their local shared memory via operating system functionalities.

is also included. For the detector operation, *run control NSM requests* are sent to the subsystems, e.g. to start and stop data taking. In addition, via the run control system, the COPPERs are configured, which in turn configure the front-end readout system of the subdetectors (PXD excluded).

To collect the log messages of the `daq_slc` processes in the run control network in a central point and forward them to a database, a `daq_slc` *log collector* process is running. It receives the messages via its NSM node connected to the run control NSM network. This application is discussed further in Chapter 6.

A more detailed documentation of `daq_slc` can be found in [5, 6]. For the documentation of NSM see [9].

2.3. Trigger System

In most bunch crossings within the interaction region no physically interesting hard scattering processes, but a predominant part of Bhabha processes ($e^+e^- \to e^+e^-$) do occur. The Bhabha process dominates the production in the e^+e^- collisions, owing to its large production cross section. Recording all available information of all possible processes would "eat up" a lot of technical resources.

At the Belle II experiment, two dedicated trigger systems are used to define the events and to reduce the data rate to a technically manageable level. First, the hardware-based *Level 1 trigger*, which defines the events and performs a loose selection. Second, the software-based *High Level Trigger (HLT)* is used for further filtering. Both trigger systems are investigated for dedicated efficiency analyses in this thesis (see Chapter 4 and Chapter 5).

In the following two sections, the two systems are introduced and their working principles are explained.

2.3.1. Level 1 Trigger

The hardware-based Level 1 trigger system is composed of several subsystems as shown in Fig. 2.4. From the front-end readout system the subdetector data is continuously streamed to the subtrigger systems in nearly real-time. There are four *subtrigger systems* based on the CDC, ECL, TOP, KLM data. In the *general reconstruction logic (GRL)*, the data of the subtrigger systems is reconstructed on a low level, e.g. it matches tracks in the CDC and clusters in the ECL. In the current Level 1 trigger setup, the CDC and ECL subtrigger systems are mainly used. The *global decision logic (GDL)* is the final logic subsystem which calculates the final Level 1 trigger decision. If the Level 1 trigger decision is positive, the trigger signal is distributed via the *trigger and timing distribution (TTD) system* back to the front-end readout system of the subdetectors. Triggered by that signal, the detector readout process for the event is started as described in Section 2.2.1. A more detailed figure of the Level 1 trigger setup is shown in Appendix A.1. [10, 11]

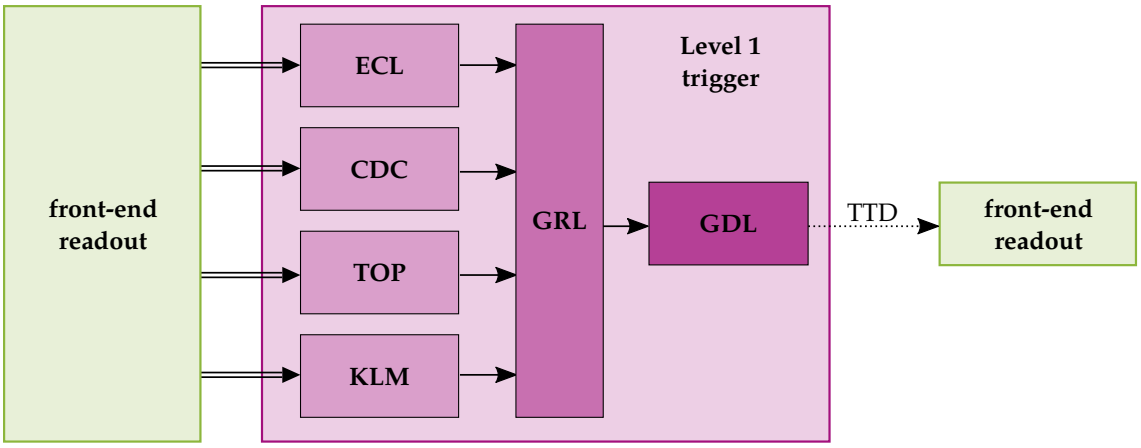


Fig. 2.4.: The Level 1 trigger system with the four subtrigger systems. Currently, the CDC and ECL subtrigger systems are mainly used.

The CDC subtrigger system aims to find hit patterns that resemble charged tracks and provides information about their momentum, position, charge, etc. As a result of the decreasing CDC efficiency in the forward and backward end-cap regions, the CDC subtrigger system is not sensitive for charged tracks in the very forward or backward directions.

The ECL subtrigger system provides fast trigger signals based on the measurement of deposited energy in the Thallium-doped CsI crystals, using two primary and complimentary trigger schemes.

- The *total energy trigger* is sensitive to physics events with a high energy deposit in the ECL.
- An *isolated cluster counting* is sensitive to events with low-energy clusters like physics events of multi-hadronic processes and/or minimum ionizing particles (MIPs).

In addition, the ECL subtrigger system is designed to select Bhabha events based on the polar and azimuthal angular alignment and the energy deposited in crystal clusters as described in Section 4.2.

The Level 1 trigger system is designed to provide a trigger signal at a fixed latency of 4.4 µs after a collision [10]. During this time, the detector data is buffered on the front-end readout system. The DAQ system is designed to operate with a Level 1 trigger rate up to 30 kHz. A Level 1 trigger rate of approx. 20 kHz is expected as stated in [11], owing to physically interesting processes.

2.3.2. High Level Trigger

In the following a distinction between the HLT as a software trigger and the HLT as a server farm is necessary. The HLT server farm consists of 10 units[3]. Each unit consists of a `hltctl`, `hltin`, `hltout`, and $16 - 20$ worker (`hltwkxy`) hosts. All hosts except the `hltctl` host are network booted. They boot via `nfs` and `PXEBoot` from the `hltctl` host. At one HLT unit an additional `roipc` host exists. A schematic overview of the HLT server farm is shown in Fig. 2.5.

It has to be mentioned that the event builder stages in the data readout chain, shown in Fig. 2.2, consists of two separate parts. This is shown in Fig. 2.6. The data transmitting and the data receiving parts are running on different hosts. At the HLT server farm, the events are passed through several stages as shown in Fig. 2.7:

- The events coming from the detector are sent by the event builder 1 TX to the event builder 1 RX, which runs on the `hltin` hosts.
- From the event builder 1 RX, the events are forwarded to the distributor process also running on the `hltin` hosts.
- The distributor process sends the events to the `hltwkxy` worker hosts.
- On the worker hosts a full reconstruction of the events is processed in thousands of parallel processes[4] on the whole HLT server farm. A software trigger result for each event is calculated. The same software framework used for offline analysis is used for the online reconstruction on the HLT (see Chapter 4 and Chapter 5).
- After the full event reconstruction on the workers the events are sent to the final collector on the `hltout` hosts.
- Here, the calculated ROIs are sent to the `roipc` which "triggers" the ONSEN to read out the PXD ROIs and send the PXD ROI data to the event builder 2 TX.
- The events are meanwhile forwarded to the event builder 2 TX where the missing PXD ROI data is added.
- Finally, the events are sent to the event builder 2 RX and the data is written on the online `storagexy` node which belongs to the respective HLT unit.

[3] Current setup in October 2020.

[4] All events are independent from each other. Therefore, they can be processed in individual parallel processes.

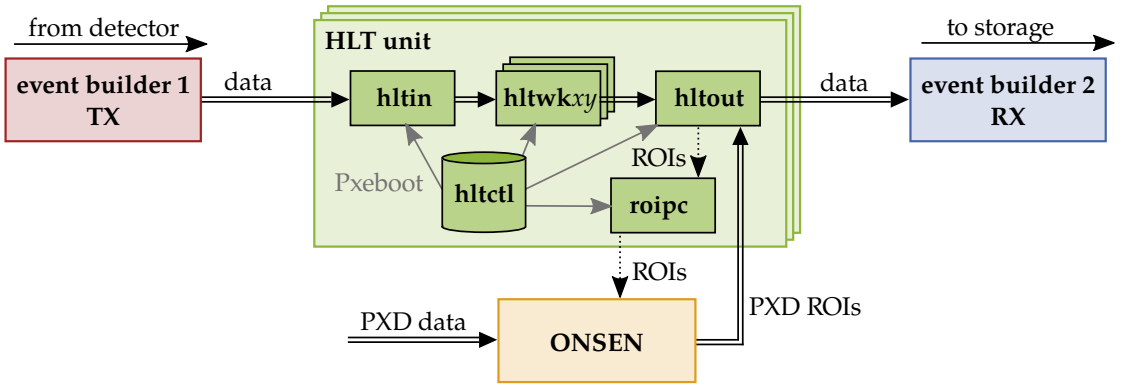


Fig. 2.5.: Technical setup of the HLT server farm. From the event builder 2 TX the events are sent to the `hltin` hosts of the HLT units. From the `hltout` hosts, the selected events are sent to the event builder 2 RX.

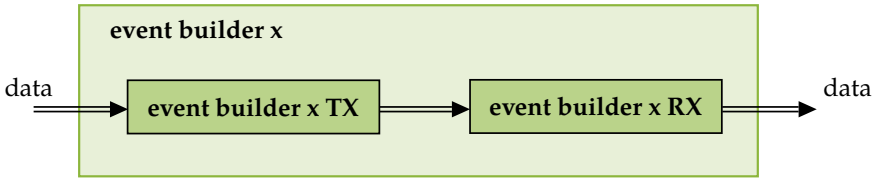


Fig. 2.6.: The two event builder stages in the detector readout chain consists of two components each. Data transmitting and data receiving parts are running on different hosts. [12]

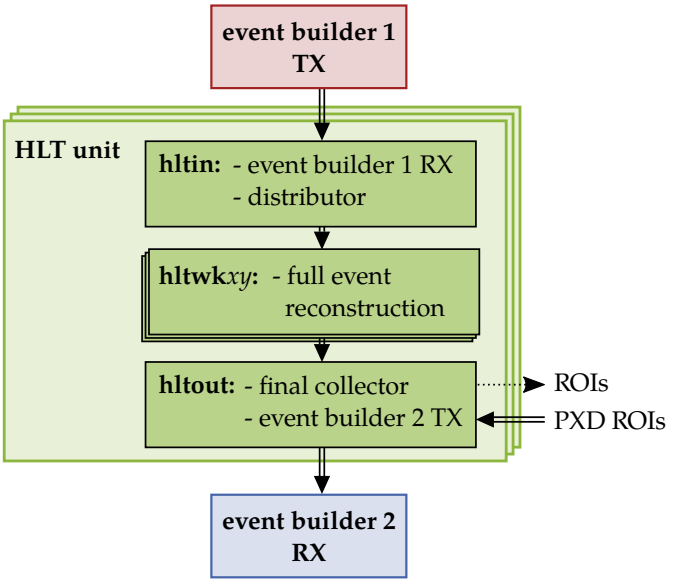


Fig. 2.7.: Overview of the processes relevant for the online processing, running on the HLT server farm. Events passing the processes from top to bottom.

The HLT output rate is expected to be 10 kHz and the mean process time for each event is on average restricted to 320 ms for 20 kHz input rate. This is reduced to 213 ms for 30 kHz input rate.

A more detailed description of the HLT can be found in [13, 14, 15].

2.4. Data Production

The timeline of the Belle II data taking up to now is separated in three phases.

- *Phase I*, performed in 2016 during commissioning of the SuperKEKB for beam environment characterization, without the Belle II detector (detector was moved out from the beamline).
- *Phase II*, during 2018, first collisions without full VXD, but *BEAST*[5] inner detector.
- *Phase III*, since 2019, the Belle II experiment is running with the full detector (only the 2nd layer of PXD is missing).

The recorded datasets of phase III are named after a specific naming schema e.g. experiment 10 proc 11 run 5610, or experiment 12 bucket 10 run 2800. It contains the experiment number, the major processing number or the bucket number, and the run number.

Each year is separated in three run periods: spring (a), summer (b), and autumn (c). Run period 2020a means the spring run period of the year 2020. The experiment number is related to a run period (or run periods) and the data taking mode as listed in [16]. Every time the data taking in the DAQ system is started, a new run begins and the run number is incremented by 1. During an experiment, a data amount of (valid) runs, sufficient for a calibration of this dataset, is collected and processed. This processed dataset is called bucket. A major processing (proc) is done with a much larger dataset, e.g., in the case of proc 11 the datasets of experiment $7 - 10$.

The data production and processing is described in [17] in more detail.

[5] Beam Exorcism for a Stable ExperimenT for the measurement of the beam background.

3. Statistical Fundamentals

This chapter describes the statistical fundamentals for calculating trigger efficiencies and their uncertainties as well as the scaling of data samples. A short formal definition of trigger efficiencies is given in Section 3.1, followed by the frequentist approach to calculate these in Section 3.2. The calculation of uncertainties is explained in Section 3.3. Furthermore, the `TEfficiency` class of the *ROOT data analysis framework* [18] is introduced in Section 3.4. How to scale simulated datasets for comparison purposes is explained in Section 3.5.

3.1. Trigger Efficiencies

The trigger system is based on the information provided by the detector or by software algorithms which processed the detector data. Due to uncertainties and inefficiencies of the detector components themselves like e.g. the energy resolution in the ECL or the tracking efficiencies in the tracking software, these uncertainties and inefficiencies are propagated to the trigger system. Finally, this leads to an inefficiency in the trigger operation.

Fig. 3.1 shows schematically the efficiency of a trigger. All events produced by the particle accelerator, through collisions of electrons and positrons, constitute the total sample space S. Let B be a set of physical interesting processes with $B \subseteq S$, which the trigger has to select. The set of events actually selected by the trigger is T with $T \subseteq S$. Due to the aforementioned inefficiencies and uncertainties, the set T will contain only a subset A with $A \subseteq B$ and $A = B \cap T$ of the events of interest and additionally an amount of background events $T \setminus B$.

For precise physical measurements it is crucial to know the accuracy of the trigger with which the events of the set B are selected. A perfect operating trigger would select the whole set B so that $A = B$ is valid.

The trigger efficiency ε is the conditional probability $P(\varepsilon \mid \boldsymbol{\theta})$ of an event passing the selection, with a number of conditions attached to its properties $\boldsymbol{\theta}$ (e.g. photon energy, track momentum, number of tracks, etc.). With the total number of events n in the set B which the trigger should select and the number of events actually selected by the trigger k in set A, the trigger efficiency $\hat{\varepsilon}$ can be estimated by

$$\hat{\varepsilon} = \frac{k}{n}. \tag{3.1}$$

This is described in Section 3.2 in more detail.

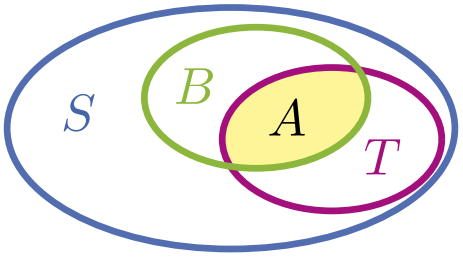


Fig. 3.1.: Schematic illustration of the trigger selection T. Set B contains the events of interest of the sample space S. The trigger selects a set T, which contains a subset A of B and additional background.

There are two main methods for determining n and k. In both methods, the trigger decision has to be stored along with the event data. One method is to store only the trigger decision, but record all events. In the subsequent analysis the trigger efficiency can be calculated by selecting the interesting subset B and comparing it with the trigger decision. The other method is to use orthogonal triggers with uncorrelated selection criteria, e.g., a trigger using the energy in the ECL and a trigger using the number of tracks in the CDC. The orthogonal trigger must also select the subset B (or at least a uniform subset of it). In the subsequent offline analysis the trigger efficiency can be calculated by comparing the set of the orthogonal triggers with the trigger decision of the trigger of interest.

There are two possible approaches to describe the trigger efficiency with statistical methods. The first is the frequentist approach, which determines the probability based on the outcome of a series of frequently occurring events, and the other is the Bayesian approach, which describes the probability that a previously known probability distribution is true in view of the observed data. A more detailed discourse can be found in [19].

In this thesis only the frequentist approach is described and used as recommended by the PDG [20]. The uncertainties of the calculated trigger efficiencies are estimated with the Clopper-Pearson method described in Section 3.3. Both the trigger efficiency and its uncertainties are calculated in this work using the ROOT class `TEfficiency` as described in Section 3.4.

3.2. Estimating the Trigger Efficiency - Frequentist Approach

A *Bernoulli experiment* is an experiment with two possible outcomes. With respect to triggers, an event can be selected by the trigger (*success*) or an event can be rejected by the trigger (*failure*). The frequentist approach interprets probability as the frequency of the outcome of a series of repeated experiments. A trigger efficiency is the frequency of successes of Bernoulli experiments when applying the *Bernoulli theorem*[1]. It contains the probability $P(k)$ to obtain k successes from n trials, where the probability of success ε can be described by the binomial distribution

$$\mathrm{Bi}(k \mid \varepsilon, n) = P(k) = \binom{n}{k} \varepsilon^k (1-\varepsilon)^{n-k}. \tag{3.2}$$

[1] In case of an infinite number of trials of Bernoulli experiments, the probability of success is given by the success frequency.

With respect to Fig. 3.1 k is the number of events of interest selected by the trigger (set A) and n is the total number of events of interest (set B). The observation of an event x selected by the trigger ($x \in A$) is given by the probability ε. A rejection of the event of interest x ($x \in \overline{B \cap A}$) is given by the probability $1 - \varepsilon$.

The random variable k can be measured by counting the number of events of interest selected by the trigger (set A) and the probability ε is the unknown parameter. To solve this problem, the trigger efficiency ε can be estimated using the maximum likelihood method (ML). This method returns the value for a parameter (or parameter list) for which the observed random variable(s) is (are) most likely. The likelihood function $L(\mathbf{k} \mid \varepsilon)$ for the unknown parameter ε is defined as

$$L(\mathbf{k} \mid \varepsilon) = \prod_{i=1}^{m} P(k_i; \varepsilon), \tag{3.3}$$

for a m times repeated measurement for the random variable k. The highest probability of observing the measured successes is given by the ML parameter $\hat{\varepsilon}$, which maximizes the probability function $L(\mathbf{k} \mid \varepsilon)$ by

$$L(\mathbf{k} \mid \hat{\varepsilon}) = \max(L(\mathbf{k} \mid \varepsilon)). \tag{3.4}$$

The maximum of the likelihood function for the parameter ε can be found by deriving the likelihood function from the unknown parameter and requiring

$$\frac{\partial L(\mathbf{k} \mid \varepsilon)}{\partial \varepsilon} \stackrel{!}{=} 0. \tag{3.5}$$

Since the log function is monotonically increasing it is more convenient to define the log-likelihood $\log L(\mathbf{k} \mid \varepsilon)$. As a result the product in the likelihood function is converted into a series of sums and the maximum log likelihood can be calculated with

$$0 \stackrel{!}{=} \frac{\partial \log L(\mathbf{k} \mid \varepsilon)}{\partial \varepsilon} = \sum_{i=1}^{m} \log \binom{n}{k_i} \left[k_i \frac{1}{\varepsilon} - (n - k_i) \frac{1}{1 - \varepsilon} \right]. \tag{3.6}$$

The square bracket for each k_i disappears when the ML estimator $\hat{\varepsilon}$ is given by

$$\hat{\varepsilon} = \frac{k_i}{n}. \tag{3.7}$$

This is the proof that Equation 3.1 is valid using a single measurement of k composed of a series of Bernoulli experiments.

3.3. Clopper-Pearson Uncertainty Interval

The construction of *confidence intervals* for frequentist statistics is based on the method originally developed by Neyman [21]. It is used to express uncertainties of an estimated parameter. The so-called *confidence level* expresses the frequency with which the confidence interval of an estimated parameter of each set of repeated random experiments covers the unknown true value.

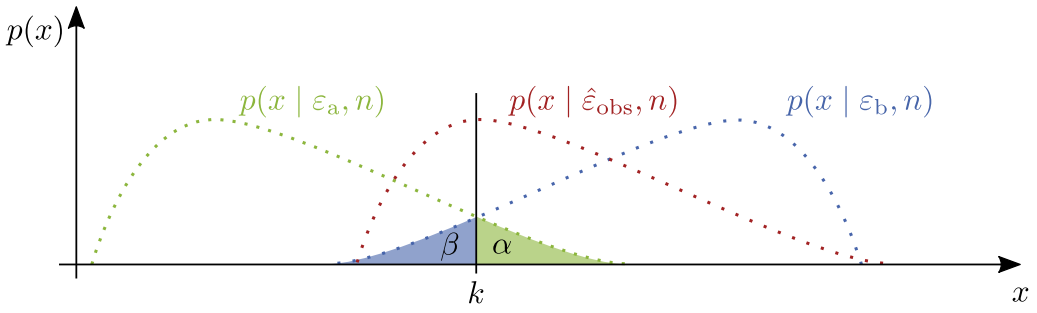


Fig. 3.2.: Schematic diagram for determining the Clopper-Pearson interval limits for a random distribution. Representation of the probability α that more events survive the selection than the number of observed k events for the success probability ε_a. As well as the probability β to observe less events than k for the success probability ε_b.

A method of calculating the confidence interval to determine an uncertainty of the estimated trigger efficiency $\hat{\varepsilon}_\mathrm{obs}$ is the construction of the Clopper-Pearson uncertainty interval for a binomial distribution (Eq. 3.2). The upper and lower bounds of the interval are determined on the basis of the number of observed k events selected by the trigger and on the confidence level C_l

$$C_\mathrm{l} = 1 - \alpha - \beta, \tag{3.8}$$

where α and β are the probabilities that a number x of selected events below or above the observed k is obtained. This is illustrated in Fig. 3.2 for ε_b as upper bound and ε_a as lower bound with $\varepsilon_\mathrm{a} < \hat{\varepsilon}_\mathrm{obs} < \varepsilon_\mathrm{b}$. The probability of obtaining less or more than the observed k selected events for a trigger efficiency ε_b or ε_a is considered using the discrete probability functions of the upper bound $p(x \mid \varepsilon_\mathrm{b}, k, n)$ and the lower bound $p(x \mid \varepsilon_\mathrm{a}, k, n)$.

To find the lower limit of the Clopper-Pearson interval, the corresponding discrete probability distribution of a certain ε_a must be found, which corresponds to the probability α and to the number of selected events x with $x > k$. The probability to observe x selected events is given by

$$P(x \geq k \mid \varepsilon_\mathrm{a}, k, n) = \frac{1 - C_\mathrm{l}}{2} = \alpha. \tag{3.9}$$

The probability of Equation 3.9 can be calculated as follows

$$\alpha = \sum_{i=k}^{\infty} \mathrm{Bi}(i \mid \varepsilon_\mathrm{a}, n) = 1 - \sum_{i=0}^{k-1} \mathrm{Bi}(i \mid \varepsilon_\mathrm{a}, n) = 1 - \mathrm{Be}(1 - \varepsilon_\mathrm{a}, n - k + 1, k), \tag{3.10}$$

using the relationship between the summation of binomial distributions and the beta distribution[2]

$$\sum_{i=0}^{k} \binom{n}{i} \varepsilon^i (1 - \varepsilon)^{n-i} = \mathrm{Be}(1 - \varepsilon, n - k, k + 1). \tag{3.11}$$

[2] $\mathrm{Be}(x \mid a, b) = \frac{1}{\mathrm{Be}(a,b)} \int_0^x u^{a-1}(1-u)^{b-1}\,\mathrm{d}u$

According to the ROOT `TEfficiency` documentation [20] the probability α that $x > k$ is observed for an efficiency ε_a is

$$\alpha = 1 - \frac{1}{\mathrm{Be}(a,b)} \int_0^{1-\varepsilon_a} t^{n-k}(1-t)^{k-1} dt \tag{3.12}$$

$$= 1 - \frac{1}{\mathrm{Be}(a,b)} \int_{\varepsilon_a}^{1} t^{k-1}(1-t)^{n-k} dt \tag{3.13}$$

$$= \frac{1}{\mathrm{Be}(a,b)} \int_0^{\varepsilon_a} t^{k-1}(1-t)^{n-k} dt \tag{3.14}$$

$$= I_{\varepsilon_a}(k, n-k+1). \tag{3.15}$$

Using the *inverse regularized incomplete beta function* $I^{-1}_{\varepsilon_\mathrm{a}}(k, n-k+1)$ the corresponding efficiency ε_a belonging to the confidence level C_l can be calculated numerically.

For the upper bound, the steps are the same as those already shown for the lower bound. The probability

$$\beta = \sum_{i=0}^{k} \mathrm{Bi}(i \mid \varepsilon_\mathrm{b}, n) = \mathrm{Be}(1-\varepsilon_\mathrm{b}, n-k, k+1) \tag{3.16}$$

is used and the incomplete beta function can be calculated from

$$\beta = \frac{1}{\mathrm{Be}(a,b)} \int_0^{1-\varepsilon_b} t^{n-k-1}(1-t)^{k} dt \tag{3.17}$$

$$= \frac{1}{\mathrm{Be}(a,b)} \int_{\varepsilon_b}^{1} t^{k}(1-t)^{n-k-1} dt \tag{3.18}$$

$$= 1 - \frac{1}{\mathrm{Be}(a,b)} \int_0^{\varepsilon_b} t^{k}(1-t)^{n-k-1} dt \tag{3.19}$$

$$1 - \beta = I_{\varepsilon_b}(k+1, n-k). \tag{3.20}$$

More detailed information about confidence intervals can be found in [22]. The Clopper-Pearson method is the default method for the frequentist approach used by the ROOT `TEfficiency` class, as described in the next section.

3.4. TEfficiency Class - ROOT

The `TEfficiency` class is provided by the ROOT data analysis framework [20]. It deals with the calculation of efficiencies and their uncertainties. Two statistical methods are available, the frequentistic method (standard) and the Bayesian method. Uncertainties can be estimated by selecting different methods of confidence intervals. The combination of several efficiencies is also possible. In this thesis the efficiencies and the Clopper-Pearson uncertainty intervals were calculated using `TEfficiency` with the frequentist method recommended by the PDG. The `TEfficiency` default confidence level of $1\,\sigma = 0.683$ was chosen.

`TEfficiency` allows the calculation of efficiencies depending on a (binned) variable by means of histograms. For this purpose two histograms are filled with the number of total events n and the number of selected events k. The efficiency in each bin is calculated by Equation 3.7.

3.5. Integrated Luminosity Scaling

In order to compare recorded data with MC simulated data during data analysis, the simulated dataset must be scaled to the integrated luminosity of the recorded dataset. Starting from the general relationship

$$N = \varepsilon \cdot \sigma_{\mathrm{p}} \cdot L_{\mathrm{int}}, \tag{3.21}$$

where N is the number of events, ε is the overall efficiency of the detector and corresponding triggers, σ_{p} is the production cross section, and L_{int} is the integrated luminosity. There are three methods to determine the scaling factor (weight) η, depending on the available information of the simulated data sample. The scaled number of events N' is obtained by

$$N' = \eta \cdot N = \varepsilon \cdot \sigma_{\mathrm{p}} \cdot \eta \cdot L_{\mathrm{int}}. \tag{3.22}$$

If the integrated luminosity of the simulated dataset $L_{\mathrm{int,MC}}$ is known, the scaling factor η can easily be calculated by

$$\eta = \frac{L_{\mathrm{int,data}}}{L_{\mathrm{int,MC}}}. \tag{3.23}$$

Sometimes, only a subset of a produced large simulated dataset is used. If the integrated luminosity of the used simulated dataset $L_{\mathrm{int,MC}}$ is unknown, but the integrated luminosity $L_{\mathrm{int,MC,total}}$ of the entire produced simulated dataset and its number of events $N_{\mathrm{MC,total}}$ is known and additionally the number of events in the used simulated dataset N_{MC} is known, then the scale factor can be calculated as follows

$$L_{\mathrm{int,data}} = \eta \cdot L_{\mathrm{int,MC}} = \eta \cdot \frac{N_{\mathrm{MC}}}{\sigma_{\mathrm{p,MC}}} \tag{3.24}$$

$$\eta = \frac{L_{\mathrm{int,data}} \cdot N_{\mathrm{MC,total}}}{L_{\mathrm{int,MC,total}} \cdot N_{\mathrm{MC}}}. \tag{3.25}$$

And finally, if the integrated luminosity of the used and entire produced simulated dataset is not known, but the number of events in the used simulated dataset N_{MC} and the production cross section $\sigma_{\mathrm{p,MC}}$ of the MC generator is known, then the scaling factor can be calculated by

$$\eta = \frac{L_{\mathrm{int,data}}}{L_{\mathrm{int,MC}}} = \frac{L_{\mathrm{int,data}}}{N_{\mathrm{MC}}}\sigma_{\mathrm{p}}. \tag{3.26}$$

4. Impact of the Level 1 Bhabha Veto to $e^+e^- \to \pi^+\pi^-(\gamma)\gamma_{ISR}$

One of the many measurements Belle II is designed for, is the precise measurement of the $e^+e^- \to \pi^+\pi^-(\gamma)$ cross section using the *initial state radiation (ISR) method*[1]. This measurement was not possible at the predecessor Belle experiment due to the low trigger efficiency with respect to $\pi^+\pi^-\gamma_{ISR}$ events caused by the "overly aggressive" 2D Bhabha veto. For Belle II, a new 3D Bhabha veto was designed for this purpose. In this chapter, the impact of the old 2D and new 3D Level 1 Bhabha vetoes are described with respect to $\pi^+\pi^-\gamma_{ISR}$ events.

Section 4.1 motivates this analysis with respect to the *anomalous magnetic moment of the muon* and Section 4.2 describes the conditions of the considered Level 1 triggers. In Section 4.3 the event selection for the analysis is explained. Finally, Section 4.4 and Section 4.5 present the results of the loss calculation for the Bhabha vetoes, which are summarized in Section 4.6. Comparisons with similar studies on previous datasets by [23] were performed.

4.1. The Belle II $e^+e^- \to \pi^+\pi^-(\gamma)$ Measurement

The theoretical calculations of the running of the QED fine structure constant $\alpha_{EM}(s)$ or the anomalous magnetic moment of the muon are limited due to the precision of the contribution of second order loop effects from *hadronic vacuum polarization*. Especially for the anomalous magnetic moment of the muon, the $e^+e^- \to \pi^+\pi^-$ process is the one that contributes most to the theoretical uncertainty. [24, 25]

4.1.1. Muon Anomalous Magnetic Moment

Loop corrections, as shown in Fig. 4.1, lead to a deviation of the gyromagnetic ratio g_μ of the muon from $g_\mu = 2$ which is parameterized as the anomalous magnetic moment a_μ of the muon with

$$a_\mu = \frac{g_\mu - 2}{2}. \tag{4.1}$$

[1] Using the ISR method, the total process is $e^+e^- \to \pi^+\pi^-(\gamma)\gamma_{ISR}$ but the ISR photon γ_{ISR} is not included in the cross section. (See Section 4.1.2)

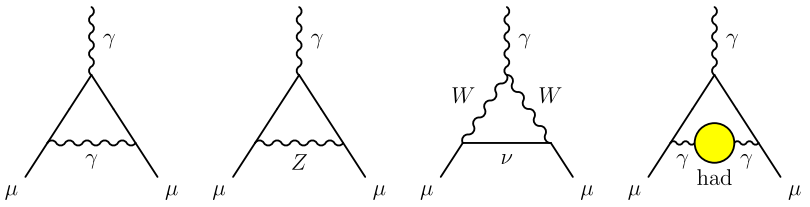


Fig. 4.1.: Contributing processes to a_μ^{SM}. The first diagram on the left side shows the first order QED contribution, followed by the next two diagrams showing the lowest-order weak contribution. Finally, the right most diagram shows the lowest-order hadronic contribution. Illustration taken from [24].

The currently observed deviation between the theoretically predicted and the experimentally measured value of the anomalous magnetic moment of the muon is $3.3\,\sigma$ [24]. In general, the Standard Model theoretical prediction is divided into three components

$$a_\mu^{\mathrm{SM}} = a_\mu^{\mathrm{QED}} + a_\mu^{\mathrm{EW}} + a_\mu^{\mathrm{Hadron}}. \tag{4.2}$$

The experimental measurement (BNL E821 Muon (g-2)) and theoretical calculation of a_μ according to the PDG in August 2019 [24] is:

$$a_\mu^{\mathrm{exp}} = 116592091(54)(33) \times 10^{-11} \tag{4.3}$$

$$a_\mu^{\mathrm{SM}} = 116591830(1)(40)(26) \times 10^{-11} \tag{4.4}$$

The theoretical prediction of the hadronic contributions requires inputs from measured production cross sections of light hadronic productions $ee \to X$. Light hadronic productions contributing the most to a_μ^{Hadron} due to the $\sim 1/s'$ dependency of the QED kernel function $K(s')$ in the dispersion integral[2], where $\sqrt{s'}$ is the energy in the center of mass system (c.m.s.). Hence, the $\rho \to \pi^+\pi^-$ resonance in the $0.6\,\mathrm{GeV} - 0.9\,\mathrm{GeV}$ region dominates. The contribution of the ρ resonance can be calculated from the dispersion integral [24, 25]

$$a_\mu^{\pi\pi(\gamma),\mathrm{LO}} = \frac{1}{4\pi^3} \int_{4m_\pi^2}^{\infty} \mathrm{d}s'\, K(s')\sigma^0_{\pi\pi(\gamma)}(s'), \tag{4.5}$$

where $\sigma^0_{\pi\pi(\gamma)}$ is the "bare" cross section without vacuum polarization but with *final state radiation (FSR)*.

4.1.2. Cross Section Measurement $\mathrm{e^+e^-} \to \pi^+\pi^-(\gamma)$ - ISR Method

To obtain a continuous spectrum of the production cross section in relation to the c.m.s. energy with a particle accelerator operated with a fixed c.m.s. energy, the ISR method is used. Only events with an initial state radiated photon $\mathrm{e^+e^-} \to \pi^+\pi^-(\gamma)\gamma_{\mathrm{ISR}}$ are considered [26, 1]. Fig. 4.2 shows the Feynman diagram of such an ISR process. In addition, higher orders of additional ISR and FSR are also included – indicated as (γ).

[2] $a_\mu^{\mathrm{Hadron,LO}} = \frac{1}{4\pi^3}\int_{4m_\pi^2}^{\infty} \mathrm{d}s'\, K(s')\sigma^0_{\mathrm{Hadron}}(s')$, where $K(s')$ is a QED kernel function, $\sqrt{s'}$ is the c.m.s. energy, and $\sigma^0_{\mathrm{Hadron}}(s')$ is the "bare" cross section of the hadronic process.

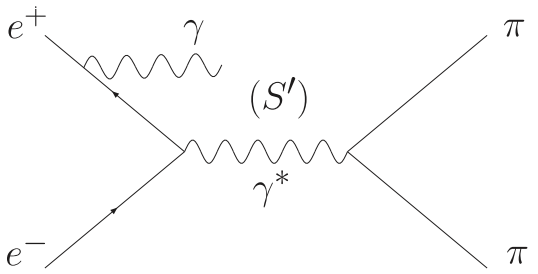


Fig. 4.2.: Production of two charged pions via an electron and positron collision with initial state radiation. Adapted from [26].

The goal is to measure the cross section used in Equation 4.5 more precisely to reduce the uncertainty of the theoretical prediction of a_μ^{SM}.

In 2012, the BABAR experiment presented the latest and most accurately measured $\mathrm{e^+e^-} \to \pi^+\pi^-(\gamma)$ production cross section [26]. With this latest measurement the total error on the anomalous muon moment a_μ^{SM} was reduced to 0.7 % of the nominal value. With 4.3 times higher statistics ($1\,\mathrm{ab}^{-1}$), it may be possible to reduce the total error to 0.4 % of the nominal value [1].

The relation of the process' "bare" cross section $\sigma_X^0(\sqrt{s'})$ and the number of observed events $N_{X\gamma_{\mathrm{ISR}}}$ is given by

$$\frac{\mathrm{d}N_{X\gamma_{\mathrm{ISR}}}}{\mathrm{d}\sqrt{s'}} = \frac{\mathrm{d}L_{\mathrm{ISR}}^{\mathrm{eff}}}{\mathrm{d}\sqrt{s'}} \cdot \epsilon_{X\gamma_{\mathrm{ISR}}}(\sqrt{s'}) \cdot \sigma_X^0(\sqrt{s'}), \tag{4.6}$$

where $\mathrm{d}L_{\mathrm{ISR}}^{\mathrm{eff}}/\mathrm{d}\sqrt{s'}$ is the effective ISR luminosity, $\epsilon_{X\gamma_{\mathrm{ISR}}}(\sqrt{s'})$ is the detection efficiency, and $\sqrt{s'}$ is the reduced mass due to the ISR photon[3]. A more detailed explanation can be found in [26]. The Level 1 trigger Bhabha vetoes contribute to the detection efficiency $\epsilon_{X\gamma_{\mathrm{ISR}}}(\sqrt{s'})$.

4.2. Level 1 Bhabha Vetoes and ECL Energy Trigger

The Bhabha process $\mathrm{e^+e^-} \to \mathrm{e^+e^-}$ dominates the production in the SuperKEKB collider due to its huge cross section. To reduce these events a simple 2D Bhabha trigger was used in the previous Belle experiment to veto these events. Without such a trigger line, the amount of data would be too large to be written to the storage system. The same 2D Bhabha veto as for Belle was used in the Belle II experiment from the beginning of phase II to the end of experiment 8 in phase III of the data taking.

The 2D Bhabha trigger expects two particles that generates two electromagnetic showers in the ECL. To distinguish the position of deposited energy in ECL clusters, the ECL is divided into 17 rings around the beam axis (Fig. 4.3). These rings do not correspond to the granularity of the ECL. Only two back-to-back ECL clusters in the θ-z plane are considered for the trigger decision. Therefore, 14 combinations of ECL ϕ rings are taken

[3] $s' = s(1 - 2E_\gamma^\star/\sqrt{s})$, where $\sqrt{s}$ is the collider c.m.s. energy and E_γ^* is the energy of the ISR photon.

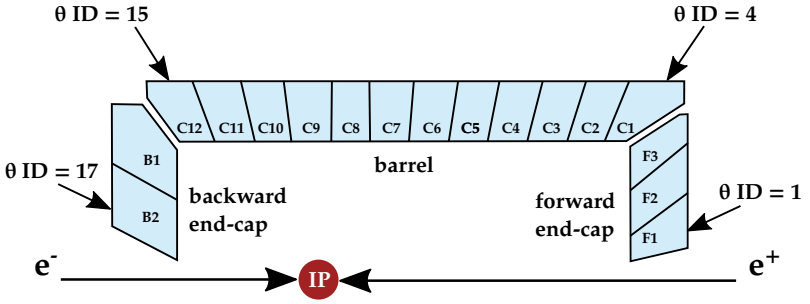


Fig. 4.3.: Schematic overview of half of the ECL ϕ rings cross section around the beam axis. Figure adapted from [27].

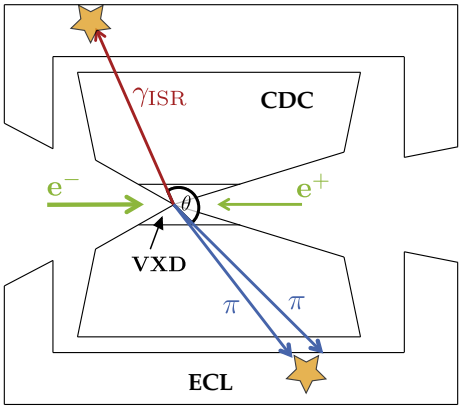


Fig. 4.4.: Schematic visualization of a $e^+e^- \to \pi^+\pi^-\gamma_{ISR}$ event depositing energy in the ECL. With this back-to-back event signature in the θ-z plane, the 2D Bhabha trigger is fired. Figure adapted from [28].

into account. If, for example, an ECL ϕ ring in the forward end-cap exceeds the energy threshold and additionally in some ECL ϕ rings of the backward end-cap, the `ecl_bha` trigger is fired. The threshold energy lies between 2.5 GeV and 4 GeV and depends on the position of the ring. In the forward direction the threshold energy is in general higher than in the backward direction due to the asymmetric beam energies. [27]

This simple 2D Bhabha veto brings a high loss of efficiency for e.g. $\pi^+\pi^-\gamma_{ISR}$ events when the two pions are collinear and back-to-back to the ISR photon. A corresponding event signature is shown in Fig. 4.4. To avoid such a false positive, the 3D Bhabha veto is used since the beginning of experiment 10. It also uses the ϕ information of the ECL clusters.

In Table 4.1 the requirements of the 3D Bhabha trigger are listed. Two ECL clusters must exist (CL1 and CL2) and the sum of the θ_{CMS} polar angles of the two clusters must be between 165° and 190°. The opening of the azimuthal angles $\Delta\phi_{CMS}$ between the two ECL clusters must be between 160° and 200°. In addition, the c.m.s. energy E_{CMS} of each cluster (CL1 and CL2) must be higher than 3 GeV and one of the clusters must have a c.m.s. energy higher than 4.5 GeV. [27]

A loose ECL trigger is the Level 1 `hie` trigger. This trigger is fired when the energy

Table 4.1.: Conditions for the 3D Bhabha Level 1 trigger, all quantities are in the c.m.s. frame:

	condition
(1)	$165^\circ < \sum \theta_{\mathrm{CMS}} < 190^\circ$
(2)	$160^\circ < \Delta\phi_{\mathrm{CMS}} < 200^\circ$
(3)	$E_{\mathrm{CMS,CL1}} > 3$ GeV and $E_{\mathrm{CMS,CL2}} > 3$ GeV and ($E_{\mathrm{CMS,CL1}} > 4.5$ GeV or $E_{\mathrm{CMS,CL2}} > 4.5$ GeV)

Table 4.2.: Recorded data samples used for the analysis of the 2D and 3D Bhabha veto:

purpose	data production	runs	int. luminosity
2D Bhabha	exp. 8, proc 11	2639,1976,2266,1539,2630,2608,1553	$413\,\mathrm{pb}^{-1}$
3D Bhabha	exp. 12, buck. 10	2849,2750,2860,2731,2844,2859,2721, 2739,2880,2845	$2382\,\mathrm{pb}^{-1}$

deposited in the ECL is greater then 1 GeV and no Bhabha trigger veto or injection veto due to particle injection into the storage rings has been triggered. In experiment 8 the Bhabha veto is induced by the 2D Bhabha trigger and from the beginning of experiment 10 the Bhabha veto is induced by the 3D Bhabha trigger.

In the current detector operation, the Bhabha triggered events are all recorded due to still sufficiently low data rates. Therefore, the loss of the Bhabha vetoes was calculated by comparing the collected dataset with the `hie` triggered data and the Bhabha triggered data as described in the following Section 4.4 and Section 4.5. This measurement was not biased by the HLT because it operates currently not in a filter mode (see Chapter 5).

4.3. Event Selection

In this analysis, experiment 8 proc 11 recorded data was used to investigate the impact of the 2D Bhabha veto on $\mathrm{e^+e^-} \to \pi^+\pi^-(\gamma)\gamma_{\mathrm{ISR}}$ events. In addition, experiment 12 bucket 10 recorded data was used to investigate the 3D Bhabha veto regarding this process as well. The used datasets are listed in Table 4.2. Contributing processes relevant for the event selection, are listed in Table 4.3. Here, simulated data were included with the exception of $ee \to ee\gamma$, which was only used for additional studies as described further down in this section. The integrated luminosity of the simulated data samples has been scaled to the recorded data integrated luminosity as described in Section 3.5. It was assumed that the efficiency of the Level 1 trigger on the simulated data is 100 %.

Table 4.3.: Simulated data used to examine contributing processes in the $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ event selection:

process	MC production	int. luminosity
[4] $ee \to KK\gamma$	MC13a	$638\,978\,\mathrm{pb}^{-1}$
[5] $ee \to \mu\mu\gamma$	MC13a	$59\,880\,\mathrm{pb}^{-1}$
[6] $ee \to \pi\pi\gamma$	MC13a	$10\,000\,\mathrm{pb}^{-1}$
[7] $ee \to ee\gamma$	MC13a	$10\,000\,\mathrm{pb}^{-1}$

The *Belle 2 software analysis framework* (`basf2`) [29] was used for the event selection to produce the n-tuples. A so-called `basf2` steering file was created to select all events with the corresponding event topology from the included datasets. The event topology consists of two oppositely charged tracks and an energy deposit in the ECL without an associated track. Therefore, the `basf2` internal `pi-:all` and `pi+:all` particle lists, as well as the `gamma:tight` particle list were used. `basf2` automatically creates all possible combinations of particles in the event that fulfill the defined criteria. If there are multiple candidates, the one with the highest ECL energy for the ISR photon candidate is chosen. The reconstructed decay chain is $\rho \to \pi^+\pi^-$ and $\gamma^* \to \rho\gamma_{\mathrm{ISR}}$.

In a second step, additional cuts were made (using `python3` and `DataFrame` objects from the `pandas` package [30]) to reduce the number of background events that still contaminated the selected events in the n-tuple. These additional cuts applied to the event properties are listed in Table 4.4.

To distinguish electrons from other charged tracks a dedicated region in the ratio E/p of the energy E and the momentum p of the charged particles is selected. It is used to separate events with pion pairs from those with electron pairs. Fig. 4.5 shows the efficiency of the E/p cut applied to both charged particles. In general, electrons have a higher E/p ratio than pions (or muons, see Chapter 5). The remaining Bhabha contamination in the recorded data sample after the E/p cut applied is expected to be about 0.056 % and is neglected.

To ensure that the event was generated by a collision of an electron and a positron the track has to originate from the interaction point (IP). This is enforced with cuts on the point of closest approach (POCA) in the r-ϕ plane $|d_0|$ and z-direction $|z_0|$. Only events with at least 5 CDC hits (`nCDCHits`) where used, and the energy of the ISR photon was required to be over 3 GeV in the c.m.s. frame. Furthermore, due to the low efficiency of the

[4] /belle/MC/release-04-00-03/DB00000757/MC13a/prod00013235/s00/e1003/4S/r00000/3000021006/mdst/sub00

[5] /belle/MC/release-04-02-04/DB00001046/MC13a/prod00013669/s00/e1003/4S/r00000/mumu/mdst/sub00/

[6] /belle/MC/release-04-00-03/DB00000757/MC13a/prod00013237/s00/e1003/4S/r00000/3110021000/mdst/sub00

[7] /belle/MC/release-04-02-04/DB00001046/MC13a/prod00013668/s00/e1003/4S/r00000/ee/mdst/sub00

[8] wrong in Belle 2 note [23]

Table 4.4.: Cuts applied to the properties of the $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ event candidates. The track fit χ^2 prob. cut was only applied to the experiment 8 data:

selection	variable name	criteria
track fit χ^2 prob.	pValue	> 0.001/ -
E/p	clusterEoP	< 0.8[8]
$\mid d_0 \mid$	abs(d0)	$< 2\,\mathrm{cm}$
$\mid z_0 \mid$	abs(z0)	$< 4\,\mathrm{cm}$
nCDCHits	nCDCHits	> 4
γ_{ISR} c.m.s. energy	useCMSFrame(E)	$> 3\,\mathrm{GeV}$
track momentum	p	$> 1\,\mathrm{GeV}$
$\theta_{\gamma_{\mathrm{ISR}}}$ angle	theta	$50^\circ < \theta_{\gamma_{\mathrm{ISR}}} < 110^\circ$
inv. mass γ^* (all part.)	M	$10\,\mathrm{GeV} < M(\pi\pi\gamma) < 11\,\mathrm{GeV}$

ECL end-cap regions, only events are used where the ISR photon is detected in the ECL barrel region by applying a cut to the polar angle of the ISR photon $\theta_{\gamma_{\mathrm{ISR}}}$. A cut on the pion momentum p in combination with a cut on the invariant mass window of the whole event removes additional background.

The track fit χ^2 probability cut was only applied to experiment 8 data to compare the results with [23]. In the analysis of the experiment 12 data this cut was removed because it caused a bad agreement between recorded and simulated data. It is a known issue in the tracking group that many events in more recent recorded data contain a low track fit χ^2 probability (close to zero) because of imperfections in calibration or noise hits due to multiple scattering [31].

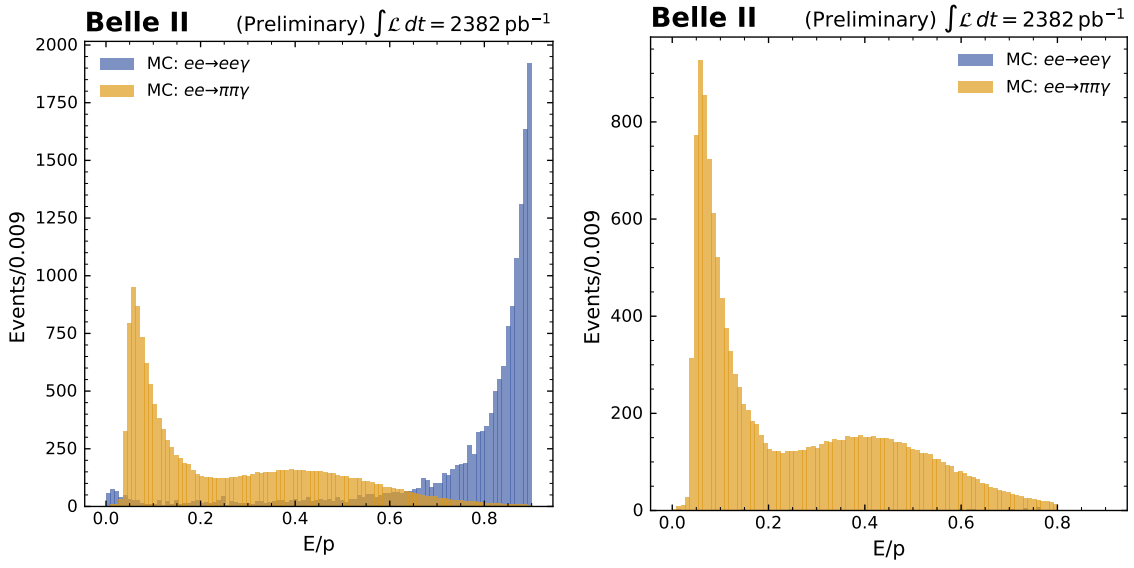


Fig. 4.5.: Simulated $ee \to \pi\pi\gamma$ and $ee \to ee\gamma$ data samples considered a) without the E/p applied and b) with E/p cut applied to both charged particles. After applying also the E/p cut only $4.4 \times 10^{-3}\,\%$ of the simulated Bhabha events remain in the selection. These are $0.056\,\%$ with respect to the total number of selected recorded data events. (Caution: Since the cut is applied to both charged particles of an event, and in the histograms (a) and (b) both particles are combined, the cut removes almost all particles with $E/p \leq 0.8$ of the $ee\gamma$ events.)

4.4. Loss of the 2D Bhabha Veto

It was analyzed to what extent the 2D Bhabha veto rejects candidates which would be accepted by the offline $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ selection. A $413\,\mathrm{pb}^{-1}$ dataset from experiment 8 was used and the result was compared with the results of an earlier analysis performed with data from early phase II [23]. Furthermore, this step validated the method used for the event selection in Section 4.3.

Fig. 4.6 shows the reconstructed invariant mass $M(\pi\pi)$ of the ρ meson. The ρ peak around 770 MeV expected by simulated data was observed in the recorded data as well. A smaller peak below 500 MeV is expected from the ϕ peak, which should actually occur at 1.02 GeV. This peak originates from two miss-identified kaon pairs with a wrong mass hypothesis (mass of pions) shifting the peak to a lower mass region. At the edge of the ρ peak to the higher masses the $\rho-\omega$ interference can be slightly observed by a low increase of the number of events. The ω mass is about 782 MeV. Considering the agreement of the recorded data to simulated data calculated by

$$\text{agreement} = \frac{N_{\mathrm{rec}}}{N_{\mathrm{sim}}}, \tag{4.7}$$

where N_{rec} is the number of recorded events and N_{sim} is the number of simulated events, the agreement in the 0 GeV to 2.5 GeV ρ mass window is 0.977. This is in good agreement with [23].

Since the events selected by the 2D Bhabha trigger have also been stored, this substantially simplifies this trigger analysis. To quantify the loss of $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ events if the 2D Bhabha trigger events were dismissed, equation

$$\text{loss} = 1 - \frac{N_{\mathrm{hie}}}{N_{\mathrm{hie|bhabha}}} \tag{4.8}$$

is used, where N_{hie} is the number of events selected by the `hie` trigger and $N_{\mathrm{hie|bhabha}}$ is the number of events selected by the `hie` or the 2D Bhabha trigger (exclusive or).

The recorded events selected by the 2D Bhabha, 3D Bhabha, or `hie` trigger are shown in Fig. 4.7a. Already here the difference between the 2D Bhabha trigger (`bhabha`) and the 3D Bhabha trigger (`bha3d`) is clearly visible. Fig. 4.7b shows the recorded events selected by the `hie` or 2D Bhabha trigger. The `hie` and the 2D Bhabha trigger covered 100 % of the selected $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ event candidates. This results in a `hie` trigger efficiency of 100 %.

For the calculation of the loss, the `TEfficiency` class described in Section 3.4 was used to calculate the efficiency term in Equation 4.8. The resulting loss due to the Bhabha 2D veto is

$$\text{loss} = 14.5^{+0.75}_{-0.72}\ \% \tag{4.9}$$

in the 0 GeV to 2.5 GeV ρ mass window. The mass region between 0 GeV and approx. 0.3 GeV is empty due to the applied cuts. As one can see from the simulated data, the expected number of "true" $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ events is around 0.3 GeV very small so that the impact in the mass region between 0 GeV and 0.3 GeV is expected to be negligibly low. Fig. 4.8b shows the calculated loss for each bin in the 0 GeV to 2.5 GeV mass region and

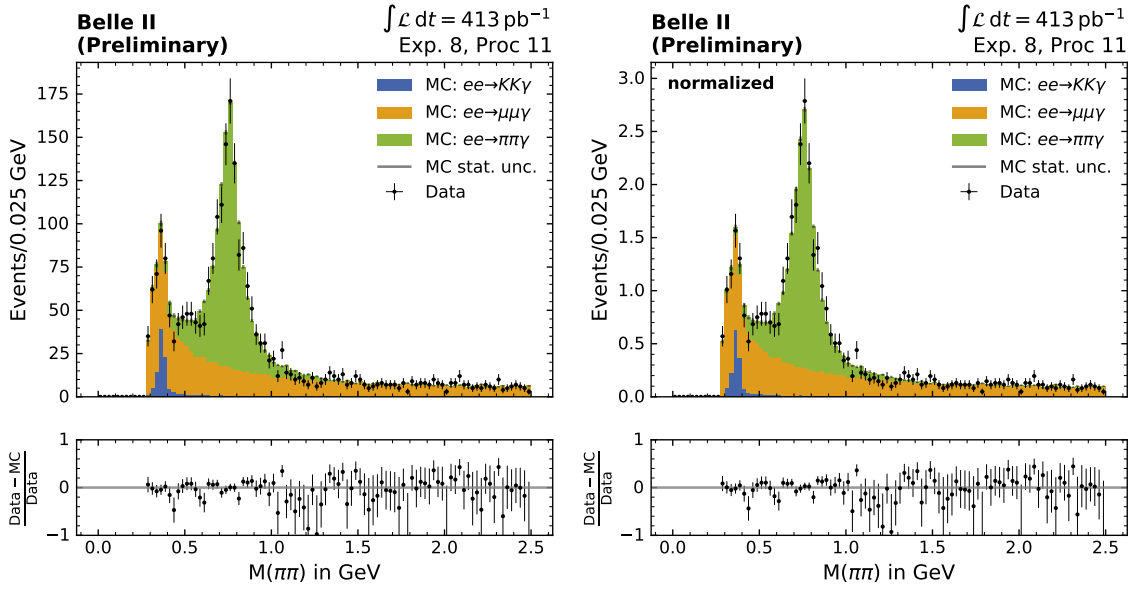
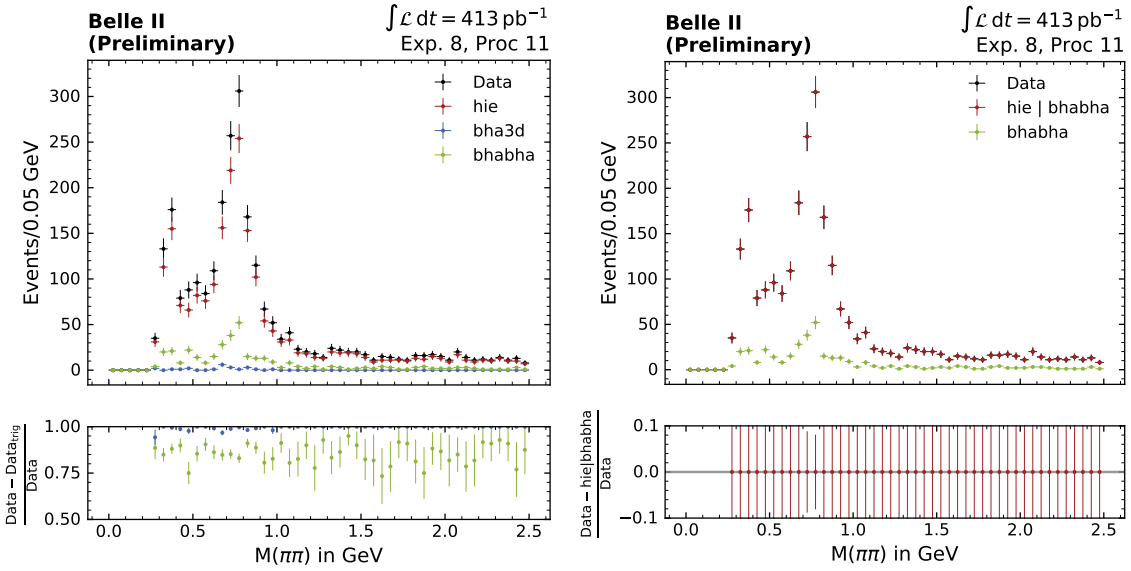


Fig. 4.6.: The reconstructed invariant mass of the ρ meson. In (a) the simulated data scaled to the data integrated luminosity is shown and in (b) it is shown normalized for the shape comparison between recorded and simulated data.



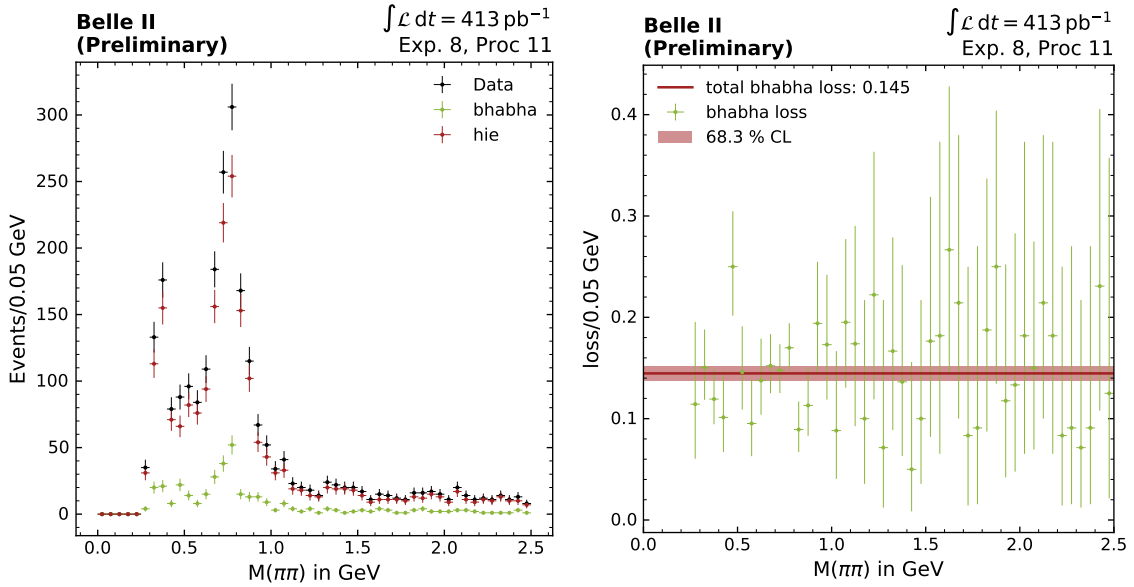
(a) **(b)**

Fig. 4.7.: Distribution of the investigated Level 1 triggers. (a) shows the events selected by the respective Level 1 trigger and (b) shows the events selected by the `hie` or 2D Bhabha (`bhabha`) trigger, which cover the entire data sample as one can see also in the pull plot.

Fig. 4.8a shows the distribution of the events passing the respective Level 1 trigger. The 2D Bhabha veto loss calculated in this analysis differs slightly from the loss of $12.3^{+0.8}_{-0.8}$ % calculated in [23]. A possible reason for the deviation is a smaller ρ mass range from 0 GeV to 2 GeV used in [23] for the calculation. Also, the data used in [23] was from a different data taking period with different detector calibrations and settings, which may cause a small difference as well.

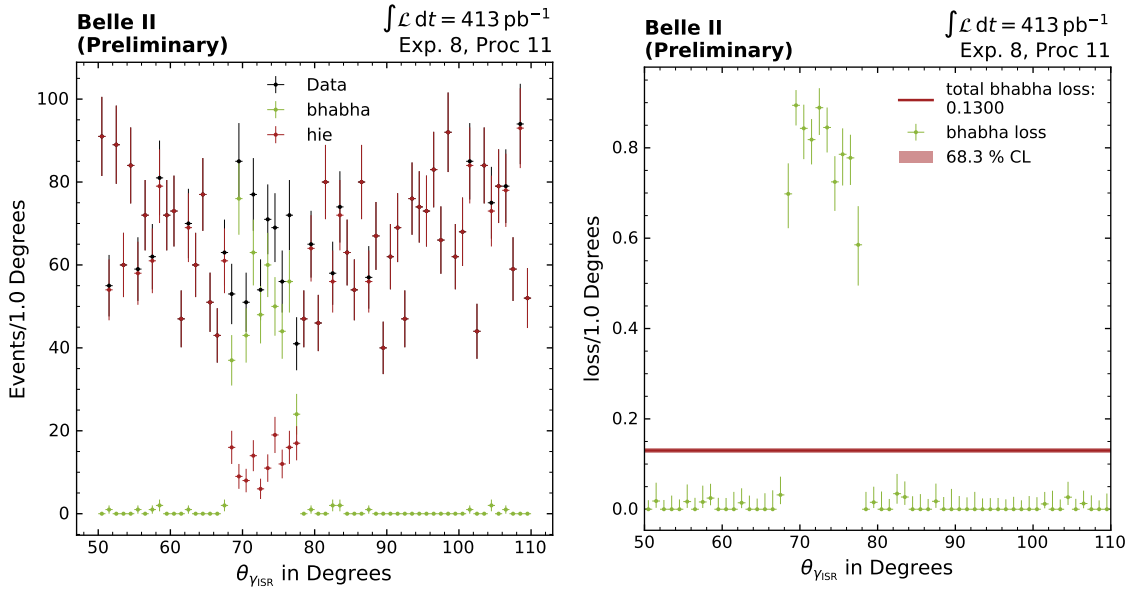
In addition, the angular distribution of the events which were selected by the 2D Bhabha trigger and the angular dependence of the loss was investigated. The loss shows a strong dependence on the polar angle $\theta_{\gamma_{\mathrm{ISR}}}$ of the ISR photon, as shown in Fig. 4.9b. Figure 4.9a shows the distribution of the events selected by the respective triggers as function of the polar angle $\theta_{\gamma_{\mathrm{ISR}}}$.

Further diagrams regarding the loss as function of event properties are shown in Appendix A.2.1.



(a) (b)

Fig. 4.8.: (a) shows the distribution of the events selected by the respective Level 1 trigger and (b) shows the calculated loss for the 2D Bhabha veto in each mass bin and the total value for the entire 0 GeV to 2.5 GeV mass range.



(a) (b)

Fig. 4.9.: (a) shows the distribution of the events selected by the respective Level 1 trigger and (b) shows the calculated loss for the 2D Bhabha veto in each polar angle $\theta_{\gamma_{\mathrm{ISR}}}$ bin in the region covered by the CDC.

4.5. Loss of the 3D Bhabha Veto

The impact of the 3D Bhabha veto on the $e^+e^- \to \pi^+\pi^-(\gamma)\gamma_{ISR}$ trigger efficiency was investigated using the same methods as described in the previous sections. However, the dataset was replaced by experiment 12 bucket 10 recorded data.

The track fit χ^2 probability cut (`pValue` cut) applied to experiment 12 data delivered different results as for the experiment 8 data. Fig. 4.10a shows the reconstructed ρ mass plot with applied cut on the track fit χ^2 probability. As one can clearly see, the statistics of the recorded data is significantly lower than expected from the simulated data samples. In this case, the agreement between recorded and simulated data calculated with Equation 4.7 is 0.756. However, the shape shown in Fig. 4.10b in the normalized histogram shows a good agreement for recorded and simulated data.

Without the cut on the track fit χ^2 probability, the agreement between recorded and simulated data calculated with Equation 4.7 is 1.046. Fig. 4.11a shows the ρ mass plot without this cut. Also the shape comparison in Fig. 4.11b between recorded and simulated data shows a good agreement. Based on this result the analysis of the 3D Bhabha veto loss was performed without the track fit χ^2 probability cut (see end of Section 4.3).

Fig. 4.12a shows the distribution of the events selected by the `hie`, `bhabha`, and `bha3d` Level 1 triggers, respectively, compared to all events selected by the Level 1 trigger as function of to the reconstructed ρ mass.

The events selected by the `hie` or 3D Bhabha triggers result in a coverage of 0.997 of the analyzed recorded dataset, in the ρ mass region between 0 GeV and 2.5 GeV, as shown in Fig. 4.12b. A further investigation of the `hie` trigger efficiency ε_{hie} on this dataset, using an orthogonal unprescaled two track trigger (`ffo`), resulted in

$$\varepsilon_{\text{hie}} = 99.50^{+0.087}_{-0.098}\ \%. \tag{4.10}$$

This `hie` trigger efficiency ε_{hie} and its uncertainties $\Delta\varepsilon_{\text{hie}}$ were taken into account in the loss calculation. Eventually, the impact of these uncertainties $\Delta\varepsilon_{\text{hie}}$ were considered as systematic uncertainty to the loss.

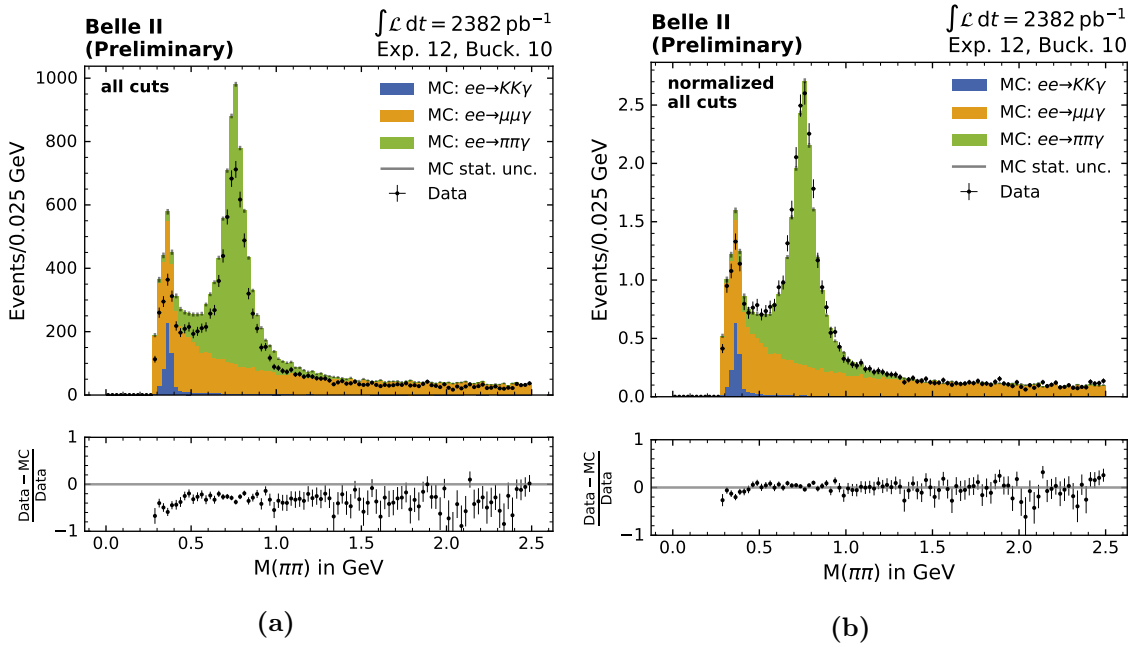
The loss of the 3D Bhabha veto is calculated similar to Section 4.4, however, the number of events selected by the `hie` trigger was corrected due to its efficiency ε_{hie}. Also the selected data in the analysis was adapted to the `bha3d` Level 1 trigger as follows

$$\text{loss} = 1 - \frac{\frac{1}{\varepsilon_{\text{hie}}} N_{\text{hie}}}{N_{\text{bha3d}} + \frac{1}{\varepsilon_{\text{hie}}} N_{\text{hie}}}, \tag{4.11}$$

where N_{hie} is the number of events selected by the `hie` trigger and N_{bha3d} is the number of events selected by the 3D Bhabha trigger.

The systematic uncertainty was calculated using

$$\begin{aligned} \Delta\text{loss}_{\text{sys}} &= \left| \frac{\partial}{\partial \varepsilon_{\text{hie}}} \text{loss} \right| \cdot \Delta\varepsilon_{\text{hie}} \\ &= \left| \frac{\frac{1}{\varepsilon_{\text{hie}}^2} N_{\text{hie}} N_{\text{bha3d}}}{\left(N_{\text{bha3d}} + \frac{1}{\varepsilon_{\text{hie}}} N_{\text{hie}} \right)^2} \right| \cdot \Delta\varepsilon_{\text{hie}}. \end{aligned} \tag{4.12}$$



(a) (b)

Fig. 4.10.: (a) shows the reconstructed ρ mass for the case that the cut on the track fit χ^2 probability of the two charged pion candidates is applied. The simulated data was scaled to the integrated luminosity of the recorded data. In (b) the simulated and recorded data are shown normalized for the shape comparison.

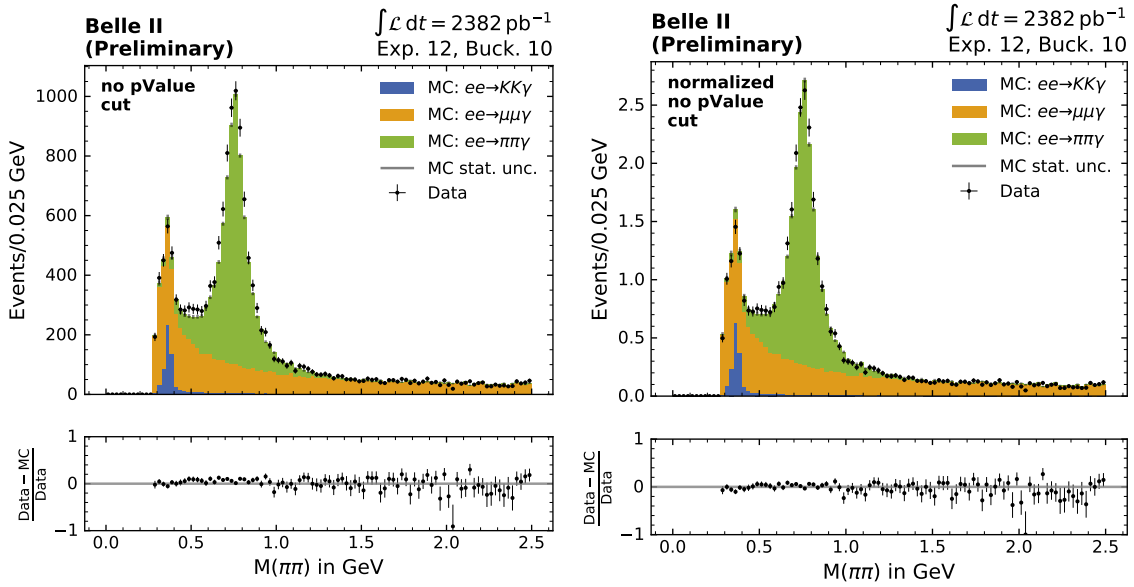
equa The total loss of the 3D Bhabha veto in the ρ mass range of 0 GeV to 2.5 GeV is

$$\text{loss} = 0.67^{+0.07226+0.00040}_{-0.06557-0.00045}\ \%, \tag{4.13}$$

where $(-0.00045,\ +0.00040)$ is the systematic uncertainty due to the Level 1 `hie` trigger efficiency uncertainty $\Delta\varepsilon_{\text{hie}}$.

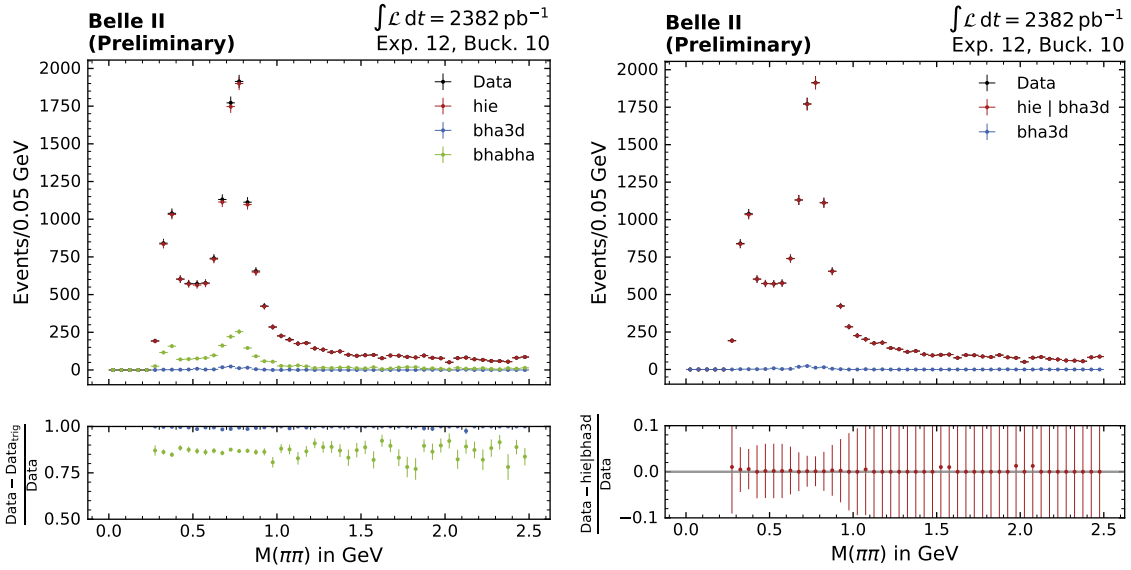
This is a reduction of the loss by a factor ~ 18 compared to the 2D Bhabha veto. Fig. 4.13b shows the distribution of the loss within the 0 GeV to 2.5 GeV ρ mass window with respect to the ρ mass distribution shown in Fig. 4.13a. The loss of the 3D Bhabha veto was estimated by [23] to be $0.6^{+0.4}_{-0.4}$ %, with a large statistical uncertainty of 66.67 % due to low statistics of phase II ($472\,\mathrm{pb}^{-1}$).

Furthermore, the angular and energy distribution of the ISR photon candidate was investigated. No significant specific regions with higher losses caused by the 3D Bhabha veto were observed. Despite a five times larger analyzed recorded dataset than [23], the statistics for such a spatial analysis were still too low due to the huge phase space and the low number of $\pi^+\pi^-\gamma_{\text{ISR}}$ events selected by the 3D Bhabha trigger. The corresponding plots are shown in Appendix A.2.2.



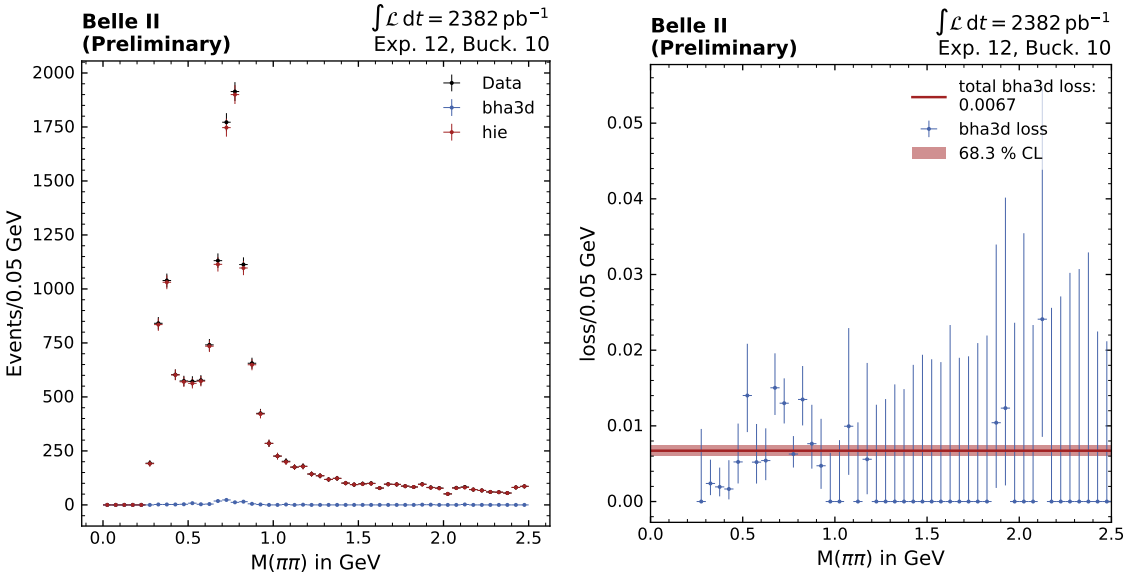
(a) **(b)**

Fig. 4.11.: (a) shows the reconstructed ρ mass without the cut on the track fit χ^2 probability (`pValue`) of the two charged pion candidates and (b) shows the simulated and recorded data normalized for the shape comparison. The removal of the track fit χ^2 probability cut improves the agreement significantly.



(a) **(b)**

Fig. 4.12.: Distribution of the investigated Level 1 triggers. (a) shows the events selected by the respective Level 1 trigger and (b) shows the events selected by the hie or 3D Bhabha (`bha3d`) trigger, which cover 0.997 of the $\pi^+\pi^-\gamma_{\mathrm{ISR}}$ event candidates selected by the Level 1 trigger.



(a) **(b)**

Fig. 4.13.: a) shows the distributions of the events selected by the respective Level 1 trigger and (b) shows the calculated loss for the 3D Bhabha veto in each mass bin and the total value for the entire 0 GeV to 2.5 GeV mass range.

4.6. Summary and Conclusion

The loss of the 2D Bhabha and 3D Bhabha veto was analyzed using recorded data of experiments 8 and 12. This resulted in $14.5^{+0.75}_{-0.72}$ % loss for the 2D Bhabha veto[9] and in $0.67^{+0.07226+0.00040}_{-0.06557-0.00045}$ % loss for the 3D Bhabha veto in a ρ mass range between 0 GeV and 2.5 GeV of the selected charged pion candidates. Finally, the 3D Bhabha veto reduces the loss of the 2D Bhabha veto about a factor of ~ 18. For future analysis and detector operation only the 3D Bhabha veto is relevant.

Only a small amount of available experiment 12 data ($2382\,\mathrm{pb}^{-1}$) was used during this work. The uncertainty on the loss can be further reduced with a larger data sample as required. Further studies with more statistics will allow a spacial analysis, e.g., the distributions of the ISR photon polar or azimuthal angles, the spacial energy distributions, or the angles between the particles. In addition, more accurate evaluations of loss correlations in the phase space will be possible.

The stability of the 3D Bhabha veto as a function of time should be investigated in further studies. For this purpose, datasets of different runs, e.g., of a run period or several run periods have to be examined and compared with respect to the 3D Bhabha veto.

[9] A systematic uncertainty was not calculated because this veto is obsolete. The 2D Bhabha veto was only considered for comparison and validation purposes.

5. HLT Efficiency of $e^+e^- \rightarrow \mu^+\mu^-$

During data taking, the HLT calculates trigger decisions for each event. Two modes of operation exist. One mode is the *filter mode*, where, according to the HLT decision, the events are either stored or irretrievably deleted. The other mode is the *monitoring mode*, where no data is dismissed. Here, each event is tagged by the HLT with its trigger decision. Currently, the HLT is operated in the monitoring mode, which allows a detailed offline analysis of the HLT efficiency.

In this chapter the results of the analysis of the HLT efficiency for the process $e^+e^- \rightarrow \mu^+\mu^-$ are presented. Section 5.1 describes the selection criteria and n-tuple generation for both the simulated and recorded data samples. Section 5.2 presents the analysis of the latest proc 11 data from experiment 10. The analysis of the older proc 10 data from experiment 8 is presented in Section 5.3 together with the investigation of the optimization of the HLT with `basf2` release 03-01-03 onwards.

5.1. Dimuon Event Selection

The n-tuples were created with a `basf2` steering file. For the selection of the two charged muons the internal `basf2` particle lists `mu+:all` and `mu-:all` were used. The best candidate, was selected by choosing the candidate with the highest sum of momentum p of both muon tracks. The two muon tracks were reconstructed within a decay chain to a virtual photon `vpho` ($\gamma^* \rightarrow \mu^-\mu^+$).

Both during the n-tuple selection and also offline (using `python3` and `DataFrame` objects from the `pandas` package), the events were additionally filtered by applying cuts on the event properties listed in Table 5.1. They are required to separate the dimuon events from the background. Especially Bhabha scattering contributes significantly to the background.

In the regions of the geometric detector acceptance [32], the production cross section of the Bhabha events is approximately 260 times larger than the production of dimuon events. The cut on the ratio of the energy E to the momentum p is used to separate muon pairs from electron pairs, as shown schematically in Fig. 5.1. Electrons have a higher E/p ratio than muons or pions. Spatial cuts on the production vertices $|d_0|$ (POCA in the x-y plane from the interaction point) and $|z_0|$ (POCA in z-direction from the interaction point) are used to ensure that the event was generated by a collision of an electron and positron. Muons are minimum ionizing particles (MIPs) in the energy region relevant for Belle II. Therefore,

Table 5.1.: Selection criteria for two track muon selection:

selection	variable name	criteria
E/p	`clusterEoP`	< 0.6
$\mid d_0 \mid$	`abs(d0)`	$< 0.5\,\text{cm}$
$\mid z_0 \mid$	`abs(z0)`	$< 2\,\text{cm}$
Entry in calorimeter	`clusterE`	$0 <$ clusterE $< 1\,\text{GeV}$
nCDCHits	`nCDCHits`	> 1
track momentum	`p`	$> 2\,\text{GeV}$
mass vpho (all part.)	`M`	$10\,\text{GeV} < M < 11\,\text{GeV}$

the cut on the energy deposited in the ECL is chosen between 0 GeV and 1 GeV. A cut on the the muon momentum $\leq 2\,\text{GeV}$ in combination with the requirement that the invariant mass of the two reconstructed muons to a virtual photon has to lie within the 10 GeV to 11 GeV window removes all the background in low momentum regions. In addition, the particles must leave at least 1 hit in the CDC.

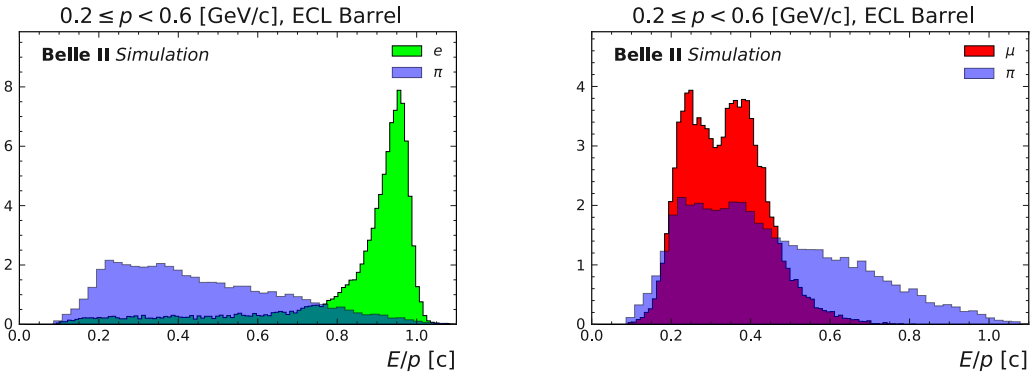


Fig. 5.1.: The separation of different particle types according to the fraction of energy E and momentum p. In the left figure the distribution for electrons and pions is shown and in the right figure the distribution for muons and pions is shown. Combining both figures one can see an electron - muon separation is possible with $E/p \sim 0.6$. Figure taken from [33].

The processes [1]$ee \to \mu\mu$, [2]$ee \to ee$, and [3]$ee \to ee\mu\mu$ were considered with respect to their contribution to the dimuon event selection. Therefore, simulated datasets of the respective processes were processed. All considered dimuon data samples (recorded and simulated) were selected with the same selection method explained in this section. Thus, a comparison between the individual datasets was possible by scaling the samples with the respective integrated luminosity as described in Section 3.5.

[1] /belle/MC/release-04-00-03/DB00000757/MC13a/prod00010385/s00/e1003/4S/r00000/mumu/mdst
[2] /belle/MC/release-04-00-03/DB00000757/MC13a/prod00012385/s00/e1003/4S/r00000/ee/mdst
[3] /belle/MC/release-04-00-03/DB00000757/MC13a/prod00012387/s00/e1003/4S/r00000/eemumu/mdst

5.2. Analyzing Data Taken With the Latest Software Trigger Release

The data processing proc 11 was the latest available processed recorded data at the time of this analysis. For this analysis run 5610[4] was chosen to study the latest available version of the HLT software trigger. A recorded dataset at the end of experiment 10 processed on the HLT with the latest available `basf2` release version was selected. Experiment 10 was the most recent processed experiment at that time. `basf2` release 04-00-04 was used for online processing on the HLT [34] and release 04-02-02 was used for the offline proc 11 processing. The dimuon selection for this analysis of the HLT efficiency was performed using `basf2` release 04-01-01.

Using the latest data processing with one of the latest physics runs, the HLT efficiency for dimuon events is in total

$$\varepsilon_{\mathrm{HLT},\mu\mu} = 99.98^{+0.010}_{-0.018}\ \%. \tag{5.1}$$

Fig. 5.2a shows the the HLT efficiency over θ_{lab} with no significant drop of the efficiency in any θ_{lab} region. The HLT efficiency is over 99.5 % in each θ_{lab} bin between 17° and 150° which is the region covered by the CDC [1]. A similar picture can be observed for the track momenta p of the muons shown in Fig. 5.2c. Most of the events are detected between ~ 4 and 7 GeV. Here, the influence of poor statistics can be observed with the large error bars below 4 and above 7 GeV.

The overall recorded/simulated data ratio is 95.7 % and within the barrel region[5] up to 97.7 %. The recorded/simulated data disagreement could be a result of level 1 trigger inefficiencies not considered or a poorly modeled detector acceptance. Especially in the end-cap regions this disagreement increases. Fig. 5.2 shows the efficiency and event distribution as function of θ_{lab} and p respectively. The distributions as functions of further variables used for the cuts as well as the software trigger efficiencies for two track dimuon events as a function of these variables are shown in Appendix A.3.1.

Bhabha scattering is in our case the background process contributing most significantly, as the simulated data selection shows. The $ee \to ee\mu\mu$ process shows no significant contribution overall because the selection cuts mostly removed this contribution. Fig. 5.2b shows that all $e^+e^- \to e^+e^-$ events are located in two single bins in the regions between the barrel and the forward and backward end-cap. In these regions, the ECL detector has a small gap, so the E/p selection fails for events with a track that traverses this gap without depositing (enough) energy in the ECL. Fig. 2.1 shows this small gap between barrel and end-cap ECL sections.

Using the $ee \to \mu\mu$ simulated data sample, the HLT efficiency for two track dimuon events is expected to be 100 % as observed.

[4] /belle/Data/proc/release-04-02-02/DB00000938/proc11/prod00013372/e0010/4S/r05610/mdst/sub00

[5] $40° \leq \theta_{\mathrm{lab}} \leq 120°$

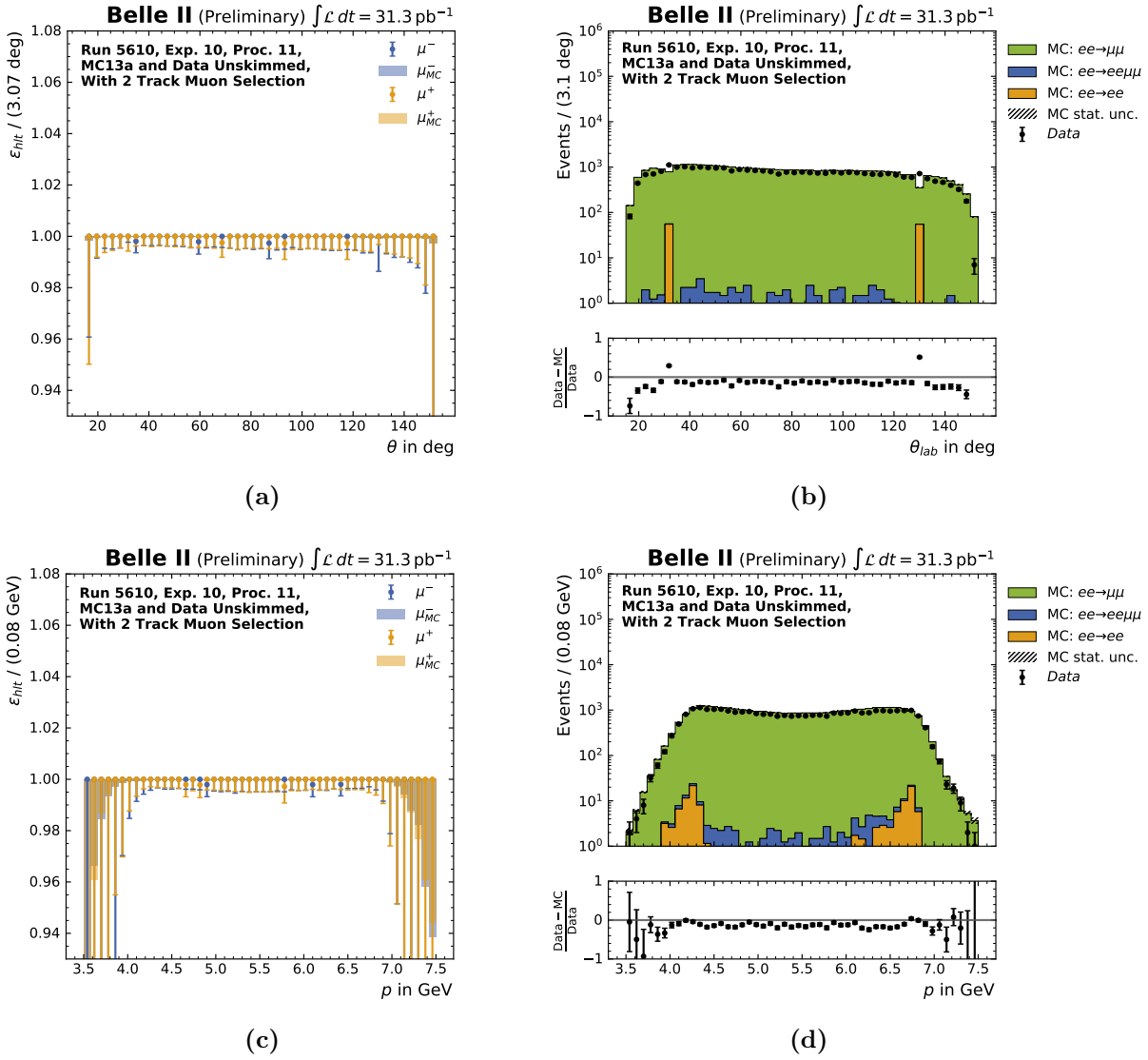


Fig. 5.2.: HLT efficiency of run 5610 in experiment 10 (a) as function of the polar angle θ_{lab} and (c) as function of the momentum p. No specific dependency of the efficiency is observable with respect to the polar angle θ_{lab} or the momentum p. The distribution of these variables are shown in (b) for polar angle and in (d) for the momentum.

5.3. Analyzing Data Taken With an Older Software Trigger Release

Initially, the HLT two track dimuon efficiency analysis was performed on the data recorded in experiment 8 with the offline processing proc 10. At that time, the recorded data of experiment 8 with offline processing proc 11 and also experiment 10 recorded data were not available. Run 1539[6] was chosen because it contained a suitable amount of data (approx. $56\,\mathrm{pb}^{-1}$). Run 1539 is one of the runs with the highest integrated luminosity in experiment 8. `basf2` release 03-01-03 was used for online processing on the HLT [34] and release 04-01-00 was used for the offline proc 10 processing. The dimuon selection for this analysis of the HLT efficiency was performed using `basf2` release 04-01-01. The overall HLT efficiency for two track dimuon events is

$$\varepsilon_{\mathrm{HLT},\mu\mu} = 89.4^{+0.175}_{-0.178}\,\%. \tag{5.2}$$

Fig. 5.3 shows the HLT efficiency for the two track dimuon events with respect to the polar angle θ_{lab} and the momentum p. In the polar region from 40° to 120° (barrel region) the HLT efficiency is between 95 % and 100 %, but it never reaches the 100 % mark. However, in the end-cap regions, the HLT efficiency drops below 20 % in the outermost areas.

Additionally, the HLT efficiency for events with a momentum p above 6.5 GeV and below 4.4 GeV decreases. These regions are correlated with the end cab regions of the polar angle variable θ_{lab}. The distributions as functions of the variables used for cuts as well as the trigger efficiency of these variables are shown in Appendix A.3.2.

To investigate the origin of the efficiency drop in the end-cap regions and the deviation from 100 % in the barrel region, the HLT rejected events were examined with respect to the HLT trigger line `selectmumu`. This trigger line is – as the name suggests – designed to select events with two muons. The particle pair is identified as a muon pair if the conditions for the `selectmumu` trigger line listed in Table 5.2 are fulfilled. A track fit result must exist, and the tracks must be oppositely charged. The POCA of the tracks must be closer than 10 cm to the interaction point along the beam axis. Each track has to leave at least 1 hit in the CDC, and each track must contain a transversal momentum in the track fit result. The track with the highest momentum must be over 3 GeV, and the track with the lowest momentum must be over 2.5 GeV. In addition, the azimuthal angle between the two tracks must be larger than 165°, and the deposited energy of one track in the ECL must lie between 0 GeV and 1 GeV.

If all conditions for the `selectmumu` trigger are met, the HLT would select the corresponding event to store it in the storage system. Due to existing prescales for a trigger, only a fraction of events corresponding to the prescale value would eventually be stored. Since no prescaling is activated for `selectmumu`, the HLT must select all events fulfilling the selection criteria. During the analysis of the discarded events and the examination of their properties it was found, that nearly all discarded events fulfilled the conditions of the `selectmumu` trigger line.

[6] /belle/Data/release-04-01-00/DB0000000748/proc10/prod00009638/e0008/4S/r01539/mdst/sub00

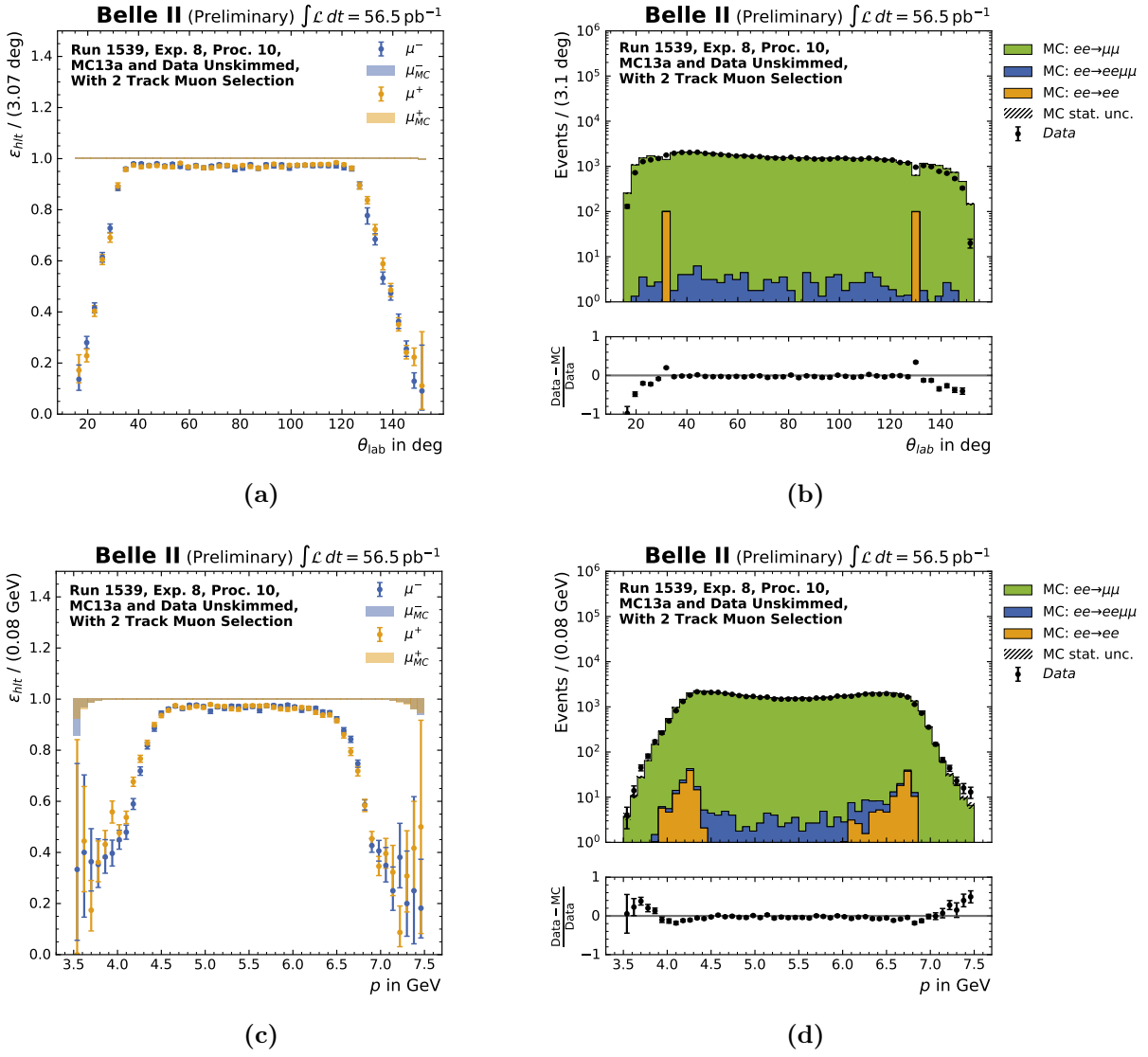


Fig. 5.3.: HLT efficiency of a run in experiment 8 (a) as function of the polar angle θ_{lab} and (c) as function of the momentum p with a significant dependency on the polar angle θ_{lab} and the momentum p distributions. The distribution of the variables are shown in (b) for the polar angle θ_{lab} and in (d) for the momentum.

Table 5.2.: Selection criteria for the selectmumu trigger line of the HLT:

event property	condition in the source code
track fit result	`True`
track charge	`track1=+1 and track2=-1` or vise versa
production vertex in z direction	`z0 < 10`
number of CDC hits for tracks	`nCDCHits > 0`
tracks must contain p_t	`True`
track with highest momentum	`highp > 3`
track with lowest momentum	`lowp > 2.5`
azimuthal angle between muon tracks	`dphi > 165`
deposited energy of tracks in ECL	`(negativeClusterSumLab > 0.` `and negativeClusterSumLab < 1.)` `or (positiveClusterSumLab > 0.` `and positiveClusterSumLab < 1.)`

An investigation of the `basf2` source code, especially the code of the software trigger module (HLT), gave some hints. Several changes were made to the software trigger module of the `basf2` release 03-01-03 used for the online processing to a newer release 03-01-04. An observed problem was that the data type of the variable `m_result` in the class `SoftwareTriggerResult` changed. It changed from `std::map<std::string, int>` to a map with a pair of `int` instead of only one `int`:
`std::map<std::string, std::pair<int, int>>`.
In the former release 03-01-03 only the prescaled HLT decision was stored for each filter line, from release 03-01-04 on both results were stored, the prescaled and the not prescaled HLT decision. Since the data processed offline were processed with release 04-01-00 that uses the new data type for the HLT result, the original information of the not prescaled decision must have been lost because it was not stored on the HLT at the time of the online analysis. However, there were values for the not prescaled decision in the data processed offline available. Consultation with the data processing team revealed that the HLT skims were recalculated offline. However, this should not affect the filter decisions, since skims and filters are calculated separately. At this point it remains unclear where the additional not prescaled decision came from. Probably the second `int` value for the unprescaled decision is just filled with the content of the memory next to the original `std::map<std::string, int>` object.

In the end this problem could not be resolved. Meanwhile, proc 11 was available with experiment 10, and the observed problems with proc 10 and experiment 8 regarding the two-track dimuon HLT efficiency had disappeared. Furthermore, the inefficiencies observed with this dataset were no longer observed in experiment 10 as already described in Section 5.2.

Table 5.3.: Overview of the datasets used to compare the software trigger efficiency of the two track dimuon events. All selection were made with `basf2` release 04-01-01. Additionally for experiment 10 run 5610 a selection was made with the latest master branch available at this time.

Exp./Run	HLT release	Proc	$\mathrm{HLT}_{\mathrm{pass}}$	$\mathrm{HLT}_{\mu\mu}$	$\mathrm{HLT}_{\mathrm{eff}}^{\mathrm{tot}}$
8/1539	03-01-03	9	13 %	0.27 %	96.3 %
8/1539	03-01-03	10	13.1 %	0.3 %	89.4 %
8/1539	03-01-03	11	13.1 %	0.3 %	89.4 %
8/2266	03-01-04	10	17.8 %	0.29 %	98.1 %
10/5610 (4-1-1)	04-00-04	11	33.7 %	0.49 %	99.9 %
10/5610 (master[7])	04-00-04	11	33.7 %	0.49 %	99.9 %

5.4. Conclusion and Summary

To investigate the HLT efficiency for the two track dimuon events, different datasets from different experiments were used. Table 5.3 shows an overview of the different datasets that were investigated.

It is noticeable that experiment 8 run 1539 proc 10 and proc 11 provide approximately 10 % less HLT efficiency for two track dimuon events than the other datasets investigated. The fact that proc 9 of experiment 8 run 1539 also gives a nearly 10 % better value is not understood. In Section 5.3 one could see a strong dependence for the trigger efficiency for the polar angle θ_{lab} with bad values in the end-cap regions.

Problems with events dismissed by the HLT but determined in offline analysis to pass the `selectmumu` trigger line were observed in experiment 8 proc 10. The origin to this issue was not resolved for experiment 8.

In the end, however, it could be shown that with the newer dataset from experiment 10, which was processed online with a much newer software release 04-00-04 and offline in proc 11 with release 04-02-02, the overall HLT efficiency for two track dimuon events is $\varepsilon_{\mathrm{HLT},\mu\mu} = 99.98^{+0.010}_{-0.018}$ % and the influence of the polar angle or the dependence of the HLT efficiency on other event properties disappeared.

[7] `basf2`: commit 8da1206bd4ce13761abc82df5aed3bd5e0ebc0e7

6. The DAQ ELK Setup

In the past, the Belle II experiment suffered from a low data acquisition efficiency (DAQ efficiency), especially in the 2019c autumn run period. A major source was the inconvenient access to the logging and monitoring data for the DAQ and subdetector experts with the existing logging and monitoring system. This repeatedly led to long recovery times for the experts in case of occurring problems. Furthermore, there was no tool to quantify the DAQ efficiency.

Therefore, a new central logging and monitoring system for the DAQ network of the Belle II experiment – the so-called DAQ ELK system – was designed and implemented based on the *Elastic Stack* [35]. A new working group was founded for the implementation in the DAQ network of the Belle II experiment.

This chapter describes the contribution made to the implementation and maintenance of the new system in the scope of this work. The previous logging and monitoring approach, how the individual DAQ applications, services, and machines were monitored, is described in Section 6.1 and it is motivated why a new approach was chosen. Furthermore, the Elastic Stack is described in detail in Section 6.2. An *Elasticsearch* [36] cluster has been set up as a central database and search engine for the log files generated in the DAQ network described in Section 6.3. In Section 6.4 a variety of collected data is presented. Section 6.5 describes the integration of the monitoring of the HLT server farm in the new system.

Basic knowledge in the administration of computer systems, as well as handling the operating system Linux are presupposed in this chapter, since they are not part of this thesis.

6.1. Former Logging and Monitoring Approach at the Belle II DAQ Network

The structure of the previous logging and monitoring architecture is shown schematically in Fig. 6.1. Log messages are generated in different ways. This can be done by any system services, by the `daq_slc` applications themselves, or by other programs required for DAQ operation like NSM. The log messages are stored in text files on the machines on which the programs and services are executed. In addition, the `daq_slc` application produced log messages are additionally transferred to a central PostgreSQL database via the log collector service included in the framework.

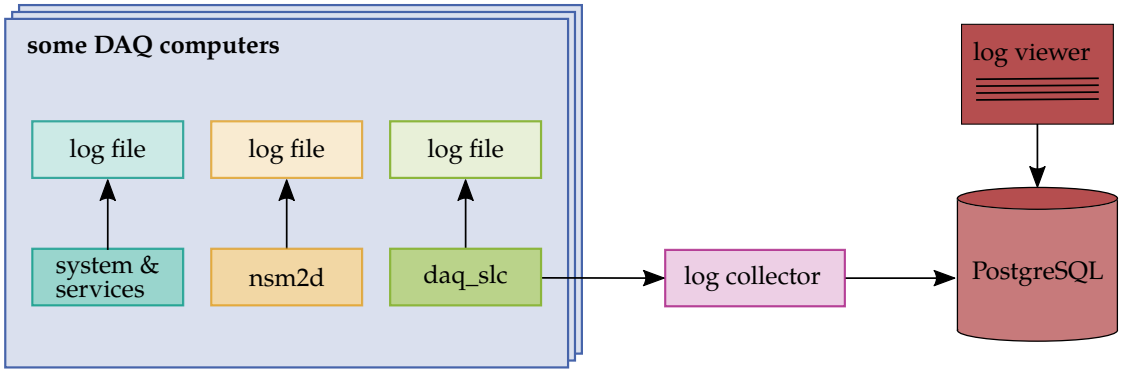


Fig. 6.1.: Schematic overview of the former logging and monitoring setup in the Belle II DAQ network. For the explanation see the text.

Running a single PostgreSQL database was not sufficient for a large scaled monitoring system in the entire DAQ network. PostgreSQL is inherently not scalable, but can be expanded to a multi-node system with increased effort and third-party tools. Furthermore, PostgreSQL is not optimized for full-text searches, which is necessary when searching log messages. Neither a generalized inverted index (GIN) which is used to build search indices for text data nor TimescaleDB [37], a PostgreSQL for time-series data, have been used. Restructuring the former system would also have caused a long downtime. This is another reason why a new system was set up in parallel.

PostgreSQL itself does not provide a front-end user interface (UI). Therefore, a simple "home-made" log viewer was available for easy display of log messages. The log viewer, however, was not able to exclude unwanted message types from the search results. It was not possible to select messages of a certain log level or origin. This is of great disadvantage if the log messages have to be examined for an unusual message before the last detector run was aborted and a flood of unwanted messages have been displayed. Another disadvantage of the log viewer was the lack of any caching mechanisms, which resulted in a strong disk I/O on the database system. A continuous operation of the log viewer resulted in a permanent connection to the database server with a request for the list of log messages every 2 seconds. In addition, a new request was also sent when the old one was not yet finished, so several requests were running at the same time and slowed down the performance. The database server always had a large overhead due to these issues, which resulted in a high load when someone used the log viewer. Log messages were only displayed in tabular form and any visualization was not possible.

Log messages, which are only stored on the individual machines, are only accessible via terminal sessions, e.g. via SSH connections, and must be retrieved individually. This makes it almost impossible to compare log messages between different machines. Furthermore, such queries are very complex and time-consuming.

The comparison of log messages and their evaluation is of great importance in troubleshooting and in finding correlations between errors. In order to improve the performance of the experiment, or rather the data acquisition and its robustness, in the future, it is necessary to monitor and evaluate the components in the DAQ network accordingly. Operating

the SuperKEKB accelerator is very cost-intensive. Therefore, the aim of the DAQ group is to improve the efficiency of data acquisition, which means avoiding downtime due to malfunctions and also reducing the time for error analysis. Early detection of errors by characteristic behavior of the components and error prediction based on observation of trends in the log data is also helpful.

Previously, a central monitoring of the HLT server farm and the adjacent storage nodes was also missing. There was no monitoring of important services such as `hltworkerd`, `filedb` or `storagerd`, which are absolutely necessary for data acquisition. Like other machines in the DAQ network, the `basf2` log output was only written to local log files. The past has also made clear the need to monitor the software versions of `basf2` and `daq_slc`. Frequently, incorrect software versions had caused problems in operation. There is also a lack of monitoring of the hardware such as memory, CPU usage, load etc. A central diagnosis of the HLT server farm has not been possible.

With *Zabbix* [38], a first step towards monitoring the DAQ network had already been taken. Zabbix used the PostgreSQL database to store the measurement data. For example, the event size in bytes sent by the event builder to the HLT is monitored.

6.2. Introduction to the Elastic Stack

The Elastic Stack was formerly called ELK Stack. ELK is composed of the three tools *Elasticsearch*, *Logstash* and *Kibana*. These three tools also form the core of the DAQ ELK system. Actually, there is a fourth tool family called *Beats* provided by Elastic. So, the system should be called ELBK but the Beats tools weren't available when the name was given to the previous ELK Stack. These additional tools are the reason why the ELK Stack was later renamed to Elastic Stack by Elastic. We kept the shorter name *ELK*.

This section describes the key elements of the Elastic Stack used in the DAQ ELK system. They are briefly introduced, with a focus on the work accomplished during this work. It does not claim to be complete.

6.2.1. Why Elastic Stack

In a network with several servers running different processes, each server generates a variety of different log messages. It is difficult to investigate the resulting large amount of log messages and separate log files distributed on each server. The Elastic Stack provides a set of open source tools that are compatible with each other to search and analyze the log messages from many servers at a central point. Since the components of the Elastic Stack are designed to be fully scalable, the monitored infrastructure can grow and the Elastic Stack will not become a "bottle neck."

This is the reason why the Elastic Stack has been well established in the IT sector and industry. Companies like Cisco, Docker, Microsoft, and Ebay rely on the Elastic Stack [39]. Accordingly, there is a strong community that is interested in the further development and maintenance of the Elastic Stack. The success of such open source software is increasing yearly.

A comprehensive documentation of the Elastic Stack tools is available on the Elastic website [35]. Also a Elastic user forum [40] which helps solving problems can be found additional to platforms like Stack Overflow [41] which are a valuable source for help and inspiration. Various blogs exist in the internet dealing with suggestions to prevent the administrators to step into pitfalls.

Beside the Elastic Stack, there are other systems available. Probably the most common is the commercial software *Splunk* [42], which is used e.g. by the ATLAS experiment [43]. Splunk only provides a free license limited by data volume ($500\,\mathrm{MB\,d^{-1}}$). When using Splunk, the costs are increasing with increasing data volume. The Elastic Stack basic license includes most of the required tools and functionalities with unlimited data volume for free. Nevertheless, support and special plug-ins such as warning functions and security packages are available when purchasing extended licenses. However, the tool for warning function, for example, can also be set up by other third-party open source tools. Therefore, the Elastic Stack basic license is sufficient for use in the DAQ network.

6.2.2. Tools of the Elastic Stack

In the following sections, the basic tools forming the core of the DAQ ELK system are introduced. The full documentation of these tools can be found in their official documentations as referred in the respective sections.

6.2.2.1. Elasticsearch

Elasticsearch is a search engine based on the *Apache Lucene* full-text search engine library [44] and acts as a database with internal analysis and aggregation. All inserted data are becoming documents, which are stored serialized in JSON format. This distinguishes this search engine from ordinary databases based on column and row structure. Indexing makes the documents quickly searchable. The indices allow a quick full-text search or all other types of searches as with numerical data. Elasticsearch can also return relevance to the search results when string patterns do not match a hundred percent.

Indices are used to store the information of the sequentially stored data, at which position in the memory they can be found. Using indexes saves the time-consuming task of going through the entire data instead of just the smaller index that points to the correct location in memory where the actual data can be found. This is comparable to an index in a book.

A full-text search engine analyzes the documents during the indexing process and creates an association between the words in an index and the original text stored sequentially on the hard disk. However, the location where the search index is stored is crucial. The access speed increases ascending with HDD, SSD, and RAM. So, if the index is kept in the much faster RAM instead of being loaded from the hard disk each time, the search performance is further increased.

A challenge for the search engine is to find out the key elements for the assignment with respect to the text to be indexed. Therefore, the search engine has to divide the text into so-called tokens and store them together with the reference to the document in the index. As a rudimentary example, the text can be divided using spaces, commas or other

separators. The remaining text fragments are just these tokens. Different combinations of phrase parts occurring in the text are also a possible tokenization.

Elasticsearch uses an inverted index data structure that sorts the index by token. In the indexed state the tokens are called terms. Textbooks use the same structure for indices. Each term is mapped to all documents to which it is related. Within a search process only the terms in the index have to be matched. As soon as the relevant term is found, the list of all relevant documents is returned. [45]

Finally, each word in the text document, is simply listed in the inverted index and the link to the document in which this word occurs is established. Documents consist of a series of fields filled with data by key-value pairs. For each of these fields an index is created in an optimized data structure. Numeric fields for example are indexed in BKD tree structures [46], a k-dimensional tree structure optimized for search efficiency and external memory access.

The detailed Elasticsearch underlying search engine and index algorithms can be found in the Apache Lucene documentation [44].

In the following the basic terminology, required in the context for the DAQ ELK implementation, is introduced.

- A machine (virtual or physical) running a single Elasticsearch service is called a *node*.
- Several Elasticsearch nodes in a network can be connected to form an Elasticsearch *cluster*. The cluster appears to the environment as a single service. This means that it is not visible from the outside which node in the cluster is processing or indexing the data. The cluster handles all this internally. Similarly, sending search requests to the cluster is also processed internally, and the result is sent to the outside world.
- As already mentioned, the documents are stored and indexed in indices. One or more indices can be assigned to an *index alias*. Each index can be divided into one or more *shards*.
- The shards are distributed to several nodes. Besides the so-called *primary shards*, which contain the indexed data, any number of replica shards with the same data can exist. However, only the primary shards are indexed and the *replica shards* are updated accordingly.

Elasticsearch provides a *representational state transfer application programming interface (REST API)* for interaction with the cluster (a single Elasticsearch node in a network is also a single-node cluster). This is a programming interface for distributed systems, which is based on the function of the *World Wide Web (WWW)* and focuses on the communication from machine to machine. A specific protocol implemented according to the REST paradigm is the *HTTP protocol* [47]. This protocol is also used for communication with the Elasticsearch cluster. An HTTP request is sent and a response is received. For example, to request a list of indexes, that the Elasticsearch cluster contains, an HTTP `GET` request can be sent to the cluster:

```
1 GET /_cat/indices
```

The cluster will answer with a list of indices. If the request fails, it will answer with an error. To index documents, for example, HTTP `PUT` or `POST` requests are sent with the respective data to the cluster.

The official documentation of Elasticsearch gives a detailed overview of the REST-API and its usage. On Linux operating systems, the program *curl* [48] is a standard tool for sending HTTP requests. The Kibana interface also provides a convenient developer function to send such requests to the Elasticsearch cluster [49].

Structured queries, full text queries, or a combination of both can be sent to the Elasticsearch cluster via the REST API. A structured query is e.g. `message` and `priority` of log messages from a `host` sorted by `timestamp`. This type of query is also used in SQL. However, full-text queries list all documents that match the search term, sorted by relevance, i.e. how well the search term matches the terms of the index. For example, the documents can be searched for *"hello, world"* and Elasticsearch returns all documents that match the search.

All search functions can be called with the JSON-like query language (Query DSL) of Elasticsearch. This query language also provides access to the aggregation and analysis functions of Elasticsearch. Documents can be counted, and averaged, and maximum and minimum numerical values can be calculated. When inserting log messages from several servers with a field containing the host name, queries can be made regarding the number of log messages of each host. E.g. the host that sends the most messages can be determined using this technique. By interpreting this information, using a time series, any anomalies can be easily detected.

Different types of Elasticsearch nodes can be set up in a cluster:

- **Data only nodes** containing primary and replica shards of indices. They can be used to index and search data.
- **Master nodes** know the state of the cluster and manages the other nodes, but only one node can be an active master node in the cluster. However, there can be several master eligible nodes which will participate in the election of a new master among themselves in case the active master node fails. Depending on the settings, the master node may or may not contain data. A stable master node is important for the stability of the cluster.
- **Coordinating nodes** are neither data nodes nor master nodes. Actually, every node is a coordinating node. If a node is neither a master node nor a data or ingest node, it is only a coordinating node. They can act as smart load balancers in a cluster, distributing search requests to relevant nodes and receiving and merging the results. Also indexing requests can be processed by this node type.
- **Ingest nodes** can be used for pre-processing data before indexing the documents in the cluster.

Adding multiple nodes to a cluster has several advantages. First, the capacity of the cluster increases and data can be distributed redundantly across multiple nodes, so that if one node fails, the lost data can be recovered by the other nodes that have a copy of the

"lost" data. Second, the additional nodes can also process search and index queries. This allows parallel processing of search queries and data indexing. The cluster performance increases. When nodes join or leave the cluster, Elasticsearch manages in the background a migration of the shards in the cluster to balance the load on the individual nodes. It also ensures that the appropriate replica shards are available. The recommendation of Elastic is: "the more nodes, the merrier."

The shard size and the number of primary shards of an index, however, strongly depends on performance issues and trade-offs. Basically, the more shards, the more resources must be spent on maintaining these indices. On the other hand, the larger the shards are, the more time is needed to migrate them to other nodes when rebalancing the cluster. With more small shards, the processing of requests per shard becomes faster, but more requests have to be made, which may finally cause fewer and therefore larger shards to be requested faster.

Elastic offers on its documentation for Elasticsearch two references for orientation on shard number and shard size. The average shard size should be between a few GB up to a few dozen GB. Especially, for time-based data, shard sizes from 20 GB to 40 GB are common. Furthermore, the number of shards on a node should not exceed 20 shards per 1 GB of available heap space.

An Elasticsearch cluster mandatorily requires a master node that manages the cluster state and transmits it to the other nodes in the cluster. The cluster state contains all information about all nodes in the cluster, all shards, indices, their mappings, templates and other metadata. Calculating and processing the cluster state can generate a large overhead. Joining and leaving nodes in the cluster is handled by the master node. The master node is elected by all nodes in the cluster that are configured to vote. When setting up a cluster with multiple nodes, the master nodes may face a problem called a *split-brain problem*.

Split Brain Problem
This problem can occur when a certain number of master eligible nodes lose the connection to the rest of the cluster and elect a new master among themselves. When the connection to the cluster is restored, two master nodes are in the cluster at the same time. This leads to an inconsistent cluster state, which can cause significant problems and data loss cannot be excluded. To prevent this, the following points should be considered:

- Recommended is an odd number of master eligible nodes.
- Set the parameter which determines how many nodes must be in communication to elect a master node. The rule is $N/2 + 1$, where N is the total number of master eligible nodes in the cluster. In the recommended case of an odd number of nodes, $N/2$ is rounded down as an integer.
- Additionally the timeout parameter when the connection to a node is failed can be increased.

Since a cluster with two nodes will fail to index new data and the cluster will lose its high availability, the above recommendations require at least three master eligible nodes to avoid the split brain situation.

The complete Elasticsearch documentation can be found in [36].

6.2.2.2. Logstash

Indexing of data in Elasticsearch can be done either by manually sending an HTTP request for indexing with the data or by using some data senders like the Elastic Beats family. If the data comes from several different sources without a consistent data structure, a separate filter pipeline such as Logstash may be required to normalize the data before indexing.

Logstash is a event processing pipeline that consists of three stages. The input stage is, where the data enters the pipeline and an event is created. Possible inputs are via a UDP port or from a Beats data sender. There is no rule in which format the data has to be available. The event is built from fields that contain the data. From the *input stage* the event is passed on to the *filter stage*. A useful filter is the so called *Grok filter*. It parses plain text from a field and recognizes patterns. This way additional fields can be created and filled with the parsed data and an unstructured text from a field can be brought into a uniformly structured and queryable form. Additionally, the fields can be manipulated by the *mutate filter*. Several filters can be applied to an event one after the other. The final stage after the filter stage is the . Here one or more outputs can be defined, where and in which format the data should be sent. For example the Elasticsearch output, which uses the convenient bulk index request to send a series of events at once.

The complete Logstash documentation can be found in [50].

6.2.2.3. Kibana

Kibana is also described as a window into the Elastic Stack. Without Kibana, the Elasticsearch cluster is a black box that responds to HTTP requests. With the Kibana front end, which is based on a web interface, the data can be analyzed and visualized. In addition, an interface for the administration of the Elasticsearch cluster is provided. Using Kibana, complicated queries can be generated automatically and sent to Elasticsearch. Dashboards can be created that display the analyzed data in clear diagrams or in tabular form at any time. The interactive selection of time periods allows a detailed analysis of the sequence of events that generate the log messages.

Kibana also provides easy access to the index lifecycle management service of Elasticsearch. Different lifecycle policies can be managed and applied to indices. Depending on conditions such as index size or age, the index can be put into an appropriate phase to consume less hardware resources, or it can be deleted. Defining phases help to keep the cluster performant over time.

For developers, a console is available that communicates with the cluster via manually created HTTP requests and provides auto-complete functionality, which is extremely useful. The console acts as a kind of notebook. Previous commands do not disappear.

End users can be provided predefined user areas and predefined dashboards with Kibana. This allows users to get special visualized tools for monitoring log files, processes and whole host systems. In Section 6.5.4 this is explained with an example as it was used in this thesis.

The complete Kibana documentation can be found in [49].

6.2.2.4. Beats

Two Beats tools are especially relevant for this work, Filebeat and Metricbeat, which are briefly introduced in this section. Both tools are data loaders that collect data from different sources and send it to a defined output, which can be either Elasticsearch or Logstash.

Filebeat is a data shipper for log files. Each log file or location to be monitored is specified as input and a so-called *harvester* is started for each individual file. The harvesters go through the contents of the log files and send the log messages to a central point where they are aggregated and sent to the configured output. When the harvester is running, it reads the log file line by line. The harvester has the advantage that it remembers which data it has already sent. This also holds true when the service is restarted. In addition, Filebeat ensures that each event is delivered to the output at least once. Filebeat waits therefore for each message to be confirmed by the output.

Metricbeat periodically collects metrics of the operating system and the services running on it and delivers them to a specific output. It allows monitoring of the services and the system. To define which metrics should be collected, appropriate modules are selected which contain the relevant information for retrieving the metrics. One module can contain several sets of metrics. For example, the system module provides access to the system metrics on which the Metricbeat service runs. It contains *metricsets* like `cpu`, `load`, `memory`, and further more. Each metricset contains a list of metrics. E.g. the `cpu` metric set contains several CPU statistics like `idle`, `irq`, `user`, `system`, `iowait`, `softirq`, `cores`, `nice`, `steal`, and `total`.

A full documentation of these tools can be found in [51] and [52].

6.3. Setup of a Central Elasticsearch Cluster

As a central database and search engine in the new DAQ ELK system, an Elasticsearch cluster has been set up, which is distributed over various virtual machines (VMs). Fig. 6.2 shows schematically the structure of the DAQ ELK Elasticsearch cluster[1]. The cluster runs on its own physical server on which the VMs are hosted. Each individual Elasticsearch node runs in its own VM. The cluster consists of a total of four Elasticsearch nodes. Among them is one coordinating node and three master eligible data nodes.

The coordinating node is responsible for forwarding search requests to the corresponding data nodes and for merging the results and finally sending it back to the requester. Kibana is connected to this node as the main interface to the Elasticsearch cluster for the user. The coordinating node acts as a smart load balancer and does not contain any data and is not master eligible. However, the three other Elasticsearch nodes are data nodes that are master eligible. I.e. one of these data nodes is selected as master node and another one as deputy. The cluster is designed in such a way that for each shard located on one node, a replica shard is located on another node. To avoid the danger of the so-called *split brain*, as described in Section 6.2.2.1, three master eligible data nodes were assigned to the cluster. Data is fed into the Elasticsearch cluster via HTTP requests and the Elasticsearch REST API and then sent to one of the three data nodes and indexed into the cluster from there.

[1] Current setup in October 2020.

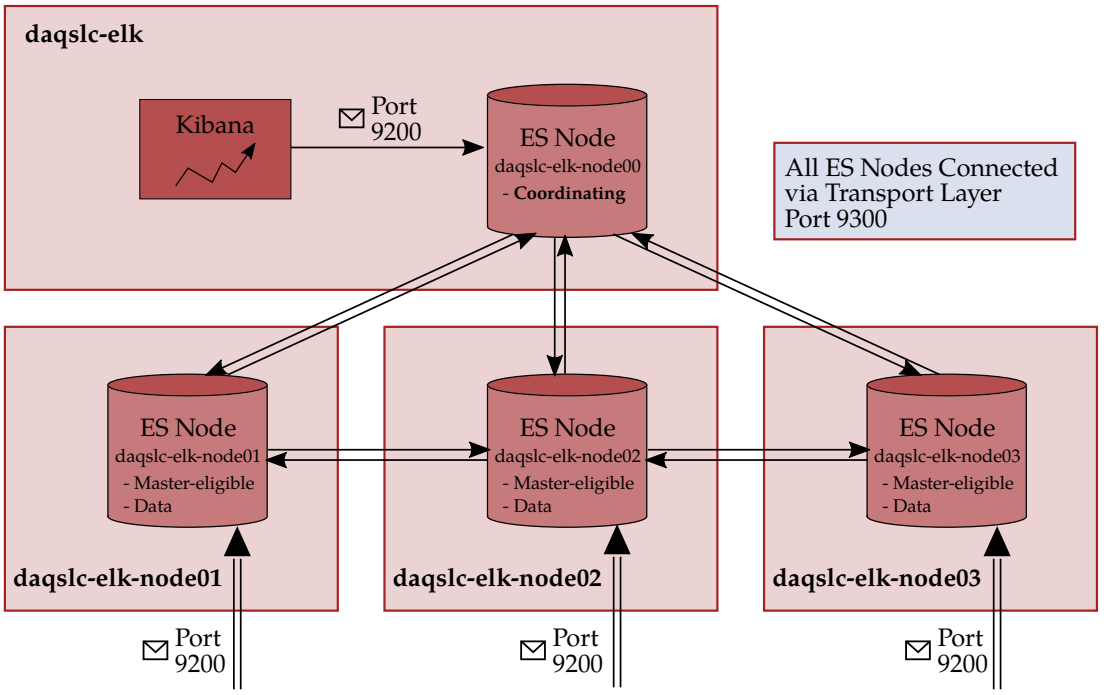


Fig. 6.2.: Four Elasticsearch nodes (ES Nodes) running in VMs forming an Elasticsearch cluster. Three ES nodes (running on the `daqslc-elk-nodexy` VMS) are data and master eligible nodes and one node is a coordinating node (running on the `daqslc-elk` VM). The Kibana interface accesses the Elasticsearch cluster via the coordinating node. Via the Transport layer is cluster managed internally. Data to be indexed in the cluster is sent to the data nodes.

The nodes in the cluster itself communicate with each other via their own transport layer. When new nodes are added to or removed from the cluster, this transport layer is used to address the new node and/or propagate the cluster state through the master node to all nodes accordingly. The transport layer is decoupled from the input and search request interfaces.

Initially, a cluster with the Elasticsearch version number 7.5 was set up. During the 2020 summer shutdown the Elasticsearch cluster was backed up via a snapshot and updated to the newer version 7.8.1. The cluster update was carried out during operation and the cluster was accessible and fully functional at all times. Finally, both the successful update and the snapshot confirmed that the chosen cluster concept works.

6.4. Collecting Log Files, Metrics, and Other Information

Elasticsearch gets its data on different channels from several sources. In this section some of the sources and collection methods are explained. Log messages and measurements related to the HLT are described in the next section.

6.4.1. Log Collector

The log collector is a service of the `daq_slc` framework and runs on a central server in the DAQ network. All log messages from the DAQ apps of the `daq_slc` framework are sent to the log collector via NSM. The log collector distributes the log messages further according to its configuration. It can be used to forward the log messages via UDP to any endpoint. The log collector had been configured to sent the output via UDP to the Logstash service, running on the `daqslc-elk` VM of the Elasticsearch cluster. Logstash formats the log messages for optimized search capabilities and inserts the data into the Elasticsearch cluster.

Log messages should include fields containing a timestamp, log level (message priority), source, and message. In the case of the log collector the source is divided into two sections, a category and a node. These result from the names of the NSM nodes from where the log message was sent. Each category and node also has an identification number (id). For the analysis of the log messages it is useful to have the node and the category information. E.g. the category DAQ contains all HLT units and workers. However, due to a bug in `daq_slc`, which could not easily be fixed due to too many dependencies, only the node name of the log message is set correctly and the category id and category name are empty. To make this information available for later analysis, a Logstash filter is used to determine the category id and category name from a known list of node names mapped to their categories. This works because the category is superior to the nodes. Here, the Logstash `translate` filter plugin is used, which obtains values from a key value pair in file formats like e.g. JSON. The output of the Logstash pipeline is directed to the Elasticsearch cluster where the messages are indexed accordingly to the modified fields.

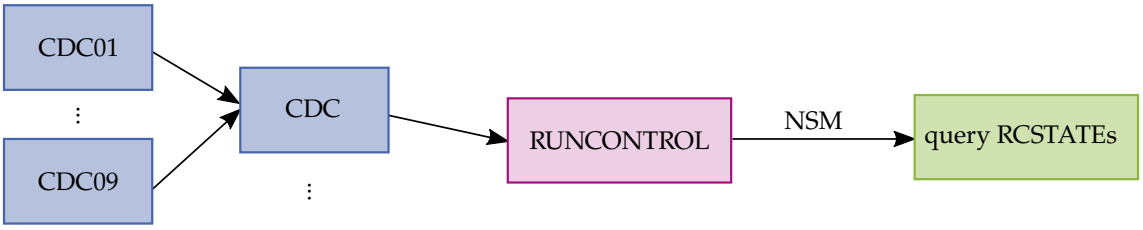


Fig. 6.3.: Querying the NSM `RUNCONTROL` node for the nodes it knows for its run control states (RCSTATEs). For example, the `CDC` NSM node is such a known node. So the `CDC` node is also queried for its known NSM nodes and their RCSTATEs. Iterating over all such NSM nodes results in a two level tree of subdetector RCSTATEs.

6.4.2. NSM Variables

The entire communication for detector operation with the `daq_slc` system runs via the NSM run control network. To make the relevant NSM variables accessible for Elasticsearch, the NSM network must be accessed with an NSM node. Therefore, `nsmd2` runs on the `daqslc-elk` VM of the Elasticsearch cluster and is connected to the run control network. The Python package `pynsm2` by M. Remnev [53] provides a convenient interface for interaction with NSM using Python. With the Python Elasticsearch client interface [54] the data can be indexed just as conveniently in the Elasticsearch cluster.

Via this interface to the run control NSM variables, e.g. the run control states of the subdetectors can be collected and transferred to Elasticsearch for later error analysis. For this purpose a separate NSM node was added to the network. The run control states of all NSM nodes in this network running under the master node `RUNCONTROL` are requested to send all NSM nodes known to it and their run control states (the state of `RUNCONTROL` is requested as well). These nodes are also queried for their run control states and additional for the nodes known to them. So the script iterates through a tree with two levels below the `RUNCONTROL` node to query the corresponding run control status information. Fig. 6.3 schematically shows an illustration of the iteration for the run control queries for a single branch of the `RUNCONTROL` node.

6.4.3. Monitoring the Metric of Elasticsearch VMs

Metricbeat was used on the virtual machines to monitor the overhead and the general operation of the virtual machines hosting the Elasticsearch cluster. The host server running the virtual machines was also monitored. Since Metricbeat also indexes the measured data in the Elasticsearch cluster, Kibana can also be used to analyze and visualize the performance of the hardware. During the setup and expansion of the DAQ ELK system, values such as memory, disk space, IO performance, and CPU usage were constantly monitored and consulted for high-performance operation of the Elasticsearch cluster.

6.4.4. Zabbix Data

Zabbix is another tool that runs on several machines in the DAQ network, mainly for monitoring network traffic. Since the data collected with Zabbix should also be available

for analysis in Kibana, it is also transferred to the Elasticsearch cluster. However, Zabbix does not support a native connection to Elasticsearch. Therefore, the data is exported to a JSON file. The JSON file is tracked by a Filebeat service which ships the data finally to the Elasticsearch cluster.

Especially, for monitoring the HLT server farm, described in Section 6.5, Zabbix is relevant in the scope of this thesis.

6.4.5. Interface to the PXD Log Messages

The slow control of the PXD is separated from the slow control network of the other subdetectors. Therefore, up to now the log messages generated in the PXD slow control were not distributed to the log collector of `daq_slc`. The PXD is managed by *EPICS* [7] in combination with *CSS* [55]. Meaning, they do not use NSM to distribute variables and send specific requests to their machines. This is handled by EPICS, an open source software tool for operating and controlling devices in physical experiments where hundreds of machines are networked together. CSS is the front end for the EPICS network, where process variables distributed over the EPICS network are visualized and users can interact with the hosts on the network via control panels.

Since NSM was not available on the PXD network, the connection to the `daq_slc` log collector was missing. Instead, they used a *STOMP-capable C++ log file library with EPICS integration* [56] to index their log messages in an internal Elasticsearch data base. This Elasticsearch data base is not accessible to the DAQ network.

In order to collect PXD log and alarm messages via the DAQ ELK system, the PXD group recently provided a gateway to their logging system using *Apache Camel* [57] and the *Spring* framework [58, 59]. These tools are running in a *Docker* container [60] on the `daqslc-elk` VM of the Elasticsearch cluster [61]. Fig. 6.4 shows schematically this setup. The PXD log messages are collected and indexed in the DAQ ELK Elasticsearch cluster in an PXD-specific index. This allows the evaluation of the log messages collected with the `daq_slc` log collector along with the log messages of the PXD network. Troubleshooting and error detection can thus be improved.

6.4.6. Monitoring the Cluster Disk Usage

In order to guarantee a high-performance operation of the Elasticsearch cluster, the disk usage for individual indices was evaluated via the Python Elasticsearch client interface [54]. Here the DAQ ELK related indices are requested by the coordinating node and the Elasticsearch cluster answers the request with a list of the indices together with the size on the disk with and without the replication shards. This made it possible to determine the exact storage requirements of each index and to track their developments. The server on which the VMs of the Elasticsearch cluster are hosted provides each data node disk space of approximately 3 TB, which means a total disk space of approximately 9 TB for the data nodes. Fig. 6.5 shows the development of disk usage from March 2020 to July 2020, when the DAQ ELK system ran in test mode and accompanied the beam run period 2020a and 2020b.

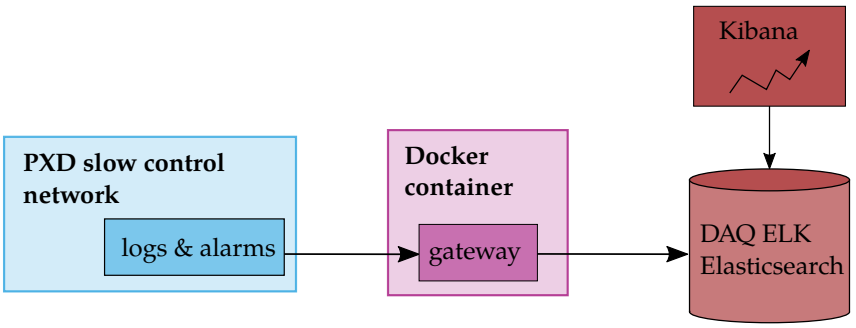


Fig. 6.4.: Schematic overview of the gateway in the PXD slow control network to distribute PXD log messages to the DAQ ELK system.

It is noticeable that in the earlier phase, the Zabbix indices (`zabbix*`) dominated with almost 80 %, and that due to the lack of any index lifecycle management, it continuously increased by several GB per day. After applying such an index lifecycle management to the Zabbix index, the rapid rise of the index was halted. As the development of the disk usage in July 2020 shows, it has even decreased in absolute numbers over the 5 months by about 100 GB, although new monitoring tasks have been added. The index lifecycle management ensures that the data is deleted after a certain time. This allows the size to be kept almost constant.

In February 2020, as part of the development of the HLT monitoring system, all worker hosts of HLT unit 7 were monitored with Metricbeat (`b2daq-hltmetricbeat*`), even without a corresponding index lifecycle management. Thus, within one month about 100 GB of measured values were collected by Metricbeat. For the entire HLT server farm this would mean a data volume of about 1 TB per month. It is obvious that a corresponding system monitoring of all HLT units is only possible with index lifecycle management. Here a solution like for the Zabbix data is already being worked on. The data is simply deleted after a certain time.

Other storage intensive indices are those of the log files of the `hltworkerd` processes (`b2daq-hlt-hltworkerd*`). Considering the log messages of the HLT units[2] 2-4 and 6-9 indexed in the Elasticsearch cluster from April 29th to July 3rd (178 GB for 7 units in 52 days), corresponds to an increase of disk usage of $\sim 0.5\,\mathrm{GB\,d^{-1}\,unit^{-1}}$.

The indices of log messages from the HLT and log collector should be stored permanently. A possibility for the future is to obtain further hardware, where old indices can be migrated by the index lifecycle management and can be accessed if necessary.

6.5. Monitoring the HLT Server Farm

Recording data from the detector is only possible if the HLT server farm works reliably. Since a central monitoring of the hardware, processes, and log messages did not previously

[2] HLT unit 1 and 5 were not able to run the Elastic Stack tools because of a corrupt JAVA installation. This was fixed with the HLT operating system upgrade on CentOS 7 during the 2020 summer shutdown.

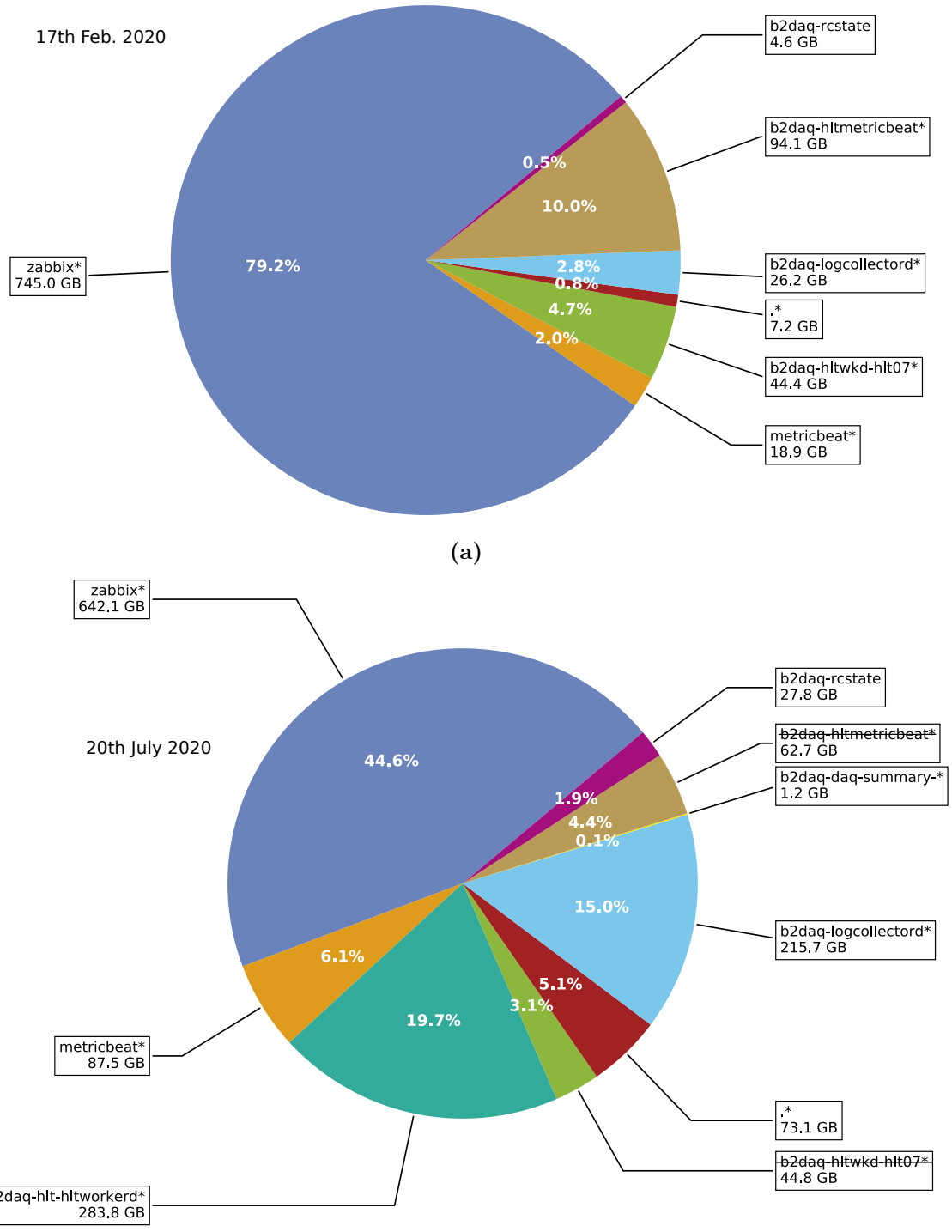


Fig. 6.5.: Disk usage of the Elasticsearch indices. The upper pie chart shows an earlier state in February 2020 with a total disk usage of $\sim$ 940 GB and the lower pie chart shows the state after the end of the physics runs in July 2020 with a total disk usage of $\sim$ 1440 GB. The crossed out indices are not longer used but still occupied some disk space.

exist, the DAQ ELK system was also designed for these tasks. This section covers the setup of the DAQ ELK system on the HLT. The functionality of the HLT was already described in Section 2.3.2.

6.5.1. Deployment and Configuration

With the installation of the DAQ ELK system on the HLT, a modern approach to orchestration was adopted. Previously, a system to manage the HLT didn't exist. Orchestration is useful when complex systems, consisting of several servers, have to be managed, configured, and maintained in a uniform way. Centrally managed systems are less prone to human errors. The HLT server farm consists of about 200 individual hosts. Therefore, it is much more practical to define a central state in which the whole system has to be and then distribute it to all machines. Once the framework is in place, the maintenance of the system is very convenient.

For the distribution and configuration of the DAQ ELK system on the HLT, the open source tool *Ansible* [62] is used. Ansible is the tool of choice because it is clientless and operates via SSH. Once a state is deployed to the clients, it is still possible to configure manually on the client systems. Especially, for test purposes in permanently changing environments, this is very useful. Proper states can easily be restored by another deployment of the playbook. Ansible is developed by RedHat which also develops the RedHat Enterprise Linux (RHEL) on which CentOS 7, running on the HLT units, is built on top, which means optimal compatibility for this tool.

The so-called playbooks are applied to the target computers from a central node. These playbooks define how the target system is supposed to look. During distribution, Ansible takes care of the implementation via SSH. Finally, a feedback is given whether the process was successful or not. Fig. 6.6 shows schematically the deployment and configuration on the HLT. From the control node on the `daqscl-elk` VM, the complete HLT ELK setup was deployed to the HLT units and the corresponding HLT hosts. To reach the HLT hosts which are not `hltctl` a jump host must be defined. Only `hltctl` is reachable from the outside by the name of the HLT unit i.e. `hlt01`, `hlt02`, etc. The other hosts (`hltwkxy`, `hltin`, `hltout`) are only via `hltctl` reachable.

Once all services were installed and configured on the HLT units, Ansible is also used to start and stop these services. No manually log in to each host and and start/stop the services via the command line is needed. Since the entire DAQ ELK system was implemented on the HLT to prevent manual login and search, this approach is continued with the orchestration.

6.5.2. Collecting and Shipping the Log Messages

Primarily relevant log messages on the HLT are those of the HLT distributor (`distributord`), worker (`hltworkerd`), and collector daemons (`finalcollectord`), as well as those of the `nsm2` daemons and the `roisender`. A quick overview of the relevant applications is listed in Table 6.1.

Filebeat takes over the collection of the corresponding log files on the corresponding HLT unit and sends them to the Logstash pipeline running on the same HLT unit. The Logstash

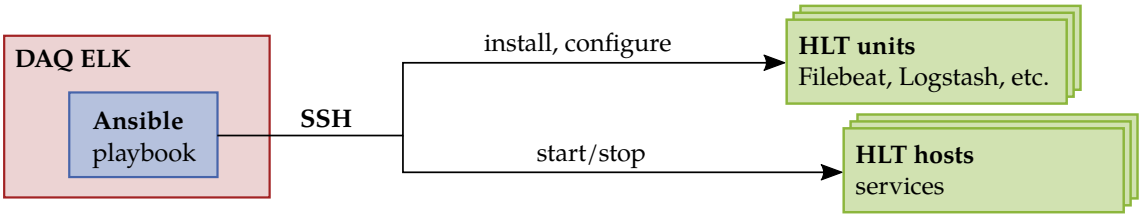


Fig. 6.6.: Deployment of the DAQ ELK related services using Ansible. The Ansible application is only installed on a DAQ ELK host and deploys the client states (HLT units and hosts) through a SSH tunnel.

Table 6.1.: List of services on the HLT whose log messages are collected and sent to the Elasticsearch cluster. [63] [64]

`daq_slc` app	task
`distributord`	Gets data from event builder 1 and sends it to the workers.
`hltworkerd`	Unpack and reconstruct the events used for filtering and data quality management.
`finalcollectord`	Receives the event data reconstructed by the workers. Sends the filtered data further to the storage. (PXD excluded)
`finalcollectord_with_roi`	Receives the event data reconstructed by the workers. Sends the filtered data further to the storage. (PXD included)
`nsm2d`	Provides the interface to the NSM network.
`roisenderd`	Collects ROIs from the HLT and sends them to the ONSEN which requests the data from PXD.

pipelines of the respective HLT units send all data to the central Elasticsearch Cluster to index the documents according to their original application that generated the log message. In the Logstash pipeline, filters are applied to add the information of the original host on which the log message was generated, which is retrieved from the HLT unit's host name. In case of `hltworkerd` log messages, the log file name containing the worker number is parsed. This allows separation between the HLT units and the HLT workers of each unit during the later analysis of the log messages. Possible corrupt HLT workers or even HLT units can be identified in this way.

6.5.3. Process Monitoring and Monitoring of Software Releases

Zabbix natively supports a monitoring tool for process existence on host systems. From a central Zabbix server, which provides a template system for configuration and monitoring natively supported functionalities, the list of processes to be monitored for their existence can be easily transferred to the respective HLT hosts. New processes can be added to and removed from the watch list at any time.

In principle it is also possible to query running processes with the Elastic tool Metricbeat, but this has the disadvantage that in case of a not running process no feedback is given by that tool, i.e. it is not explicitly clear whether the monitoring process is not running or the monitored process is not running. So it is not possible to actively indicate that a monitored process is not running. Zabbix, however, gives feedback even if the monitored process is not running. For later performance analysis this is an advantage because the statistics can be built from the negative feedbacks as well.

The positive and negative feedbacks are sent to the Elasticsearch cluster every 60 seconds and are indexed together with the host information. With the help of the Kibana visualization tools and the dashboards, a central process monitoring for the HLT was created. Fig. 6.7 shows an example of the process monitoring visualized in a heat map. It is clearly visible when which process was not running. In case of a recent problem, it is directly observable which processes are affected.

In the past, problems have been caused by using wrong software releases. For this reason the softlinks to the releases of the Belle II `externals`, Belle II `tools` from the Belle II Software repository [65], and `daq_slc` are monitored using Zabbix. With an appropriate visualization in Kibana this allows for a simple central control of the relevant software versions used on each single HLT unit.

6.5.4. Visualization and Analysis

A series of dashboards were set up using the Kibana interface. All the above mentioned information were visualized for the DAQ and HLT experts. Fig. 6.8 shows an organigram of the dedicated HLT dashboards. Two separate main scopes exist: One is the scope for the DAQ shifters with a brief overview of necessary daemons which have to be running on the HLT during the data taking process. If daemons died this is visualized here and the shifter can execute the necessary recovery applications. This is optimized to minimize the downtime during data taking. The other scope is created for the HLT experts providing more detailed information. Several subscopes are provided for different problematic issues.

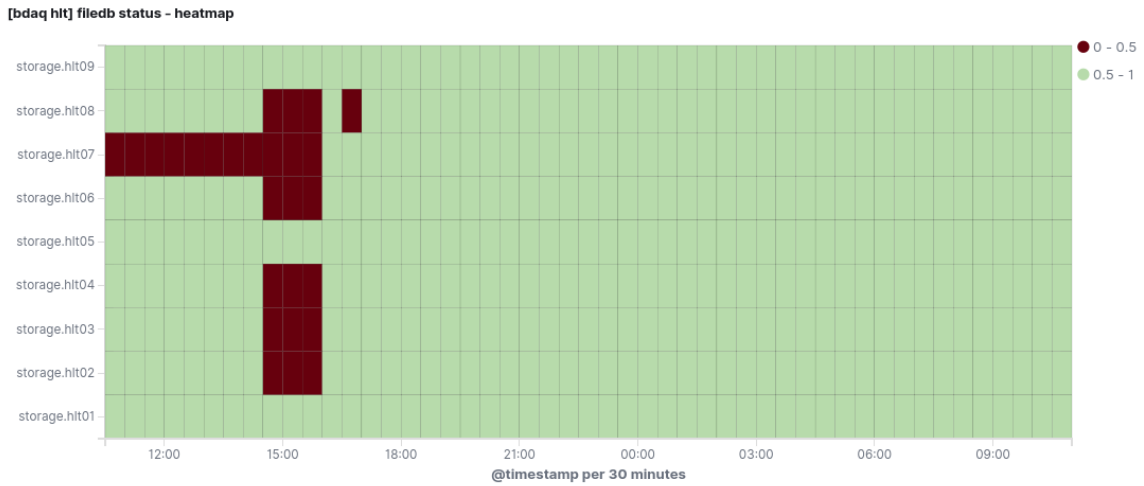


Fig. 6.7.: Example of the process monitoring on the HLT. Here, the `filedb` process is monitored using Zabbix. It checks every 60 seconds if the corresponding process is running or not and writes as result a 1 or 0 in the dedicated field of the Elasticsearch index. This field is evaluated with Kibana in a heat map graphic as illustrated.

Critical unpacker errors are displayed and the occurrence on the respective host is visualized. Messages from the log collector of the run control network are displayed and can be easily inspected and filtered. For each HLT unit, the software release symbolic link targets are listed that point to the releases in use. Occurrence of `hltworkerd` log messages and their distribution according to the respective HLT units as well as a table with filtered current `ERROR` and `FATAL` log messages are presented. The NSM2 subscope visualizes the status of the `nsm2d` processes for the nodes connected to the run control network and the nodes connected to the internal HLT network. Additionally, the log messages of the NSM daemons are listed and can be filtered specifically by HLT unit and/or HLT host.

6.6. In Action - Resolving PXD Readout Error

During the start of the DAQ ELK project, an unsolved problem was observed several times with respect to the PXD. During data acquisition, the PXD got stuck in error mode due to overriding the internal event buffer in the PXD readout boards (*ONSEN*). It was not clear whether this problem was due to a malfunction in the PXD system or of the HLT, which is responsible for triggering the readout of the PXD ROIs via the ROISender (`roisenderd`). Therefore, the `roisenderd` log output was analyzed in Kibana. The `roisenderd` log output contains messages with the number of events currently in the internal buffer of the ONSEN waiting for an HLT decision. If the HLT accepts an event, the PXD ROI data is sent to event builder 2 and the event is removed from the buffer, otherwise it is only removed. So in normal operation the number of these missing events should fluctuate around a constant value.

Fig. 6.9 illustrates the number of missing events from the `roisenderd` log output. It can be clearly seen that the number of missing events increased linearly with time. After this

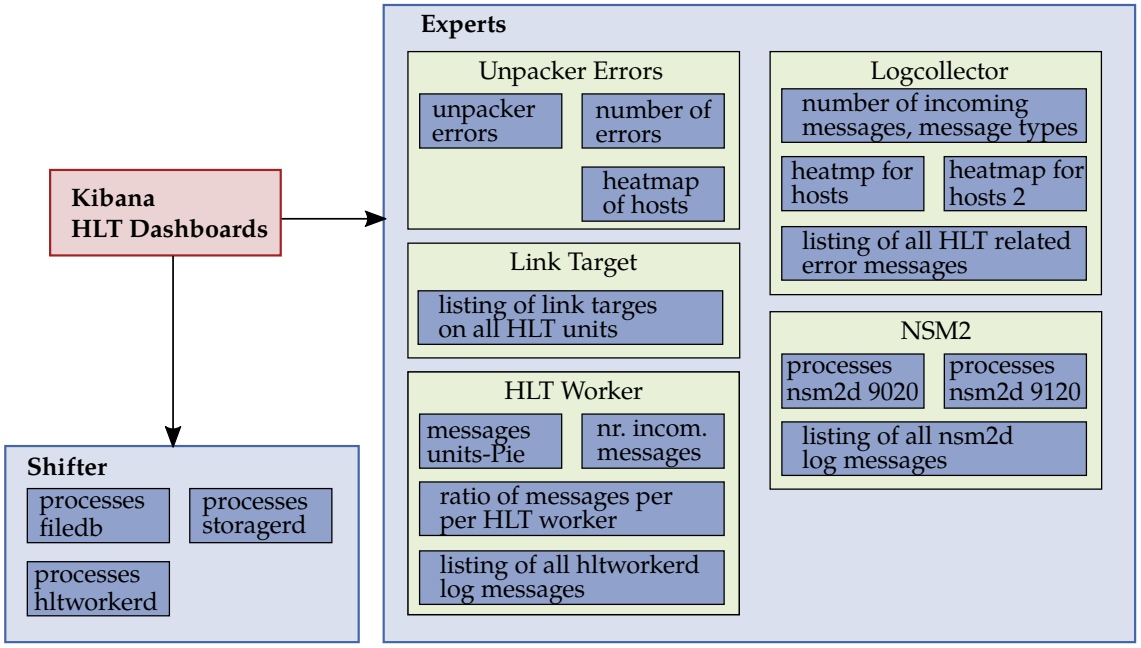


Fig. 6.8.: Schematic overview of the HLT dashboards in Kibana. Two separate scopes exist. The shifter scope, containing a short overview of crucial processes running and a detailed expert scope. See the explanation in the text.

observation it was clear that the HLT silently dropped some events during its processing, which led to the buffer override in the ONSEN, as mentioned above. Already in the early stages of setting up the DAQ ELK system, the power was demonstrated and it showed that the setup of such a system was long overdue. With this visualization in Kibana, the loss of events can be quickly determined by checking this plot.

6.7. Backup System and Lifecycle Management

As already mentioned, the DAQ ELK system ran in test mode during the beam run from March 2020 to July 2020. Despite the very good results, the system still has one major vulnerability. There exists no backup system in case of hardware failure of the server running the virtual machines of the Elasticsearch cluster and most of the other DAQ ELK services as described in the last sections. This point was discussed in the course of the 2020 summer shutdown. A suitable solution is currently being searched for.

Fig. 6.10 shows one of the possible concepts that is currently being considered. Distributing the corresponding data and master eligible nodes to three physically separate servers would still ensure that the system would be up and running in case of a failure of one of the physical hosts. Even the data can still be duplicated in the cluster after the data was recovered from the replica shards. If the physical server 1 fails, the coordinating node would have disappeared, but this is not essential. Such a node can either be started spontaneously on one of the two remaining hosts or it can be left out entirely. The recommendation of the Elasticsearch documentation is not to have too many coordinating nodes in the cluster.

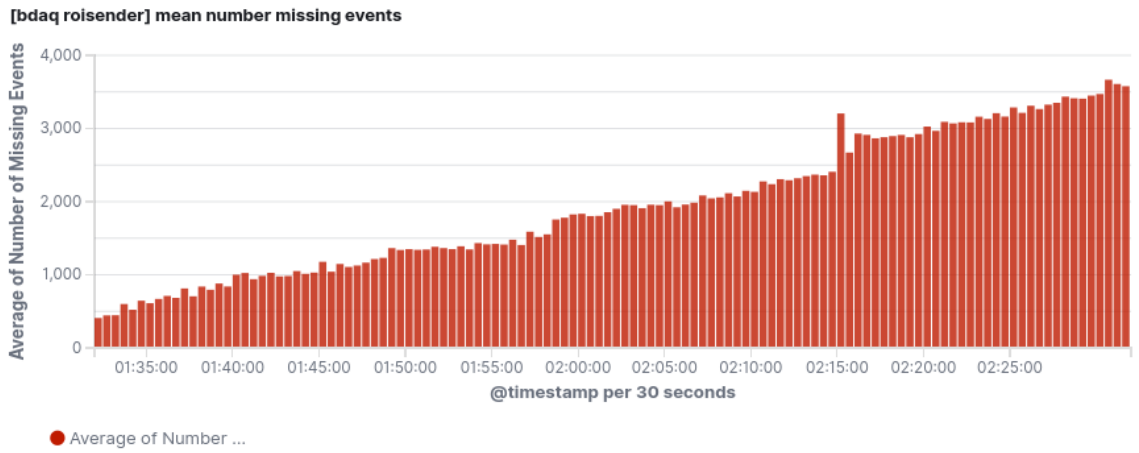


Fig. 6.9.: Analyzed ROISender log output using a Kibana visualization tool.

The Kibana service can also be hosted on one of the other servers and access the cluster from there. Nevertheless, the cluster would all the time be available for data indexing.

Furthermore, as depicted in Fig. 6.10, the data nodes are divided into hot and cold nodes. Elasticsearch allows the division of the active indices into three phases: hot, warm, cold. These phases can be used to define how much and which resources are available to the indices of the respective phase. Indices of a colder phase can also be shrinked. All this can be defined by the index lifecycle management of Elasticsearch.

In addition to reliability, the additional hardware and the additional nodes also bring better performance, as additional replica shards can also be used to process search requests. Three independent master eligible nodes would also relieve the data nodes, which, in addition to indexing and query processing, do not have to be additionally burdened with the load of the master node. Only one of the master eligible nodes is actually elected as master, and one is elected as its deputy.

6.8. Conclusion and Outlook

Within the scope of this work, a major contribution was made to setting up a new system that allows central monitoring of log messages and system parameters in the Belle II DAQ network. The DAQ ELK system is based on tools provided by the Elastic Stack, an open source system established in the IT industry used for e.g. monitoring data centers. That such a system was long overdue for the operation of the Belle II detector was demonstrated by its rapid success and immediate roll-out into test operation on the DAQ network and in the Belle II control room. Once the fail-safety problem has been solved, the project will officially go into production mode.

During the physics run from March to July 2020 the system was tested in operation and continuously adapted to the needs and performance of the system. Fig. 6.11 shows a schematic overview of the setup of the DAQ ELK system in autumn 2020. The individual components were explained separately in this chapter.

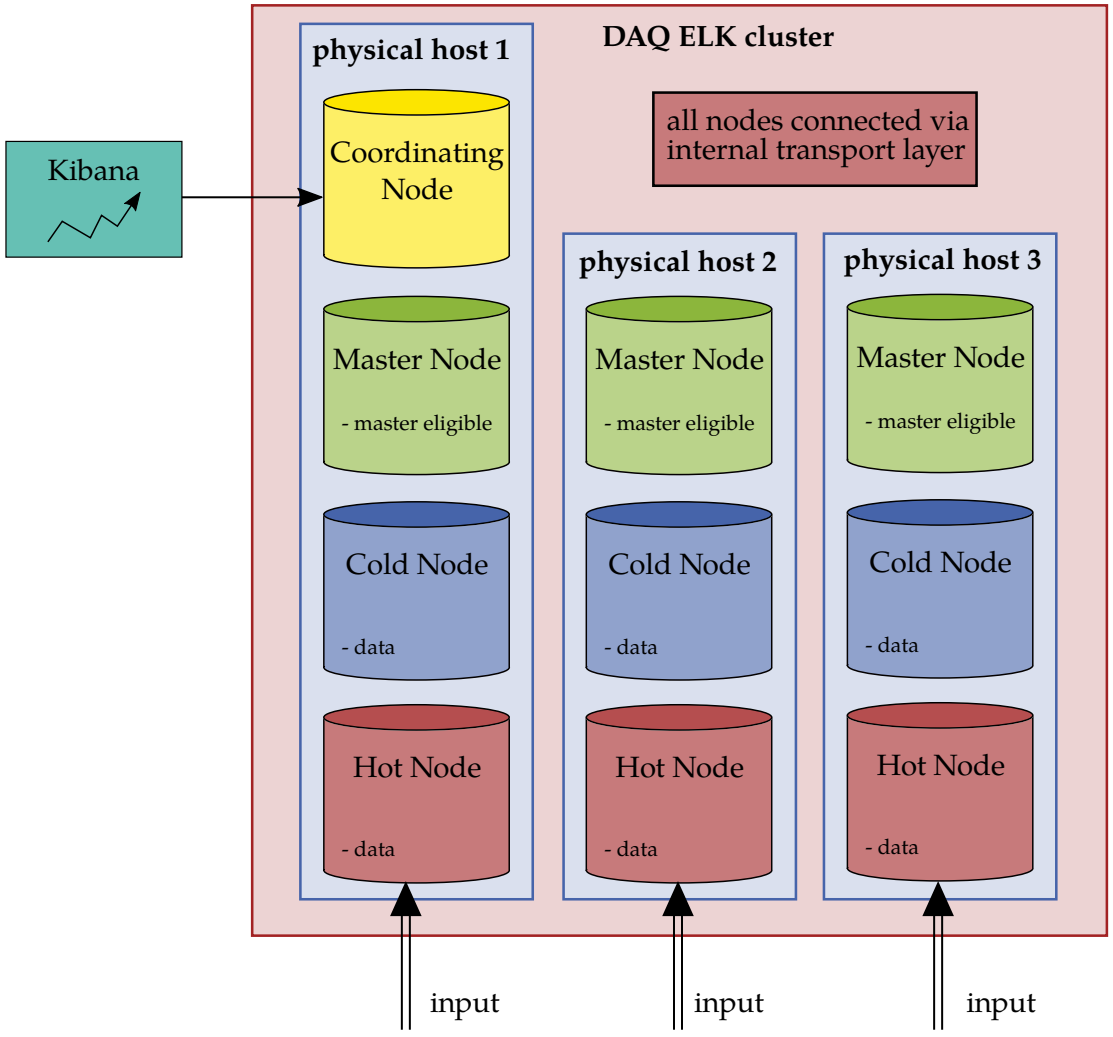


Fig. 6.10.: Possible future concept to guarantee the fail-safe operation of the DAQ ELK system. Three physically separate servers running the virtual machines for the Elasticsearch cluster. Hot, (warm), cold nodes can be used for index lifecycle management to relieve the cluster. Separate master nodes can also increase the performance of the cluster (see text).

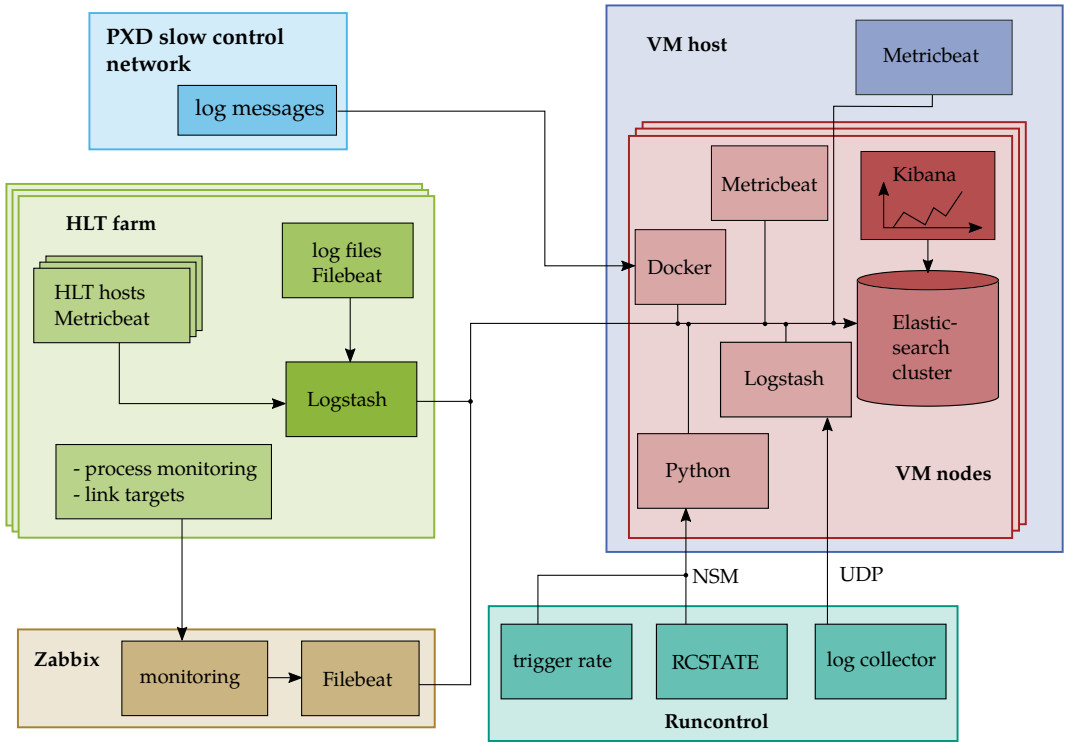
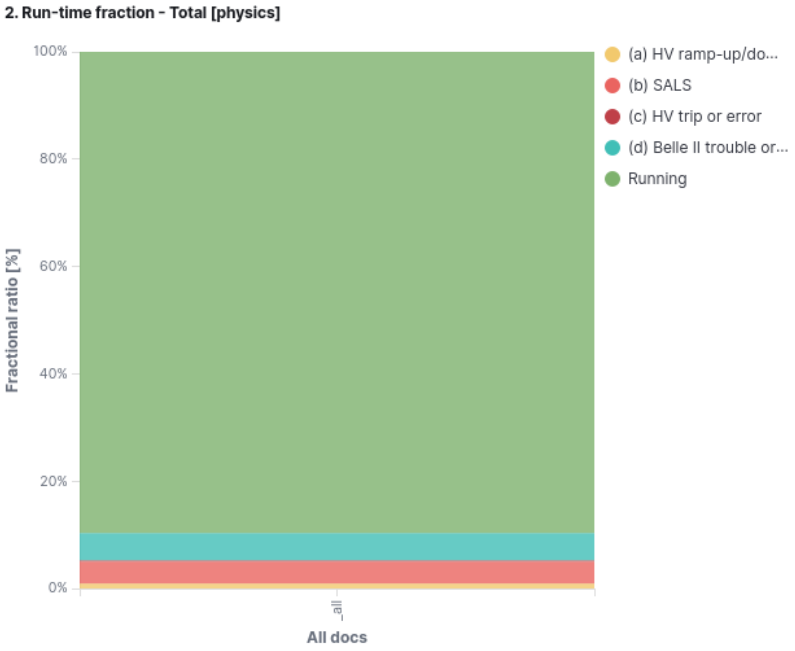


Fig. 6.11.: Schematic overview of the DAQ ELK system. Collecting log messages from the log collector from the run control network and form the entire HLT server farm. Process monitoring of the mandatory processes on the HLT and monitoring of the symbolic links pointing to the software releases using Zabbix. The PXD log messages are collected via a gateway from the PXD slow control network. NSM variables can be collected via an NSM Python interface. Hardware and system metrics can be monitored with Metricbeat.

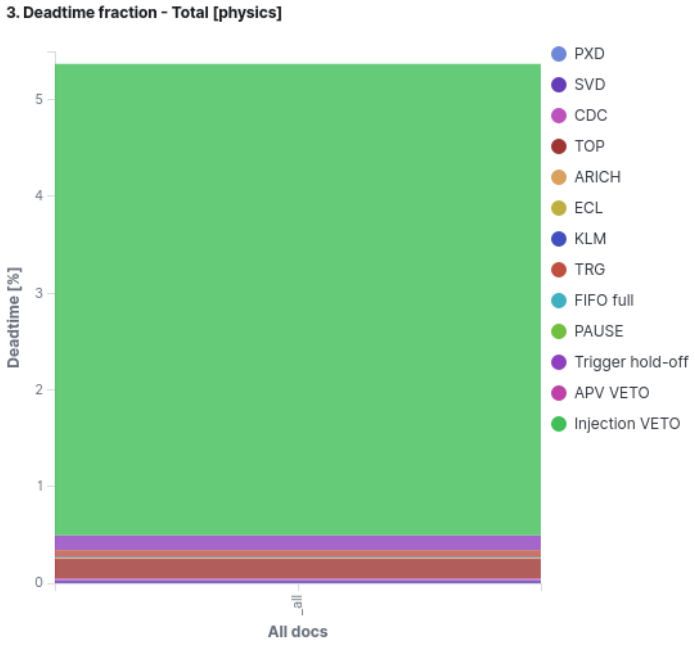
The project was welcomed and approved by the experiment's technical board. In the meantime, the subdetector groups have set up dashboards in Kibana to analyze the log messages of their detector components. In the Belle II control room, there are displays showing Kibana e.g. to visualize the current triggering rate, which can be used to immediately determine if a problem has occurred but the run control system has not yet entered the error mode. Kibana is also used by the daily run meeting to analyze and discuss the subdetector performance as shown in Fig. A.17 in the appendix. It is clearly visible which subdetectors are responsible for the most downtime and based on this information an action plan can be developed.

Based on the DAQ ELK system a fundamental DAQ efficiency calculation tool was implemented by DAQ experts using Kibana. Sources causing inefficiencies can be now investigated in detail and nailed down. Fig. 6.12 shows two visualizations to monitor the DAQ efficiency.

The system architecture of the DAQ ELK system is easily expandable. New functionalities can be added and existing ones improved at any time. The cluster can be scaled as needed. There now exists a solid foundation on which logging and monitoring can be built for the next years. It is a system that can be properly maintained and configured as required.



(a)



(b)

Fig. 6.12.: Kibana visualizations to determine the Belle II DAQ efficiency for physics runs in the time period 1st of May to 1st of July 2020. (a) shows the fraction of operational state. With a fraction of 89.9 % the experiment was in a running state (HV and luminosity enabled). (b) shows the dead time fraction of the running state. It is visible that the injection veto occupies 4.9 % of the running state.

7. Summary and Conclusion

In this thesis three independent topics were worked on: The impact of the Level 1 Bhabha veto to $e^+e^- \to \pi^+\pi^-(\gamma)\gamma_{\mathrm{ISR}}$, the HLT efficiency of $e^+e^- \to \mu^+\mu^-$ and the DAQ ELK setup, on which more detailed summaries and conclusions are given in each of the three dedicated chapters. This chapter provides a short summary of the overall results and achievements.

First, the loss of the old Belle 2D Bhabha veto and the new Belle II 3D Bhabha veto was analyzed using recorded data of experiment 8 and experiment 12. For the future, Bhabha events will be vetoed using the 3D Bhabha veto only. The analysis resulted in a loss of

$$\mathrm{loss}_{\mathrm{bha3d}} = 0.67^{+0.07226+0.00040}_{-0.06557-0.00045}\ \% \tag{7.1}$$

for the 3D Bhabha veto in a ρ mass range between 0 GeV and 2.5 GeV of the selected charged pion candidates. A dataset of ($2382\,\mathrm{pb}^{-1}$) of experiment 12 data was analyzed. The 3D Bhabha veto reduces the loss of the old Belle 2D Bhabha veto by a factor of approx. 18.

The uncertainty on the loss can be further reduced with a larger data sample as required. Further studies with more statistics will allow a spacial analysis, e.g., the distributions of the ISR photon polar or azimuthal angles, the spacial energy distributions, or the angles between the particles. The stability of the 3D Bhabha veto as a function of time should be investigated in further studies. For this purpose, datasets of different runs, e.g., of a run period or several run periods have to be examined and compared with respect to the 3D Bhabha veto.

Second, the HLT efficiency was investigated for two track dimuon events. Different data sets from different experiments were considered. The overall HLT efficiency for two track dimuon event data, processed online on the HLT with release 04-00-04 and offline in the latest major processing proc 11 with release 04-02-02, is

$$\varepsilon_{\mathrm{HLT},\mu\mu} = 99.98^{+0.010}_{-0.018}\ \%. \tag{7.2}$$

The influence of event properties, e.g. the polar angle or the momentum, on the HLT efficiency, as observed in previous software releases, disappeared.

Third, a major contribution was made to setting up a new system that allows central monitoring of log messages and system parameters in the Belle II DAQ network. The DAQ ELK

system is based on tools provided by the Elastic Stack, an open source system established in the IT industry used for e.g. monitoring data centers. The HLT was integrated to a central logging and monitoring approach for the first time. That such a system was long overdue for the operation of the Belle II detector was demonstrated by its rapid success and immediate roll-out into test operation on the DAQ network and in the Belle II control room.

During the physics run from March to July 2020 the system was tested in operation and continuously adapted to the needs and performance of the system. The project was welcomed and approved by the experiment's technical board. Once the fail-safety problem has been solved, the project will officially go into production mode.

Subdetector groups have already set up dashboards in Kibana to analyze the log messages of their detector components controlled by the DAQ slow control system. Additionally, a gateway to the PXD log and alarm messages was installed. In the Belle II control room, there are displays showing Kibana e.g. to visualize the current triggering rate, which can be used to immediately determine if a problem has occurred but the run control system has not yet entered the error state. In the daily run meeting the subdetector performances are discussed, now using the new DAQ ELK system. DAQ experts implemented a tool for fundamental DAQ efficiency calculations using the new system. Sources causing inefficiencies can be now investigated in detail and "nailed down."

The system architecture of the DAQ ELK system is easily expandable. New functionalities can be added and existing ones improved at any time. The implemented cluster can be scaled as needed. Now, a solid foundation on which logging and monitoring can be built for the next years exists. A system that can be properly maintained and configured as required.

Danksagung

Zum Abschluss möchte ich Allen danken, die zur Fertigstellung dieser Arbeit beigetragen haben.

Insbesondere danke ich Prof. Dr. Günter Quast und Prof. Dr. Florian Bernlochner für die Übernahme des Referats und des Korreferats.

Bei Dr. Markus Prim möchte ich mich für die persönliche Betreuung bedanken und dass ich ihn jederzeit um Rat fragen konnte.

Bei Prof. Dr. Christopher Hearty und Dr. Torben Ferber bedanke ich mich für die gute Zusammenarbeit bei den Trigger-Studien.

Für die gute Zusammenarbeit am DAQ-ELK-Projekt möchte ich mich bei Dr. Takuto Kunigo und bei Dr. Soh Yamagata Suzuki bedanken, die mir immer mit Rat und Tat zur Seite standen. Außerdem bedanke ich mich bei Dr. Nils Braun, Dr. Oskar Hartbich und Dr. Björn Spruck, die ich immer mit Fragen zum DAQ-System und zum Detektor löchern konnte.

Ich danke dem ETP und vor allem Dr. Pablo Goldenzweig, dass sie mir den Forschungsaufenthalt in Japan am KEK ermöglicht haben. Dr. Goldenzweig danke ich auch insbesondere für seinen Einsatz und seine Unterstützung für diese Arbeit. Frau Bräunling danke ich außerdem für die Hilfe bei allen verwaltungstechnischen Anliegen.

Zuletzt möchte ich mich noch bei der gesamten Belle-II-Arbeitsgruppe am ETP (und den zwei Toppies :)) für die breite Unterstützung und die tolle Zeit bedanken.

Bibliography

[1] E. Kou et al., "The Belle II Physics Book," *arXiv:1808.10567 [hep-ex]* (2019) .

[2] Z. Doležal, S. Uno, et al., "Belle II technical design report," *KEK Report 2010-1* (Oct. 2010) .

[3] M. Nakao, T. Higuchi, R. Itoh, and S. Y. Suzuki, "Data acquisition system for Belle II," *Journal of Instrumentation* **5** no. 12, (Dec, 2010) C12004–C12004.

[4] "PostgreSQL: The World's Most Advanced Open Source Relational Database." https://www.postgresql.org/. [Online; accessed 18-September-2020].

[5] T. Konno, R. Itoh, M. Nakao, S. Y. Suzuki, and S. Yamada, "The Slow Control and Data Quality Monitoring System for the Belle II Experiment," *IEEE Transactions on Nuclear Science* **62** no. 3, (2015) 897–902.

[6] T. Konno, "Tutorial for slow control software," 2018. https://confluence.desy.de/download/attachments/35009102/SC_tutorial_20181119.pdf?version=1&modificationDate=1542608541287&api=v2. [Online; accessed 09-October-2020; restricted access].

[7] "The Experimental Physics and Industrial Contrl System." https://epics-controls.org. [Online; accessed 18-September-2020].

[8] "The EPICS Archiver Appliance." https://slacmshankar.github.io/epicsarchiver_docs/index.html. [Online; accessed 21-October-2020].

[9] M. Nakao, "NSM2 – Network Shared Memory," 2013. https://confluence.desy.de/download/attachments/34043577/nsm2.pdf?version=1&modificationDate=1467384029712&api=v2. [Online; accessed 21-Oktober-2020; restricted access].

[10] Y. Iwasaki, "Belle II Trigger System," 2020. https://confluence.desy.de/download/attachments/107067367/Trigger%20System%2020200219.ai?version=1&modificationDate=1583409150805&api=v2. [Online; accessed 20-Oktober-2020; restricted access].

[11] Y. Iwasaki, B. Cheon, E. Won, X. Gao, L. Macchiarulo, K. Nishimura, and G. Varner, "Level 1 Trigger System for the Belle II Experiment," *IEEE Transactions on Nuclear Science* **58** no. 4, (2011) 1807–1815.

[12] S. Y. Suzuki, S. Yamada, R. Itoh, M. Nakao, T. Konno, T. Higuchi, and K. Nakamura, "The Three-Level Event Building Systemfor the Belle II Experiment," *IEEE Transactions on Nuclear Science* **62** no. 3, (2015) 1162–1168. https://confluence.desy.de/display/BI/Core+Computing+Library?preview=%2F134209238%2F134213874%2FThree-Level+Event+Building+System+for+the+Belle+II+Experiment.pdf.

[13] N. Braun, "Combinatorial Kalman Filter and High Level Trigger Reconstruction for the Belle II Experiment," 2019. https://www.springer.com/de/book/9783030249960.

[14] N. Braun, "ZMQ HLT operations," 2019. https://confluence.desy.de/display/BI/ZMQ+HLT+operations. [Online; accessed 20-October-2020; restricted access].

[15] N. Braun, "HLT Data Transportation with ZMQ," 2019. https://confluence.desy.de/display/BI/HLT+Data+Transportation+with+ZMQ. [Online; accessed 20-October-2020; restricted access].

[16] "Experiment numbering." https://confluence.desy.de/display/BI/Experiment+numbering. [Online; accessed 22-October-2020; restricted access].

[17] "Phase 3 data." https://confluence.desy.de/display/BI/Phase+3+data. [Online; accessed 22-October-2020; restricted access].

[18] R. Brun and F. Rademakers, "ROOT: An object oriented data analysis framework," *Nucl. Instrum. Meth. A* **389** (1997) 81–86. https://root.cern.ch/.

[19] D. Casadei, "Estimating the selection efficiency," *arXiv:0908.0130 [physics.data-an]* (2012) .

[20] CERN ROOT, "TEfficiency Class Reference." https://root.cern.ch/doc/master/classTEfficiency.html. [Online; accessed 30-August-2020].

[21] J. Neyman and J. Harold, "Outline of a Theory of Statistical Estimation Based on the Classical Theory of Probability," *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* **236:333–380** (1937) .

[22] G. Cowan, *Statistical Data Analysis*. Oxford Universty Press, 1998.

[23] Y. Maeda, "Analysis of light hadron production with initial state radiation using full data sample of the phase2 operation," *BELLE2-NOTE-PH-2018-030* (2018) . https://docs.belle2.org/record/1110/files/phase2full_pipigamma_v1_2.pdf. Draft Version 1.1 [restricted access].

[24] M. e. a. Tanabashi and Particle Data Group, "Review of Particle Physics," *Physical Review D* **98** no. 3, 030001. https://link.aps.org/doi/10.1103/PhysRevD.98.030001.

[25] T. B. et al., "The Muon (g-2) Theory Value: Present and Future," *arXiv:1311.2198v1 [hep-ph]* (2013) . https://arxiv.org/pdf/1311.2198.pdf.

[26] J. P. Lees et al. (the BABAR Collaboration), "Precise Measurement of the $e^+e^- \to \pi^+\pi^-(\gamma)$ Cross Section with the Initial-State Radiation Method at BABAR," *arXiv:1205.2228v1 [hep-ex]* (2012) .

[27] "Bhabha trigger." https://confluence.desy.de/display/BI/Bhabha+trigger. [Online; accessed 27-October-2020; restricted access].

[28] Y. Maeda, "Measurement of cross section of light hadron production in e^+e^- collisions in the Belle II experiment," *BELLE2-TALK-DRAFT-2018-129, BELLE2-TALK-CONF-2018-148* (2018) . https://docs.belle2.org/record/1249/files/BELLE2-TALK-CONF-2018-148.pdf. [Online; accessed 27-October-2020].

[29] "Documentation basf2 framework,". https://software.belle2.org/. [Online; accessed 28-October-2020; restricted access].

[30] "pandas,". https://pandas.pydata.org/. [Online; accessed 28-October-2020].

[31] P. Rados et al., "Tracking Efficiency with Taus,". https://indico.belle2.org/event/2325/contributions/11153/attachments/5572/8627/tau_eff_29May2020.pdf. [Slide 18; Online; accessed 28-October-2020].

[32] T. Ferber, P. Urquijo, "Overview of the Belle II Physics Generators," *BELLE2-NOTE-PH-2015-006* (2016) . [restricted access].

[33] M. Milesi, J. Tan2, P. Urquijo, "Lepton identification in Belle II using observables from the electromagnetic calorimeter and precision trackers," *BELLE2-CONF-PROC-2020-005* (2020) . [restricted access].

[34] "Use of Releases." https://confluence.desy.de/display/BI/Use+of+Releases. [Online; accessed 09-September-2020; restricted access].

[35] "Elastic Stack and Product Documentation." https://www.elastic.co/guide/index.html. [Online; accessed 22-September-2020].

[36] "Elasticsearch Reference." https://www.elastic.co/guide/en/elasticsearch/reference/7.8/index.html. [Online; accessed 17-September-2020].

[37] "PostgreSQL for time-series." https://www.timescale.com/. [Online; accessed 12-October-2020].

[38] "Zabbix." https://www.zabbix.com/. [Online; accessed 12-Oktober-2020].

[39] "Elastic customer stories of all shapes and sizes." https://www.elastic.co/customers/success-stories?usecase=enterprise-search. [Online; accessed 16-September-2020].

[40] "Elastic Discussion Forums." https://discuss.elastic.co/. [Online; accessed 22-September-2020].

[41] "Stack Overflow." https://stackoverflow.com/. [Online; accessed 22-September-2020].

[42] "Splunk." https://www.splunk.com/. [Online; accessed 22-September-2020].

[43] "Experience With Splunk for Archiving and Visualisation of Operational Data in ATLAS TDAQ System." https://cds.cern.ch/record/2304202/files/ATL-DAQ-SLIDE-2018-087.pdf. [Online; accessed 22-September-2020].

[44] "Apache Lucene 7.4.0 Documentation." https://lucene.apache.org/core/7_4_0/index.html. [Online; 16-September-2020].

[45] M. Ingebrigtsen, "Indexing for Beginners." https://www.elastic.co/blog/found-indexing-for-beginners-part1/. [Online; accessed 22-September-2020].

[46] O. Procopiuc, P. Agarwal, L. Arge, and J. Vitter, "Bkd-Tree: A Dynamic Scalable kd-Tree,".

[47] "Hypertext Transfer Protocol Version 2 (HTTP/2)." https://tools.ietf.org/html/rfc7540. [Online; accessed 22-September-2020].

[48] D. Stenberg, "Everything curl." https://ec.haxx.se/. [Online; accessed 22-September-2020].

[49] "Kibana Guide." https://www.elastic.co/guide/en/kibana/7.8/index.html. [Online; accessed 17-September-2020].

[50] "Logstash Reference." https://www.elastic.co/guide/en/logstash/7.9/index.html. [Online; accessed 17-September-2020].

[51] "Filebeat Referenc." https://www.elastic.co/guide/en/beats/filebeat/7.8/index.html. [Online; accessed 17-September-2020].

[52] "Metricbeat Reference." https://www.elastic.co/guide/en/beats/metricbeat/7.8/index.html. [Online; accessed 17-September-2020].

[53] Mikhail Remnev, "nsm2-pynsm2," 2019. https://stash.desy.de/users/remnev/repos/nsm2-pynsm2. [Online; accessed 13-September-2020; restricted access].

[54] Elastic, "Python Elasticsearch Client," 2020. https://elasticsearch-py.readthedocs.io/en/master/. [Online; accessed 13-September-2020].

[55] "Control System Studio (CSS)." http://controlsystemstudio.org/. [Online; accessed 21-September-2020].

[56] "STOMP-capable C++ logfile library with EPICS integration." https://github.com/sus-ziti-uni-hd/epics-Logfile. [Online; accessed 21-September-2020].

[57] "Apache Camel." https://https://camel.apache.org. [Online; accessed 20-September-2020].

[58] "Spring." https://https://https://spring.io. [Online; accessed 20-September-2020].

[59] M. Ritzert, "Improving User Information by Interfacing the Slowcontrl's Log and Alarm Systems to a Flexible Chat Platform," *17th Int. Conf. on Acc. and Large Exp. Physics Control Systems* (2019) . http://icalepcs2019.vrws.de/papers/mompr002.pdf. [Online; accessed 17-Oktober-2020].

[60] "Docker." https://www.docker.com/. [Online; accessed 21-September-2020].

[61] "PXD Log Interface." https://confluence.desy.de/display/BI/PXD+Log+Interface. [Online; accessed 21-September-2020; restricted access].

[62] RedHat, "Ansible Documentation," 2020. https://docs.ansible.com/ansible/latest/index.html. [Online; accessed 14-September-2020].

[63] "ZMQ HLT operations." https://confluence.desy.de/display/BI/ZMQ+HLT+operations. [Online; accessed 20-September-2020; restricted access].

[64] "ROISender." https://confluence.desy.de/display/BI/ROISender. [Online; accessed 20-September-2020; restricted access].

[65] "Belle II Software." https://stash.desy.de/projects/B2/repos/software/browse. [Online; accessed 20-September-2020; restricted access].

A. Appendix

A.1. Experimental Setup

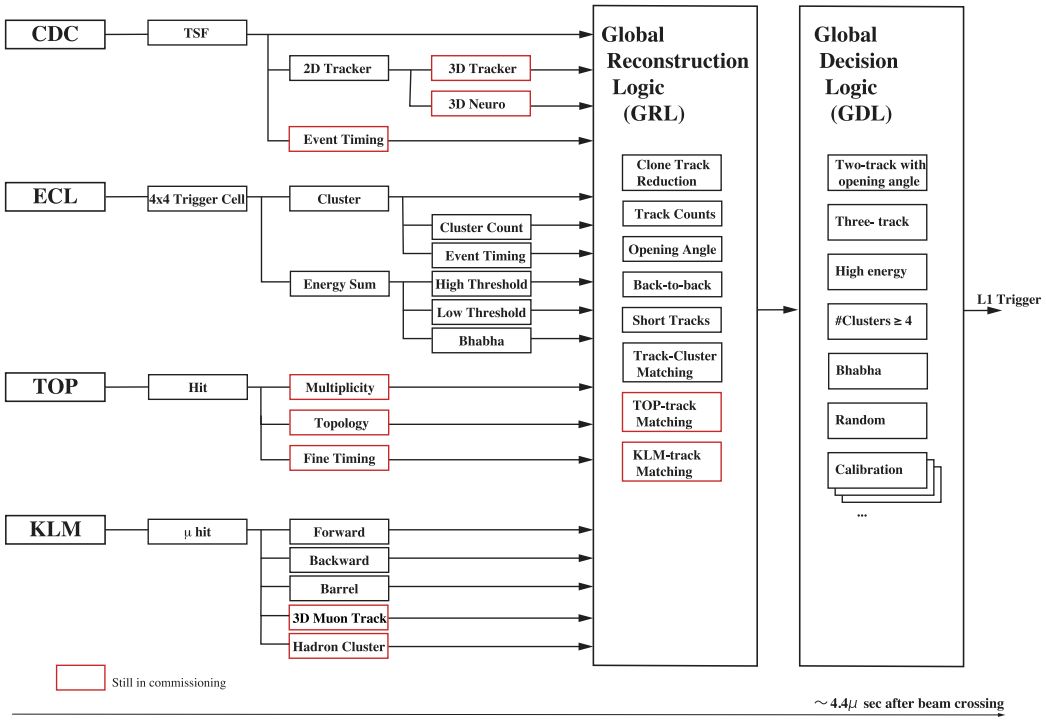


Fig. A.1.: Level 1 trigger system with the subtrigger systems. Currently, the CDC and ECL subtrigger system are mainly used. Figure adapted from [10].

A.2. Plots for Level 1 Bhabha Veto Loss

Additional plots of the distributions of the selected Level 1 triggers as functions of particle properties and the calculated loss with respect to the 2D Bhabha or 3D Bhabha veto are listed in this section. Furthermore, for the 3D Bhabha veto the angular and energy distribution of the ISR photon candidate are shown, as well as the distribution of the loss in the phase space.

A.2.1. 2D Bhabha Veto

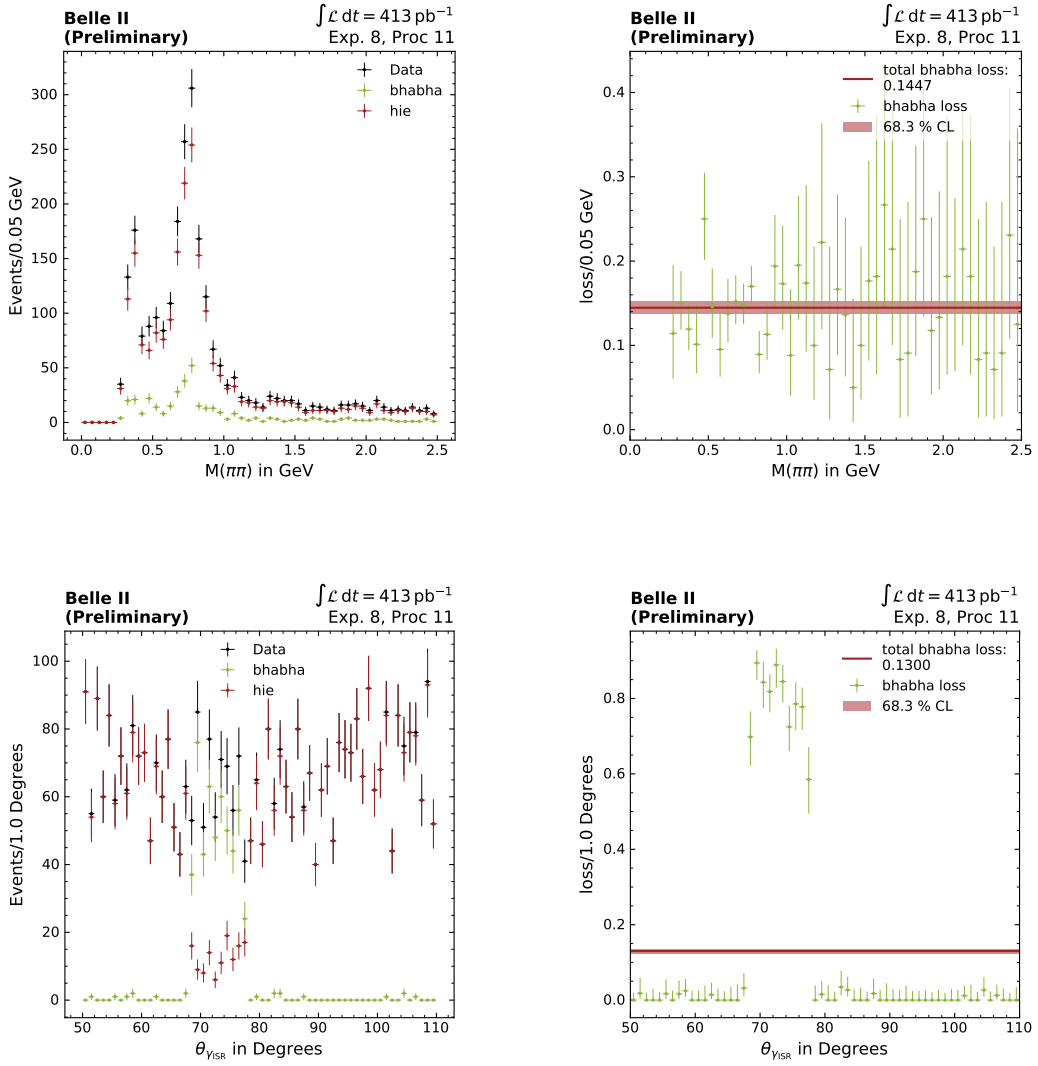


Fig. A.2.: Plots in the left column show the distributions of the events selected by the respective Level 1 trigger as function of the variables used for the event selection. The plots in the right column show the calculated loss distributions as functions of these variables.

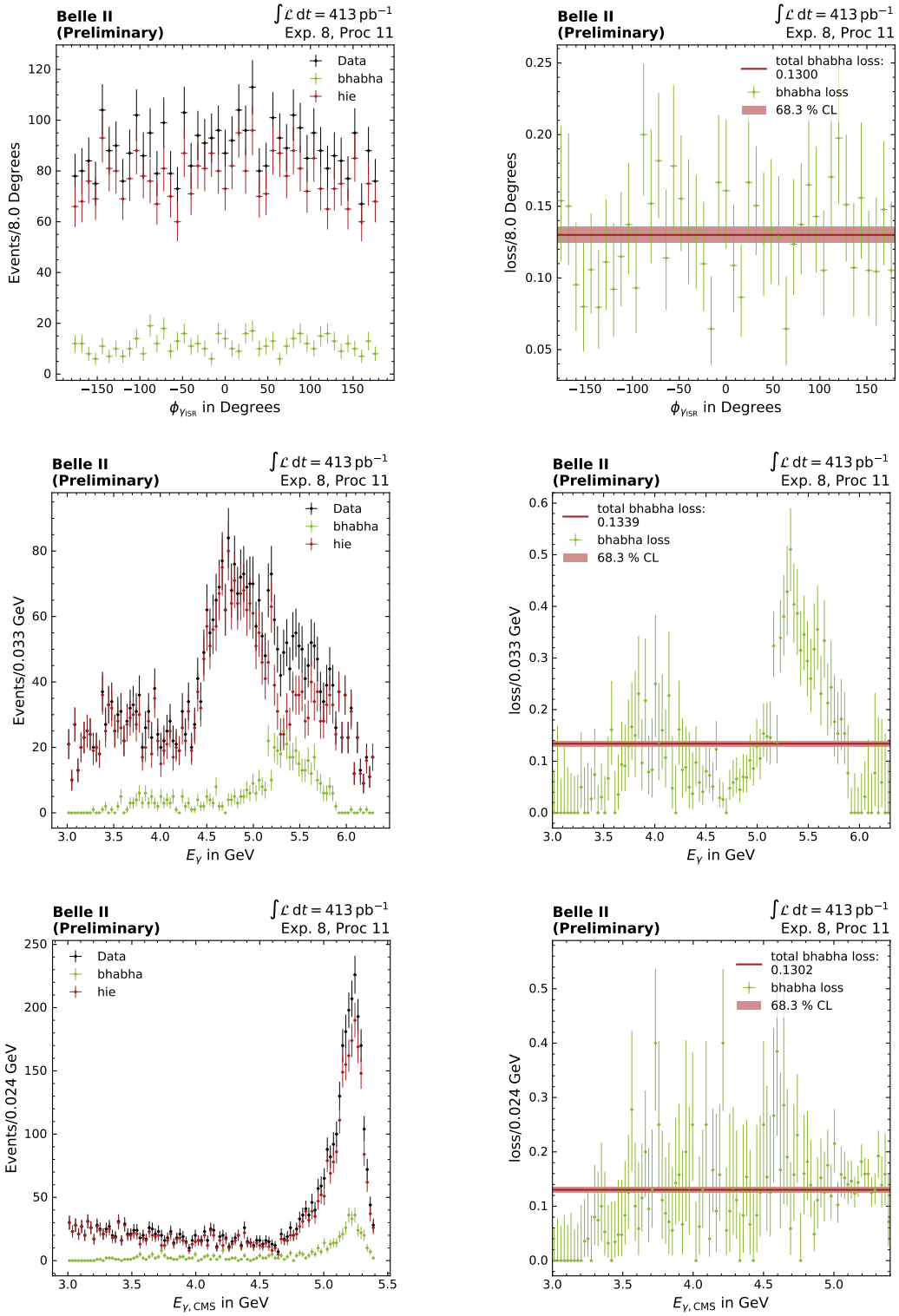


Fig. A.3.: Plots in the left column show the distributions of the events selected by the respective Level 1 trigger as function of the variables used for the event selection. The plots in the right column show the calculated loss distributions as functions of these variables.

A.2.2. 3D Bhabha Veto

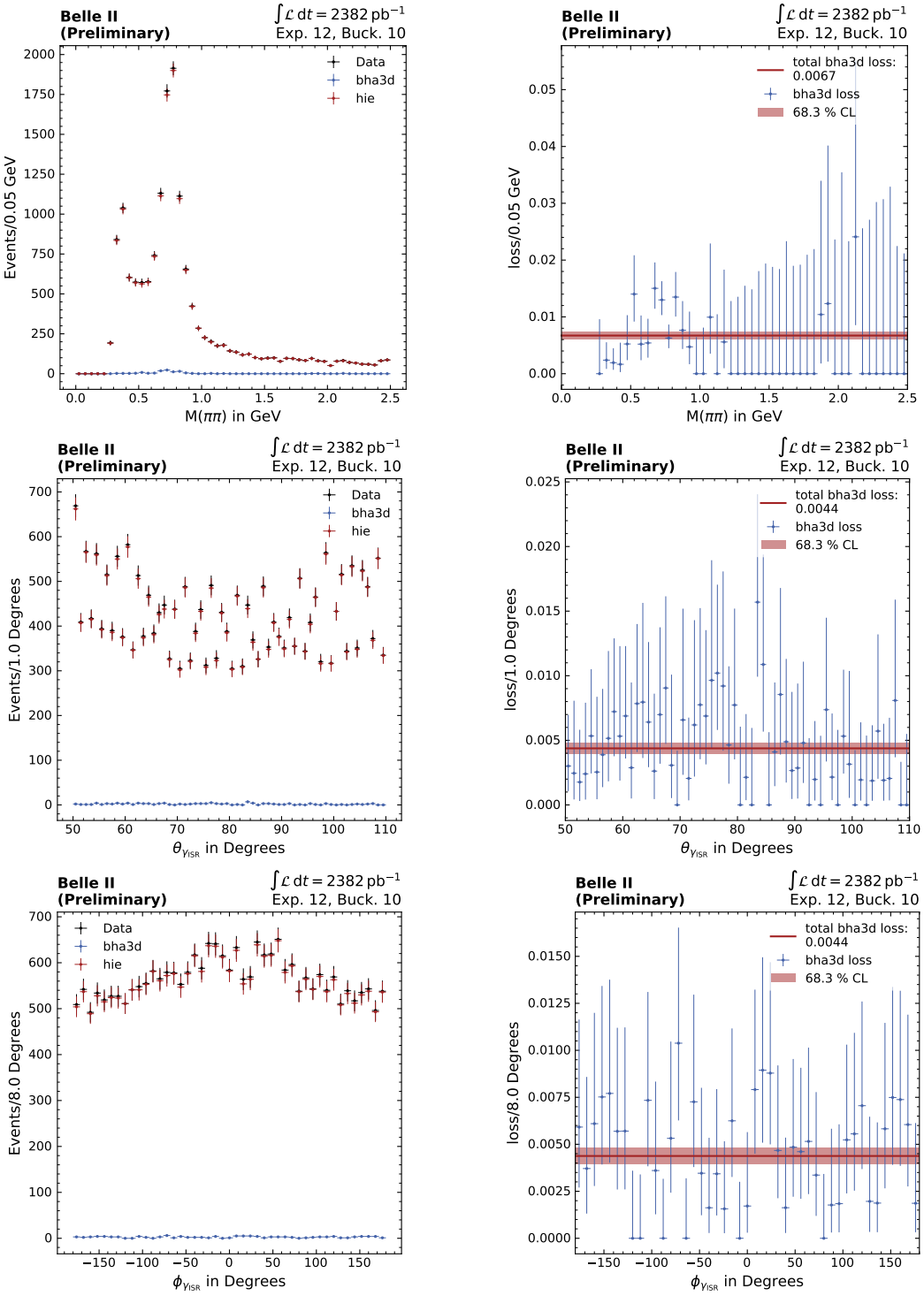


Fig. A.3.: Plots in the left column show the distributions of the events selected by the respective Level 1 trigger as function of the variables used for the event selection. The plots in the right column show the calculated loss distributions as functions of these variables.

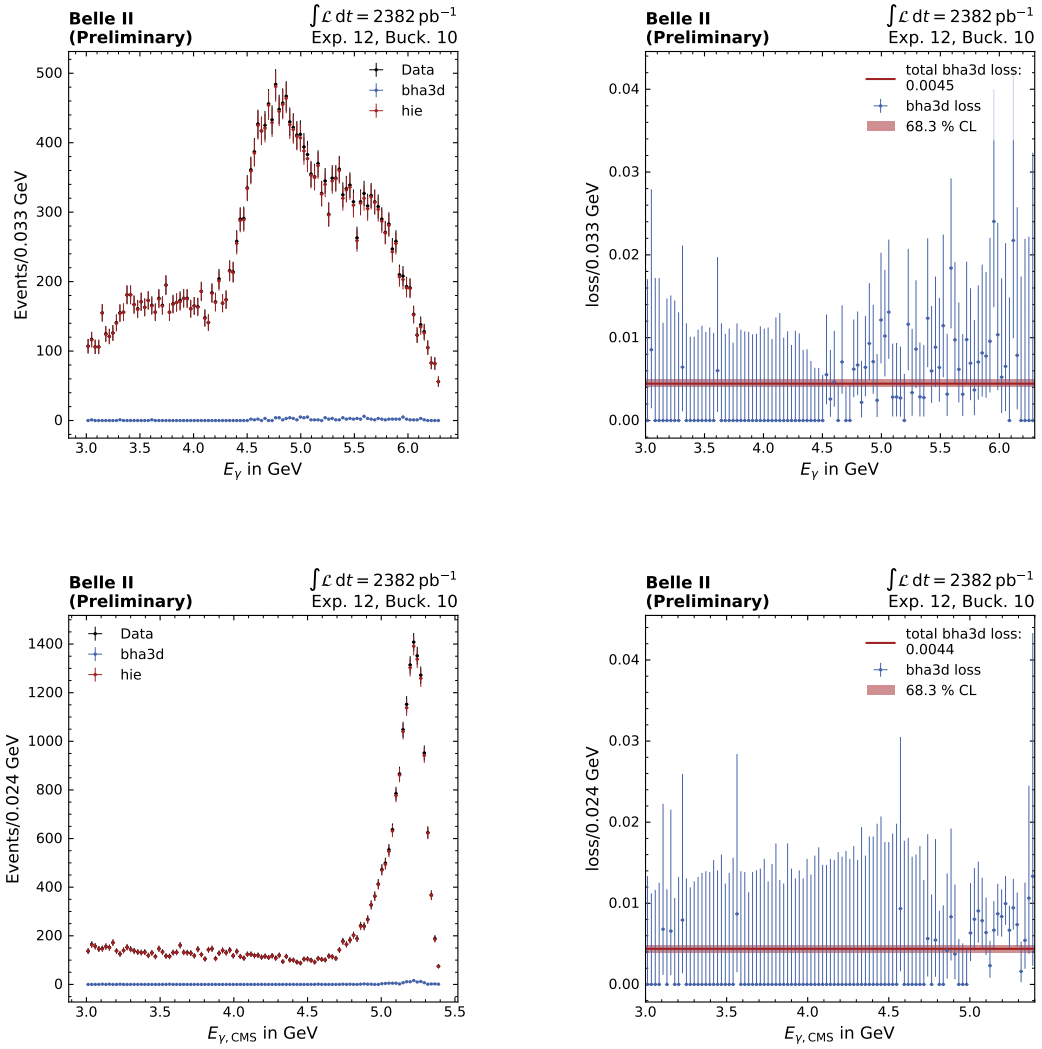


Fig. A.4.: Plots in the left column show the distributions of the events selected by the respective Level 1 trigger as function of the variables used for the event selection. The plots in the right column show the calculated loss distributions as functions of these variables.

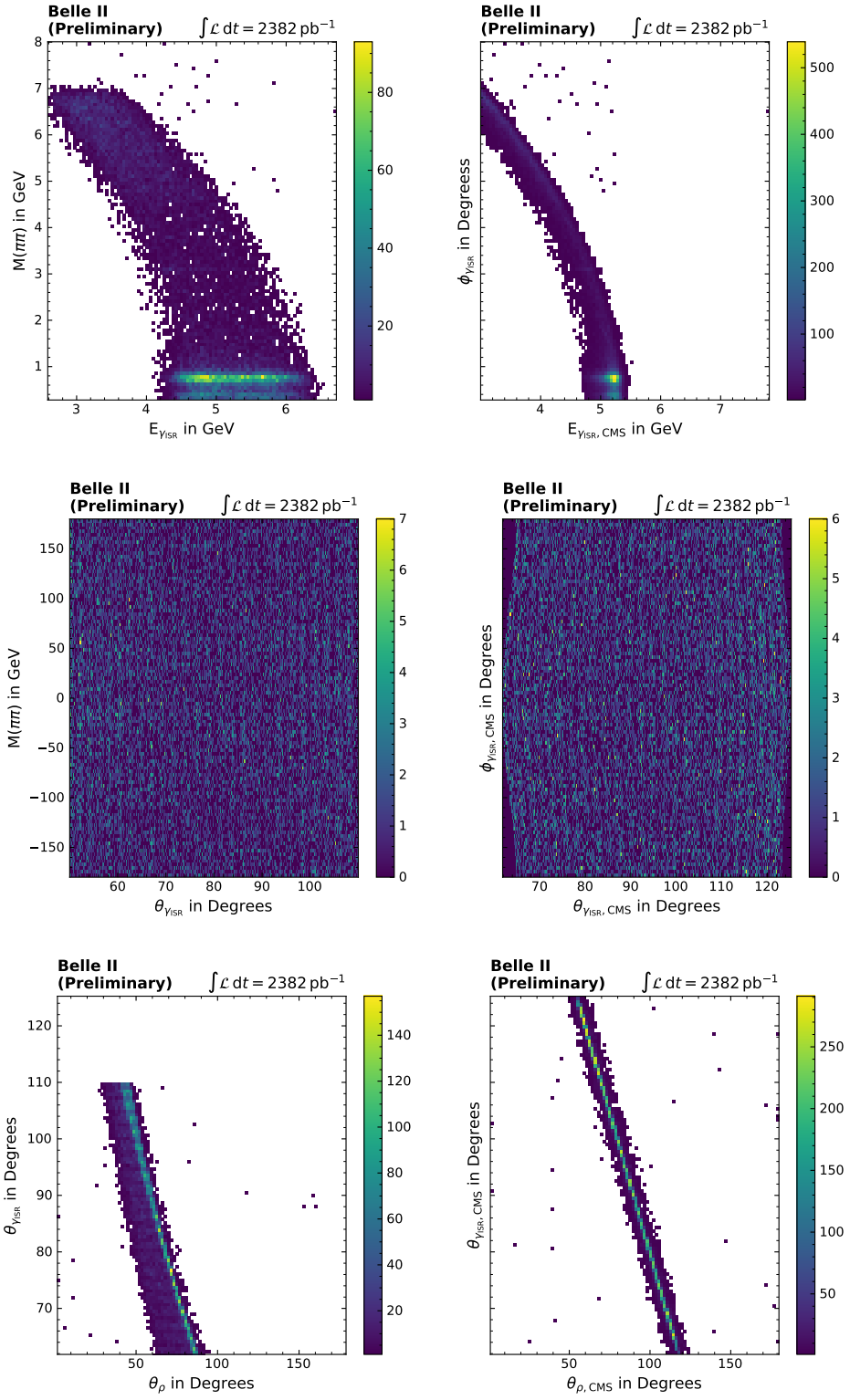


Fig. A.5.: Plots in the left column show 2D histograms of the correlations between particle properties in the lab frame. In the right column the same particle properties are shown in the c.m.s. frame.

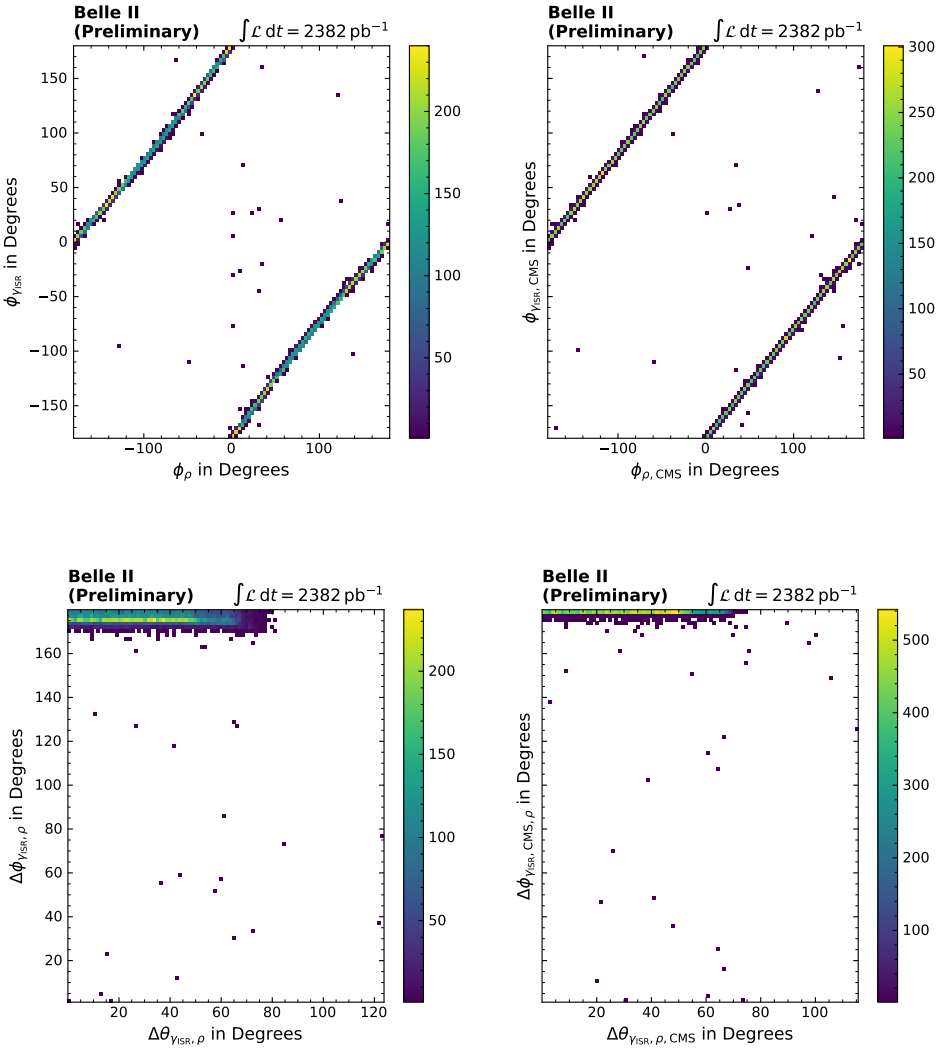


Fig. A.6.: Plots in the left column show 2D histograms of the correlations between particle properties in the lab frame. In the right column the same particle properties are shown in the c.m.s. frame.

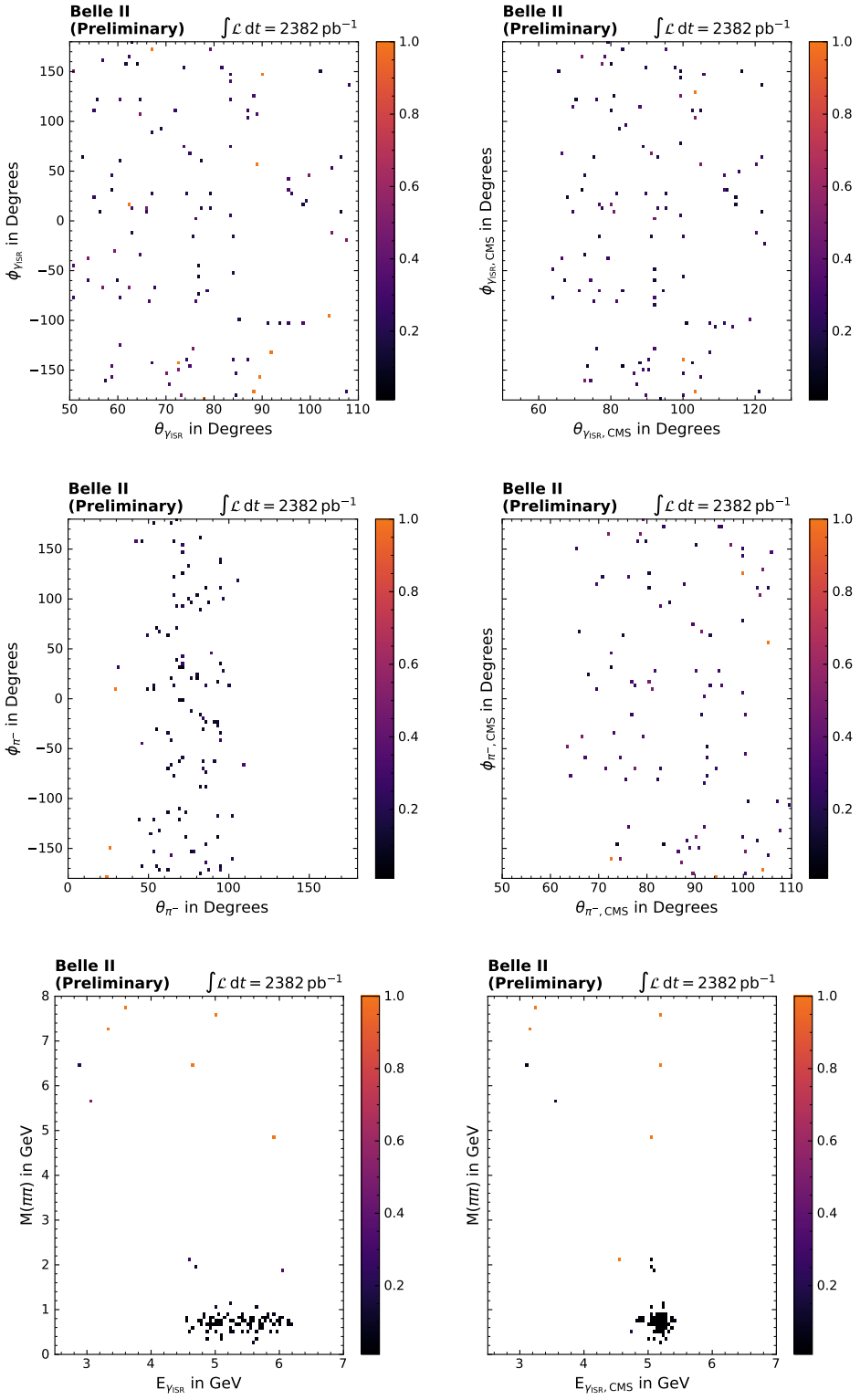


Fig. A.7.: Plots in the left column show the loss correlations in the phase space of the particles in the lab frame. In the right column the same phase space is shown in the c.m.s. frame.

A.3. Plots for HLT Two-Track Muon Efficiency

Additional HLT efficiency plots as functions of particle properties, as well as the distribution of the number of events as functions of these properties are shown in this section for experiment 10 and experiment 8 analyses.

A.3.1. Experiment 10 - Proc 11

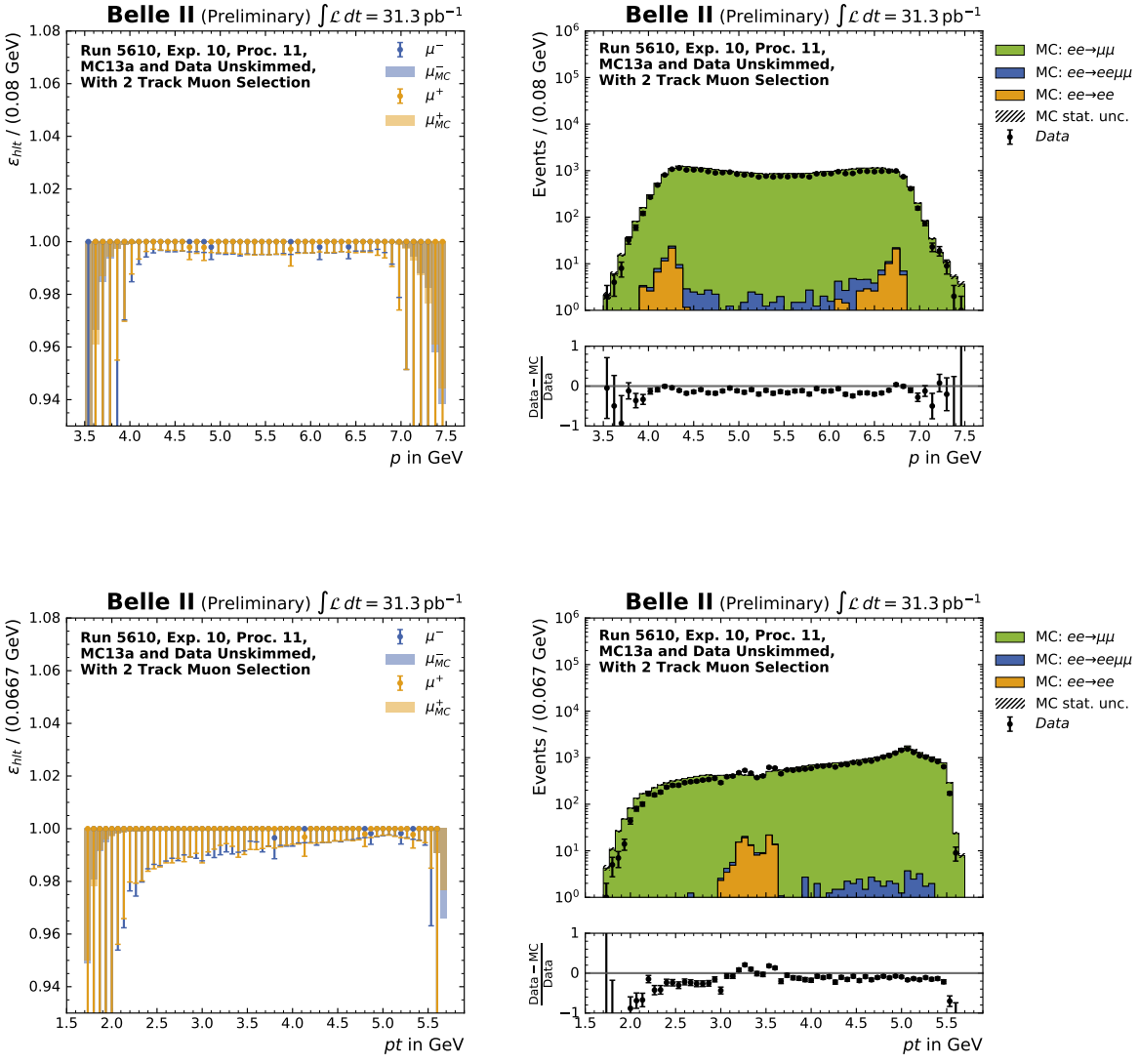


Fig. A.7.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

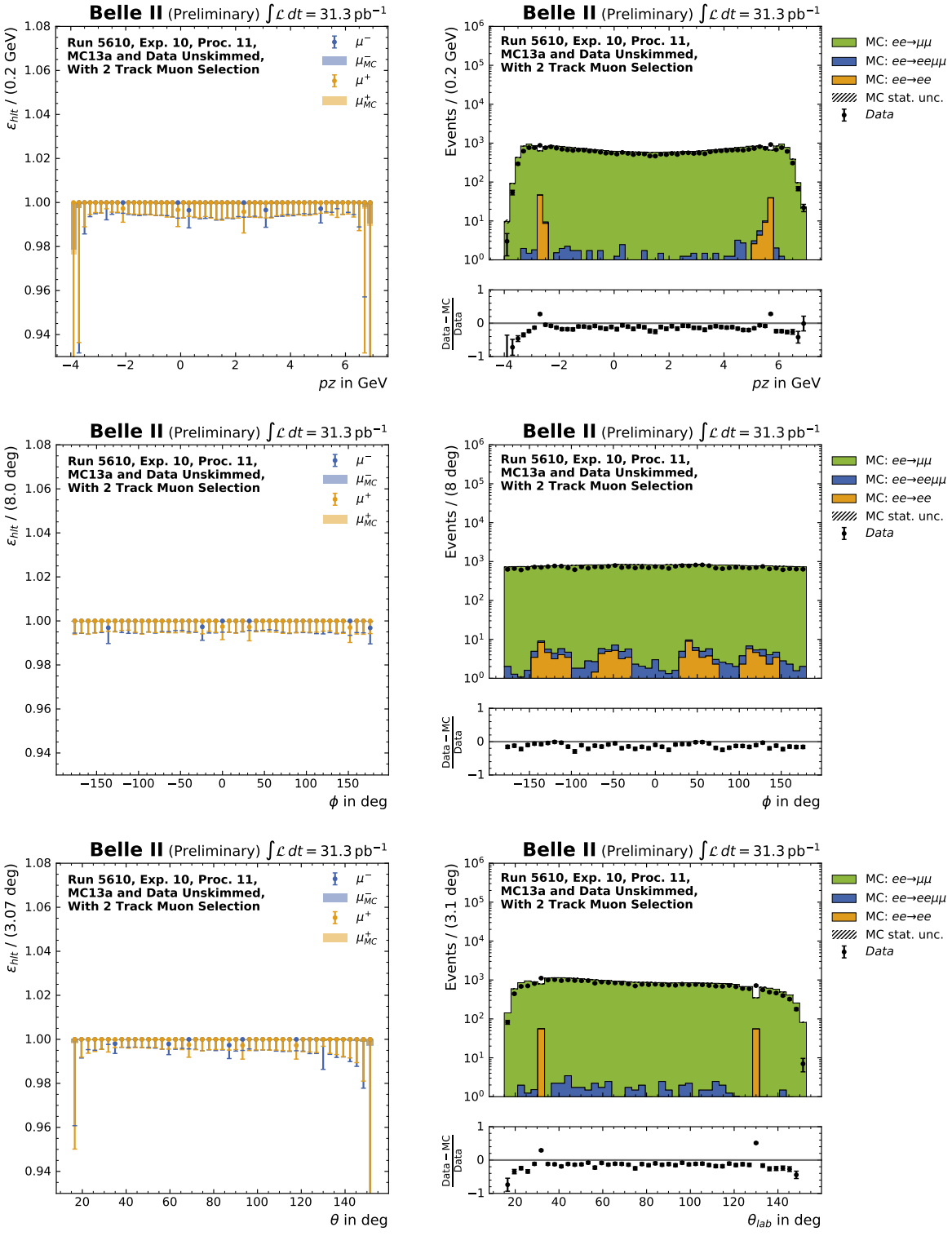


Fig. A.8.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

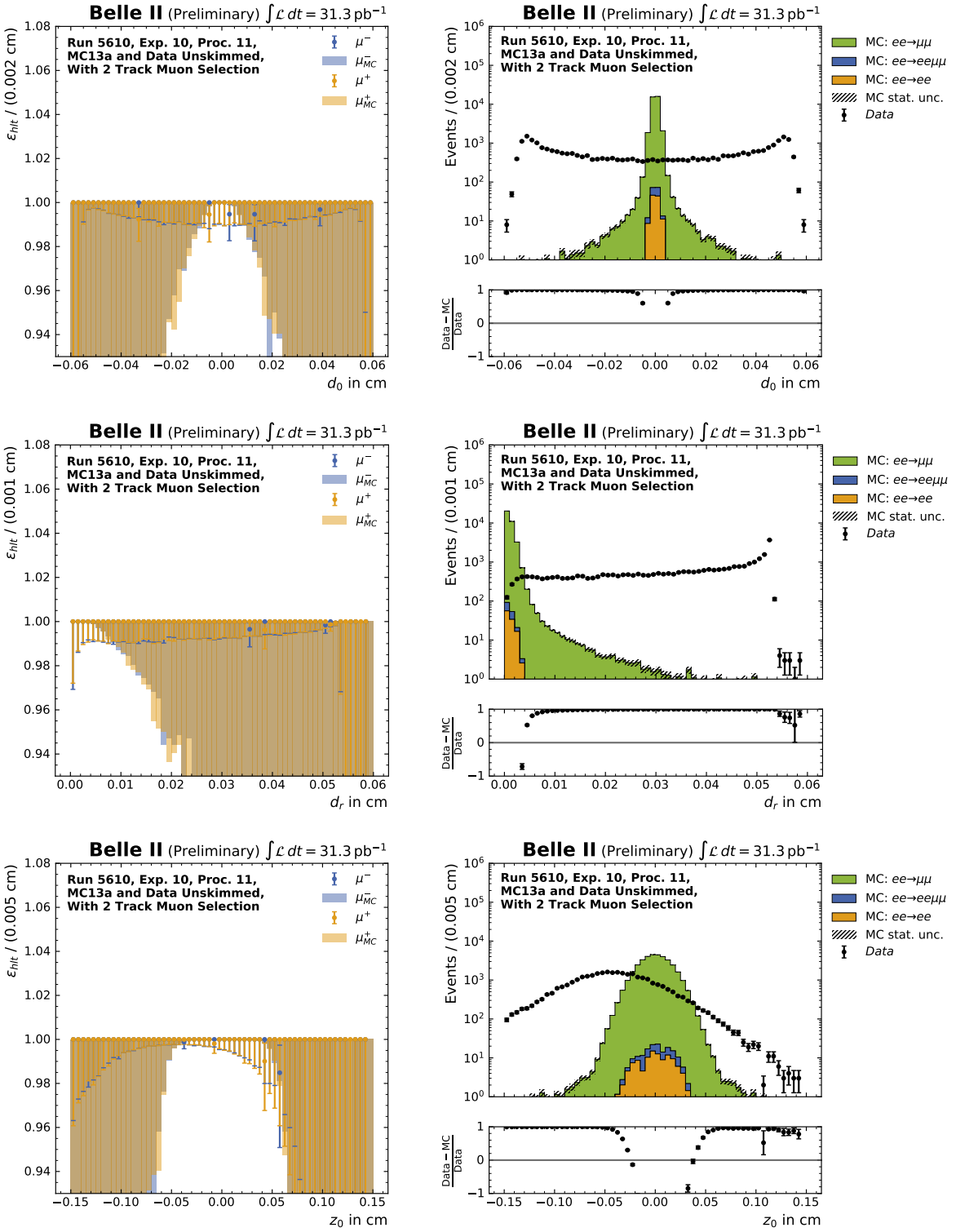


Fig. A.9.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

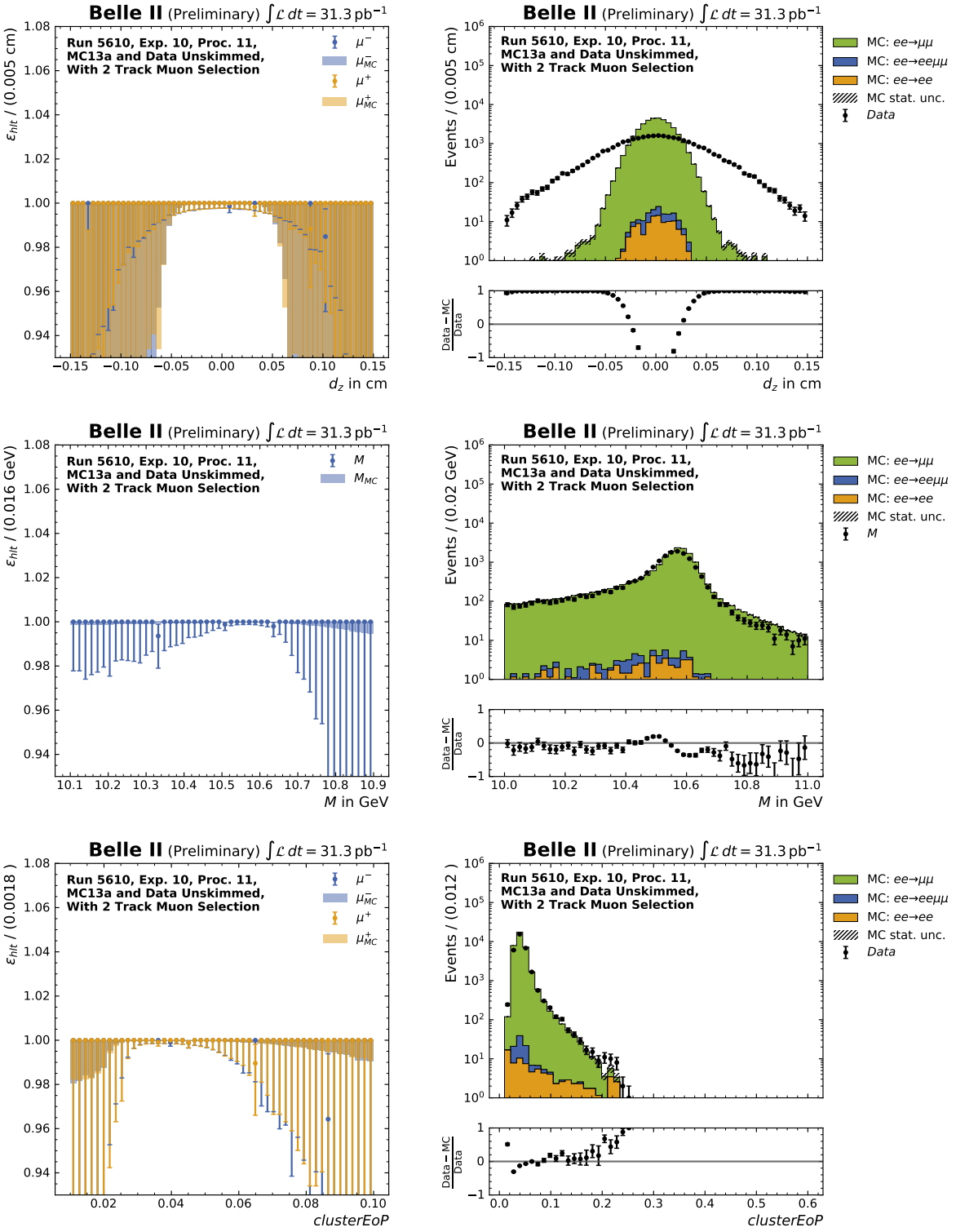


Fig. A.10.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

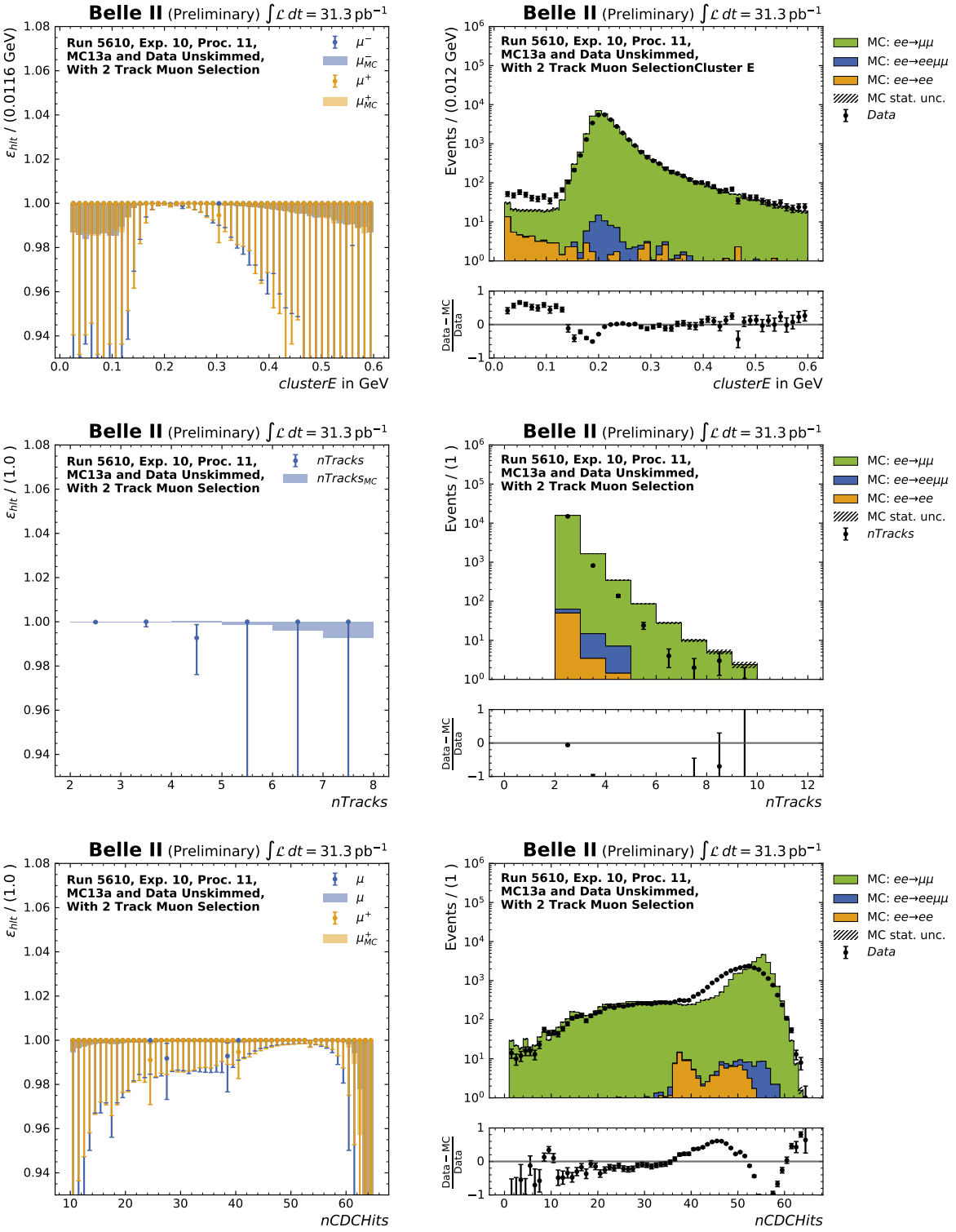


Fig. A.11.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

A.3.2. Experiment 8 - Proc 10

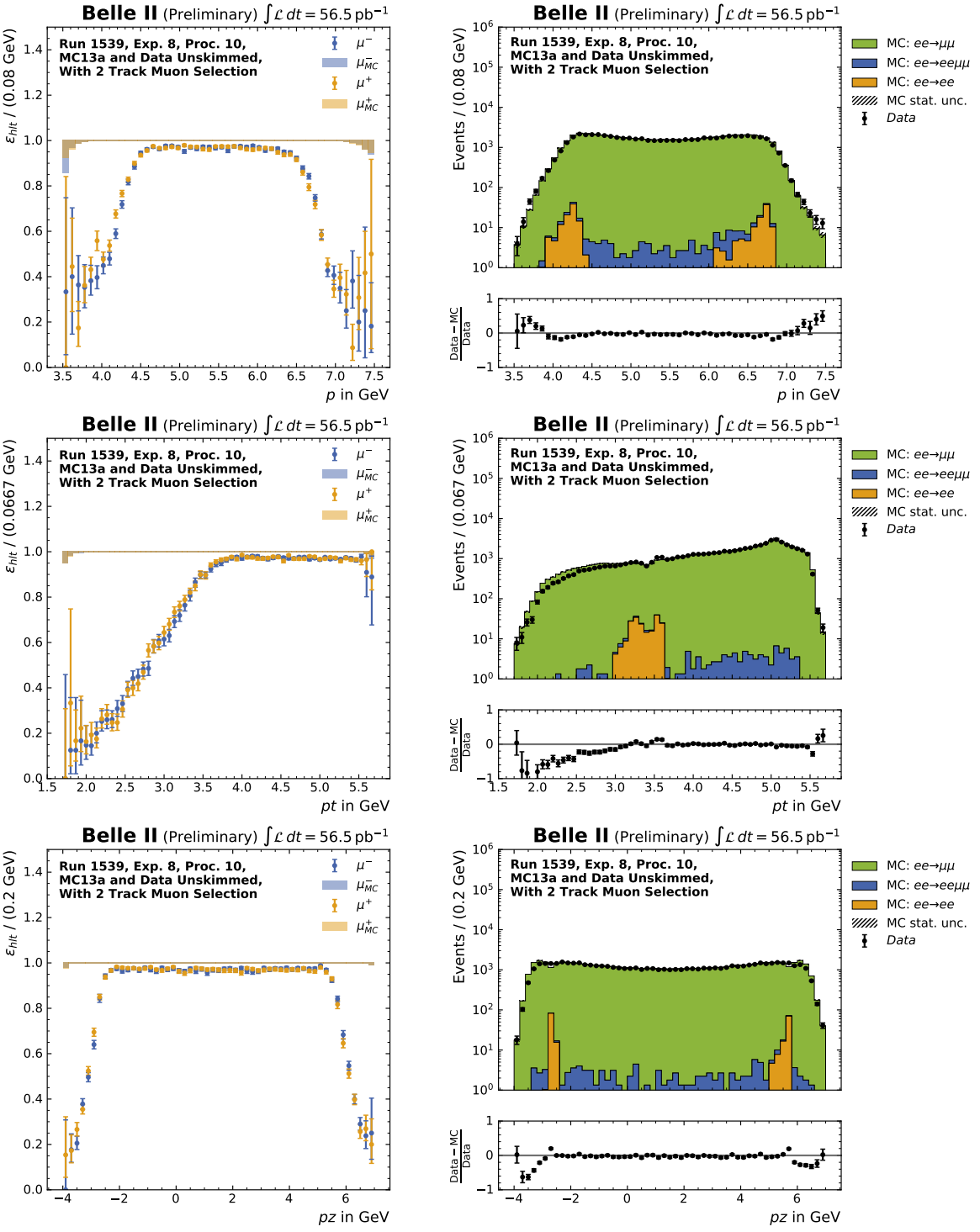


Fig. A.12.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

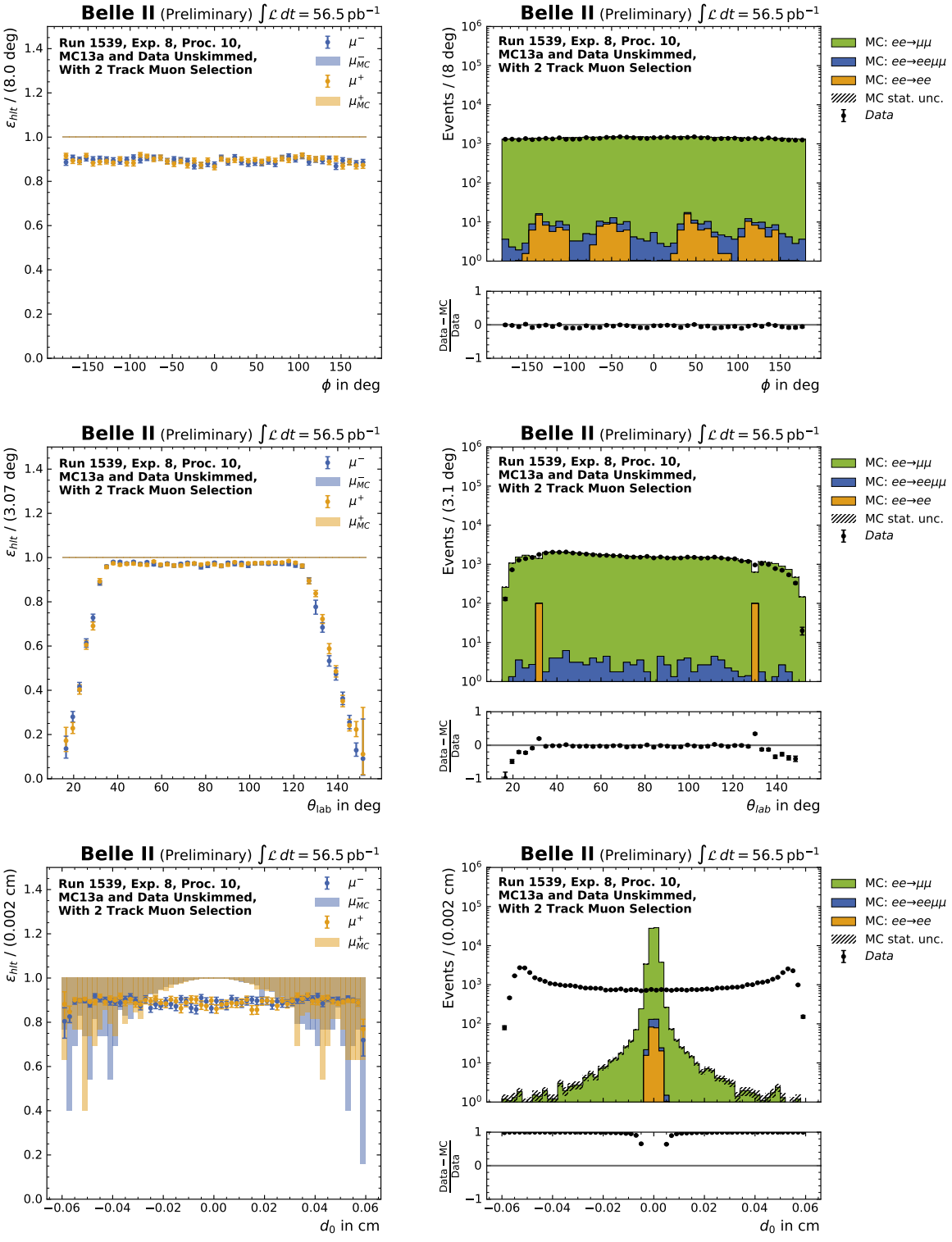


Fig. A.13.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

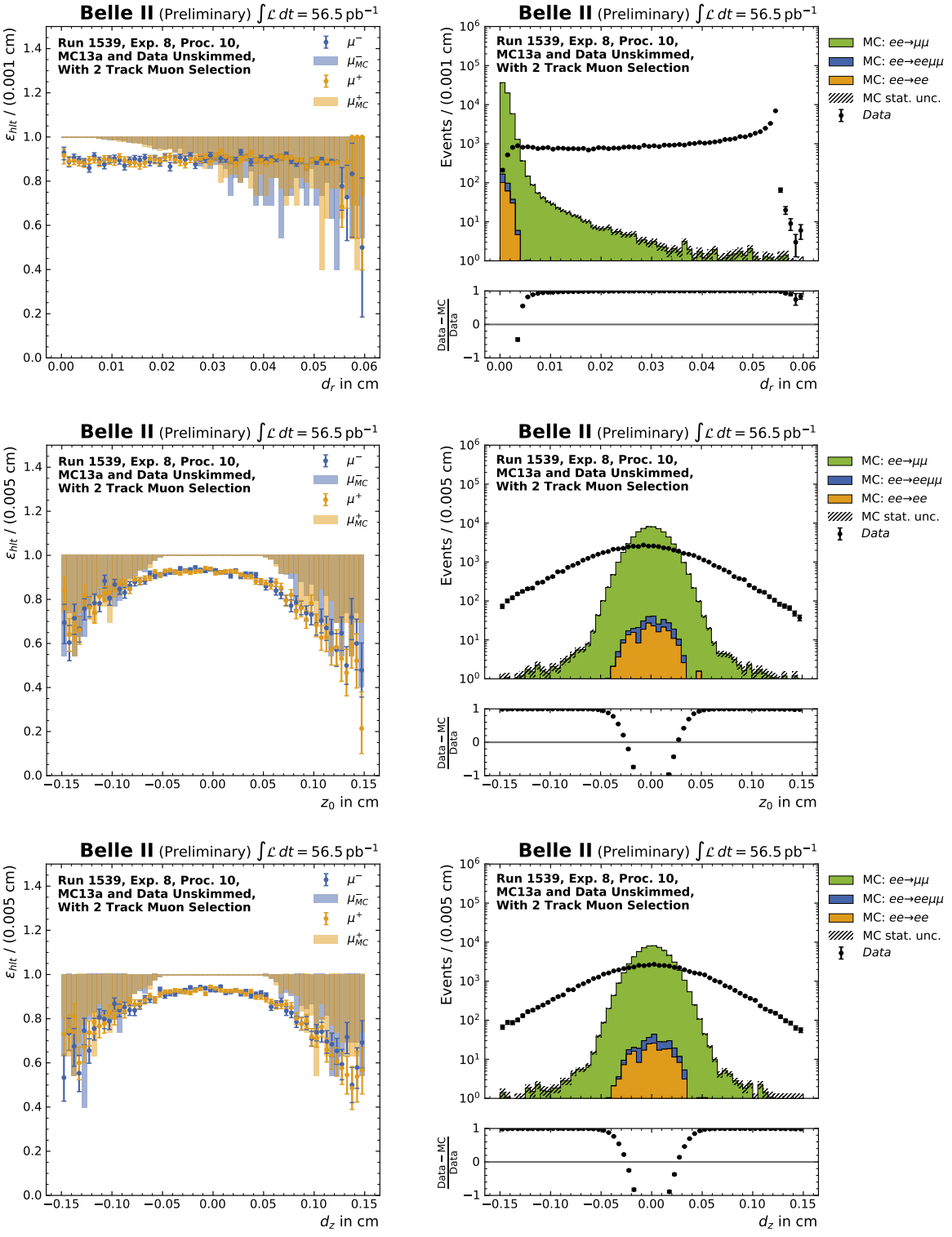


Fig. A.14.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

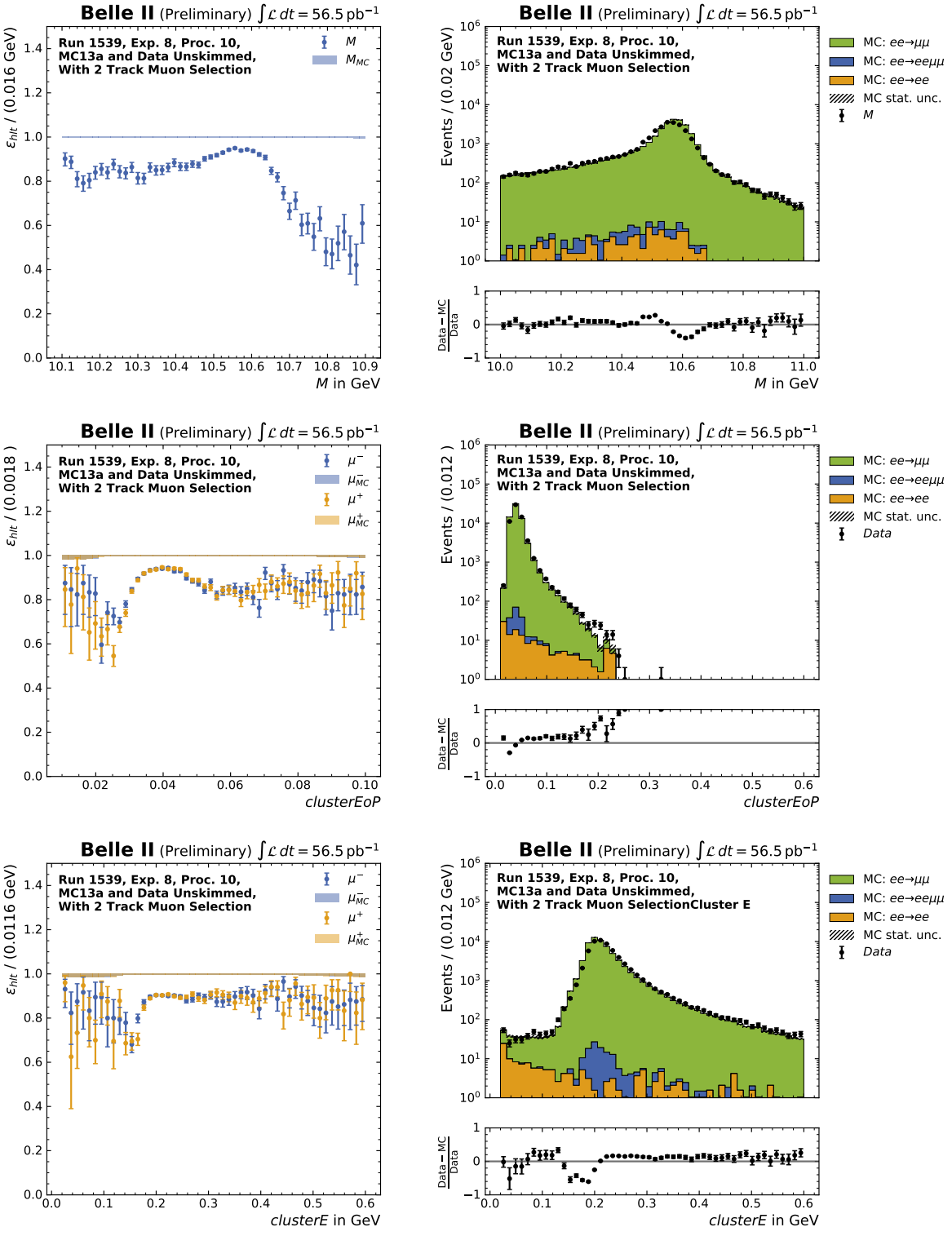


Fig. A.15.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

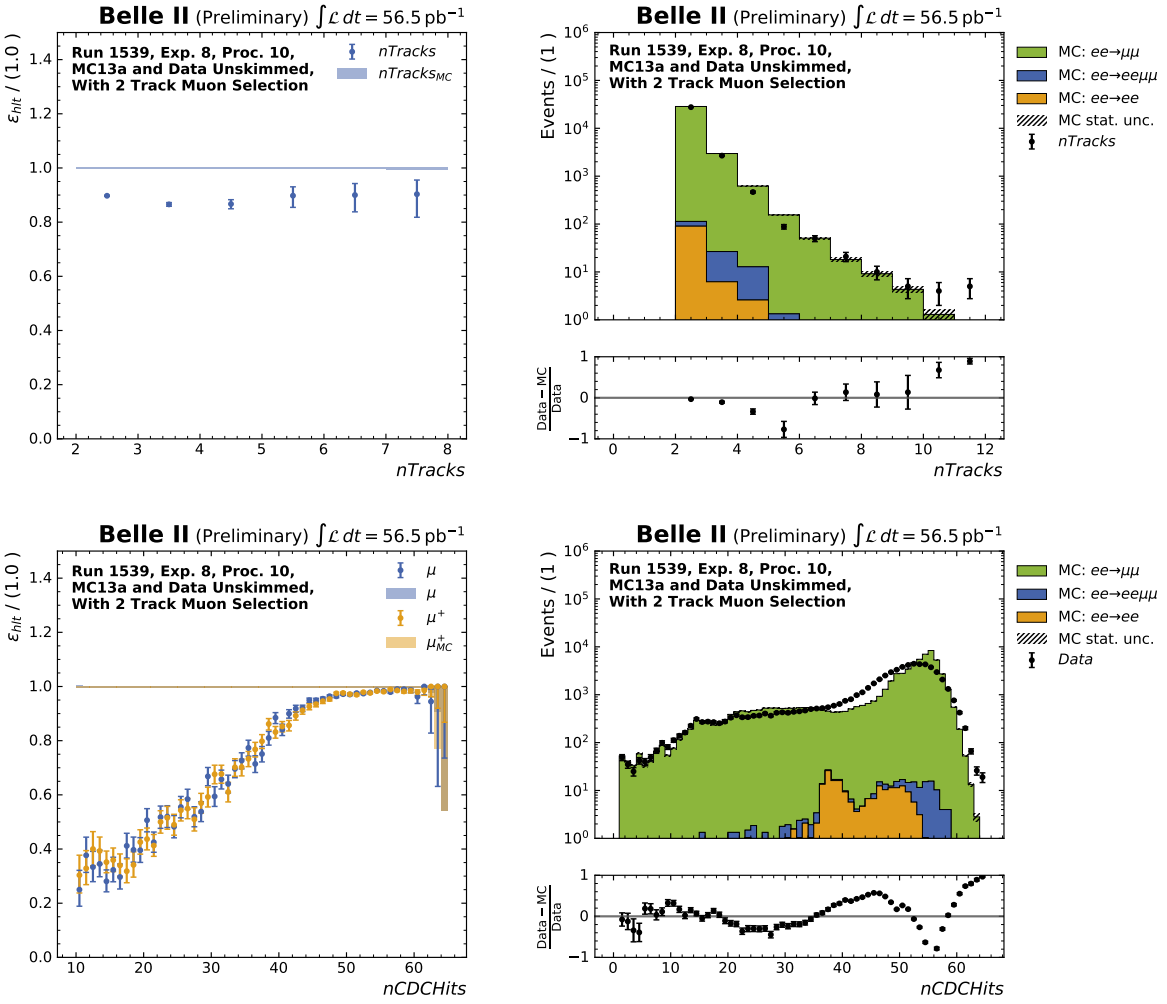


Fig. A.16.: The plots in the left column show the HLT efficiency as functions of particle properties. The plots in the right column show the distribution of number of events as function of these particle properties. All plots are in the lab frame.

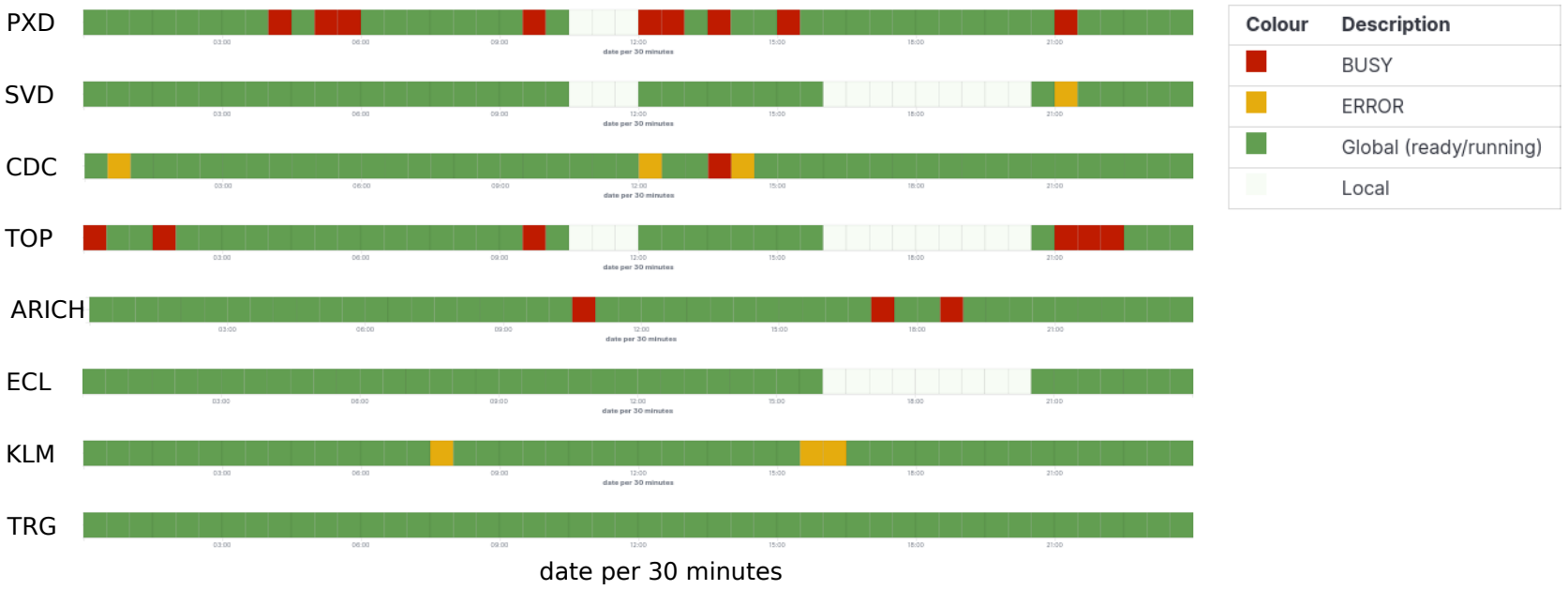


Fig. A.17.: Kibana visualization of the state of the subdetectors shown in every days run meeting.