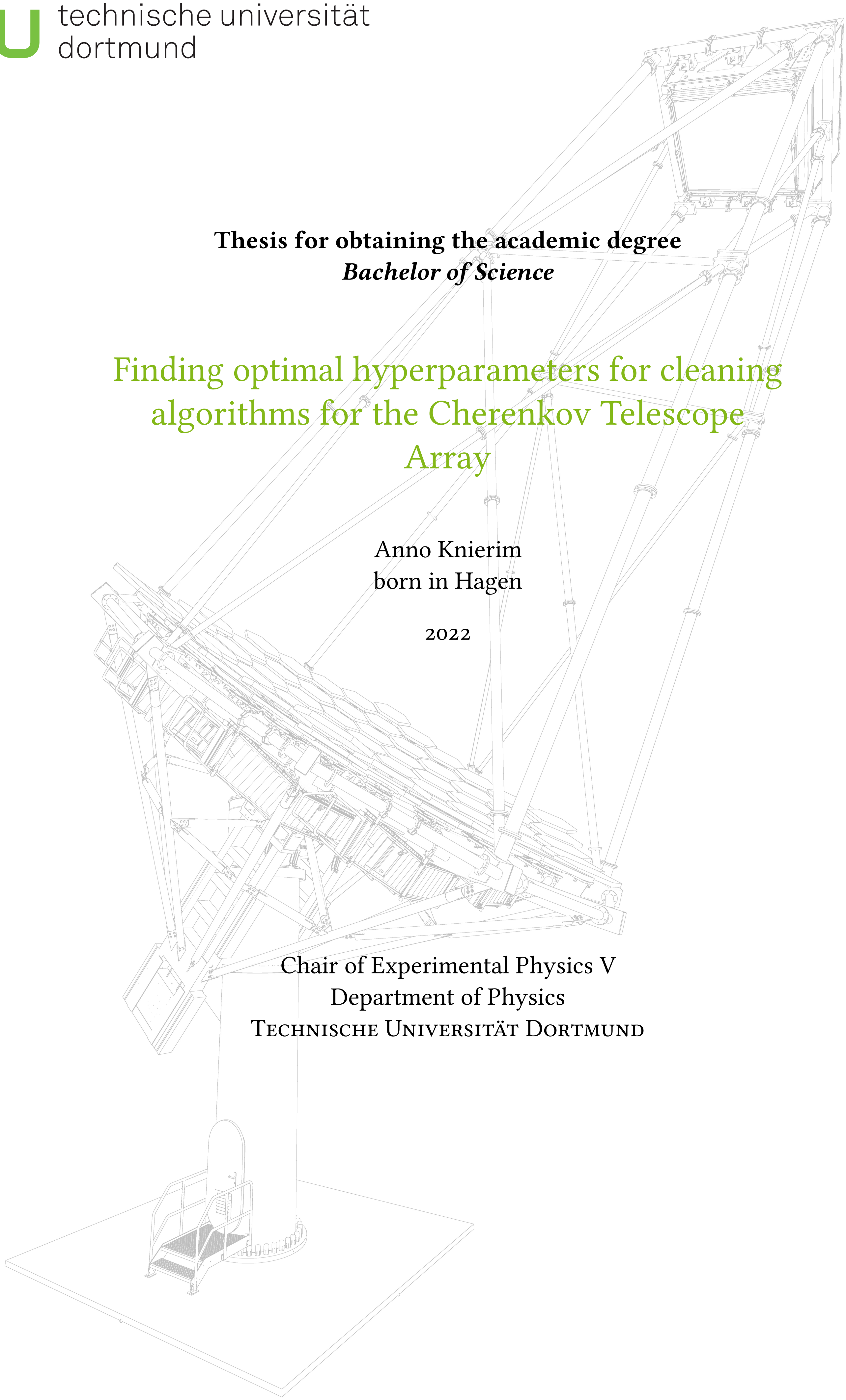
technische universität
dortmund

Thesis for obtaining the academic degree
Bachelor of Science

Finding optimal hyperparameters for cleaning algorithms for the Cherenkov Telescope Array



Anno Knierim
born in Hagen

2022

Chair of Experimental Physics V
Department of Physics
Technische Universität Dortmund

Reviewer: Prof. Dr. Dr. Wolfgang Rhode
Co-reviewer: Dr. Dirk Wiedner
Submission date: 7 September 2022

This thesis is set in Libertinus (Serif, Sans and Math) and Fira Code typeset using LaTeX with LuaTeX from TeXLive 2021.
Title graphic by Kai Brügge [9].

Contents

Gamma-Ray Astronomy

1

Astronomy, being one of the oldest sciences, is a vast field of study dating back to the earliest days of civilization, where astronomers have been studying the stars and the planets to understand the universe. It is, therefore, no surprise that astronomy spawned a great number of discoveries throughout the centuries. Whereas first observations were made by eye only, we now have access to a multitude of experiments and telescopes that deepen our understanding of the universe. With the discovery of cosmic rays (CR) by Victor Hess in the early 20th century, the new field of astroparticle physics was born [25].

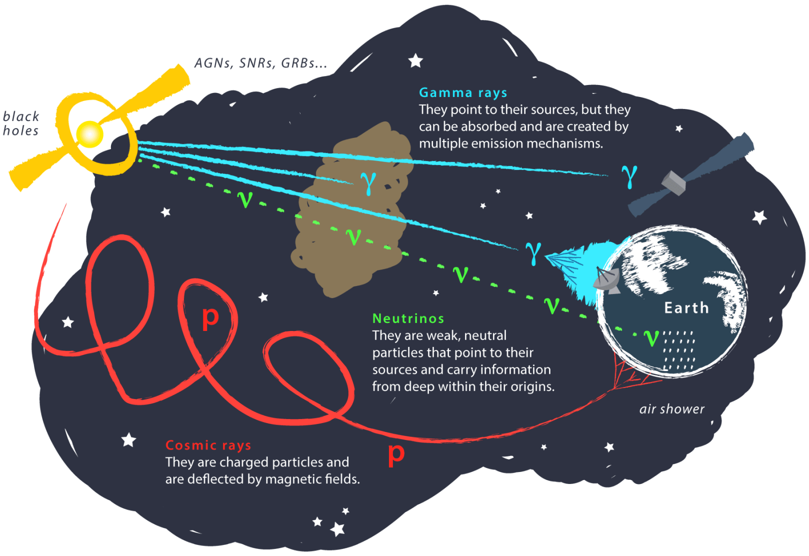


Figure 1.1: Different types of cosmic messengers on their way to Earth. Charged particles like protons and electrons are deflected by magnetic fields and therefore making it hard to pinpoint the source. Only the origin of photons and neutrinos can be reconstructed directly since they are uncharged particles and therefore travel in straight lines. However, photons can be absorbed or created in multiple mechanisms. Since neutrinos only rarely interact with matter via the weak force, their detection is significantly harder than that of photons [5].

Only 50 years after Hess' discovery, in 1961, Explorer XI [24], the first satellite experiment to measure gamma rays from space was launched, providing measurements of gamma rays above 50 MeV, thus starting gamma-ray astronomy from space-based experiments. Just a few years earlier neutrinos were already discovered by Cowan and Reines [12], with solar neutrinos being discovered by Davis et. al. [16]

in the 1960s at the Homestake experiment. The first detection of neutrinos from a supernova was made by the Kamiokande experiment in 1987 [22]. The most recent discovery of a cosmic messenger has been that of gravitational waves in 2015 [1].

Cosmic messengers come in different types, that are either charged or uncharged, as shown in Figure 1.1. CR like electrons, protons or atomic nuclei are charged particles making it difficult to trace back their origin as they are deflected by cosmic electromagnetic fields. Uncharged particles like photons or neutrinos, however, travel in straight lines, making it easier to reconstruct their origins.

While photons can be absorbed by dust clouds on their way to Earth, they are easier to detect than neutrinos, the latter having a very small cross-section and interacting only weakly with matter. This means that neutrino detectors and experiments, such as IceCube [4] or Super-Kamiokande [29], have to be large enough to detect neutrinos in sufficient quantities.

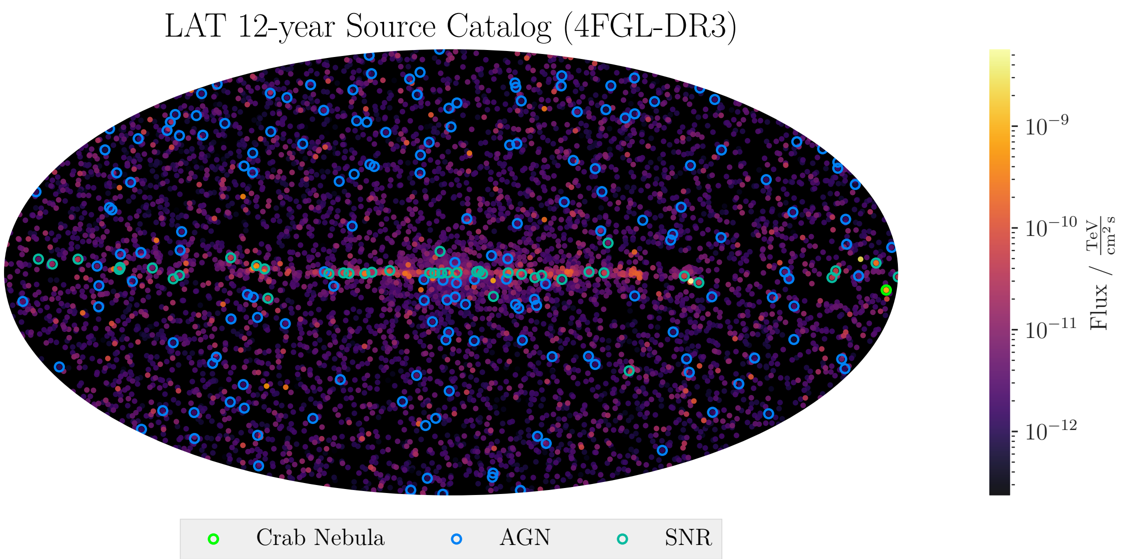


Figure 1.2: Mollweide projection of the *Fermi* Large Area Telescope (*Fermi*-LAT) 4FGL-DR3 catalog data of gamma-ray sources. The sky map shows the flux of gamma-ray sources in TeV/(cm^2 s) observed over a span of 12 years from the experiment's launch in 2008. Note the concentration of high-energy (HE) gamma-ray sources around the galactic plane. Additionally, the Crab Nebula (bright green circle, right hand side)—being a standard candle gamma-ray astronomy—is marked along with supernova remnants (SNRs) (darker green/turquoise circles) and active galactic nuclei (AGNs) (blue circles) that *Fermi*-LAT observed [2, 3]. Notice, that the AGNs shown here are but a subset of all AGNs observed by the experiment—the non-blazar active galaxies. They were chosen to show at least some other sources alongside the SNRs without obstructing the plot too much.

In recent years, gamma-ray astronomy has become an important research field in astroparticle physics. The term gamma ray is generally denoted as photons with energies above 100 keV [18]. Due to this high-energy nature, gamma rays pose some of the most powerful cosmic messengers in the universe and since photons at such energies cannot be produced by thermal processes, their origin has to be described by higher order processes involving charged particles. Some of the brightest sources of gamma rays in our galaxy are supernova remnants (SNRs) such as the Crab Nebular. The Crab Nebula, in particular, poses as a so-called standard candle with its constant flux of high-energy gamma rays (HEGR), allowing to test the performance of new experiments and telescopes. Other sources for gamma rays include gamma ray bursts (GRBs), Pulsars and active galactic nuclei (AGNs), the latter being one of the most common types of extragalactic gamma-ray sources. AGNs themselves can be classified further depending on the existence of a relativistic jet, the viewing angle of the observer or how bright they are.

Gamma rays can be detected by a variety of experiments, either ground- or space-based. Space-based experiments like the *Fermi*-LAT are usually more sensitive to lower energies, but their performance is lower for higher energies, as their effective collection area is comparatively small. Ground-based experiments, however, are more sensitive to higher energies, although they have to rely on the scattering of secondary particles in extensive air showers (EASs) induced by the primary particle in Earth's atmosphere. The gamma-ray sky as observed by *Fermi*-LAT over a span of 12 years is shown in Figure 1.2.

For the past two decades, ground-based Imaging Air Cherenkov Telescope (IACT) experiments like the Major Atmospheric Gamma-Ray Imaging Cherenkov (MAGIC) telescopes, the Very Energetic Radiation Imaging Telescope Array System (VERITAS) and the High Energy Stereoscopic System (H. E. S. S.) have been monitoring these very-high-energy gamma rays (VHE gamma rays) to gain an understanding of their production. This allowed us to determine different source classes inside and outside our galaxy, with the most important source class inside our galaxy being SNRs such as the Crab Nebula.

IACTs and the Cherenkov Telescope Array

2

Most modern gamma-ray observations are performed with either space-based experiments or with Imaging Air Cherenkov Telescopes (IACTs), which are ground-based telescopes or arrays of telescopes that use the Cherenkov light emitted by EASs in the atmosphere. In the following sections I will introduce IACTs and the Cherenkov Telescope Array (CTA) and explain the mechanisms that make it possible to observe gamma rays with these types of experiments.

2.1 Imaging Air Cherenkov Telescopes

Because of their ground-based setup, IACTs are taking advantage of the Earth's atmosphere to get a larger effective area than any space-based instrument. This is especially helpful for energies above 100 GeV, where the gamma-ray flux is low compared to lower energies that have higher fluxes and the effective collection area of even larger space-based instruments such as *Fermi*-LAT is too small [18, p. 256]. This means, that space-based experiments with their small effective area will see fewer high-energy events compared to ground-based experiments. The cosmic ray flux in Figure 2.1 shows this well: The flux decreases rapidly with higher energies, with only one high-energy photon per day and square meter reaching Earth from the Crab Nebula [26] and even fewer photons for energies in the EeV domain, where only 1 particle per square kilometer per year reaches Earth.

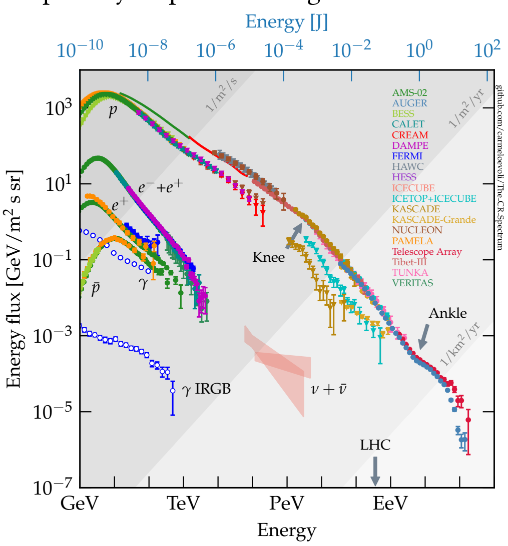
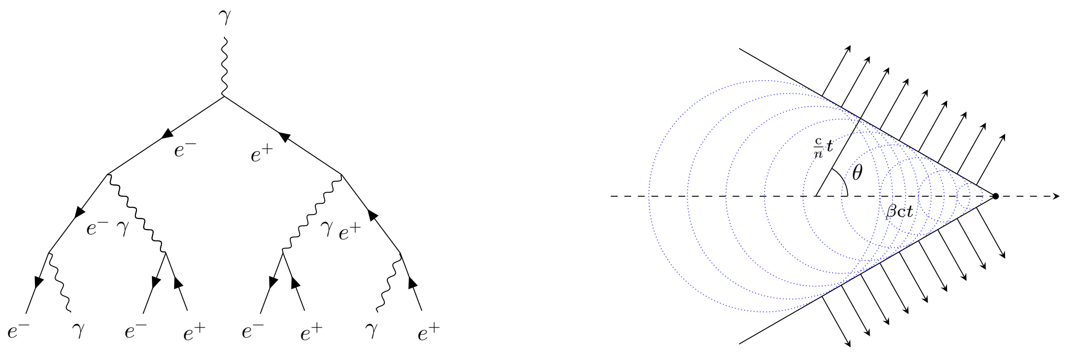


Figure 2.1: The cosmic ray flux as a function of energy. For very high energies the flux becomes very small with only 1 particle per square meter per year reaching Earth. For even higher energies in the domain of EeV this flux becomes 1 particle per square kilometer per year [17].

For high-energy gamma rays, Earth's atmosphere is opaque, so ground-based experiments rely on cascades of sub-particles in so-called EASs. For gamma-ray astronomy, the electromagnetic component of these EASs, shown in Figure 2.2a, is of interest, as these are produced by gamma rays. When a gamma ray interacts with Earth's atmosphere, it decays into an electron and a positron via pair production. These charged particles then emit more photons via bremsstrahlung. These photons, again, produce more charged particles, which in turn emit more photons. This process continues until any of these processes reaches energies below a certain threshold, i. e. 1022 keV for the pair production process. For these processes to happen, the particles have to be in the fields of the atoms of the Earth's atmosphere. As the particles travel at speeds faster than light in the atmosphere, they emit Cherenkov light at a fixed angle θ with respect to the refraction index n of the atmosphere and the factor $\beta = {}^{v}/_{c}$. The angle can be determined trigonometrically as

$$\cos\theta = \frac{1}{\beta n}, \tag{2.1}$$

as can be seen in Figure 2.2b.



(a) Simplified first steps of the Heitler model of the electromagnetic component of an extensive air shower. A gamma ray induces an electron and a positron via pair production that then emit more photons via bremsstrahlung.

(b) Cherenkov radiation (outward-pointing arrows) from a charged particle traveling at uniform velocity, faster than the speed of light in the medium. The dashed line shows the path of the particle over time.

Figure 2.2: The electromagnetic component of an extensive air shower (see (a)) is induced by gamma rays that then decay into electrons and positrons via pair production. These charged particles will emit more gammas via bremsstrahlung, which again will produce new electrons and positrons. This will continue until e. g. the energy of the gammas is below the threshold of 1022 keV for pair production. Another process is the production of Cherenkov light from the electrons and positrons, which is shown in (b). The Cherenkov light is the result of the charged particles traveling faster than the speed of light in the atmosphere, thus emitting photons in a fixed angle θ w. r. t. the refraction index n of the medium and the factor β.

Cherenkov light emitted by EASs can then be collected by an IACT's mirrors and detected by its camera within a timeframe of an order of nanoseconds. The resulting image will show the shape of the air shower and can be used to determine the shower's primary particles' properties and reconstruct its origin. As the hadronic component of EASs produce electromagnetic subshowers, IACTs have a dominant hadronic background, which has to be separated from the gamma ray-induced showers, for example with machine learning algorithms. This work, however, does not focus on these algorithms, but instead on cleaning algorithms that are used to separate the signal from the electronic noise coming from the camera as well as from the Night Sky Background (NSB). This is a vital step in the analysis of an event, as only a fraction of all pixels in the camera frame are part of the signal.

2.2 The Cherenkov Telescope Array

CTA is a new generation of IACTs that will consist of two sites, one of which will be built at the Observatorio del Roque de los Muchachos (ORM) on the Canarian island of La Palma while the other will be built in the southern hemisphere at the European Southern Observatorys (ESO) Paranal Observatory in the Atacama desert of northern Chile. In this work, I will focus on the northern array, also called CTA north, on La Palma.

CTA consists of three types of telescopes, shown in Figure 2.3, each being sensitive to a different energy range [13]:

Large-Sized Telescope (LST): The largest telescopes in CTA have primary deflector diameter of 23 m with a field of view of 4.3 deg and are sensitive to energy ranges from 20 GeV to 150 GeV.

Medium-Sized Telescope (MST): The MSTs have a primary deflector diameter of 11.5 m with a field of view of 7.7 deg for their NectarCam and are sensitive to energy ranges from 150 GeV to 5 TeV.

Small-Sized Telescope (SST): Being the smallest telescope type in CTA, the SSTs have a primary deflector diameter of 4.3 m, a secondary deflector diameter of 1.8 m with a field of view of 10.5 deg and are sensitive to energy ranges from 5 TeV to 300 TeV.

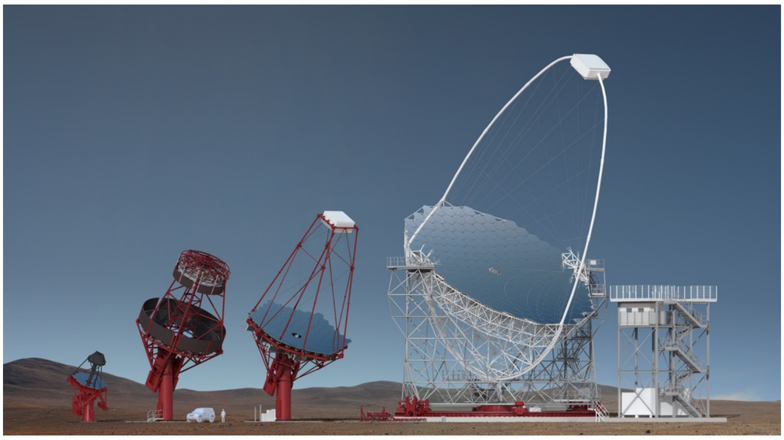


Figure 2.3: A rendered image of the 3 telescope prototypes in CTA. From left: The SST, two types of MST (SCT and MST) and the LST. This work will not feature the SCT, as the used simulation data contains only LST and MST data for the northern array. Image credit: Gabriel Pérez Diaz, IAC [31].

Since the southern hemisphere has a better view of the galactic center than the northern hemisphere, only CTA south will feature the SSTs along the other two telescope types. CTA north will only feature MSTs and LSTs. In its full configuration, CTA would feature 70 SSTs, 25 MSTs and 4 LSTs at CTA south, and 15 MSTs and 4 LSTs at CTA north. Due to financial limitations, however, the reduced layout—called Alpha Configuration— is the current official layout for both CTA north and south. This work will focus on the northern site, consisting of 9 MSTs and 4 LSTs [14]. Figure 2.4 shows the planned Alpha Configuration layout of both sites of CTA where CTA south is displayed with its 37 SSTs, 14 MSTs and the 4 LSTs in the center [15].

The MST's and LST's cameras are based on photo multiplier tube (PMT) photodetectors, while the SST's camera is based on a silicon photomultiplier (SiPM) photodetector. Both detector types provide fast

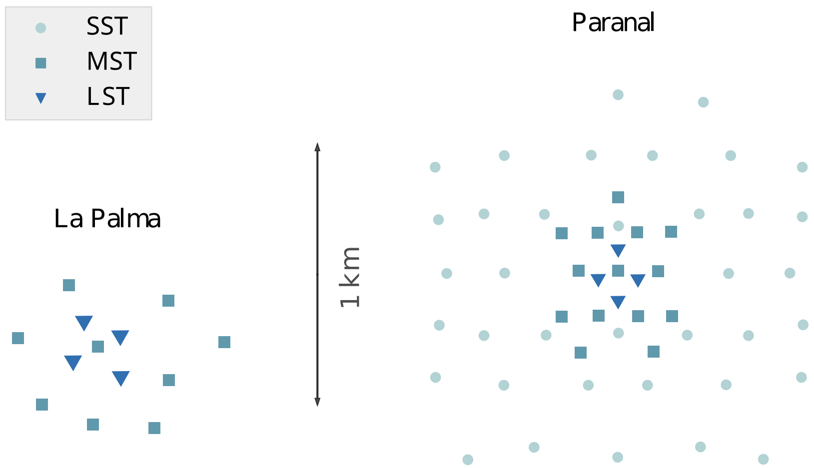


Figure 2.4: Alpha Configuration layout of the CTA north and CTA south site. CTA south would feature 37 SSTs, 14 MSTs and 4 LSTs [15] in it's Alpha Configuration, while CTA north would feature 9 MSTs and 4 LSTs [14]. This figure was adapted from Kai Brügge in his Ph.D. thesis [9].

response times in the order of nanoseconds. PMTs can detect small amounts of electromagnetic radiation due to their amplification process. However, this amplification process also amplifies the noise of the detector and also needs higher voltages than SiPMs. The latter, on the other hand, needs to be cooled, as SiPMs are prone to have a higher dark current at higher temperatures. The MST and LST will feature a camera with 1855 pixels.

Together with gravitational wave and neutrino observatories as well as with other photon observatories, CTA will probe the universe in its high-energy (HE) domain, allowing to study a broad field of key questions in astrophysics. As such, CTA's scientific goals can be divided into three categories [11]:

Origin and Role of relativistic Cosmic Particles: CTA will try to find the sources of HE cosmic particle acceleration in the universe. This includes the search for the mechanisms that accelerate cosmic particles. Also, CTA will try to answer what role these particles play in feedback on star formation and galaxy evolution.

Extreme Environments: CTA aims to deepen our understanding of the processes that are at work close to neutron stars and black holes, the characteristics of relativistic jets and pulsar wind nebula (PWN). Also, it will probe how intense radiation fields and magnetic fields are and how these would evolve over cosmic time scales.

Frontiers in Physics: CTA will try to answer questions about the nature of dark matter and its distribution in the universe. As such, the existence of axion-like particles will be probed. Furthermore, CTA will search for quantum gravitational effects on photon propagation.

With CTA probing the very-high-energy (VHE) gamma-ray sky and a resolution outperforming predecessors, like H. E. S. S. or MAGIC, it will be able to observe a broad variety of sources, allowing insights into interaction processes of VHE gamma rays as well as their origin.

Data processing with ctapipe 3

The analysis in this work was done with the open-source low-level data processing software for CTA, `ctapipe` [23]. The version used is the development version `0.15.1.dev166+gf26107f`, henceforth shortened to version `0.15.1`. The goal of `ctapipe` is to provide a complete analysis framework ranging from data calibration and image extraction to the reconstruction of events and the analysis of their properties. In this chapter, I will first describe how the data were simulated in section 3.1 and then, in section 3.2, explain the different data levels and `ctapipe`'s various analysis steps. In section 3.3 I will explain the full pipeline created for this work.

3.1 Data simulation

In its current in-development state, `ctapipe` is used and tested with simulated data, as the software can only be sufficiently tested when the output is compared to truth data. This data is created with the help of Monte Carlo (MC) simulations. The software used for these simulations is called Cosmic Ray Simulations for Kascade (CORSIKA) [21] and allows for detailed simulation of EASs initiated by high-energy cosmic rays (HECRs). The showers are observed by a virtual IACT array, the `sim_telarray` [8]. The resulting so-called `simtel` data is used in `ctapipe` and can be processed as described in section 3.2 and Figure 3.1. Some of the data properties are listed in Table 3.1.

This data is the basis for the analysis in this work. A total of 20 `simtel` runs of diffuse gamma data were used for the initial analysis, i. e. the testing of different parameters as described in section 4.2. The individual runs were processed into **DL1** files and merged before being processed for each cleaning setting.

Once promising settings are found, a larger dataset containing 987 runs is processed with these selected settings, allowing for more statistics and a better comparison of the cleaning algorithms.

Table 3.1: Simtel data properties of the CORSIKA simulation used for the datasets in this works analysis.

	Diffuse Gamma data
Energy range / TeV	0.003–330
Zenith angle / °	20
View cone angle / °	10
Number of showers	50 000
Spectral index	−2.0
Maximum scatter range / m	1900

3.2 Data Levels in `ctapipe`

There are several data levels in `ctapipe`, spanning from the raw data **R0** to the reconstructed events **DL2**, with the raw data levels being denoted by an **R** and the calibrated data levels by a **D**. Figure 3.1 shows a simplified overview of the data levels and the analysis steps. The raw data level **R0** is the data that comes directly from the photodetectors. The time-resolved signal of the data is calibrated from **R0** to **R1**. Then the data volume gets reduced (**R1** →**DL0**) by a selection of waveforms.

The **DL0** data level is the first level being stored and also the first level to be processed from the simulation datasets used in this work. From **DL0** to **DL1a**, the images of the data are extracted from the time pulses, which are then cleaned by a cleaning algorithm, allowing for parametrization of the events (**DL1a** →**DL1b**). The parametrized events can then be reconstructed (**DL1b** →**DL2**) and stored on the **DL2** data level.

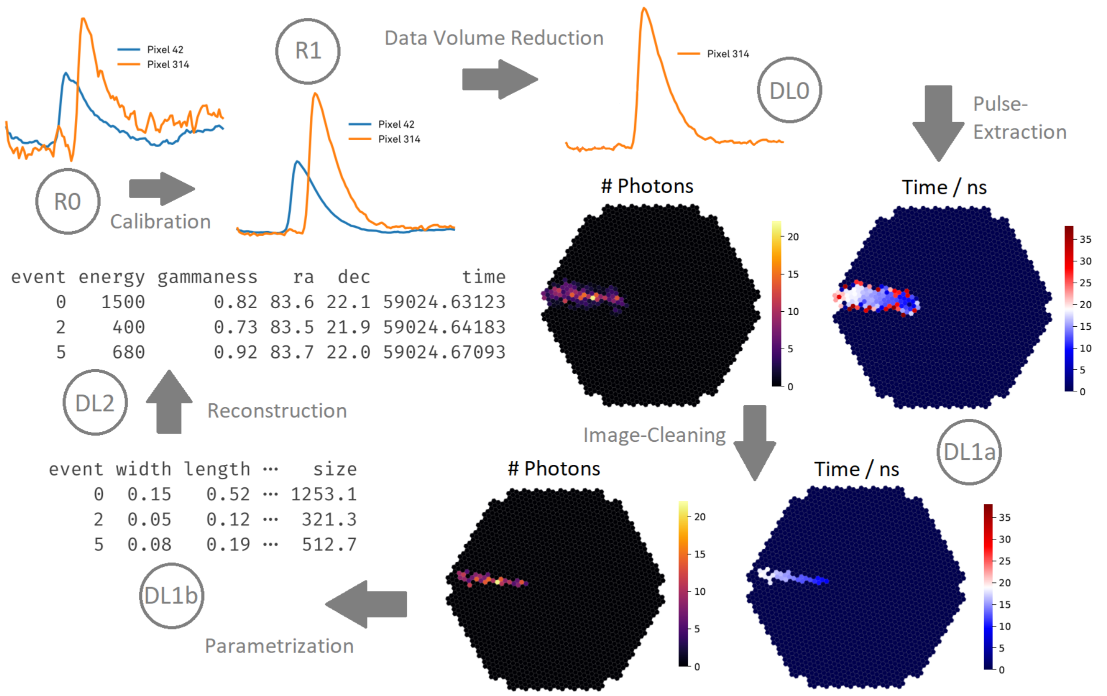


Figure 3.1: Data levels in `ctapipe`. Raw data levels are denoted by an **R** and calibrated data levels by a **D**. The raw data first gets calibrated (**R0** →**R1**) and then reduced in volume by selecting waveforms (**R1** →**DL0**). From there the images are extracted (**DL0** →**DL1a**) and cleaned with a cleaning algorithm. This allows for parametrization of the events (**DL1a** →**DL1b**). The parametrized events can then be reconstructed (**DL1b** →**DL2**) [26, 19].

3.3 This work's data processing pipeline

In this work, the data processing pipeline is as follows: First, several simulation data runs are selected and processed with `ctapipe`. Then, the datasets are merged and serve as a basis for a re-run of the combined dataset with various parameter combinations described in chapter 4.

The settings for `ctapipe` are saved in two configuration files: First, a file, that sets up the cleaning process and what data to write to the output file. For the preprocessing step of this work's pipeline, the cleaning settings are not relevant, as this step only serves to create a merged dataset. The second file contains a list of all the allowed telescope IDs. This allows a selection of the telescopes that are used in the analysis. This is especially helpful if one wants to only analyze MST-related data, like in this work.

For the re-run of the combined dataset, the cleaning settings are used[1], however, as this step is the heart of this work's search for the optimal hyperparameters. The configuration files used are shown in appendix 1.

Each resulting dataset can then be processed on an array or telescope data level, resulting in **DL1a** image plots, values for the angular resolution and the efficiency as well as metrics for the performance. An in-detail description of how this helps to compare the performance of the different cleaning algorithms can be found in section 4.2. A schematic overview of the data processing pipeline of this work is shown in Figure 3.2.

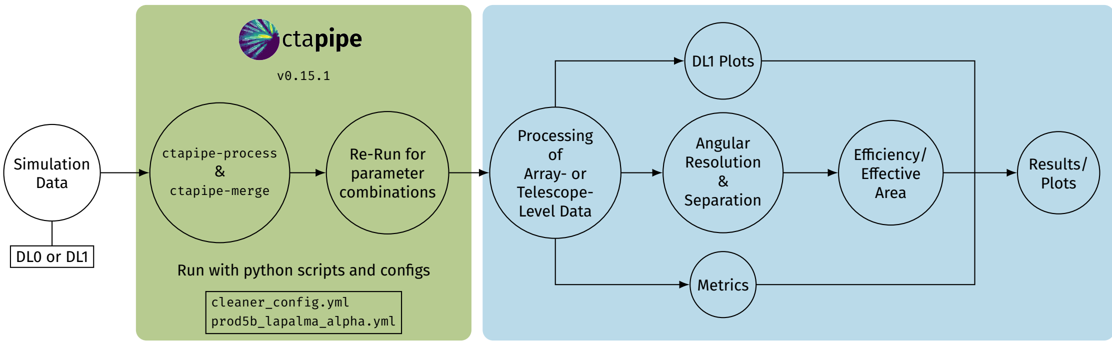


Figure 3.2: Schematic overview of the data pipeline used for this work. Single runs of the simulation data are processed with `ctapipe-process` and then merged via the tool `ctapipe-merge` (green shaded area). The merged data is then processed on the array or telescope data level resulting in scores for metrics as well as **DL1** `images` and plots for the angular resolution and the effective area (blue shaded area).

[1] As opposed to the preprocessing step

Finding Optimal Hyperparameters for the Cleaning Algorithms 4

This work aims to find optimal hyperparameters for the cleaning algorithms used in `ctapipe`. Optimizing the cleaning step of the **DL1** data level in section 3.2 can most certainly lead to a better reconstruction of the events in a dataset. Therefore, a tweaking of the available parameters of each cleaner is necessary. In this chapter, I will first introduce the cleaning algorithms and their parameters in section 4.1 and then describe the procedure to find optimal hyperparameters in section 4.2.

4.1 Cleaning Algorithms

Version `0.15.1` of `ctapipe` features four cleaning algorithms, two of which are time-based. The `TailcutsImageCleaner` algorithm is the most basic algorithm of the four and serves as a good starting point for the development of new cleaning algorithms. Its first step is to select all pixels that are above a certain threshold, the core threshold Q_c. These pixels are the core part of the signal and are the brightest. The `TailcutsImageCleaner` algorithm then selects all pixels that are above the boundary threshold Q_b and are neighboring the core pixels. A visualization of the algorithm is shown in Figure 4.1 for the default values of the algorithm.

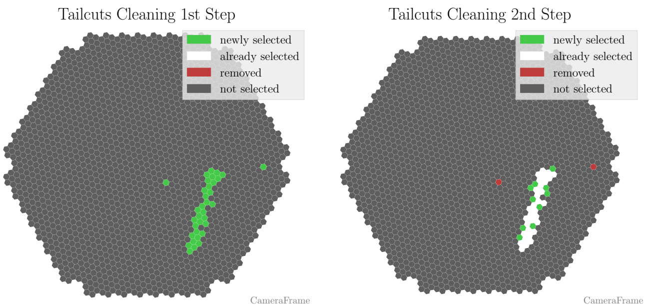


Figure 4.1: Visualization of the `TailcutsImageCleaner` algorithm for a MST NectarCam image. First, all pixels above the core threshold are selected. Then, all pixels neighboring the core pixels that are above the boundary threshold are selected.

The `MARSImageCleaner` algorithm [**mars**] is very much based on the `TailcutsImageCleaner` algorithm and features an additional step, in that it also selects all neighbors of a neighbor of a core pixel, if they are above the boundary threshold. The three steps for the `MARSImageCleaner` algorithm are shown in Figure 4.2 for the default values of the algorithm.

The `FACTImageCleaner` algorithm is a time-based cleaning algorithm that first selects all pixels that are above the `core_threshold`. Then, all pixels that have less than N neighbors are removed. The

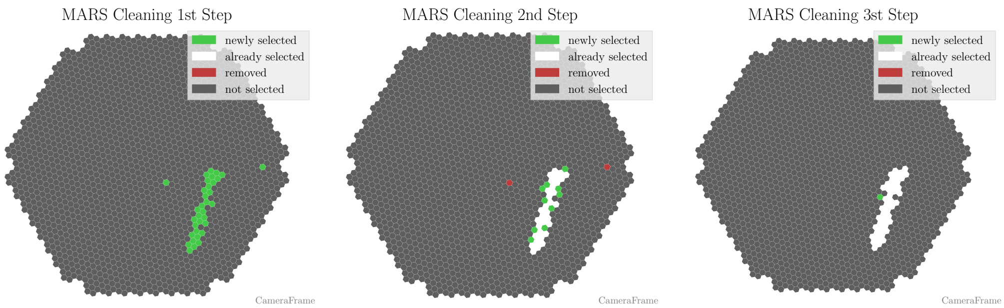


Figure 4.2: Visualization of the `MARSImageCleaner` algorithm for a MST NectarCam image. The first two steps are identical to the `TailcutsImageCleaner` algorithm. The third step selects all neighbors of a neighbor of a core pixel, if they are above the `boundary_threshold`.

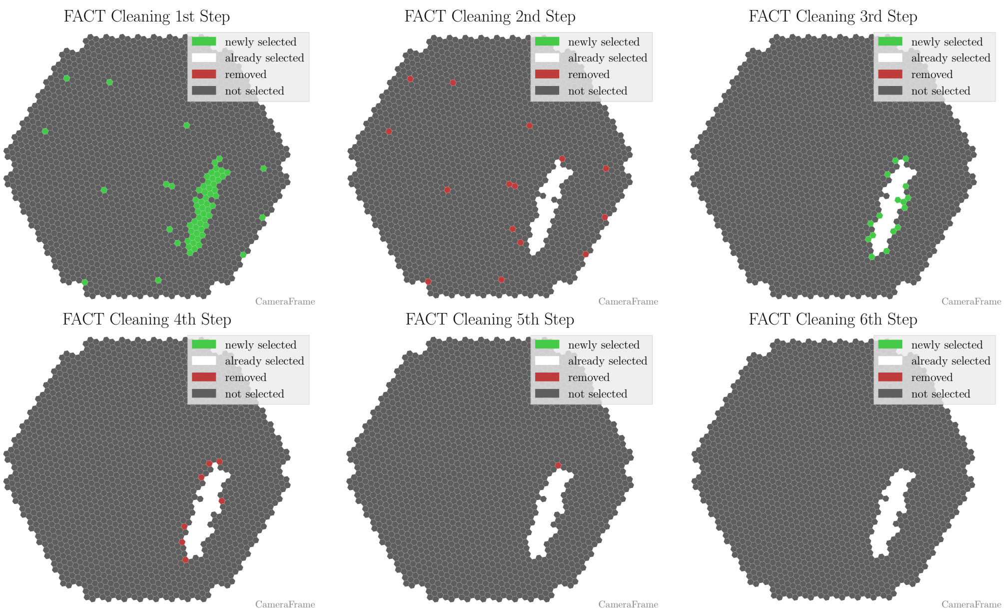


Figure 4.3: Visualization of the `FACTImageCleaner` algorithm [**fact**] for a MST Nectar-Cam image. The first step selects all pixels above the `core_threshold`. The second step removes all pixels that have less than N neighbors. The third step selects all pixels neighboring the remaining pixels that are above the `boundary_threshold`. The fourth step removes all pixels that have less than N neighbors, that have arrived within a given timeframe. The fifth and sixth steps are analogous to steps two and four.

min_number_neighbors parameter is set to 2 by default. For the third step, all pixels neighboring the remaining pixels that are above the boundary_threshold are selected. After that, all pixels that have less than N neighbors and that have arrived within a given timeframe are removed. The time_limit parameter is set to 5 ns by default. Again, all pixels with less than N neighbors are removed. The last step once again removes pixels with less than N neighbors, arriving within the given time limit. The visualization of the algorithm is shown in Figure 4.3 for the default values of the algorithm.

The TimeConstrainedImageCleaner algorithm is another time-based algorithm, coming from the MAGIC collaboration. It first selects all pixels that are above the core_threshold [**tcc**]. Then, all pixels that have less than N neighbors are removed. The min_number_neighbors parameter is set to 1 by default. After that, all core pixels whose arrival times are within a given timeframe of the average arrival time. This time_limit_core parameter is set to 4.5 ns by default. As a fourth step, the TimeConstrainedImageCleaner algorithm finds all pixels above the boundary_threshold. Then, all pixels with less than N neighbors arriving within a given timeframe are removed. This time_limit_boundary parameter is set to 1.5 ns by default. The visualization of the algorithm is shown in Figure 4.4 for the default values of the algorithm.

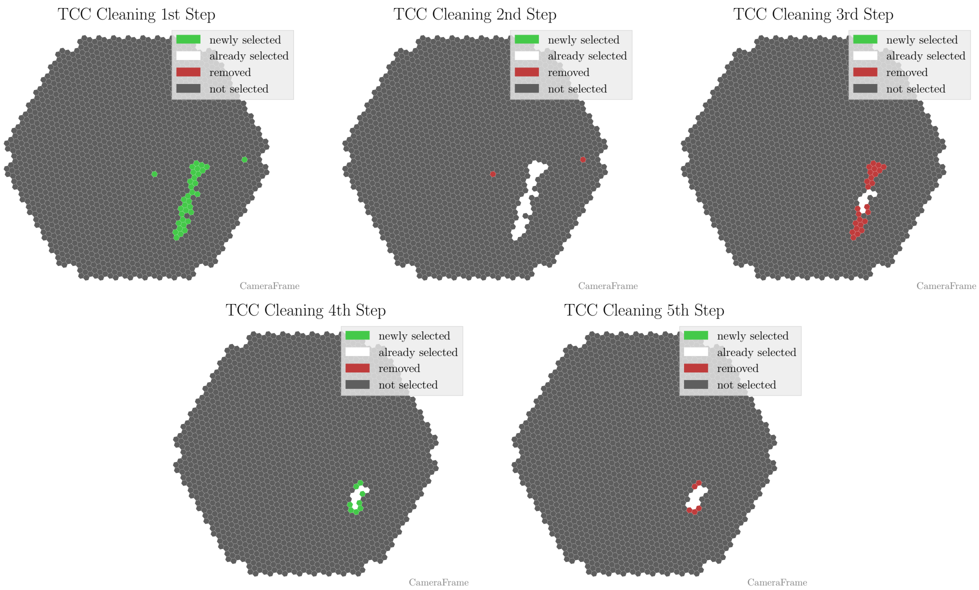


Figure 4.4: Visualization of the TimeConstrainedImageCleaner algorithm for a MST NectarCam image. The first step selects all pixels above the core_threshold. The second step removes all pixels that have less than N neighbors. The third step selects all pixels whose arrival times are within a given timeframe of the time_limit_core parameter. The fourth step selects all pixels above the boundary_threshold. The fifth step removes all pixels with less than N neighbors, that have arrived within a given timeframe of the time_limit_boundary parameter.

4.2 Hyperparameters

The hyperparameters of each cleaning algorithm are set in a specific configuration file. There, the user can set and change the parameters, shown in Table 4.1. TailcutsImageCleaner and MARSImageCleaner can be set-up with only three parameters: the `picture_threshold`, the `boundary_threshold` and `min_number_picture_neighbors`. The time-based algorithms have additional parameters: First of all a `time_limit` for `FACTImageCleaner` and then a `time_limit_core` as well as a `time_limit_boundary` for `TimeConstrainedImageCleaner`. The full configuration files for the default settings of each cleaning

Table 4.1: The four cleaning algorithms and their hyperparameters. Being the most basic algorithms, `TailcutsImageCleaner` and `MARSImageCleaner` have only three parameters, while the `FACTImageCleaner` and `TimeConstrainedImageCleaner` algorithms have one and two additional time-based parameters, respectively. This table shows the default values and, if the parameters have them, also their units, as they are implemented in the `ctapipe` source code for version `0.15.1`.

Cleaning Algorithm	Hyperparameter	Default Values
`TailcutsImageCleaner`	`picture_threshold`	7 p.e.
	`boundary_threshold`	5 p.e.
	`min_number_picture_neighbors`	0
`MARSImageCleaner`	`picture_threshold`	7 p.e.
	`boundary_threshold`	5 p.e.
	`min_number_picture_neighbors`	0
`FACTImageCleaner`	`picture_threshold`	4 p.e.
	`boundary_threshold`	2 p.e.
	`time_limit`	5 ns
	`min_number_picture_neighbors`	2
`TimeConstrainedImageCleaner`	`picture_threshold`	7 p.e.
	`boundary_threshold`	5 p.e.
	`time_limit_core`	4.5 ns
	`time_limit_boundary`	1.5 ns
	`min_number_picture_neighbors`	1

algorithm are listed in appendix 1. There, the configuration files for the allowed telescopes as described in section 3.3 are shown, too.

To find the optimal hyperparameters, I used the `ParameterGrid` class from the `sklearn.model_selection` module in `scikit-learn` [28]. The `ParameterGrid` class is useful to create a dictionary of all possible combinations of a list of given parameters. These combinations are then written to a config file and processed with `ctapipe`. The output is hundreds of datasets equal to the number of combinations of the given parameters, each with a different setting for the cleaning performed on the data. The listing below shows the parameters fed into the `ParameterGrid`:

```
common_params = {
    "picture_quantiles": (0.995, 0.999, 0.9992, 0.9995, 0.9997, 0.9999),
    "boundary_threshold_ratio": (1/4, 1/3, 1/2, 2/3, 3/4),
    "min_number_picture_neighbors": (1, 2, 3, 4, 5)
}
fact_params = {
    "time_limit": (1.0, 2.0, 4.0, 5.0, 6.0, 10.0, 12.0)
}
tcc_params = {
    "time_limit_core": (9.0, 12.0, 15.0, 18.0, 20.0),
    "time_limit_boundary" = (4.5, 9.0, 12.0, 15.0)
}
```

The name `common_params` is used here to refer to the dictionary of parameters that are common to all cleaning algorithms, while `fact_params` and `tcc_params` denote the additional hyperparameters exclusive to `FACTImageCleaner` and `TimeConstrainedImageCleaner`. The `picture_quantiles` parameter determines how many percent of all pixels will be below the `picture_threshold`. This is especially useful, as this results in a more precise value for the threshold[1]. The `boundary_threshold_ratio` sets the ratio of the `boundary_threshold` w. r. t. the `picture_threshold`. The other parameters, `min_number_picture_neighbors` and the time limits are the same as explained before and are, therefore, absolute values.

The parameters result in 150 combinations for `TailcutsImageCleaner` and `MARSImageCleaner` each, 1050 combinations for `FACTImageCleaner` and 3000 combinations for `TimeConstrainedImageCleaner`. Since it's impossible to find the optimal settings just by looking at the cleaned images of the datasets, a combined metric is necessary to evaluate each cleaner's performance for the parameter combinations.

In this work, I chose to first look at the efficiency

$$\mathrm{Eff} = \frac{n_{\mathrm{reco}}}{n_{\mathrm{total}}}, \tag{4.1}$$

where n_{reco} is the number of reconstructed events and n_{total} the total number of events in the dataset. By taking the mean of the efficiency over all energy bins of the whole dataset, one can sort the datasets by setting upper and lower bounds for the efficiency. The intervals lengths were chosen as 0.05, i. e. 5 %, resulting in a total of 20 intervals.

For each interval, the mean angular resolution is calculated. This is achieved by determining the 68 % containment of the angular distance distribution, i. e. the angular distance between the reconstructed origin and the true origin of the shower. This angular distance, or separation, is calculated with the `astropy.coordinates.angular_separation` function from `astropy` [6, 7]. The dataset with the lowest mean angular resolution is the best performing for each interval. Not all events will ever be properly reconstructed for any parameter combination, so there won't be a mean angular resolution for all intervals, but only a subset. Also, the mean angular resolution may be higher when more events were reconstructed. As a result, a trade-off between efficiency and angular resolution is necessary.

For the performance analysis of the algorithms, I calculated metrics such as, but not limited to, the True Positive Rate (tpr) or recall, the False Positive Rate (fpr) or fall-out and the True Negative Rate (tnr) or specificity. A list of all used metrics can be seen in Table 4.2.

Table 4.2: The metrics used for this works analysis as well as their calculation. Right: Confusion matrix for a binary (positive/negative) classification.

Metric	Calculation
True Positive Rate (tpr)	$\frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}$
False Positive Rate (fpr)	$\frac{\mathrm{fp}}{\mathrm{fp}+\mathrm{tn}}$
True Negative Rate (tnr)	$\frac{\mathrm{tn}}{\mathrm{tn}+\mathrm{fp}}$
False Negative Rate (fnr)	$\frac{\mathrm{fn}}{\mathrm{fn}+\mathrm{tp}}$
Positive Predictive Value (ppv)	$\frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fp}}$
Accuracy (acc)	$\frac{\mathrm{tp}+\mathrm{tn}}{\mathrm{tp}+\mathrm{fp}+\mathrm{tn}+\mathrm{fn}}$
Balanced Accuracy (ba)	$\frac{\mathrm{tpr}+\mathrm{tnr}}{2}$

		Prediction	
		positive	negative
Label	pos.	True Positive (tp)	False Negative (fn)
	neg.	False Positive (fp)	True Negative (tn)

[1] As opposed to setting the thresholds by hand, e. g. 4 to 10 in 0.5 increments

Results 5

In this chapter, the results of the analysis are presented. First I present the results of the efficiency analysis in section 5.1. The initial tests of parameter combinations are based on a small dataset consisting of 20 runs, i. e. 12 668 events. This is due to the number of parameter combinations that are tested, as more runs would increase the processing time immensely. Furthermore, I present the results of the angular resolution for a combined metric with the efficiency. Then, in section 5.2, the metrics of each resulting combination of hyperparameters are presented. In section 5.3, the performance of each cleaning algorithm compared to the default settings is presented. Finally, a comparison of the different cleaning algorithms is presented in section 5.4.

5.1 Analysis of the efficiency and the angular resolution

To narrow down possible candidates for the optimal hyperparameters, I first analyzed the efficiency of the different cleaning algorithms. The efficiency is determined by the number of events that are reconstructed after cleaning. For this work I chose 20 intervals within 0 and 1 with a step size of 0.05. The mean efficiency is then calculated as the mean of Equation 4.1. For each interval those datasets are selected, where the mean efficiency lies between the lower and upper bound of the interval. Then the minimum angular resolution is determined for each interval. The parameters of these datasets are then selected to be the optimal parameters for each cleaning algorithm. This not only allows for a comparison of the cleaners but also a decision on a trade-off between the efficiency and the angular resolution, namely having a better angular resolution, but a lower efficiency or a higher efficiency but a higher and therefore worse angular resolution. The results for the mean efficiency are listed in Table 5.1 and the results for the mean angular resolution in Table 5.2.

As one can see, not all cleaning algorithms have valid values for each interval, peaking at a maximum of around 45 % to 50 % of successfully reconstructed events. This is because not all events are stereo events, i. e. events, where two or more telescopes were triggered. The remaining events are therefore mono events and do not contribute to either the efficiency or the angular resolution. The efficiency would be higher, of course, for a full-array analysis, but that would mean also including LSTs data, which for this work would not help find the optimal parameters, since it is better to analyze the telescope types separately. The reason for the latter is, that optimizing the telescopes by type would lead to better results for the hyperparameters.

From the tables, one can see that there is a clear trade-off in choosing between efficiency and angular resolution. As such, for further comparison of the cleaning algorithms, the corresponding datasets for the efficiency and the angular resolution are plotted in Figure 5.1 for the intervals [0.25, 0.30] and [0.45, 0.50]. The reason for this is that these are the minimum and maximum intervals w. r. t. the efficiency, where all cleaners have valid values.

Furthermore, the mean angular resolution is plotted against the efficiency in Figure 5.2.

Table 5.1: The results of the analysis for the mean efficiency of each cleaning algorithm. The efficiency is calculated as the ratio of the number of reconstructed events n_{reco} and the number of total events n_{total}. The table lists the lower and upper limits of each efficiency interval. The efficiency is then calculated as the mean over the whole energy range of the dataset and each listed efficiency is the one where the mean angular resolution is minimal for the given interval. Notice how not all cleaning algorithms have valid results for all efficiency intervals.

		Mean Efficiency			
$\mathrm{Eff}_{\mathrm{lower}}$	$\mathrm{Eff}_{\mathrm{upper}}$	`tailcuts`	`mars`	`fact`	`tcc`
0.00	0.05			0.034	
0.05	0.10			0.051	
0.10	0.15			0.117	
0.15	0.20			0.163	
0.20	0.25			0.218	0.211
0.25	0.30	0.263	0.265	0.287	0.273
0.30	0.35	0.313	0.315	0.321	0.316
0.35	0.40	0.362	0.364	0.369	0.390
0.40	0.45	0.402	0.403	0.426	0.426
0.45	0.50	0.451	0.463	0.466	0.455
0.50	0.55		0.501		

Table 5.2: The results of the analysis for the mean angular resolution of each cleaning algorithm. The table lists the lower and upper limits of each efficiency interval. The angular resolution listed is the minimum mean angular resolution of the respective efficiency interval. The corresponding efficiency values are listed in Table 5.1. Notice how not all cleaning algorithms have valid results for all efficiency intervals.

		Mean Angular Resolution			
$\mathrm{Eff}_{\mathrm{lower}}$	$\mathrm{Eff}_{\mathrm{upper}}$	`tailcuts`	`mars`	`fact`	`tcc`
0.00	0.05			0.244	
0.05	0.10			0.297	
0.10	0.15			0.420	
0.15	0.20			0.428	
0.20	0.25			0.477	0.413
0.25	0.30	0.358	0.291	0.415	0.396
0.30	0.35	0.308	0.282	0.367	0.340
0.35	0.40	0.334	0.332	0.366	0.386
0.40	0.45	0.357	0.343	0.365	0.383
0.45	0.50	0.395	0.395	0.404	0.390
0.50	0.55		1.301		

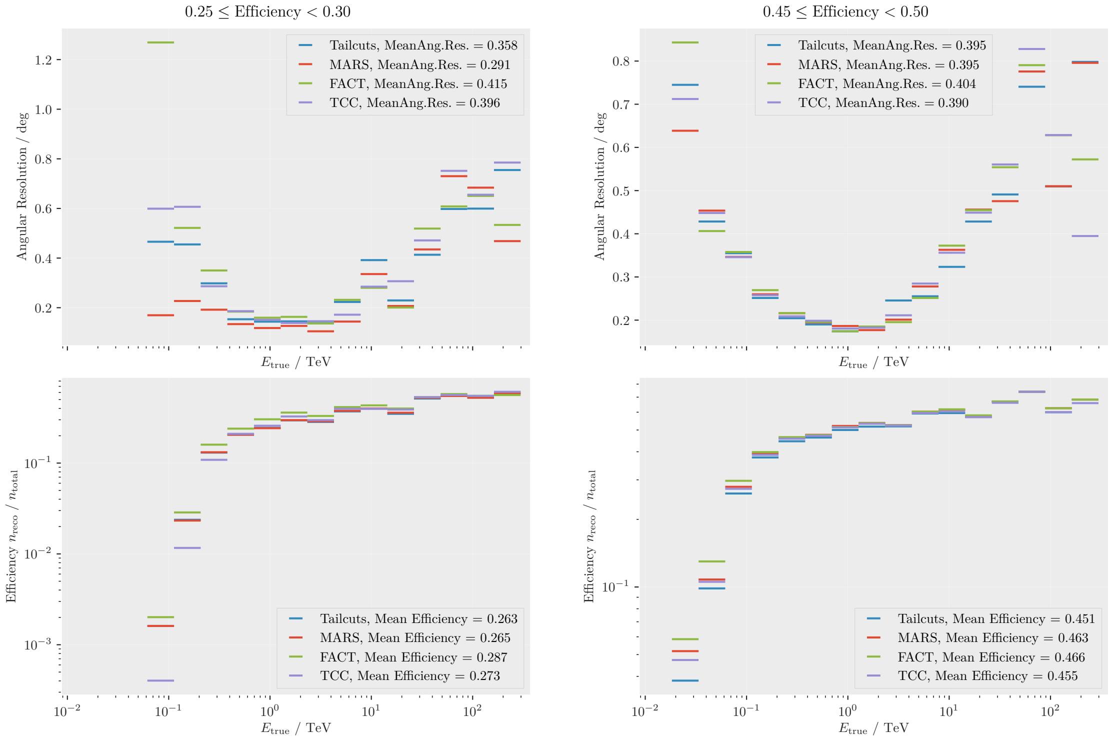


Figure 5.1: Mean angular resolution and efficiency for the MST simulation binned per energy.

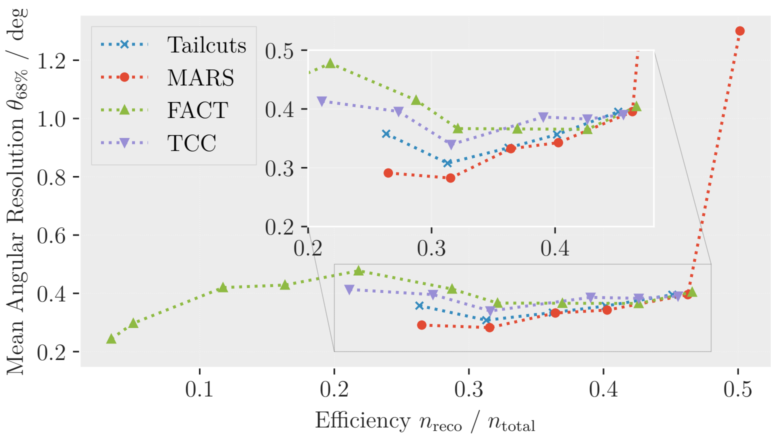


Figure 5.2: Angular resolution against efficiency.

5.2 Metrics of the cleaning algorithms

5.3 Performance compared to the default settings

5.4 Comparison of the cleaning algorithms

6 Conclusions and Outlook

Bibliography

1. B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration Collaboration). "Observation of Gravitational Waves from a Binary Black Hole Merger." *Phys. Rev. Lett.* 116, 6 2016, page 061102.
   DOI: 10.1103/PhysRevLett.116.061102. https://link.aps.org/doi/10.1103/PhysRevLett.116.061102
2. S. Abdollahi et al. "*Fermi* Large Area Telescope Fourth Source Catalog." *The Astrophysical Journal Supplement Series* 247:1, 2020, page 33.
   DOI: 10.3847/1538-4365/ab6bcb. https://doi.org/10.3847/1538-4365/ab6bcb
3. S. Abdollahi et al. "Incremental *Fermi* Large Area Telescope Fourth Source Catalog." *The Astrophysical Journal Supplement Series* 260:2, 2022, page 53.
   DOI: 10.3847/1538-4365/ac6751. ARXIV: 2201.11184 [astro-ph.HE]. https://doi.org/10.3847%2F1538-4365%2Fac6751
4. A. Achterberg et al. "First year performance of the IceCube neutrino telescope." *Astroparticle Physics* 26:3, 2006, pages 155–173.
   DOI: 10.1016/j.astropartphys.2006.06.007. ARXIV: astro-ph/0604450 [astro-ph]
5. J. A. Aguilar, J. Yang, and S. Bravo. *Neutrinos and gamma rays, a partnership to explore the extreme universe*. IceCube/WIPAC. 2016.
   https://icecube.wisc.edu/news/view/455 visited on 2022-08-05
6. Astropy Collaboration et al. "Astropy: A community Python package for astronomy." *Astronomy & Astrophysics* 558, A33, 2013, A33.
   DOI: 10.1051/0004-6361/201322068. ARXIV: 1307.6212 [astro-ph.IM]
7. Astropy Collaboration et al. "The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package." *The Astronomical Journal* 156:3, 123, 2018, page 123.
   DOI: 10.3847/1538-3881/aabc4f. ARXIV: 1801.02634 [astro-ph.IM]
8. K. Bernlöhr. "Simulation of imaging atmospheric Cherenkov telescopes with CORSIKA and sim_telarray." *Astroparticle Physics* 30:3, 2008, pages 149–158.
   DOI: https://doi.org/10.1016/j.astropartphys.2008.07.009. https://www.sciencedirect.com/science/article/pii/S0927650508000972
9. K. Brügge. "Unmasking the Gamma-Ray Sky. Comprehensive and Reproducible Analysis for Cherenkov Telescopes." PhD thesis. Technische Universität Dortmund, 2019
10. T. A. Caswell et al. *matplotlib/matplotlib: REL: v3.5.1*. Version v3.5.1. 2021.
    DOI: 10.5281/zenodo.5773480. https://doi.org/10.5281/zenodo.5773480

11. Cherenkov Telescope Array Consortium et al. *Science with the Cherenkov Telescope Array*. WORLD SCIENTIFIC, 2019.
DOI: 10.1142/10986. ARXIV: 1709.07997v2 [astro-ph.IM]

12. C. L. Cowan et al. "Detection of the Free Neutrino: a Confirmation." *Science* 124:3212, 1956, pages 103–104.
DOI: 10.1126/science.124.3212.103. https://www.science.org/doi/abs/10.1126/science.124.3212.103

13. *CTA Technology*. Cherenkov Telescope Array Consortium.
https://www.cta-observatory.org/wp-content/uploads/2019/12/CTA-Specifications_v08_formatted.pdf visited on 2022-08-10

14. *CTAO's site in the northern hemisphere*. Cherenkov Telescope Array Consortium.
https://www.cta-observatory.org/about/array-locations/la-palma/ visited on 2022-08-10

15. *CTAO's site in the southern hemisphere*. Cherenkov Telescope Array Consortium.
https://www.cta-observatory.org/about/array-locations/chile/ visited on 2022-08-10

16. R. Davis. "Solar Neutrinos. II. Experimental." *Phys. Rev. Lett.* 12, 11 1964, pages 303–305.
DOI: 10.1103/PhysRevLett.12.303. https://link.aps.org/doi/10.1103/PhysRevLett.12.303

17. C. Evoli. *The Cosmic-Ray Energy Spectrum*. 2020.
DOI: 10.5281/zenodo.4396125. https://doi.org/10.5281/zenodo.4396125

18. S. Funk. "Ground- and Space-Based Gamma-Ray Astronomy." *Annual Review of Nuclear and Particle Science* 65:1, 2015, pages 245–277.
DOI: 10.1146/annurev-nucl-102014-022036. https://doi.org/10.1146/annurev-nucl-102014-022036

19. J. Hackfeld. "Analyzing the Data Volume Reduction for the LST-1 Prototype of the Cherenkov Telescope Array." MA thesis. Ruhr-Universität Bochum, 2021

20. C. R. Harris et al. "Array programming with NumPy." *Nature* 585:7825, 2020, pages 357–362.
DOI: 10.1038/s41586-020-2649-2. https://doi.org/10.1038/s41586-020-2649-2

21. D. Heck et al. *CORSIKA: a Monte Carlo code to simulate extensive air showers*. 1998.
https://ui.adsabs.harvard.edu/abs/1998cmcc.book.....H.
Note: Provided by the SAO/NASA Astrophysics Data System

22. K. Hirata et al. "Observation of a neutrino burst from the supernova SN1987A." *Phys. Rev. Lett.* 58, 14 1987, pages 1490–1493.
DOI: 10.1103/PhysRevLett.58.1490. https://link.aps.org/doi/10.1103/PhysRevLett.58.1490

23. K. Kosack et al. *cta-observatory/ctapipe: v0.15.0 – 2022-05-23*. Version v0.15.0. 2022.
DOI: 10.5281/zenodo.6572720. https://doi.org/10.5281/zenodo.6572720

24. W. L. Kraushaar and G. W. Clark. "Search for Primary Cosmic Gamma Rays with the Satellite Explorer XI." *Phys. Rev. Lett.* 8, 3 1962, pages 106–109.
DOI: 10.1103/PhysRevLett.8.106. https://link.aps.org/doi/10.1103/PhysRevLett.8.106

25. M. S. Longair. *High energy astrophysics: an informal introduction for students of physics and astronomy*. Cambridge University Press, Cambridge [England] ; New York, 1981.
ISBN: 9780521235136, 9780521280136

26. M. Nöthe. "Monitoring the High Energy Universe. Open, Reproducible, Machine Learning Based Analysis for the first G-APD Cherenkov Telescope." PhD thesis. Technische Universität Dortmund, 2020

27. M. Nöthe et al. *cta-observatory/pyirf: v0.7.0 – 2022-04-19*. Version v0.7.0. 2022. DOI: 10.5281/zenodo.6469902. https://doi.org/10.5281/zenodo.6469902

28. F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research* 12, 2011, pages 2825–2830

29. Y. Suzuki. "The Super-Kamiokande experiment." *The European Physical Journal C* 79, 2019. DOI: 10.1140/epjc/s10052-019-6796-2

30. T. pandas development team. *pandas-dev/pandas: Pandas*. 2020. DOI: 10.5281/zenodo.3509134. https://doi.org/10.5281/zenodo.3509134

31. *The technology behind the next generation very-high energy gamma-ray detector*. Cherenkov Telescope Array Consortium. https://www.cta-observatory.org/project/technology/ visited on 2022-08-11

Glossary

acc Accuracy. 15

AGN active galactic nucleus. 2, 3

ba Balanced Accuracy. 15

CORSIKA Cosmic Ray Simulations for Kascade. 8

CR cosmic rays. 1, 2

CTA Cherenkov Telescope Array. 4, 6–8

EAS extensive air shower. 3–5, 8

ESO European Southern Observatory. 6

Fermi-LAT *Fermi* Large Area Telescope. 2–4

fn False Negative. 15

fnr False Negative Rate. 15

fp False Positive. 15

fpr False Positive Rate. 15

GRB gamma ray burst. 3

HE high-energy. 2, 7

HECR high-energy cosmic ray. 8

HEGR high-energy gamma rays. 3

H. E. S. S. High Energy Stereoscopic System. 3, 7

IACT Imaging Air Cherenkov Telescope. 3–6, 8

LST Large-Sized Telescope. 6, 7, 16, 30

MAGIC Major Atmospheric Gamma-Ray Imaging Cherenkov. 3, 7, 13

MC Monte Carlo. 8

MST Medium-Sized Telescope. 6, 7, 10–13, 30

NSB Night Sky Background. 5

ORM Observatorio del Roque de los Muchachos. 6

PMT photo multiplier tube. 6, 7

ppv Positive Predictive Value. 15

PWN pulsar wind nebula. 7

SiPM silicon photomultiplier. 6, 7

SNR supernova remnant. 2, 3

SST Small-Sized Telescope. 6, 7

tn True Negative. 15

tnr True Negative Rate. 15

tp True Positive. 15

tpr True Positive Rate. 15

VERITAS Very Energetic Radiation Imaging Telescope Array System. 3

VHE very-high-energy. 7

VHE gamma rays very-high-energy gamma rays. 3, 7

Appendix

1 Configurations for `ctapipe`

The listing below shows the configuration file used for preprocessing the datasets, i. e. before the application of the different cleaning settings.

```
DataWriter:
  transform_image: true
  transform_peak_time: true
  write_images: true
  write_parameters: false
  write_raw_waveforms: false
  write_showers: true

ProcessorTool:
  progress_bar: true

CameraCalibrator:
  image_extractor_type: NeighborPeakWindowSum

ImageProcessor:
  image_cleaner_type: TailcutsImageCleaner

  TailcutsImageCleaner:
    picture_threshold_pe:
      - [type, ”LST*”, 8.5]
      - [type, ”MST*NectarCam”, 9.0]
    boundary_threshold_pe:
      - [type, ”LST*”, 4.75]
      - [type, ”MST*NectarCam”, 4.5]
    keep_isolated_pixels: false
    min_picture_neighbors: 2

  ImageQualityQuery:
    quality_criteria:
      - [”enough_pixels”, ”np.count_nonzero(image) > 2”]
      - [”enough_charge”, ”image.sum() > 50”]

ShowerProcessor:
  reconstructor_type: HillasReconstructor
  HillasReconstructor:
    StereoQualityQuery:
      quality_criteria:
        - [enough intensity, ”parameters.hillas.intensity > 50”]
        - [Positive width, ”parameters.hillas.width.value > 0”]
        - [enough pixels, ”parameters.morphology.num_pixels > 3”]
        - [not clipped, ”parameters.leakage.intensity_width_2 < 0.5”]
```

```yaml
SimTelEventSource:
  focal_length_choice: effective
  skip_calibration_events: true
```

The listings below show the contents of the default configuration files used with `ctapipe` for each cleaning algorithm respectively.

`TailcutsImageCleaner`:

```yaml
DataWriter:
  transform_image: true
  transform_peak_time: true
  write_images: false
  write_parameters: true
  write_raw_waveforms: false
  write_showers: true

ProcessorTool:
  progress_bar: true

CameraCalibrator:
  image_extractor_type: NeighborPeakWindowSum

ImageProcessor:
  image_cleaner_type: TailcutsImageCleaner

  TailcutsImageCleaner:
    picture_threshold_pe:
      - [type, "LST*", 8.5]
      - [type, "MST*NectarCam", 9.0]
    boundary_threshold_pe:
      - [type, "LST*", 4.75]
      - [type, "MST*NectarCam", 4.5]
    keep_isolated_pixels: false
    min_picture_neighbors: 2

  ImageQualityQuery:
    quality_criteria:
      - ["enough_pixels", "np.count_nonzero(image) > 2"]
      - ["enough_charge", "image.sum() > 50"]

ShowerProcessor:
  reconstructor_type: HillasReconstructor
  HillasReconstructor:
    StereoQualityQuery:
      quality_criteria:
        - [enough intensity, "parameters.hillas.intensity > 50"]
        - [Positive width, "parameters.hillas.width.value > 0"]
        - [enough pixels, "parameters.morphology.num_pixels > 3"]
        - [not clipped, "parameters.leakage.intensity_width_2 < 0.5"]
```

`MARSImageCleaner`:

```yaml
DataWriter:
  transform_image: true
  transform_peak_time: true
  write_images: false
```

```yaml
  write_parameters: true
  write_raw_waveforms: false
  write_showers: true

ProcessorTool:
  progress_bar: true

CameraCalibrator:
  image_extractor_type: NeighborPeakWindowSum

ImageProcessor:
  image_cleaner_type: MARSImageCleaner

  MARSImageCleaner:
    picture_threshold_pe:
      - [type, ”LST*”, 8.5]
      - [type, ”MST*NectarCam”, 9.0]
    boundary_threshold_pe:
      - [type, ”LST*”, 4.75]
      - [type, ”MST*NectarCam”, 4.5]
    keep_isolated_pixels: false
    min_picture_neighbors: 2

  ImageQualityQuery:
    quality_criteria:
      - [”enough_pixels”, ”np.count_nonzero(image) > 2”]
      - [”enough_charge”, ”image.sum() > 50”]

ShowerProcessor:
  reconstructor_type: HillasReconstructor
  HillasReconstructor:
    StereoQualityQuery:
      quality_criteria:
        - [enough intensity, ”parameters.hillas.intensity > 50”]
        - [Positive width, ”parameters.hillas.width.value > 0”]
        - [enough pixels, ”parameters.morphology.num_pixels > 3”]
        - [not clipped, ”parameters.leakage.intensity_width_2 < 0.5”]
```

FACTImageCleaner:

```yaml
DataWriter:
  transform_image: true
  transform_peak_time: true
  write_images: false
  write_parameters: true
  write_raw_waveforms: false
  write_showers: true

ProcessorTool:
  progress_bar: true

CameraCalibrator:
  image_extractor_type: NeighborPeakWindowSum

ImageProcessor:
  image_cleaner_type: FACTImageCleaner

  FACTImageCleaner:
    picture_threshold_pe:
```

```
      - [type, "LST*", 8.5]
      - [type, "MST*NectarCam", 9.0]
    boundary_threshold_pe:
      - [type, "LST*", 4.75]
      - [type, "MST*NectarCam", 4.5]
    keep_isolated_pixels: false
    min_picture_neighbors: 2

  ImageQualityQuery:
    quality_criteria:
      - ["enough_pixels", "np.count_nonzero(image) > 2"]
      - ["enough_charge", "image.sum() > 50"]

ShowerProcessor:
  reconstructor_type: HillasReconstructor
  HillasReconstructor:
    StereoQualityQuery:
      quality_criteria:
        - [enough intensity, "parameters.hillas.intensity > 50"]
        - [Positive width, "parameters.hillas.width.value > 0"]
        - [enough pixels, "parameters.morphology.num_pixels > 3"]
        - [not clipped, "parameters.leakage.intensity_width_2 < 0.5"]
```

`TimeConstrainedImageCleaner`:

```
DataWriter:
  transform_image: true
  transform_peak_time: true
  write_images: false
  write_parameters: true
  write_raw_waveforms: false
  write_showers: true

ProcessorTool:
  progress_bar: true

CameraCalibrator:
  image_extractor_type: NeighborPeakWindowSum

ImageProcessor:
  image_cleaner_type: TimeConstrainedImageCleaner

  TimeConstrainedImageCleaner:
    picture_threshold_pe:
      - [type, "LST*", 8.5]
      - [type, "MST*NectarCam", 9.0]
    boundary_threshold_pe:
      - [type, "LST*", 4.75]
      - [type, "MST*NectarCam", 4.5]
    keep_isolated_pixels: false
    min_picture_neighbors: 2

  ImageQualityQuery:
    quality_criteria:
      - ["enough_pixels", "np.count_nonzero(image) > 2"]
      - ["enough_charge", "image.sum() > 50"]

ShowerProcessor:
```

```
reconstructor_type: HillasReconstructor
HillasReconstructor:
  StereoQualityQuery:
    quality_criteria:
      - [enough intensity, "parameters.hillas.intensity > 50"]
      - [Positive width, "parameters.hillas.width.value > 0"]
      - [enough pixels, "parameters.morphology.num_pixels > 3"]
      - [not clipped, "parameters.leakage.intensity_width_2 < 0.5"]
```

The following listing shows the contents of the `prod5b_lapalma_alpha.yml` configuration file, which is used to set the allowed telescope IDs for `ctapipe-process`.

```
# telescope ids of the 4 LST + 9 MST La Palma alpha configuration in Prod5b
# (out of 84 telescopes total)
EventSource:
  allowed_tels: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 19, 35]
```

Since the comparison of the cleaning algorithms also required differentiating between MSTs and LSTs, the following two listings show the contents of the `prod5b_lapalma_mst.yml` and `prod5b_lapalma_lst.yml` configuration files, respectively:

MSTs:

```
# telescope ids of the 9 MST of the La Palma alpha configuration
# in Prod5b
EventSource:
  allowed_tels: [5, 6, 7, 8, 9, 10, 11, 19, 35]
```

LSTs:

```
# telescope ids of the 4 LST of the La Palma alpha configuration
# in Prod5b
EventSource:
  allowed_tels: [1,2,3,4]
```

2 Software used

This work was written and built with LaTeX and LuaTeX from TeXLive 2021 on both Windows 10 and Ubuntu 20.04. Also, I relied heavily on the python programming language, of which the most important libraries used for this work are listed below.

- `numpy` [20]
- `pandas` [30]
- `matplotlib` [10]
- `astropy` [6, 7]

- `pyirf` [27]
- `scikit-learn` [28]

For the processing of the datasets, I used the development version of `ctapipe`, more specifically, version `0.15.1.dev166+gf26107f`. A complete listing of all used python libraries can be found below:

```
name: cta-dev
channels:
  - anaconda
  - defaults
  - conda-forge
dependencies:
  - sphinx_rtd_theme=1.0.0
  - sphinx=3.5.4
  - jupyter=1.0.0
  - matplotlib=3.5.1
  - iminuit=2.11.2
  - tqdm=4.64.0
  - traitlets=5.1.1
  - joblib=1.1.0
  - sphinx-automodapi=0.14.1
  - pytest-runner=6.0.0
  - pytables=3.7.0
  - tomli=2.0.1
  - psutil=5.9.0
  - pytest-cov=3.0.0
  - python=3.8.13
  - pyyaml=6.0
  - numba=0.55.1
  - pytest=7.1.2
  - pip=22.0.4
  - jinja2=3.0.3
  - xz=5.2.5
  - nbsphinx=0.8.8
  - scipy=1.8.0
  - cython=0.29.28
  - eventio=1.9.1
  - pandas=1.4.2
    - zlib=1.2.11
    - black=22.3.0
    - h5py=3.6.0
    - numpydoc=1.2.1
    - vitables=3.0.2
    - wheel=0.37.1
    - ipywidgets=7.7.0
    - scikit-learn=1.0.2
    - graphviz=3.0.0
    - bokeh=2.4.2
    - zstandard=0.17.0
    - pre-commit=2.18.1
    - numpy=1.21.6
    - astropy=5.0.4
    - setuptools=62.1.0
    - ipython=8.2.0
    - jupyterlab=3.3.4
    - seaborn=0.11.2
    - ipympl=0.9.1
    - ca-certificates=2022.6.15
    - certifi=2022.6.15
    - openssl=3.0.5
    - gammapy=0.20
    - pylint=2.14.3
    - ctapipe=0.15.1
    - colour=0.1.5
    - flake8=4.0.1
  - pip:
      - pyirf==0.7.0
      - pytimedinput==2.0.1
      - runipy==0.1.5
      - setuptools-scm==6.4.2
```

Acknowledgments

Eidesstattliche Versicherung

(Affidavit)

Name, Vorname
(surname, first name)

Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

☐ Masterarbeit
(Master's thesis)

Titel
(Title)

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.	I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Ort, Datum
(place, date)

Unterschrift
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Ort, Datum
(place, date)

Unterschrift
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**