Bachelor Degree Project

ALTO Timing Calibration
- Calibration of the ALTO detector array
  based on cosmic-ray simulations

Author: Sotirios-Ilias Tsivras
Supervisor: Dr. Yvonne Becherini
Co-Supervisors: Dr. Michael Punch
  Dr. Satyendra Thoudam
Examiner: Dr. Peter Kuijer
Semester: VT 2018
Subject: Physics
Abstract

This thesis describes a timing calibration method for the detector array of the ALTO experiment. ALTO is a project currently at the prototype phase that aims to build a gamma-ray astronomical observatory at high-altitude in the Southern hemisphere. ALTO can be assumed as a hybrid system as each detector consists of a Water Cherenkov Detector (WCD) on top of a Scintillator Detector (SD), providing an increased signal to background discrimination compared to other WCD arrays.

ALTO is planned to complement the Very-High-Energy (VHE) observations by the High Altitude Water Cherenkov (HAWC) gamma ray observatory that collects data from the Northern sky. By the time the full array of 1242 detectors is installed to the proposed site, ALTO together with HAWC and the future Cherenkov Telescope Array (CTA) will serve as a state-of-the-art detection system for VHE gamma-rays combining the WCD and the Imaging Atmospheric Cherenkov Telescope (IACT) techniques.

When a VHE gamma-ray or cosmic-ray enters the Earth’s atmosphere, it initiates an Extensive Air Shower (EAS). These particles are sampled by the detector array and by checking the arrival times of nearby tanks, the method reveals whether a detector suffers from a time-offset.

The data analyzed in this thesis derive from CORSIKA (COsmic Ray SImulation for KAscade) and GEANT4 (GEometry ANd Tracking) simulations of cosmic-ray events within the energy range of 1–1.6 TeV, which mainly consist of protons. The high flux of this particular type of cosmic-rays, gives us a tool to statistically evaluate the results generated by the proposed timing calibration method.

In the framework of this thesis, I have written code in Python programming language in order to develop the timing calibration method. The method identifies detectors that suffer from time-offsets and improves the reconstruction accuracy of the ALTO detector array. Different Python packages were used to execute different tasks: astropy to read-filter-present-write large datasets, numpy (Numerical Python) to make datasets comprehensive to functions, scipy (Scientific Python) to develop our models, sympy (Symbolic Python) to find geometrical correlations and matplotlib (Mathematical Plotting Library) to draw figures and diagrams.

The current version of the method achieves sub-nanosecond accuracy. The next step is to make the timing calibration more intelligent in order to correct itself. This self-correction includes an agile adaptation to the data acquired for long periods of time, in order to make different compromises at different time intervals.

Keywords: ALTO, Extensive Air Shower, Water Cherenkov Detector, Scintillator Detector, HAWC, IACT, CORSIKA, GEANT4, ROOT, Python
Preface

This thesis is a report for the ALTO project organized by the Astroparticle research group of the Linnaeus University. I am thankful for my supervisor, Yvonne Becherini, who gave me the opportunity to study Astrophysics and extend my knowledge in programming with Python. I would also like to thank my co-supervisors, Michael Punch and Satyendra Thoudam, for providing me the necessary computational tools in order to complete my tasks.
1 Cosmic Rays and Very-High-Energy Gamma Rays 2
  1.1 Cosmic Rays .................................................. 2
  1.2 Sources of High Energy Particles .......................... 3
  1.3 Very High Energy Gamma Rays .............................. 4
  1.4 A model for extensive air showers .......................... 5
  1.5 Detection techniques ......................................... 6
  1.6 Gamma and Cosmic Ray Shower Footprints .................. 7

2 The ALTO project 9
  2.1 Site location .................................................. 9
  2.2 Geometrical properties of the ALTO array .................. 10
  2.3 Detector description .......................................... 13
  2.4 The shape of the shower-front ............................... 14
  2.5 Simulation of events and reconstructed parameters .......... 14
  2.6 The flux of background hadronic events .................... 17
  2.7 Advantages of the ALTO experiment ........................ 19
  2.8 Current and future state of ALTO ........................... 19

3 Timing Calibration in the ALTO project 23
  3.1 The timing-calibration problem ............................. 23
  3.2 The White-Rabbit technology ................................ 24
  3.3 The timing-calibration method ............................... 25
  3.4 The cosmic ray event simulations ........................... 26
  3.5 The Python ecosystem ........................................ 27
  3.6 The Astropy project .......................................... 29
  3.7 From ROOT to Astropy ....................................... 30
  3.8 The detector array .......................................... 32
  3.9 Creating the final table .................................... 33
  3.10 Looping over the events .................................... 34
  3.11 Plane Fit ..................................................... 35
  3.12 Optimized plane solution ................................... 38
  3.13 Generating residuals for a subset of detectors .......... 40
  3.14 Offset identification ....................................... 41
  3.15 Calculating the residuals for different R-N sets ......... 41

4 Results and further discussion 45
  4.1 Residual analysis for a subset of detectors ................ 45
  4.2 Further discussion about the timing calibration method .... 46
  4.3 Further research for the ALTO project ...................... 50
  4.4 Conclusion .................................................. 51
A Abbreviations A
B Code listings C
Bibliography G
Chapter 1

Cosmic Rays and Very-High-Energy Gamma Rays

“A mechanism that consumes matter, can also accelerate”

Astronomy is one of the oldest branches of science. It differs from other sciences as the experimental tests are not carried out in the laboratory, but from observations of extreme states of matter found in the Universe that are impossible to create here on Earth.

*Very High Energy (VHE) gamma-ray Astronomy* deals with the study of the Universe in photons of energies above \(~30\ \text{GeV}\) (Gigaelectronvolt). These photons are generated by violent astrophysical processes, and they can be detected by measuring cascades of secondary low-energy particles, known as Extensive Air Showers (EAS), generated from their interactions with the molecules in the Earth’s atmosphere.

Air showers mainly consist of electrons, positrons and photons, together with muons, pions, and kaons if the primary is a cosmic-ray particle. These particles can be detected on the ground directly using particle detectors which can be Water Cherenkov Detectors (WCDs) or Scintillation Detectors (SDs). An EAS can also be detected indirectly using Imaging Atmospheric Cherenkov Telescopes (IACTs) as the cascade of charged particles generates a flash of Cherenkov radiation lasting between 5 and 20 nanoseconds (ns).

*ALTO* is a particle detector array based on WCD technique, similar to the High Altitude Water Cherenkov Gamma-Ray Observatory (HAWC) experiment in Mexico but combined with SDs, for VHE gamma-ray astronomy in the Southern Hemisphere. The array, currently at the prototype phase, will be installed at an altitude of \(~5\ \text{km}\) above sea level, and it will be capable of measuring VHE gamma rays above about 200 GeV [1].

It will consist of more than a thousand detector units distributed over an area of 160 m in diameter with each unit consisting of a WCD and a SD. With its wide field-of-view and almost 100\% duty cycle, ALTO will continuously observe VHE gamma rays from a variety of astrophysical sources. It will also continuously monitor activities of the Galactic centre region and act as an alert system for the Southern part of the upcoming Cherenkov Telescope Array (CTA) experiment.

This thesis describes a method to identify and calibrate systematic offsets in the signal arrival times recorded at the WCDs. An accurate signal arrival time is necessary to achieve a high accuracy in the reconstruction of the arrival direction of the primary gamma rays.
The method is based on using relative signal arrival times between the detectors from the background cosmic-ray events.

The study will use data from a dedicated ALTO simulation performed by the Astroparticle Physics group at Linnaeus University.

1.1 Cosmic Rays

Cosmic rays are high energy particles that constantly reach the Earth from all directions. They mostly consist of protons and helium nuclei, with a small fraction of heavier nuclei and electrons. When they were first discovered by Victor Hess in 1912, it was assumed to be electromagnetic radiation. However, during the 1930’s it was found that they were electrically charged as they were affected by the Earth’s magnetic field. The highest energy cosmic rays, measured so far, have several million times more energy than the highest energy currently achieved with the Large Hadron Collider (LHC) at CERN [6].

Although the origin of cosmic rays is not exactly known, it is believed that particles up to $10^6$ eV observed at the Earth are originated from our Sun (solar), while particles within the energy range of $10^6$–$10^{18}$ eV are produced in our Milky Way Galaxy. However, cosmic rays of energies higher than $10^{18}$ eV are most likely to be extragalactic [6].

In our Galaxy, supernova remnants are considered as the most plausible sources of cosmic rays. Theoretically, it has been established that supernova shocks can accelerate particles via the first-order Fermi acceleration mechanism. This mechanism predicts a maximum energy of cosmic rays of $Z \times 10^{15}$ eV, where $Z$ is the charge of the nuclei. This shows that even for the iron nuclei ($Z=36$), the maximum energy that can be accelerated by supernova shocks is below $10^{17}$ eV [7].

To be compared, the highest energy cosmic ray events ever detected by ground-based detectors, have energies of the order of $3 \times 10^{20}$ eV [6]. The origin of the highest energy cosmic rays is even more unclear. Possible sources include Active Galactic Nuclei (AGNs) and Gamma-Ray Bursts (GRBs).

Cosmic rays travelling through space for large distances are losing much of their initial energy in collisions with Cosmic Microwave Background (CMB) radiation. The existence of cosmic rays with energies greater than $10^{20}$ eV implies an extragalactic source near to us. However, experimental observations with astrophysical detectors do not point to any already-known astrophysical object.

The nature of cosmic ray propagation in the Galaxy and in the intergalactic medium is also not fully understood. The chemical composition of cosmic rays is similar to the abundances of elements in our solar system, with some exceptions particularly for the light elements Lithium, Beryllium and Boron which are present in overabundances. In addition, measurement of abundances of radioactive isotopes, such as $^{10}$Be, can provide information of the residence time of cosmic rays in the Galaxy [28].

Combining all available information, astrophysicists came to the conclusion that the highest energy extragalactic cosmic rays are possibly accelerated by AGNs. An AGN is
1.2 SOURCES OF HIGH ENERGY PARTICLES

found at the centre of galaxies and consists of a super-massive black hole that accretes material from a surrounding disc.

1.2 Sources of High Energy Particles

Based on the observations of numerous sources of VHE gamma rays, the major candidates that can accelerate particles to very high energies are listed below:

1. **Supernovae remnants:** Supernova explosions are transient astronomical events occurring at the last evolutionary stages of the life of massive stars. Such cataclysmic events happen every 30 years within our Galaxy. The ejecta material expelled by the explosion can generate shock waves in the surrounding medium, and accelerate particles to relativistic energies through the diffusive shock acceleration mechanism. Turbulence generated on both sides (upstream and downstream) of the shock allows charged particles to repeatedly cross the shock front and gain energy in every crossing. If this mechanism is the dominant source of cosmic ray acceleration in the Galaxy, it would require an energy transfer of 10% of the total kinetic energy released by an explosion into cosmic rays [9].

2. **Active Galactic Nuclei (AGN):** These objects, such as Markarian 421, are one of the dominant sources detected in VHE gamma rays. Their core consists of a supermassive black hole and the radiation is powered from the accretion of matter into the central compact core region. Particle acceleration takes place in the relativistic jets that stretch up to several kiloparsecs away from a core, generating VHE gamma rays. The observed flux at the Earth depends on the orientation of the jet with respect to the line of sight. The class of AGN which has jets pointing directly towards the Earth is called blazar [7].

3. **Gamma ray bursts (GRBs):** They are the most energetic phenomena in the known Universe. Short gamma-ray bursts last a few seconds and they are initiated from the collapse of two neutron stars. On the other hand, long gamma-ray bursts may last for several minutes and they are generated when a massive star collapses to a neutron star or a black hole. The sources of GRBs are billions of light years (ly) away from Earth, implying that these phenomena are extremely energetic and rare. However, a subclass of GRBs called soft gamma repeater originate from objects inside the Milky Way [7].

4. **Pulsars:** They are rapidly rotating neutron stars that exhibit large electromagnetic fields. Their radiation comes as two narrow beams of light, emerging in opposite directions, similar to a lighthouse as seen from a ship. When charged particles are in the appropriate range they are accelerated, and this mechanism is assumed to be the main source of very-high energy gamma rays. Observations show that pulsars accelerate mostly electrons [7].

5. **Big Bang relics:** These objects include super-symmetric dark matter, monopoles and cosmic strings, and they are objects that cosmologists infer as long as the mysteries about cosmic rays remain unclear. Cosmic strings are one-dimensional topological defects created during a symmetry breaking phase transition in the early universe, while monopoles are elementary particles with one magnetic pole. If
monopoles exist in our Universe, they may decay or annihilate to generate energy in the form of gamma rays [9].

1.3 Very High Energy Gamma Rays

Most of the radiation that exists in the Universe comes from thermal processes. An example of a thermal process, is the fusion inside our Sun which radiates energy by burning Hydrogen to Helium. However, it is impossible for this kind of process to generate VHE gamma rays and only non-thermal mechanisms can do so. The part of the Universe that includes all these unknown processes, is called the Non-Thermal Universe and a depiction of it is given in Fig. 1.1 [10]. Cosmic rays play a key role of studying the Universe as their energy density is so high that can be compared with that of starlight as seen from Earth.

VHE gamma-rays can be produced by the following two processes:

- interaction of cosmic-ray nuclei with matter producing $\pi^0$ meson which decays into two gamma rays
- interaction of cosmic-ray electrons with matter producing radiation and inverse Compton scattering of background photons by the electrons

The final flux of gamma rays is proportional to three parameters:

- the efficiency of the source to create cosmic rays
- the initial energy of each of the generated charged particles
- the efficiency of the medium to transform particle energy into photons

In most cases, the above mentioned interactions take place in the medium right after the cosmic rays are accelerated. While cosmic-rays will be deflected or scattered in the Interstellar Medium (ISM), photons propagate unaffected. As a result, if these photons are finally detected from Earth, they may reveal the direction of their origin as they propagate unaffected by electromagnetic fields.
1.4 A model for extensive air showers

Extensive air showers develop in a random way and depending on the initial energy of cosmic ray particle and the altitude of the observer, the number of photons and particles may exceed $10^{10}$ in number. Electromagnetic cascades include electrons, positrons and photons, while proton cascades include pions, photons, muons and neutrinos.

For electromagnetic cascades, the first interaction of the VHE gamma ray with the atmosphere creates two photons through pair production process (electron – positron pair). The electron and positron pair then radiate via bremsstrahlung. After n steps, the cascade reaches a maximum size of $N_{\text{max}} = 2^n$ particles, after which the process starts to diminish as particles with energy less than a critical value (approximately 80 MeV) get attenuated in the atmosphere.

On the other hand, the first interaction of cosmic rays with the air molecules produce charged and neutral pions. Neutral pions immediately decay into two photons giving electromagnetic showers similar to those of primary gamma rays. Charged pions decay into muons and neutrinos. The muons can further decay into electrons/positrons, or they reach the ground, depending on their energy. Simulations show that the final ratio of photons to electrons is similar to a pure electromagnetic cascade. In the particular case where the primary hadron is a proton, about one third of the primary energy goes into electromagnetic showers and the rest is carried by charged pions [11].

The maximum size of the shower is proportional to the initial energy of primary particle. The penetration length also increases logarithmically with energy and depends on the air density at the location of the observer. Fig. 1.2 gives a simplified version of Heitler’s model for electromagnetic showers on the left schematic, while the right schematic shows an extension to hadronic showers where the dashed lines indicate neutral pions which quickly decay yielding electromagnetic sub-showers.
CHAPTER 1. COSMIC RAYS AND VERY-HIGH-ENERGY GAMMA RAYS

1.5 Detection techniques

Cosmic rays entering the Earth’s atmosphere interact with the air molecules and initiate particle cascades, called *extensive air showers*. Depending on the type and energy of the initial particle, showers develop as a combination of electromagnetic cascades and hadronic multi-particle production. As we will see later in the following chapters, a crucial difference between the cascades initiated by gamma-rays and hadrons is that the latter are muon-rich.

When a high-energy photon reaches the upper atmosphere, it initiates a cascade of secondary particles comprising mainly electrons, positrons and low-energy photons. Simulations show that photons greatly outnumber electrons due to the high energy loss rate of electrons in the atmosphere. The secondary particles can be directly or indirectly detected with the following methods:

- **Measuring Cherenkov radiation**: when charged particles propagate through a medium faster than light, the medium generates a cone of electromagnetic radiation. A particle well-above the Cherenkov threshold energy creates nearly 20 photons per meter in air, mainly within the ultraviolet (UV) range of the electromagnetic spectrum. Instruments which detect gamma-rays within the energy range of $50 \text{ GeV} - 50 \text{ TeV}$ (Tera Electron Volt) based on this effect are called *Imaging Air Cherenkov Telescopes* (IACTs, Fig. 1.3 (left) [9].

- **Direct Detection of the Air Showers**: as an air shower reaches the ground, it may extend over an area of hundreds of square meters and have a thickness of several meters, depending on the primary energy and the observation level. The particles can be detected using an array of Water Cherenkov Detectors and Scintillation Detectors which record light signals produced inside the detector volume using photomultiplier tubes (PMTs). Fig. 1.3 (right) shows a picture of the HAWC WCD array in Mexico [6][8].

- **Hybrid Detection**: a hybrid system combines several IACTs and one WCDA in order to take advantage of the strengths of both techniques. For example, low energy cosmic rays may not reach the ground but can be detected with IACTs, but a WCDA may operate during daytime where IACTs are highly affected by Sun or Moon-light. [6]
1.6 Gamma and Cosmic Ray Shower Footprints

A challenging task in gamma-ray astronomy is to distinguish the electromagnetic events from the dominant cosmic-ray background. In general, cosmic rays “break apart” along their path initiating a number of sub-showers, as the generated pions carry large transverse momentum. This results in a more random and messy “footprint” of particles on the ground.

An example of the differences between electromagnetic and hadronic cascades can be seen in Fig. 1.4, derived from shower simulations of the ALTO array. For the gamma-ray shower, the main impact is recorded within a set of neighbouring tanks. However, in a proton shower, high energy secondary particles can be detected in multiple distant locations within the detector array.
Chapter 2

The ALTO project

“The background hadronic events may serve as a standard calibration source”

ALTO is a project for a wide field-of-view air shower detector array dedicated to explore the gamma-ray sky at energies higher than 200 GeV. It is planned to be installed in the Southern Hemisphere at an altitude of \( \sim 5 \text{ km} \) and will complement the operation of the HAWC Gamma-ray observatory which continuously collects data from the Northern sky [12].

ALTO belongs to the technological “family” of Water Cherenkov Detector (WCD) arrays, consisting of 1242 tanks distributed over a circular area of 160 m in diameter. However, it will be a hybrid detector array with liquid Scintillation Detectors (SDs) installed underneath the WCDs.

Taking into account its location and closed-packed arrangement, it will provide an improved sensitivity, better angular resolution and lower energy threshold compared to the HAWC detector array. The operation of ALTO in the Southern hemisphere will complement the observations made by HAWC, and it will also serve as an all-sky monitor for the Southern part of the future Cherenkov Telescope Array (CTA).

2.1 Site location

One possible location for ALTO is the Atacama desert in Chile, on a plateau 5.1 km above sea level. The desert covers 1000 km strip of land west of the Andes mountains and most of it consists of stony terrain, salt lakes and felsic lava. It is one of the highest and driest places on Earth where flat areas are available for installing large detector arrays for astronomical observations.

The same site hosts two other major experiments operated by the European Southern Observatory (ESO): La Silla and Paranal Observatories. It also includes the ALMA (Atacama Millimeter/submillimeter Array) project which is currently the largest telescope in the world. ALMA consists of sixty-six 12-metre and 7-metre diameter telescopes, operating in an international association searching to answer some elementary questions regarding the “Dark Universe” [30].
CHAPTER 2. THE ALTO PROJECT

From an astronomer’s point of view, the site is unique for the following reasons:

- it has a very high altitude
- it has a clear sky as clouds are nearly non-existent
- it lacks from light pollution and radio interference from nearby cities

A second possible site is located 200 km away in North Western Argentina, in the Alto Chorillos, Puna desert. This is a site above 4.8 km with existing infrastructure, since the Large Latin American Millimeter Array (LLAMA) and QUBIC experiments will be located there, and a close-by small town, San Antonio de los Cobres, at 30 km, with hotels, workshops, etc., and further away the region’s capital, Salta. The main advantages for this site are the existing infrastructure, easier access to water, the nearby town, and Argentinian support [31].

2.2 Geometrical properties of the ALTO array

The perimeter of the detector array forms a polygon, as we can see from the line connecting the boundary detectors in Fig. 2.1. However, it can be approximated by a circle $C$ with a centre at the position $[x_0, y_0]$ and a radius $R$. The centre of the array can be calculated using a function implemented in Python programming language that calculates the minimum distances between a point and a given set of x-y coordinates (pairwise_distances_argmin function of the sklearn.metrics module [34]). By giving as input to the function the x-y positions of detectors, the function may return the centre of one cluster consisting of these points: $x_0 = -0.304 \text{ m}$ and $y_0 = 0.061 \text{ m}$.

Moreover, we may computationally find the approximate shape of the array, by plotting the line connecting the detectors on the borders of the array (using the ConvexHull method of the scipy.spatial module [18]). Then we may represent each detector as a 2D Point in Python (using the sympy module [19]) and calculate the average distance between the boundary detectors and the centre: we find that the average radius of the circle $C$ is $R = 82.2 \text{ m}$.

As Fig. 2.2 shows, the detectors are arranged in cluster of 6. Detectors in each cluster will share a common electronics and readout system [12].

As we will see in the next chapter, CORSIKA and GEANT4 packages are used to simulate cosmic-ray and gamma-ray events for the ALTO experiment. Fig. 2.3 shows the footprint of a simulated cosmic-ray event of energy $1.3 \text{ TeV}$ observed with the ALTO array.

In Fig. 2.3, the filled colour represents the distribution of shower particles detected by the array. As we will see in the next chapter, for executing the method described in this thesis, we will select one particular detector, for example detector 775, and process only the signals of its neighbours within a circle of radius $R = 16 \text{ m}$. 
2.2. GEOMETRICAL PROPERTIES OF THE ALTO ARRAY

Figure 2.1: Basic geometrical properties of the array where each point shows the position of a unit on the ground.

Figure 2.2: 3D-representation of a cluster of six units consisting of WCD (Water Cherenkov Detector) tanks and SLD (Scintillator Layer Detector) boxes [12].
Figure 2.3: Footprint of cosmic ray event, showing the selection region size of 15 m used for the local plane fitting.
2.3 Detector description

Each WCD is a hexagonal tank of size $4.15 \, \text{m}$ width and $2.5 \, \text{m}$ height, and is made of composite material of $5 \, \text{cm}$ thickness which is made of carbon fibre sandwiched with polyvinyl chloride. The composite material provides high strength and low weight. The volume of the water inside the tanks is $26.6 \, \text{m}^3$ and its overall weight is approximately $27 \, \text{tonnes}$. At the bottom of each WCD there is an 8-inch Hamamatsu photomultiplier tube (PMT) R5912-100 that detects Cherenkov photons initiated from the passage of cascade particles. In order to increase the light collection efficiency, a high-reflectivity crown is placed adjacent to the photocathode area of the PMT as depicted in Fig. 2.4. The inside of the WCD is made from a non-reflective material in order to preserve the timing information of the arriving particles [13].

The Scintillator Detector (SD) is $3 \, \text{m}$ in diameter, has a thickness of $5.6 \, \text{cm}$ and it is mounted underneath the WCD. It is planned to be made of aluminium sheets and the scintillating material is an organic liquid, Linear Alkyl Benzene, mixed with small quantities of wavelength shifter powders. An 8-inch Hamamatsu PMT R5912-20, placed at the top of the scintillator tank facing downwards, will collect the scintillation light produced from the passage of a charged particle. Contrary to the WCD, the inner surface will be as reflective as possible in order to maximize the light collection efficiency [13].

A concrete table is placed between the WCD and the SD, having a height of $25 \, \text{cm}$ and a weight of $6.6 \, \text{tonnes}$, which acts as a support and as additional shielding of the SD,
so that barely the muons can reach it. Taking also into account that each WCD rests on this concrete table with 3 concrete pillars, the overall weight of each detector unit will be nearly 35 tonnes.

2.4 The shape of the shower-front

When simulating a shower cascade, the arrival time of the signal and the total signal (integrated charge) in each detector are obtained. Only information collected by the WCDs are used to determine the arrival direction of the primary particle and the position of the shower axis (shower core). Signals from the SDs are used to improve the signal-over-background discrimination, thereby to improve the sensitivity.

For each shower event, the relative signal arrival times between the WCDs are obtained. The relative times correspond to the relative distances travelled by the shower particles with respect to the shower front. For this study, the value corresponding to the arrival time of the first photon is used for the arrival time value. In studies of the gamma-ray angular reconstruction, the LnU group has shown that using the time from a fit to the maximum of the waveform gives a more accurate result, however the time corresponding to the first photons should be sufficient on average for timing studies. This could be refined in subsequent work.

The shape of the shower wave-front approaches an asymptotic cone at a few tens of metres from the shower axis. For the analysis presented in this thesis, we will make two assumptions: the wave-front is a plane (called the “shower plane”) and the particles in the air shower travel towards the ground in the direction normal to the wave-front. It will be explained in the next chapter that by comparing the real data acquired from a WCD detector array with that theoretically expected, one may find timing offsets in the operation of the system.

In Fig. 2.5 a theoretical model of the shower plane is given where \([x_i, y_i]\) is the position of the detector \(i\), \([x_c, y_c]\) is the position of the shower core, \(d_i\) is the vertical distance of detector \(i\) from the shower plane and \(\theta\) is the zenith angle of the arriving primary particle. As we may see in the following chapter, the timing calibration focuses on calculating the \(d_i\) parameter for each detector of a particular cosmic ray event.

2.5 Simulation of events and reconstructed parameters

CORSIKA and GEANT4 are the simulation packages for studying the shower development and the passage of particles through matter correspondingly. The code is specifically parametrized for ALTO according to its altitude at the Chile site, the geometry of the array and the materials used for the construction of each detector. An additional routine then translates the photons collected by the PMTs into a signal waveform by taking into account the gain variation and transit time spread of the PMT.

One major difference between cosmic-ray and gamma-ray showers is the presence of a much larger fraction of muons in cosmic-ray showers. Muons suffer less interaction, and hence they can easily pass through both the WCD and the SD. Therefore, cosmic-ray
2.5. SIMULATION OF EVENTS AND RECONSTRUCTED PARAMETERS

initiated showers are expected to trigger more SD than gamma-ray showers, and this information can be used for the signal-over-background separation.

In Fig. 2.6, two examples of detector response to different particles are given: the left corresponds to a single muon of 1 GeV and the right to an electron of the same energy. It is significant to notice the generation of Cherenkov (aqua) and scintillation photons (red), as it is one of key parameters that makes the ALTO experiment unique: background proton initiated showers generate much more muons and so they can be easily discriminated from gamma rays [13].

For each cosmic ray event the relative arrival time of the signal and the total signal in each detector are determined. The time array is used to reconstruct the arrival direction of the air shower and the signal to determine the position of the shower core.

In Fig. 2.7 and 2.8 the relative time and signal are shown using colour maps. For each detector, the arrival time \( (\text{Trigger}_t) \) is reduced by the minimum trigger time and the signal \( (\text{Trigger}_s) \) is divided by the minimum signal recorded by the array for a specific event:

\[
\text{Trigger}_t = \text{Trigger}_t - \min(\text{Trigger}_t)
\]

\[
\text{Trigger}_s = \text{Trigger}_s / \min(\text{Trigger}_s)
\]

Fig. 2.7 shows the detectors that were hit by a shower particle, and Fig. 2.8 shows time evolution of the shower front and the position of the shower core. The left part of each figure presents the arrival time and the right the recorded signal.
Figure 2.6: GEANT4 simulation results for a 1 GeV muon (left) and a 1 GeV electron (right). The light blue traces correspond to Cherenkov photon; the red to scintillation photons; and the dark blue to secondary energetic gamma-rays. The lower plots show the simulated waveforms from the WCD (light blue) and SD (red), in photoelectrons after smearing by the PMT response. The figure is from a paper in preparation from the Astroparticle research group at Linnaeus University.

Figure 2.7: Colour-map presentation of results for Event 608, where the left hand plot shows the relative arrival time of the WCDs, and the right-hand shows the relative charge.
2.6. THE FLUX OF BACKGROUND HADRONIC EVENTS

Figure 2.8: Contour plot for Event 144, with the left showing the contours for the relative arrival times (indicating the direction of arrival) and the right showing the contours for the charge (indicating the core position).

It is important to mention that the expected reconstruction accuracy of the ALTO experiment is approximately 0.75 m for the shower core and 0.2° for the arrival direction (before applying gamma-hadron separation cuts), for cosmic rays of 10 TeV energy.

2.6 The flux of background hadronic events

As we plan to use cosmic rays for the timing calibration of ALTO, it is important to estimate the expected flux of protons (the dominant cosmic-ray species) that falls within the size of the ALTO array. Figure 2.9 shows the spectra for different cosmic-ray species taken from the Particle Data Booklet. The spectra follow a power-law behaviour above ∼10 GeV [14]. The proton spectrum can be represented by following equation:

\[
I_N(E) \approx 1.8 \times 10^4 / (E/1 \text{ GeV})^{-2.7} \text{nucleons/m}^2/\text{sec/sr/GeV (Eq.2.1)},
\]

where \(I_N(E)\) is the intensity of primary nucleons and \(E\) is the energy per nucleon [14]. In order to calculate the integral of flux over the energy range of 1–1.6 TeV we may use the quad function implemented in Python (scipy.integrate module) [20]. Nuclei arriving in the upper atmosphere with such energies are very significant to the timing calibration method that will be analyzed, as it offers an abundant and stable input to our WCDs in order to test the arriving time of signals between neighbouring tanks.

Taking into account the area occupied by the ALTO array which is \(A = \pi R^2 = 20106 \text{ m}^2\), the flux of nuclei coming from a zenith angle \(Z \leq 20°\) is calculated as:

\[
\approx 500 \text{ nuclei/second}
\]
Figure 2.9: Flux of nuclei of primary cosmic rays [14]. The elements heavier than Hydrogen are scaled for clarity, however, at the energies considered for ALTO (around a TeV) the protons (Hydrogen) to Helium ratio is about 1.4, with the remaining elements being negligible.
2.7 Advantages of the ALTO experiment

During the last decade, very-high-energy gamma-ray astronomy has seen a significant progress due to contribution from IACTs. However, these types of observatories are able to discover small-sized and slowly-varying gamma-ray emission due to their limited duty cycle and narrow field-of-view. ALTO belongs to another family of detectors, the Water Cherenkov Detector arrays which in general have the advantage of almost 100% duty cycle and a much larger Field-of-View (FoV) compared to IACTs [1]. For example, HESS has a FoV of maximum 5 degrees [36] while ALTO is expected to have a 30 degrees FoV.

As we have already mentioned, ALTO will be installed in the Southern Hemisphere, and it will provide a continuous monitoring of the southern sky for very-high-energy gamma-ray sources and the Galactic centre region.

On the other hand, depending on the energy and the type of the cosmic ray, there is an altitude above the Earth’s ground where the number of air shower particles reaches a maximum. This altitude is nearly 10 km for primary energies of 100GeV - 1TeV [14]. The higher the altitude of an observatory, the closer the approach to this point and the lower the energy threshold needed to record a signal. ALTO is planned to be installed at an altitude of $\sim 5$ km (almost 1 km higher than HAWC) which gives an extra advantage of about 40% lower energy threshold with respect to HAWC [12].

The small size of the tanks and their hexagonal shape offers a close-packed arrangement, which results in a fine sampling of the shower footprint giving better discrimination between gamma-ray and cosmic-ray showers. Moreover, the scintillator layer mounted underneath the water detector, will serve as a muon detector and improve the signal to background discrimination. Finally, ASIC Analogue Memories allows the read-out if a trigger condition has passed, allowing the digitization to be performed at a slower rate and in a more energy efficient way.

All these novel characteristics give a unique sensitivity and angular resolution, making ALTO a brilliant tool to study rapid transients such as coalescing neutron stars and black holes (from which there are spectacular recent gravitational wave signals measured), gamma-ray bursts, extended emissions such as Fermi Bubbles and extragalactic structures like nearby AGN. Other important scientific topics include the indirect detection of Dark Matter in Dwarf Galaxies and estimation of Extragalactic Background Light from the energy dependent absorption spectra of AGN [12][13].

2.8 Current and future state of ALTO

At the time of writing of this thesis, the construction of two prototype units at the Linnaeus University campus in Växjö (Sweden) is at the final stage (see Fig. 2.10). The first signals from the WCDs of these two units have been recorded since the 25th of May (see Fig. 2.11), with a single-detector rate of about 3 kHz and a coincidence rate of about 8 Hz. The full prototype setup will include several scintillator boxes that will be used to study the detector responses and for timing calibration tests.
Figure 2.10: Construction of the first ALTO WCD prototypes: the left picture shows the tanks installed at the Linnaeus University Campus (Växjö) and the right picture shows the crown on top of the PMT inside the WCD.

Figure 2.11: First signals (small and large) from the prototype WCDs: the photo-electrons captured by the PMTs are transformed into mV signals by the electronics. Red and green correspond to the signals from each tank, where a coincidence is required to digitize the signals.
Simulation efforts focus on optimizing the signal to background discrimination in order to achieve an energy threshold lower than 200 GeV, an angular resolution of the order of 0.1° and an increase in sensitivity by a factor of 5 or more compared to the HAWC experiment.

Future work involves the testing of the prototype units, further optimizations on the detector design and adaptation of electronics to the required timing and waveform precisions. Possibly during 2019, a full cluster of 6 units is planned to be directly installed on the chosen site in the Southern hemisphere.
Chapter 3

Timing Calibration in the ALTO project

“The problem of randomness requires an agile method”

Time synchronization to a precision of less than one nanosecond is very common in modern Astroparticle physics experiments where hundreds of data-acquisition stations (DAQs) need to precisely timestamp incoming sensor signals. Particularly for ALTO, the key quantity used in determining the arrival direction of the primary particle is the relative arrival times of particles in the air shower between the detectors. The relative arrival times, along with the amount of charge recorded in each detector, are also essential for determining the position of the shower axis (shower core or impact parameter) on the ground. Finally, it serves also as one of the elements to help to distinguish the gamma-ray signal events from the hadronic background, based on the quality of the timing fit to the particle wave-front.

3.1 The timing-calibration problem

Ground based air shower experiments focus on identifying the origin of very-high energy particles. In general, this can be determined by the relative times at which the particles comprising an Extensive Air Shower (EAS) reach the ground and for a water detector array, by examining the trigger times or time of maximum of the signals of sensors within each tank.

Figure 3.1 shows the profile of a simulated cosmic-ray event as seen by the ALTO array. The trigger times of the detectors can be found from the upper contour plot. One may correctly assume that the shower cascade arrived from a direction of -90° and travels across the y-axis. Moreover, from the second contour plot one may see that the shower core is found around the position \([x,y] = [-50\text{ m}, -50\text{ m}]\). The geometry of the tanks in the ALTO detector array offers a close packed arrangement which increases the accuracy of our measurements for the Azimuth angle, the Zenith angle and the initial energy of the primary particle - the reconstruction parameters.

The photodetector installed in each water detector detects the Cherenkov light emitted by fast moving particles traversing the medium, and sends its signal to the cluster electronics for each 6 units over cables, for acquisition as digital waveforms. However, the overall system may suffer from offsets occurring from temperature conditions which affect the transmission time in cables. This can be expressed for short periods of time as a constant offset added to the true trigger time and has a severe impact in sub-nanosecond
precision experiments, such as ALTO.

As a result, for a precise reconstruction it is necessary to develop a procedure by which the relative arrival times of typical signals from neighbouring tanks are calibrated. Taking into account that the shower front can be approximated by a plane and that the flux of background proton events is high and stable, we could execute our procedure for different plane dimensions in order to select the most optimum solution.

3.2 The White-Rabbit technology

The recent White-Rabbit (WR) technology developed at CERN, has offered a standard of time stamping and smart triggering with excellent performance. WR is a fully deterministic Ethernet-based network for general purpose data transfer that may synchronize hundreds of nodes with sub-nanosecond accuracy over fibre lengths of several kilometres, auto-calibrating so as to compensate environmental fluctuations [15]. Since the WR system has the advantage of being open hardware, it may be adapted to a great variety of
3.3. THE TIMING-CALIBRATION METHOD

For the ALTO observatory, the Cherenkov signal from shower particles is digitized when a certain level is exceeded – the trigger level. The time at which the trigger level occurs is tagged using the WR time stamping technology, to give the series of trigger times. WR sends a central clock to each cluster of tanks over fibre-optic cables and continually monitors the return journey time on the fibre, to allow each tank’s trigger time-tag to be correct relatively to better than nanosecond. As a result, the system up to the WR time-stamping card can be said to auto-calibrate.

Using the same Python function for calculating the centre of the array (Scikit Learn module [34]), we may group the detectors by a number of clusters according to the geometry of ALTO observatory. Fig. 3.2 gives a demonstration of how the WR technology is performed, by dividing the detector array into 207 clusters of 6 detectors.

3.3 The timing-calibration method

We have already mentioned that up to the White-Rabbit stamping card, the ALTO observatory is auto-calibrated. However, the time between the arrival of the Cherenkov photons on the light-detector in the tank and that of the signal exceeding the trigger threshold is not calibrated by the White-Rabbit. This time is made up of the transit time of the electrons in the dynode chain of the PMT, followed by the transit time in the cables going to the electronics of each cluster of 6 units. It is usually quite stable, but may vary slowly.
based on temperature conditions (transmission time in cables), or may change in a step if the high-voltage of the light detector is changed. In order to measure this added relative offset, it is necessary to develop a procedure by which the relative arrival times of typical signals from neighbouring tanks can be calibrated.

A solution could be based on the trigger times from the “particle-front” arriving from an intense gamma ray source at a known celestial position, or more flexibly using the background hadronic events. The latter have the advantage of being much more abundant and stable, but the disadvantage of coming from the whole sky, so some elementary direction reconstruction must be done to find the expected relative arrival times to be compared to the measurements, in an iterative procedure. Although both approaches can be studied using the existing simulations of the ALTO observatory, this thesis will focus on analyzing data from background hadronic events.

The initial data used to develop the timing calibration method were generated using CORSIKA for simulating the development of Extensive Air Showers, GEANT4 for simulating the passage of particles through the detector and ROOT for storing and reading the output files. For the timing calibration study, we develop several additional functions based on Python programming language, in order to extract information, present the results and identify system bottlenecks.

3.4 The cosmic ray event simulations

During the planning phase of an experiment dedicated to cosmic rays, a detailed theoretical model of the cascade created when a high energy primary particle enters the atmosphere is required. CORSIKA (COsmic Ray SImulations for KAscade) is a simulation code for EAS development in the Earth’s atmosphere. The code takes into account realistic models of the Earth’s atmosphere and magnetic field. For the data used in the present study, we choose the U.S. standard atmosphere, the magnetic field and the observation level for the ALMA site, the hadronic interaction model based on QGSJET-II-04 at high energies and FLUKA for energies below 200 GeV, and the electromagnetic interaction model based on EGS4 [5].

On the other hand, the propagation of the shower particles through matter is simulated with GEANT4 (GEometry ANd Tracking), a platform based on sophisticated Monte Carlo methods. The code takes into account the geometry of the array, the design of the detector unit, all the material properties such as density and refractive index as well as reflectivity and absorption of optical photons as a function of wavelength. The code also includes all the important interaction properties for different types of particles passing through the detector volume [4].

After giving CORSIKA and GEANT4 the configuration parameters for the ALTO project, simulations of cosmic ray events and gamma ray events over a very wide range of energies were carried out. For this thesis, only a file of cosmic ray events near the Zenith within the energy range of 1–1.6 TeV was used, since this is well above the expected threshold of ALTO, and so consists of events where there are many Cherenkov tanks which give a signal above the trigger level in the array. The expected signals generated by the detector array are stored in this file using the format from ROOT data analysis.
3.5. THE PYTHON ECOSYSTEM

ROOT is a scientific framework, mainly written in the C++ programming language, however, it uses several modules written in Python and R. It was initially developed in FORTRAN for the CERN experiment, but now it contains modules for astronomy and data mining applications [2].

As a result, the core functions of ROOT are being used for reading the initial file containing data of the ALTO detector array, where these are called by Python modules developed to read such files [32]. The analysis from then on is done using Python modules and routines developed here. The flow of information is represented schematically in Fig. 3.3.

3.5 The Python ecosystem

Python is a powerful interpreted programming language that runs on almost all platforms. It has a clean syntax, dynamic typing, high-level built-in data types and brings to the scientific computing a great number of free packages that include specialized tool-kits for fast computations and advanced visualizations.

Scientific computing may include matrix operations, integration, differentiation and statistics. By default, Python does not perform these operations except some basic mathematical functions. For solving advanced problems with sophisticated and efficient algorithms, Python comes with a great number of external packages. The core Python functions and these packages comprise the Python ecosystem [17]. The most significant tools that will be used for the completion of this thesis are:

- **Numpy and Scipy**: Numpy (Numerical Python) and Scipy (Scientific Python) are two of the most powerful Python packages that perform fast and accurate scientific computations. Numpy is assumed to be the basis of many other Python packages and allows developers to define and use multidimensional arrays. On the other hand, Scipy offers user friendly and efficient routines for linear algebra, optimization, signal and image processing that can be applied in almost all scientific areas.

- **Matplotlib – Mpld3 – Mayavi**: In order to visualize the results obtained in array form with the above mentioned packages, Matplotlib (Mathematics Plotting Library) generates high quality interactive graphs in a variety of formats. Although it originates from MATLAB graphics, it is developed in a “Pythonic” object-oriented way, and makes heavy use of Numpy. Mpld3 extends the Matplotlib functionality by using D3js, a Javascript library that generates interactive data visualizations and transforms common graphs in html files. On the other hand, Mayavi complements Matplotlib as it enables 3D scientific data visualization.

- **Sympy**: According to the syntax of Python, each variable has to be assigned with a value before it is used in a statement. Sympy (Symbolic Python) is a library that enables symbolic mathematics, which means that a variable can be defined unevaluated as a mathematical object. Moreover, the sympy.geometry module [23] allows to generate 3D geometrical entities and query for information and relationships between these objects, such as the distance between a point and a plane defined on a 3D Cartesian coordinate system.
Figure 3.3: The flow of information: the layers represent the observations of physical phenomena (layer 1), our simulations (layer 2) and visualization of the results (layer 3). The arrows show the flow of data from CORSIKA and GEANT4 to Python [4][5][6].
In this particular research, data generated from simulations of cosmic ray events are stored in numpy arrays. These arrays are imported to scipy and sympy functions, in order to extract the information we need for solving the timing calibration problem. Information is grouped for each event and results are visualized using Matplotlib (for histograms and 3D graphs) and Mpld3 (for viewing the detector parameters).

3.6 The Astropy project

The Python programming language is the fastest growing in the astronomy community. However, the packages developed by small groups of developers had little coordination and homogeneity. The Astropy project is an open-source and community-developed core Python library that includes advanced computational tools for Astronomy. Its main purpose is to specify general purpose Python packages such as Numpy and Scipy, in order to provide functionality to astronomy researchers [3][25].

Similarly to CORSIKA, GEANT4 and ROOT, Astropy minimizes code duplication and provides adequate documentation for new users. As we may see in the next chapter, although it provides high-level modules and functions that reduce programming effort and make code more readable, it performs operations on large datasets more slowly compared to Numpy and Scipy. Within the framework of this thesis, the functions of table and io modules of Astropy are going to be used. The following paragraph, gives a brief description of the core modules included in Astropy project:

- **astropy.units**: It provides unit conversion and decomposition into base units.
- **astropy.time**: It gives the opportunity to define and transform variables into different time standards, such as Universal Time, International Atomic Time and Julian Date.
- **astropy.coordinates**: It is a flexible Python coordinates library, providing an Application Programming Interface (API) for a straightforward transformation between coordinate systems.
- **astropy.table**: The Table class is a high-level wrapper to Numpy and gives the opportunity to define arrays of heterogeneous elements. Users may easily remove or add columns, mask values and export arrays to csv files. Moreover, the NDData class provides the tools for element manipulation of n-dimensional arrays. It also offers the transformation of Table objects into numpy arrays and pandas dataframes.
- **astropy.io.fits**: The Flexible Image Transport System (FITS) is an open standard that defines a digital file format for storage, transmission and processing of scientific images.
- **astropy.io.ascii**: It provides ASCII file manipulation including functions to read and write Table objects into files with ASCII-based formats.
- **astropy.wcs**: It contains utilities for managing World Coordinate Systems (WCS) transformations in FITS files.
• **astropy.cosmology**: It contains classes for simulating widely used cosmological models and functions for calculating some important parameters. Any given cosmology is represented with a class and an instance has attributes such as the Hubble parameter, CMB temperature etc.

• **astropy.io.votable**: Virtual Observatory (VO) tables is a new format introduced by the International Virtual Observatory Alliance for storing tables in XML format. This module supports read and write functionalities of VOTable files.

### 3.7 From ROOT to Astropy

Using CORSIKA and GEANT4 and setting-up the code specifically for the ALTO observatory, the Astroparticle Physics group at Linnaeus University generated the `output.root` file used here, which contains simulations of cosmic ray events within the energy range of $1–1.6 \text{ TeV}$ in the Zenith range from $0–21^\circ$. This file contains the recorded signals of the detectors over an operational period of approximately 7 seconds, in the form of a hierarchical structure – trees: one tree corresponds to one event, the branches are the detectors and the leaves are the parameters of the recorded signals.

**Structure of the data in the file:**

* EventTree:
  - Header
  - event_no
  - primaryID
  - energy
  - zFirstInteract
  - zenith
  - azimuth
  - coreX
  - coreY
  * Det0
    - detector/i:
      - dt_water/F:
      - count_water[600]/F:
      - dt_scint/F:
      - count_scint[600]/F
  ...
  * DetN

Having correctly installed ROOT and Python on a local Linux machine one can read the initial `output.root` using the statements in Listing B.1.

The generated histogram in Fig. 3.4 shows that detector 775 recorded a signal for over 100 events, with the arrival time varying between 1500 and 1700 ns. For the rest of the events included in the root file (approximately 3700) this detector was not hit by a particle.

At this point, one can store the data contained inside the root file in Numpy arrays as in Listing B.2.

The values generated using this script are equivalent to raw photoelectrons multiplied with the corresponding quantum efficiency. As a result, a routine has been developed to add a pulse shape from a template based on the rise and fall times of the PMT, to add the Gaussian fluctuations of the transit time in the dynode chain (the “transit time spread”, provided by the manufacturer), the Gaussian fluctuations in the intensity (also provided by the manufacturer), and to include the Poissonian fluctuations in the number of photons detected, all to build up the waveform which should be the result at the output of the PMT.
3.7. FROM ROOT TO ASTROPY

Figure 3.4: The “dt_water” bin for detector 775 for the events in the file used, which are the offsets of the first photo-electron seen with respect to the event’s zero time. The peak at 0 corresponds to events where this tank is not hit.

This routine then provides the time of the maximum of the resulting signal - $T_{\text{max}}$ - after the arrival of the first photon, as well as the integrated signal seen (SIGNAL). The $T_{\text{max}}$ value is added to the relative time that a detector saw the first photon ($\Delta T$) to give the time of the maximum of the waveform relative to a fixed previous time for the whole array $\Delta T_{\text{Pulse}}$:

$$\Delta T_{\text{Pulse}} = \Delta T + T_{\text{max}}$$

After transforming the Numpy array into an Astropy table, our initial file consists of 6 columns: EVENT_ID is the cosmic ray event, EVENT_EN is the energy of the primary particle, DETECTOR_ID is the detector of ALTO observatory, DELTA_T ($\Delta T$) is the time the sensor records the first photon (which gives the start time of the histogram containing the photon arrival times), $T_{\text{max}}$ ($T_{\text{max}}$) is the time within the resulting window that the waveform gets a maximum value (after the photons create a signals going through the PMT dynode chain) and SIGNAL is the magnitude of the recorded signal.

Sample of the initial output_root table

<table>
<thead>
<tr>
<th>EVENT_ID</th>
<th>EVENT_EN</th>
<th>DETECTOR_ID</th>
<th>DELTA_T</th>
<th>T_max</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>667</td>
<td>1506.5</td>
<td>29.0</td>
<td>17211.124</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>775</td>
<td>1522.0</td>
<td>24.0</td>
<td>621.899</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>452</td>
<td>1506.0</td>
<td>29.5</td>
<td>14037.185</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>454</td>
<td>1506.5</td>
<td>29.0</td>
<td>2889.433</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>455</td>
<td>1499.5</td>
<td>29.5</td>
<td>6850.172</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>555</td>
<td>1524.5</td>
<td>29.5</td>
<td>1587.188</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>655</td>
<td>1578.5</td>
<td>29.5</td>
<td>1322.175</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>865</td>
<td>1575.5</td>
<td>27.5</td>
<td>530.928</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>616</td>
<td>1539.5</td>
<td>29.5</td>
<td>617.417</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>827</td>
<td>1531.0</td>
<td>29.0</td>
<td>7543.656</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>828</td>
<td>1529.5</td>
<td>29.0</td>
<td>23111.712</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>933</td>
<td>1535.0</td>
<td>30.5</td>
<td>623.024</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>936</td>
<td>1545.5</td>
<td>28.5</td>
<td>3399.232</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>940</td>
<td>1558.5</td>
<td>28.5</td>
<td>885.473</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>1109</td>
<td>1533.5</td>
<td>27.0</td>
<td>2265.53</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>1111</td>
<td>1534.5</td>
<td>29.5</td>
<td>812.216</td>
</tr>
<tr>
<td>984</td>
<td>1276.07</td>
<td>1192</td>
<td>1573.0</td>
<td>29.5</td>
<td>754.177</td>
</tr>
</tbody>
</table>
CHAPTER 3. TIMING CALIBRATION IN THE ALTO PROJECT

3.8 The detector array

The detectors of ALTO observatory are positioned to form an array, extended over an approximate circle of 82.2 m radius. It can be seen in Fig. 3.5 that the distance between neighbouring tanks can vary between 3.6 and 6.5 m. The exact position of each detector is stored in the alto_xy.txt file and will be handled as an Astropy Table. Figure 3.5 is a depiction of the ALTO observatory generated using Matplotlib.

Sample of the initial alto_xy table

```
<Table length=1242>
<table>
<thead>
<tr>
<th>ID</th>
<th>EAST</th>
<th>NORTH</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>int32</td>
<td>float64</td>
<td>float64</td>
<td>float64</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>0</td>
<td>-43.8</td>
<td>-74.239</td>
<td>0.0</td>
</tr>
<tr>
<td>1</td>
<td>-40.2</td>
<td>-74.239</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>-42.0</td>
<td>-71.121</td>
<td>0.0</td>
</tr>
<tr>
<td>3</td>
<td>-38.4</td>
<td>-71.121</td>
<td>0.0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1238</td>
<td>37.2</td>
<td>71.773</td>
<td>0.0</td>
</tr>
<tr>
<td>1239</td>
<td>40.8</td>
<td>71.773</td>
<td>0.0</td>
</tr>
<tr>
<td>1240</td>
<td>39.0</td>
<td>74.891</td>
<td>0.0</td>
</tr>
<tr>
<td>1241</td>
<td>42.6</td>
<td>74.891</td>
<td>0.0</td>
</tr>
</tbody>
</table>
```

As can be seen in the next steps of this thesis, the timing calibration is based on the assumption that although the wave-front of a cosmic ray event is a hyperbola, it can be approximated as a sum of locally developed planes. One may ask the question “what is
the total number of these small planes that make up the hyperbolic shape of the shower front?”. In order to give a correct answer, we should make a compromise between the following two facts: the plane dimensions must be small enough in order not to approach a hyperbola, but also large enough to include as many detectors as possible. To the author’s point of view, it is similar to the compromise we make when trying to numerically calculate a definite integral of a function: break the interval into \( n \) subintervals, where \( n \) must be large enough for the function to be constant and at the same time small enough for the execution to be fast.

By analyzing the simulations, one may prove that the optimum wave-front approximation is a plane that extends over a circle area of radius \( R = 16 \text{ m} \) (see section 3.15) and calculations are going to be based on the Euclidean distances between detectors consisting a subset.

**Sympy** offers the distance function applied between two points in the 3D space, however, it is more efficient to store a table \( D \) of 1242 x 1242 size, where the element \( D_{ij} \) is the distance between detectors \( i \) and \( j \). The **Spatial** module of **Scipy** offers the distance function [18] that takes the x-y coordinates of a set of points and returns their distances. The Python code in Listing B.3 generates the desired table.

As one may expect, the diagonal elements of the generated table are zeros as they represent the distance between a detector and itself. If the table is stored as a “.txt” file, it has a size of 27 Megabytes and it takes approximately 14 seconds of computational time to be loaded from memory and represented as an astropy table object. However, due to the fact that distance calculations are executed thousands of times while running an instance of the plane fit method, it is more effective to load this large file once and retrieve its elements in almost zero time.

**Sample of the scipy_distance table**

<table>
<thead>
<tr>
<th>ID1</th>
<th>ID2</th>
<th>ID3</th>
<th>ID4</th>
<th>...</th>
<th>ID1238</th>
<th>ID1239</th>
<th>ID1240</th>
<th>ID1241</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID1</td>
<td>0.00</td>
<td>3.59</td>
<td>3.60</td>
<td>6.23</td>
<td>ID1238</td>
<td>ID1239</td>
<td>ID1240</td>
<td>ID1241</td>
</tr>
<tr>
<td>ID2</td>
<td>3.59</td>
<td>0.00</td>
<td>3.60</td>
<td>3.60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID3</td>
<td>3.60</td>
<td>3.60</td>
<td>0.00</td>
<td>3.60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID1239</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.59</td>
<td>0.00</td>
<td>3.60</td>
<td>3.60</td>
</tr>
<tr>
<td>ID1240</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.60</td>
<td>3.60</td>
<td>0.00</td>
<td>3.60</td>
</tr>
<tr>
<td>ID1241</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.23</td>
<td>3.60</td>
<td>3.60</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**3.9 Creating the final table**

For solving the timing calibration problem, one must loop over the events and search for detectors that seem to be “out of phase” with their neighbours. In order to achieve this, a table that stores combined data from both the event simulations and the the detector relative position tables, needs to be created. The script in Listing B.4 (see Fig. 3.6 for Unified Modelling Language - UML - equivalent) takes the initial \( \text{output_root} \) table, and adds 2 extra columns of the x-y detector position, according to the \( \text{alto_xy} \) table.

**Sample of the final Astro_table**

<table>
<thead>
<tr>
<th>EVENT_ID</th>
<th>EVENT_EN</th>
<th>DETECTOR_ID</th>
<th>...</th>
<th>SIGNAL</th>
<th>DETECTOR_X_POS</th>
<th>DETECTOR_Y_POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------</td>
<td>--------</td>
<td>------------</td>
<td>-----</td>
<td>--------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>1025</td>
<td>1178.22</td>
<td>667</td>
<td>...</td>
<td>17211.124</td>
<td>-11.4</td>
<td>4.223</td>
</tr>
</tbody>
</table>
3.10 Looping over the events

The appropriate data have been stored in the final_astro_table, so we may now start extracting useful information. The initial data can be grouped according to the EVENT_ID or DETECTOR_ID columns (using the `group_by` function of the `astropy.table` module in Python) in order to examine the data for a particular cosmic-ray event or a detector correspondingly. Figure 3.8 shows the distribution of cosmic ray events over the detector array (i.e. the participation rate of tanks in events). From our simulations, we may see that the central tanks are expected to record signals more frequently than those found on the boundaries of the array. The Fig. 3.9 is a histogram based on the inverse grouping, as it depicts the total number of detectors hit by a particle of the shower per event and gives an average value of 140 for this range in energy and Zenith angle.
3.11 Plane Fit

We have mentioned in the previous chapters that when a cosmic ray enters the Earth’s atmosphere, it initiates an EAS. The development of an EAS highly depends on the type of the cosmic ray, its primary energy and the altitude of the observatory. Moreover, as we have seen in the beginning of this chapter, the path of each of the generated sub-particles is a random walk and can be simulated using CORSIKA. The overall shower front of the cascade has a hyperbolic shape, however, it can be locally approximated with a plane. In the particular case of a detector array, the plane is defined by selecting a subset of neighbouring tanks and taking account not only their x-y-z position but also their relative trigger times.

In mathematical terms, a plane in x-y-z space is defined by the following equation:

\[ Ax + By + Cz + D = 0 \]

If the normal vector to the plane is \( \mathbf{v} = [a, b, c] \) and a point within the 3D-space is \( \mathbf{p} = [x_0, y_0, z_0] \), then the signed distance from the point to the plane is:

\[ D = \frac{Ax_0 + By_0 + Cz_0 + D}{\sqrt{A^2 + B^2 + C^2}} \]

The detectors that record a signal for each cosmic ray event can be easily plotted in a 3D plane using their x-y-z positions. However, the relative trigger times of the tanks may be used to give us a sense of the development of the shower front: the difference between the trigger times can be transformed from nanoseconds to meters as they represent the
CHAPTER 3. TIMING CALIBRATION IN THE ALTO PROJECT

Figure 3.8: The distribution of events over the detector array: each point represents a WCD and its colour indicates the number of times the particular detector was triggered by a shower particle (for events included in the simulation file).

Figure 3.9: Histogram showing the distribution of the number of detectors hit by a shower particle per event in the file used (the average value is 140 detectors per event).
3.11. PLANE FIT

Figure 3.10: Defining a plane in 3D: the plane is defined by the nonzero vector \( \mathbf{n} \) normal to the plane and a point on its surface [16].

Figure 3.11: The UML diagram of the plane fit method.

distance travelled by the shower particles before entering the medium inside each tank. As a result we need to define a function that simulates a plane and calculates the optimum parameters that best fit to our simulations.

The script in Listing B.5 defines the plane_ABCt function that takes as input the Azimuth and Zenith angle of a vector normal to a plane and the Time array of the detectors for a specific event. The Time is transformed from nanoseconds into meters by multiplying the column with the speed of light (included in \texttt{scipy.constants} module) and the method returns the vertical distances of each detector array to the plane. On a first approach, we shall guess the plane parameters that could fit our data and later on we are going to make use of a least-squares method (implemented in Python) in order to retrieve the optimum plane solution that suits best to our simulations.
3.12 Optimized plane solution

The \texttt{leastsq} method of the \texttt{scipy.optimize} module will be used to minimize the sum of the squares between the function used to model the plane equation and the data representing the simulations of the cosmic ray events. The arguments passed to leastsq are the function to be minimized, a first guess of the solution and some additional parameters [21][26].

As we have already mentioned, in order to calculate the plane parameters that best fit to cosmic ray event simulations, we should take into account not only the x-y-z positions of the detectors but also the relative trigger times of recorded signals. After transforming the time array into distances, the leastsq method will calculate the plane parameters that minimize the sum of distances between the plane and the detectors. In our particular case, we insert to the \texttt{leastsq} method the \texttt{plane_ABCt} function, an initial estimate of the vector \texttt{p0} consisting of Azimuth-Zenith-Average(Time) parameters and the \texttt{Time} array according to our simulations. Afterwards, we use the returned parameters to calculate the orthogonal residual distances between that optimum plane and the detectors with their corresponding times.

The code implemented in Python is performing the following tasks:

1. Read data of the \texttt{final_astro_table} and group them according to \texttt{EVENT_ID} column
2. For each event select randomly a detector (DS) which was hit by a particle
3. Calculate the Euclidean distance between DS and all other detectors in the array
4. Select data only for detectors close to DS, found within a radius of 16 meters
5. Use the \texttt{leastsq} method to calculate the plane parameters that fit best to our data, without taking into account the trigger time of DS (assumed to be affected by an offset)
6. Apply the \texttt{plane_ABCt} method to calculate the residuals of all detectors to the plane
7. Generate a grid near the selected detector, calculate the interpolated values and plot the plane to evaluate the results

The script in Listing B.6 gives the statements in Python that performed the above-mentioned tasks.

Figure 3.12 plots the optimized plane solution and the x-y-time position of each detector in the 3D plane for event 764. The scattered point in red is the detector being examined by the timing calibration method, and the points in blue are its neighbours found within a radius of 16 m. The third ($c \times \Delta T$) axis of this graph is based on the assumption that the trigger time of each detector is equivalent to the distance travelled by the shower particle which moves at nearly the speed of light and penetrates the medium inside the corresponding detector. The \textbf{blue segments} represent the residuals which are to the perpendicular distances of each detector to the optimum plane solution.
Figure 3.12: The optimized plane solution for a specific event: the magenta coloured area represents the optimum plane solution, the red coloured point is the detector under consideration and the blue coloured points are its neighbours. The blue segments starting from each point and ending to the plane are the residuals.
3.13 Generating residuals for a subset of detectors

In this section we select a subset of detectors and generate statistics regarding their residuals. In Fig. 3.14, we show the relative position of a selected group of detectors D and some characteristics regarding their relative x-y position and the number of times they have been penetrated by a shower particle for a time period of approximately 7 seconds.

As we have already mentioned, although the overall shower front is rather complex to simulate, locally in the neighbourhood of each detector it can be approximated by a plane. How good this approximation is, depends on the dimensions of the plane and the number of detectors penetrated by a shower particle that are included within its boundaries.

Taking into account that a plane needs at least 3 points in order to be defined, one may claim a plane with radius corresponding to the maximum distance between two neighbouring tanks $R = 6.5$ m will satisfy the problem constraints as according to the array
geometry, a disk with a radius larger than 4 metres always includes at least 5 tanks.

However, this is not a correct approach, as the effectiveness of the timing calibration method depends also on the overall number of events we take into account: the more the number of events, the more accurate the statistics regarding the residuals. As we will see in the next chapter, in order to balance the accuracy of the plane approximation and the statistical correctness of the generated results, the residuals should be generated for a plane (disk) with radius $R = 16$ m and the minimum number of detectors required will be $N = 4$ detectors. We should mention that the generated residuals are expressed in meters, and a residual of 0.3 m is equivalent to 1 ns time difference, as it corresponds to the path of the particle moving nearly at the speed of light.

### 3.14 Offset identification

As we have previously explained, due to temperature conditions the transmission time of the signals may vary and this will finally have an impact expressed as a offset added to the trigger time of a particular detector for a sequence of cosmic ray events. Fortunately, this offset is expected to be constant over short periods of time and as a result, it could be possible to be identified.

The plane fit method described in the previous steps, will help us find a detector that could be affected by an offset. As we have already seen in Figure 3.13, the average value of the residuals generated from event simulations corresponding to a few minutes of data acquisition of a real observatory, is approximately zero. However, if a detector suffers from an offset, this average value is expected to be equal to that offset, as the latter is assumed be constant for a time period less than 1 minute. This can be parallelized with trying to identify the position of a nail in a floor, where multiple carpets are vertically stacked over it: the nail is expected to be at the position where the surface of each carpet curves.

In order to test the accuracy of the plane fit method, we shall randomly select a detector from group D and add a constant value of 1 ns on the trigger time column of the final_astro_table. We shall then execute the code multiple times to evaluate the results. Figure 3.15 gives the ID of the selected detector in which the offset was added and the table of the generated residuals.

As we may easily see, the detector which has an offset has also an average residual greater than 0.2 m. As a result, the code we have already developed may be used in an inverse manner: gather the data of cosmic ray events over a period of several minutes, generate the residuals for all detectors and if the average value exceeds 0.2 m, mark it as “affected by an offset”.

### 3.15 Calculating the residuals for different R-N sets

In this section we are going to justify our initial assumption that the optimum parameters for executing the plane fit method is $R = 16$ m for the radius around the selected detector and $N = 4$ for the minimum number of detectors. For this purpose, we select detectors
Figure 3.15: As for Fig. 3.14, but showing the residuals after adding an offset to detectors 773, 774, 775 respectively (shown by the red points).
3.15. Calculating the residuals for different R-N sets

Figure 3.16: The final UML diagram of the method.
In Python programming language this can be performed in one script by defining the finite sets $D = \{773, 774, 775\}$, $R = \{10 \text{ m}, 16 \text{ m}, 22 \text{ m}\}$ and $N = \{3, 4, 5\}$ of sympy: when multiplying these sets we may execute the same method for all possible combinations of these elements. The output can be saved as an astropy table object which can be grouped by $R$ and $N$ values in order to visualize the differences between the obtained results.

As we may see from the Table 3.1 table only for the set $[R, N] = [16, 4]$ the residuals generated are lower than 0.1 m (equivalent to 0.3 ns) for all three selected detectors.
Chapter 4

Results and further discussion

“The calibration is based on the space-time correlation of detectors”

4.1 Residual analysis for a subset of detectors

In order to evaluate the timing calibration method described in chapter 3, a subset of detectors will be selected and their residuals generated from simulations of cosmic ray events will be examined. For this purpose, we define a set $D$ of detectors with DETECTOR_ID within the set $\{771, 772, 773, 774, 775\}$ which have been penetrated by a shower particle $N$ times in the simulation file used, as can be seen in Fig. 4.1.

The relative arrival times of the shower particles in each detector is passed to the `plane_fit` method described in the previous chapter and the residuals are saved in different text files. On these files, we perform statistical tests that will evaluate the accuracy of the timing calibration method.

The tests will be based on the following functions of the Python programming language:

- `hist(x)`: The function is included in the `matplotlib.pyplot` module and draws a histogram of $x$. It can be parametrized to plot different bins of $x$ in a vertical or horizontal orientation.

- `norm(x)`: This function is included in the `scipy.stats` module and it returns the mean and standard deviation of an array $x$.

- `norm.pdf(x,mean,std)`: This gives the probability density function defined in the standardized form according to the mean and standard deviation values of a set $x$.

- `gaussian_kde(x)`: The kernel density estimation (KDE) is a method to evaluate the probability density function of a variable in a non-parametric way and includes automatic bandwidth determination.

In the following figures, we handle the residuals of each detector as an 1D array, pass it as input to each of the function described above and plot the corresponding results.

The code developed in chapter 3 shows that if we add an $1\text{ ns}$ offset to the array of the arrival time of signals, we would get the same results with an average residual value
Figure 4.1: The selected subset of detectors are the red-blue colored points. Their color reveals the number of events they were triggered by a shower particle according to the simulation file that was used.

Increased by 0.3 m. Fig. 4.3 repeats the statistical evaluation for detector 771.

By comparing the generated graph in Fig. 4.3 with the corresponding one in Fig. 4.2, we may easily confirm that the average residual value is increased by 0.3 m. On the other hand, the standard deviation is the same as the offset added to the time array is a constant value.

4.2 Further discussion about the timing calibration method

The method analyzed in this thesis is based on the assumption that for a time interval of less than one minute, only one detector within the array would be prone to an offset. However, the same problem may occur in two or more detectors at the same period of time.

According to the plane_fit method, in order to examine each detector we should filter our initial data in order to include only its neighbouring detectors, found within a radius of \( R = 16 \text{ m} \) with respect to the selected one. As a result, if the detectors that suffer from constant offsets are being positioned at distances greater than \( R \), the method will still generate accurate results.

On the other hand, in the particular case where two or more of these detectors are spatially correlated (their distance is less than 16 m), the method will generate a solution of the optimized plane that differs from the theoretically expected one that describes the
4.2. FURTHER DISCUSSION ABOUT THE TIMING CALIBRATION METHOD

Figure 4.2: Statistical evaluation of results for the selected set of detectors: the upper part of each figure gives the histogram of the residual distribution and the lower part plots the distributions normalized to 1. The green curve is a normal distribution while the red curve is the Gaussian KDE (Kernel Density Estimator). Mean of the normal distribution is used as the main result for the timing calibration discussed here.
CHAPTER 4. RESULTS AND FURTHER DISCUSSION

Figure 4.3: Statistical evaluation of results adding 1 ns offset to detector 771. The upper figure gives the histogram of the residual distribution and the lower part plots the distributions normalized to 1. The green curve is a normal distribution while the red curve is the Gaussian KDE (Kernel Density Estimator).

development of the air shower-front. In order to clarify the above mentioned two cases, Fig. 4.4 gives a depiction of the ALTO array where more than two detectors suffer from a time offset: in the first one detectors lie at distances larger than \( R \) in the second one detectors are spatially correlated.

In order to demonstrate the impact of having two nearby detectors suffering the same offset, we repeat the plane fit method by adding 1 ns to the time array of both detectors 771 and 775. We may see in Fig. 4.5 that the average residual has a lower value and this is reasonable as the detector 775 “pulls” the optimized plane solution to higher values.

The accuracy of the method when two nearby detectors suffer a time offset simultaneously will be examined at the next phase of the ALTO project, when a cluster of six detectors will be installed on the proposed site. However, it seems to depend on the following set of parameters:

- The radius \( R \) selected in the plane_fit method: The larger the radius, the higher the probability to include more detectors that suffer from an offset.

- The minimum number of detectors \( N \) needed to execute the plane_fit method: The larger the \( N \), the lower the impact of one problematic detector to another as its statistical weight in calculating the residuals is decreased.

- The overall number of events \( N_e \) for which both detectors were triggered: If the nearby detectors recorded a signal for different cosmic ray events, then they are time-uncorrelated and we may still apply the proposed method in this thesis.

In order to quantify the significance of the third parameter (\( N_e \)) described above, we create a table that shows how frequent is the case where two or three detectors of the set [771, 772, 773] are triggered by the same cosmic-ray event. For this purpose, we
4.2. FURTHER DISCUSSION ABOUT THE TIMING CALIBRATION METHOD

Figure 4.4: Spatially uncorrelated (left) vs. spatially correlated (right) set of detectors: the detectors under consideration are coloured red and their neighbours are magenta.

Figure 4.5: Statistical evaluation of results adding $1 \text{ ns}$ offset to both detectors 771 and to a neighbouring one: the average residual is reduced by 0.03m as the optimized plane solution is "pulled" to higher values. The upper figure gives the histogram of the residual distribution and the lower part plots the distributions normalized to 1. The green curve is a normal distribution while the red curve is the Gaussian KDE (Kernel Density Estimator).
group our initial data based on the detector ID and transform the table into a \textbf{FiniteSet} of \texttt{sympy}. This gives a \texttt{dset} of cosmic-ray events where detector \texttt{d} was triggered by an air-shower particle. Finally, we use the \texttt{intersect} function to find common events between two or more detectors.

<table>
<thead>
<tr>
<th></th>
<th>\texttt{771}_{set}</th>
<th>\texttt{772}_{set}</th>
<th>\texttt{773}_{set}</th>
<th>\texttt{771}<em>{set} \cap \texttt{772}</em>{set}</th>
<th>\texttt{771}<em>{set} \cap \texttt{772}</em>{set} \cap \texttt{773}_{set}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{Number of events}</td>
<td>54</td>
<td>40</td>
<td>48</td>
<td>20</td>
<td>14</td>
</tr>
<tr>
<td>\textbf{Events set}</td>
<td>{12, 97, 148, ..., }</td>
<td>{82, 322, 375, ..., }</td>
<td>{50, 97, 248, ..., }</td>
<td>{375, 608, 737, ..., }</td>
<td>{608, 737, 925, ..., }</td>
</tr>
</tbody>
</table>

As we may see that in the particular case where two detectors suffer simultaneously from an offset over a period of 7 seconds, approximately one-third of the data acquired correspond to cosmic-ray events where both detectors are hit by a particle. As a result, the plane fit method described above will generate correct results.

\subsection*{4.3 Further research for the ALTO project}

Other research topics that should be examined before ALTO enters its full operational phase, are described in the following paragraphs:

- **Array geometry:** The detectors consisting the ALTO array are positioned in a close packed arrangement that offers a fine sampling of the air shower particles generated by cosmic rays. However, an experiment dedicated to very-high-energy gamma-ray Astronomy, has to make a compromise between its sensitivity and the overall number of detected events. For example, by increasing the minimum distance between nearby detectors from 3.6 m to 5 m, we increase the overall area covered by the ALTO array (and as a result, the number of recorded cosmic ray events), with the cost of a reduced sensitivity of the reconstructed parameters. Taking also into account the expected life-cycle of the ALTO project, a method should be developed to evaluate the overall information generated from collecting data with different geometries of the initial array.

- **Machine learning algorithms:** As we have already seen in this thesis, compromises have to be made also for solving the timing calibration method in an efficient way: by selecting a smaller radius (\textit{R}) around a detector and a higher value for the minimum number of detectors (\textit{N}) needed to execute the method, we achieve a better approximation to a plane at the cost of an increased time interval of the data acquisition. The latter may have an effect on our initial hypothesis that the offset added to the arrival time is constant. As a result, a machine learning algorithm that calculates the average residuals for different combinations of \textit{R}-\textit{N} values and selects the optimum set should also be developed.

- **Offset analysis:** By analyzing more real data, it could be possible to gain knowledge about the profile of this offset particularly for the ALTO experiment. For example, as we have mentioned previously in this chapter for a time duration of less than one minute, detectors within the array may suffer from constant offsets of different magnitudes. These offsets can be further examined over a period of several weeks
or months, in order to identify other parameters that may have an impact on the response of each detector.

- Code optimization: Minimising the execution time for the method is also crucial. For example, as we have already mentioned in chapter 3, the simulations of cosmic-ray events are loaded as `astropy.table` objects. Astropy provides user-friendly methods for performing table operations and a more elegant view of the arrays. However, when running the same routines using `numpy arrays` or `pandas dataframes`, the execution time is reduced approximately by a half. As a result, in order to perform an online calibration, we have to carefully examine the type of variables used in our methods.

- Power Generation: The energy consumption of the array is another important issue, as off-grid photovoltaic (PV) systems are going to be used. As the power consumption of each cluster of detectors will be approximately 100 W, more than 200 solar panels are going to be used. The PV array is sensitive to shadowing effects and the solar charge controllers have to be programmed in order to maximize the charge-discharge cycles of batteries.

## 4.4 Conclusion

The timing calibration method described in this thesis offers a sub-nanosecond accuracy not only in the particular case of one detector, but also when two or more detectors suffer from a time offset. The method becomes inaccurate when the affected detectors are correlated in position and time, which happens when they are positioned closer than 16 m and they are triggered by the same cosmic-ray events.

In the next few years, it is hoped that the ALTO project will enter its operational phase. The first cluster of detectors is expected to record thousands of cosmic-ray events of energies above a few hundred GeV each day. In a time period of less than one month, the size of the data recorded could reach the order of several Gigabytes. However, based on the experience gained with the prototype, the sampling frequency can be reduced while still keeping good timing characteristics by doing a fit of the maximum of the waveform, and also the integrated charge can be calculated on-the-fly or at least on-site, with these allowing the reduction of the data volume by a factor of a few hundred.

In order to perform an effective timing calibration method and to gain more experience before entering the next stage of installing the overall 207 clusters of the ALTO array, sophisticated algorithms that perform fast computations with efficient memory consumption should be developed. These algorithms should be able to identify not only constant offsets in the detector trigger times, but also temporary changes in the trigger times resulting from other factors like temperature changes and PMT voltage modifications, in order to achieve a sub-nanosecond accuracy in the timing calibration.
# Appendix A

## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Expansion</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGN</td>
<td>Active Galactic Nuclei</td>
<td>3</td>
</tr>
<tr>
<td>ALMA</td>
<td>Atacama Millimeter/Submillimeter Array</td>
<td>9</td>
</tr>
<tr>
<td>CMB</td>
<td>Cosmic Microwave Background</td>
<td>2</td>
</tr>
<tr>
<td>CORSIKA</td>
<td>Cosmic Ray Simulations for KAscade</td>
<td>26</td>
</tr>
<tr>
<td>CTA</td>
<td>Cherenkov Telescope Array</td>
<td>1</td>
</tr>
<tr>
<td>DAQs</td>
<td>Data Acquisition Stations</td>
<td>23</td>
</tr>
<tr>
<td>EAS</td>
<td>Extensive Air Shower</td>
<td>23</td>
</tr>
<tr>
<td>ESO</td>
<td>European Southern Observatory</td>
<td>9</td>
</tr>
<tr>
<td>GEANT4</td>
<td>Geometry And Tracking</td>
<td>26</td>
</tr>
<tr>
<td>GRB</td>
<td>Gamma Ray Burst</td>
<td>3</td>
</tr>
<tr>
<td>HAWC</td>
<td>High Altitude Water Cherenkov Gamma-Ray Observatory</td>
<td>1</td>
</tr>
<tr>
<td>HESS</td>
<td>High Energy Stereoscopic System</td>
<td>6</td>
</tr>
<tr>
<td>IACT</td>
<td>Imaging Atmospheric Cherenkov Telescope</td>
<td>6</td>
</tr>
<tr>
<td>Matplotlib</td>
<td>Mathematics Plotting Library</td>
<td>27</td>
</tr>
<tr>
<td>Numpy</td>
<td>Numerical Python</td>
<td>27</td>
</tr>
<tr>
<td>PMT</td>
<td>Photomultiplier Tube</td>
<td>13</td>
</tr>
<tr>
<td>PV</td>
<td>Photovoltaic</td>
<td>51</td>
</tr>
<tr>
<td>Sklearn</td>
<td>Scientific Toolkits for Machine Learning</td>
<td>10</td>
</tr>
<tr>
<td>Scipy</td>
<td>Scientific Python</td>
<td>27</td>
</tr>
<tr>
<td>SD</td>
<td>Scintillation Detector</td>
<td>13</td>
</tr>
<tr>
<td>Sympy</td>
<td>Symbolic Python</td>
<td>27</td>
</tr>
<tr>
<td>TeV</td>
<td>Tera Electron Volt</td>
<td>6</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
<td>33</td>
</tr>
<tr>
<td>VHE</td>
<td>Very High Energy</td>
<td>1</td>
</tr>
<tr>
<td>WCD</td>
<td>Water Cherenkov Detector</td>
<td>1</td>
</tr>
<tr>
<td>WRT</td>
<td>White-Rabbit Technology</td>
<td>24</td>
</tr>
</tbody>
</table>
Appendix B

Code listings

Listing B.1: Reading ROOT files in Python

```python
import ROOT
f = ROOT.TFile("/output.root")
t = f.Get("EventTree")
t.Draw("Det0.dt_water") # generates a dt plot for detector 0 for all events
\label{ROOT_read}
```

Listing B.2: Storing in Numpy arrays

```python
for i,event in enumerate(t):
    print(event.dt_water)
    vals = np.append(vals,event.dt_water)
    if event.dt_water>0:
        print(np.array(event.count_water,''))
```

Listing B.3: Generate table of distances

```python
from scipy.spatial import distance
from astropy.table import Table
file_path = "alto/alto_xy.txt"
detector_xy = np.loadtxt(file_path)
det_xy_east_west = detector_xy[:,1:3]
distance_array = distance.cdist(det_xy_east_west, det_xy_east_west, 'euclidean')
```

```
### Develop the Table that stores distances between all Detectors
###
from scipy.spatial import distance
from astropy.table import Table

file_path = "alto/alto_xy.txt"
detector_xy = np.loadtxt(file_path)
det_xy_east_west = detector_xy[:,1:3]
distance_array = distance.cdist(det_xy_east_west, det_xy_east_west, ‘euclidean’)

### Results: distance_array[i][j] = the euclidean distance between detectors i and j
###
```

# -*- coding: utf-8 -*-

```python
# Created on Tue Mar 13 10:11:25 2018
Add detector x-y coordinates to the initial output_root.txt file

from astropy.io import ascii
from astropy.table import Column
import numpy as np
def add_xy_coords():
alto_xy = ascii.read(‘alto_xy.txt’)
output_root = ascii.read(‘output_root.txt’)
det_x_pos = Column(np.arange(len(output_root)), name=’DETECTOR_X_POS’).astype(float)
det_y_pos = Column(np.arange(len(output_root)), name=’DETECTOR_Y_POS’).astype(float)
for i in range(len(output_root)):
det_x_pos[i] = alto_xy[alto_xy[‘ID’] == output_root[i][‘DETECTOR_ID’]][‘EAST’]
```
APPENDIX B. CODE LISTINGS

Listing B.4: Add detector positions to output_root table

```python
def add_xy_coords():
    det_y_pos = alto_xy[alto_xy['ID'] == output_root['DETECTOR_ID']]['NORTH']
    print(output_root['DETECTOR_ID'])
    print(det_y_pos)
    output_root.add_column(det_y_pos)
    return output_root

if __name__ == '__main__':
    output_root = add_xy_coords()
    ascii.write(output_root, 'final_astro_table.txt')
```

Listing B.5: Function defining the plane

```python
from scipy.optimize import leastsq
from scipy import interpolate

azR, zenR, t0 = 0., 0., np.mean(Time_excl)  # first guess of parameters
p0 = np.array([azR, zenR, t0])
# Tanks_excl & Time_excl = data from output_root file - DS is excluded
sol = leastsq(plane_ABCt, p0, args=(None, Tanks_excl, Time_excl))[0]
# call the plane_ABCt function to calculate the optimum residual values
T_resid = plane_ABCt(sol, None, Tanks, Time)

def plane_ABCt_interpolate(params, signal, XYZ, Time, event_id, d_exc_id, ax):
    Az, Zen, t = params[:3]
    A = np.cos(-Az)*np.sin(Zen)
    B = np.sin(-Az)*np.sin(Zen)
    C = np.cos(Zen)
    A, B, C = np.array([A, B, C])
    distance = Time*constants.c/1e9 - (A*XYZ[:,0]+B*XYZ[:,1]+C*XYZ[:,2]) - t
    detector_excluded_x = XYZ[d_exc_id][0]
    detector_excluded_y = XYZ[d_exc_id][1]
    print(detector_excluded_y)
    # Grid to cover XY domain
    x = np.arange(min(XYZ[:,0]), max(XYZ[:,0]), 0.1)
    y = np.arange(min(XYZ[:,1]), max(XYZ[:,1]), 0.1)
    X, Y = np.meshgrid(x, y)
    z = A*X + B*Y + t
    detector_excluded_x = XYZ[d_exc_id][0]
    detector_excluded_y = XYZ[d_exc_id][1]
    print(detector_excluded_y)
    # Used to evaluate the generated results
    f = interpolate.interp2d(x, y, z, kind='cubic')
```
Listing B.6: Find the plane giving minimum times to detectors
Bibliography


