Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2009, Article ID 737689, 15 pages
doi:10.1155/2009/737689

Research Article

Hardware Architecture for Pattern Recognition in Gamma-Ray Experiment

Sonia Khatchadourian,[1] Jean-Christophe Prévotet,[2] and Lounis Kessal[1]

[1] *ETIS, CNRS UMR 8051, ENSEA, University of Cergy-Pontoise, 6, Avenue du Ponceau, 95014 Cergy-Pontoise, France*
[2] *IETR, CNRS UMR 6164, INSA de Rennes, 20, Avenue des buttes de Coësmes, 35043 Rennes, France*

Correspondence should be addressed to Sonia Khatchadourian, sonia.khatchadourian@ensea.fr

Received 19 March 2009; Accepted 21 July 2009

Recommended by Ahmet T. Erdogan

The HESS project has been running successfully for seven years. In order to take into account the sensitivity increase of the entire project in its second phase, a new trigger scheme is proposed. This trigger is based on a neural system that extracts the interesting features of the incoming images and rejects the background more efficiently than classical solutions. In this article, we present the basic principles of the algorithms as well as their hardware implementation in FPGAs (Field Programmable Gate Arrays).

1. Introduction

For many years, the study of gamma photons have led scientists to understand more deeply the complex processes that occur in the Universe, for example, remnants of supernova explosions, cosmic rays interactions with interstellar gas, and so forth. In the 1960s, it has finally been possible to develop efficient measuring instruments to detect gamma-ray emissions, thus enabling to validate the theoretical concepts. Most of these instruments were built in order to identify the direction of gammas rays. Since gamma photons are not deflected by interstellar magnetic fields, it becomes possible to determine the position of the source accurately. In this context, Imaging Atmospheric Cherenkov Telescopes constitute the most sensitive technique for the observation of high-energy gamma-rays. Such telescopes provide a large effective collection area and achieve excellent angular and energy resolution for detailed studies of cosmic objects. The technique relies upon Cherenkov light produced by the secondary particles once the gamma-ray interacts with the atmosphere at about 10 km of altitude. It results a shower of secondary particles, that also may interact with the atmosphere producing other particles according to well-known physical rules. By detecting shower particles (electrons, muons, protons), it is then possible to reconstruct the initial event and determine the precise location of a source within the Universe.

In order to determine the nature of the shower, it is important to analyze its composition, that is, determine the types of particles that have been produced during the interaction with the atmosphere. This is performed by studying the different images that are collected by the telescopes and that are generally representative of the particle type. For example, gamma-ray showers usually have thin, high-density structures. On the other hand, protons are quite broad with low density.

The major problem in these experiments is that the number of images to be collected is generally huge and the complete storage of all events is impossible. This is mainly due to the fact that data-storage capacity is limited and that it is impossible to keep track of all incoming images for off-line analysis.

In order to circumvent this issue, a trigger system is often used to select the events that are interesting (from a physicist's point of view). This processing must be performed in real time and is very tightly constrained in terms of latency since it is compatible with the data acquisition rate of the cameras. The role of such triggering system is to rapidly decide whether an event is to be recorded for further studies or rejected by the system.

The organization of this paper is given as follows: the context of our work is presented in Section 2. Section 3 describes the algorithms that are envisaged in order to build a new trigger system. Considerations on hardware implementations are then provided in Section 4 and Section 5 describes the results in terms of timing and resource usage.

2. The HESS Project

The High-Energy Stereoscopic System (HESS) is a system of imaging Cherenkov telescopes that strive to investigate cosmic gamma rays in the 100 GeV to 100 TeV energy range [1]. It is located in Namibia at an altitude of 1800 m where the optical quality is excellent. The Phase-I of this project went into operation in Summer 2002 and consists of four Large Cherenkov Telescopes (LCT), each of 107 m^2 mirror area in order to provide a good stereoscopic viewing of the air showers. The telescopes are arranged on a square of 120 m sides, enabling thus to optimize the collection area.

The cameras of the four telescopes serve to capture and record the Cherenkov images of air showers. They have excellent resolution since the pixel size is very small: each camera is equipped of 960 photomultiplier tubes (PMTs) that are assimilated to pixels.

An efficient trigger scheme has also been designed in order to reject background such as the light of the night sky that interferes with measurements. Next sections describe both phases of the project in terms of triggering issues.

2.1. Phase-I. The trigger system of the HESS Phase-I project is devised in order to make use of the stereoscopic approach: simultaneous observation of interesting images must be required in order to store a specific event [2]. This coincidence requirement reduces the rate of background events, that is, events that may be assimilated to night sky noise. It is composed of two separate levels (L1 and the central trigger).

At the first level, a basic threshold is applied on signals collected by the camera. A trigger occurs if the signals in M pixels within a 64-pixel sector of the camera exceed a value of N photoelectrons. This enables to get rid of isolated pixels and thus to eliminate the noise. The pixel signals are sampled using 1 GHZ Analogue Ring Samplers (ARSs) [3] with a ring buffer depth of 128 cells. Following a camera trigger, the ring buffer is stopped and its content is digitized, summed and written in an FPGA buffer. After read-out, the camera is ready for the next event, and further processing may be performed including the transmission of data via optical cable to the PC processor farm located in the control building.

The Central Trigger System (CTS) consists in implementing the coincidence between the four telescopes. It identifies the status of the telescopes and writes this information as well as an absolute time (measured by a GPS) into an FIFO (First-in First-out) memory for each system coincidence. Once the data have been written in this FIFO, the CTS is ready to process new incoming events, about 330 nanoseconds after the coincidence occurred. The FIFO memory has a depth of 16000 events and is read out asynchronously. A schematic illustration of the HESS trigger system is depicted in Figure 1.

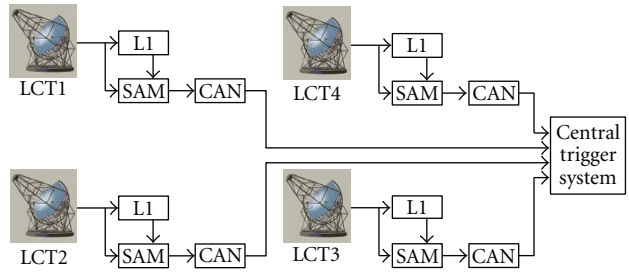

Figure 1: Schematic of the HESS Trigger System.

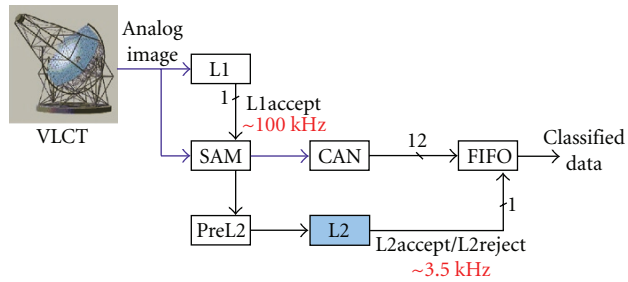

Figure 2: Schematic of the VLCT Trigger System.

2.2. Phase-II. Since its inception in 2002, the HESS project keeps on delivering very significant results. In this very promising context, researchers of the collaboration have decided to improve the initial project by adding a new Very Large Central Telescope (VLCT) in the middle of the four existing ones. This new telescope should permit to increase the sensitivity of the global system as well as improving resolution for high-energy particles. It is composed of 2048 pixels which represent the energy of the incident event.

Considering the new approach, the quantity of data to be collected would drastically increase, and it becomes necessary to build a new trigger system in order to be compatible with the new requirements of the project.

One of the most challenging objectives of the HESS project is to detect particles which energy are below 50 GeV. In this energy range, it is not conceivable to use all telescopes (since the smallest ones cannot trigger), and only the fifth telescope may be used in a monoscopic mode.

The structure of the new triggering system is depicted in Figure 2. Data coming from the VLCT camera consist of 2048 pixels values which are first stored in a Serial Analog Memory (SAM). In parallel, data are also sent to a level 1 trigger (L1) whose structure is described in Section 2.1. The L1 trigger applies a basic analog threshold on the pixel's values and generates a binary signal indicating whether an event has to be kept (L1accept) or rejected (L1reject). In the case where an event is accepted, the entire image is converted into digital patterns. These data are stored in FIFO memories until a L2accept/L2reject signal coming from a second level trigger (L2) is generated.

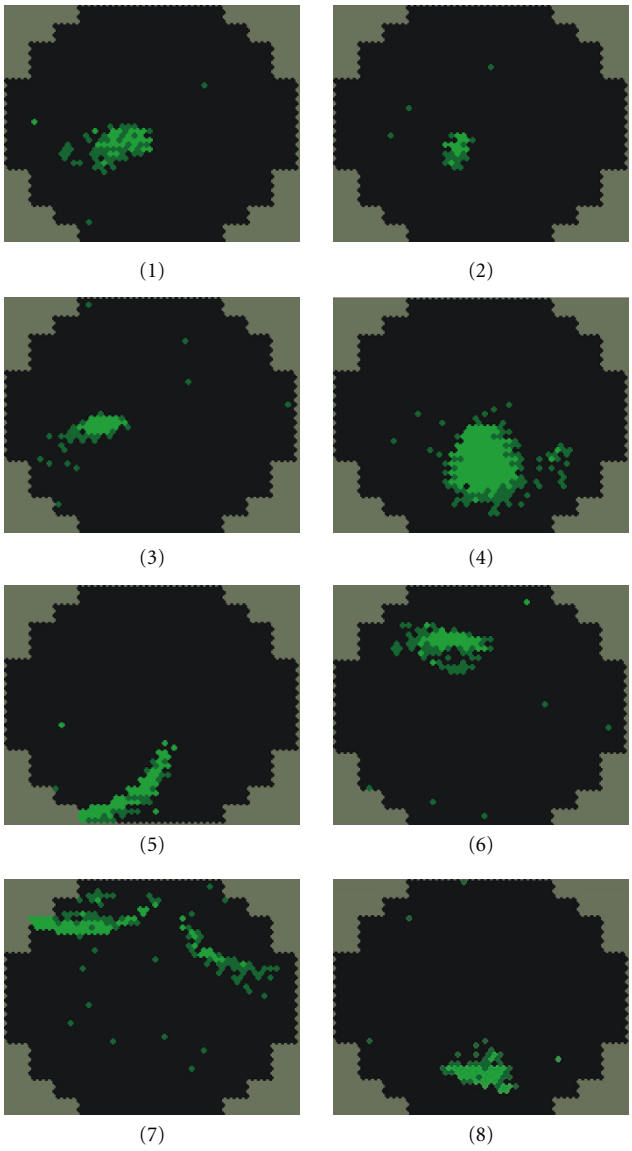

Figure 3: Gamma (1–4), muon (5-6), and proton (7-8) images of different energies.

In parallel, data are sent to the PreL2 stage which thresholds the incoming pixels according to 3 energy levels. Each pixel value is coded into 2 bits corresponding to 3 states of energies. These images are then sent to the L2 Trigger. L1 and L2 trigger decisions are expected at average rates of 100 KHz and 3.5 KHz, respectively. Examples of simulated images are depicted in Figure 3.

3. The HESS2 L2 Triggering System

In order to cope with the new performances of the HESS Phase-II system, an efficient L2 trigger scheme is currently being built. Like all triggers, it aims to provide a decision regarding the interest of a particular event. In this context, two parallel studies have been led in order to identify the best algorithms to implement at that level. The first study relied on the Hillas parameters which are seen as a classical solution in astrophysics pattern recognition. The second study that has been envisaged is to use pattern recognition tools such as neural networks associated with an intelligent preprocessing. Both approaches are described in the next sections.

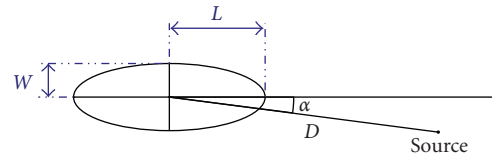

Figure 4: Overview of Hillas moments.

3.1. The First Approach

3.1.1. Hillas Parameters. Hillas parametrization has been introduced in [4]. The retained method consists in isolating image descriptors which are based on image shape parameters such as length (L) and width (W) as well as an angle (α). The α angle represents the angle of the image with the direction of the emitting source location (see Figure 4). This approach globally considers that gamma's signatures are mainly elliptical in shape whereas other particle's signatures are most irregular. This assumption is often the case in practice. Nevertheless, signatures strongly depend on the distance between the impact point of the ray shower and the telescope. This may lead to various types of images for the same event nature and constitutes a real challenge for identification (see Figure 3).

3.1.2. The Classifier. In this first approach, the classifier consists in applying thresholds on the hillas parameters (or a combination of these parameters) computed on the incoming images in order to distinguish gamma signatures between all collected images. One of the best parameters that have been identified as a good discriminator is the Center of Gravity (CoG). This parameter represents the center of gravity of all illuminated pixels within the ellipse.

In this case, the recognition of particles is performed according the following rule:

(a) if $CoG < t$, then the event is recognized as a gamma particle;

(b) if $CoG \geq t$ and $\alpha < 20$ deg, then the event is recognized as a gamma particle;

(c) otherwise, the event is rejected.

t is a parameter which is adjusted according to the data set.

The major drawback of such approach is that the considered thresholds consist of constant values. Thus, a lack of flexibility is to be deplored. For example, it does not allow to take into consideration the various conditions of the experiment that may have a significant impact on the shape of signatures.

3.2. Intelligent Preprocessing. The second studied approach aims to make use of algorithms that already brought significant results in terms of pattern recognition. Neural networks are good candidates because they are a powerful computational model. On the other hand, their inherent parallelism makes them suitable for a hardware implementation. Although used in different fields of physics, these algorithms based on neural networks have successfully been implemented and have already proved their efficiency [5, 6]. Typical applications include particle recognition in tracking systems, event classification problems, off-line reconstruction of event, and online trigger in High-Energy Physics.

From the assumption that neural networks may be useful in such experiments, we have proposed a new Level 2 (L2) trigger system enabling to implement rather complex processing on the incoming images. The major issue with neural networks resides in the learning phase which strives to identify optimal parameters (weights) in order to solve the given problem. This is true when considering unsupervised learning in which representative patterns have to be iteratively presented to the network in a first learning phase until the global error has reached a predefined value.

One of the most important drawbacks of this type of algorithms is that the number of weights strongly depends on the dimensionality of the problem which is often unknown in practice. This implies to find the optimal structure of the network (number of neurons, number of layers) in order to solve the problem.

Moreover, the curse of dimensionality [7] constitutes another challenge when dealing with neural networks. This problem expresses a correlation between the size of the network and the number of examples to furnish. This relation is exponential, that is, if the network's size becomes significant, the number of training examples may become relatively huge. This cannot be considered in practice.

In order to reduce the size of the network, it is possible to simplify its, task that is, reduce the dimensionality of the problem. In this case, a preprocessing step aims at finding correlations on data and at applying basic transformations in order to ease the resolution. In this study, we advise to use an "intelligent" preprocessing based on the extraction of the intrinsic features of the incoming images.

The structure of the proposed L2 trigger is depicted in Figure 5. It is divided into three stages. A rejection step aims to eliminate isolated pixels and small images that cannot be processed by the system. A second step consists in applying a preprocessing on incoming data. Finally, the classifier takes the decision according to the nature of the event to identify. These different steps are described in the following sections.

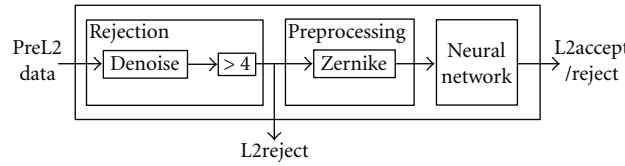

Figure 5: Schematic of the HESS Phase-II Trigger System.

3.2.1. The Rejection Step. The rejection step has two significant roles. First, it aims to remove isolated pixels that are typically due to background. These pixels are eliminated by applying a filtering mask on the entire image in order to keep the only relevant information, that is, clusters of pixels. This consists in testing the neighborhood of each pixel of the image. As the image has an hexagonal mesh grid, a hexagonal neighborhood is used. The direct neighborhood of each pixel of the image is tested. If none of the 6 neighbors are activated, the corresponding central pixel is considered as isolated and deactivated. Second, the rejection step permits to eliminate particles that cannot be distinguished by the classifier. Very small images (<4 pixels) are discarded since they contain poor information that cannot be deciphered.

3.2.2. The Preprocessing Step. The envisaged system is based on a preprocessing step whose role consists in applying basic transformations on incoming images in order to isolate the main characteristics of a given image. The most important role of the preprocessing is to guarantee invariance in orientation (rotation and translation) of the incoming images. Since signatures of particles within the image depend on the impact point of an incident particle, the image may result in a series of pixels located wherever on the telescope. Without using a preprocessing stage based on orientation invariance, the 2048 inputs of the classifier would completely differ from an image to another although the basic shape of the particle would remain the same.

The retained preprocessing is based on the use of Zernike moments. These moments are mainly considered in shape reconstruction [8] and can be easily made invariant to changes in objects orientation. They are defined as a set of orthogonal functions based on complex polynomials originally introduced in [9]. Zernike polynomials can be expressed as

$$V_{pq}(r,\theta) = R_{pq}(r)e^{iq\theta}, \tag{1}$$

where $i = \sqrt{(-1)}$, p is a nonnegative integer, q is apositive integer such as p-q is even and $p \leq q$, r: length of a vector from the origin to a point (x, y) such as $r \leq 1$, that is, $r = \sqrt{x^2 + y^2}/r_{\max}$ where $r_{\max} = \max\sqrt{x^2 + y^2}$, θ: angle between the x axis and the vector extending from the origin to a point(x, y).

$R_{pq}(r)$: Zernike polynomial defined as:

$$R_{pq}(r) = \sum_{k=q,|p-k|\text{even}}^{p} B_{pqk}r^k \tag{2}$$

with $B_{pqk} = (-1)^{(p-k)/2}(((p+k)/2)!/((p-k)/2)!((k+|q|)/2)!((k-|q|)/2)!)$.

Zernike moments Z_{pq} are expressed according to

$$Z_{pq} = \frac{p+1}{\pi}\sum_x\sum_y I(x,y)\,V_{pq}^*(r,\theta), \tag{3}$$

where $I(x,y)$ refers to the pixels' value of coordinates(x,y). The rotation invariance property of Zernike moments is due to the intrinsic nature of such moments. In order to guarantee translation invariance as well, it is necessary to align the center of the object to the center of the unit circle. This may be performed by changing the coordinates x and y of each processing point by the coordinates $x - x_0$ and $y - y_0$ where x_0 and y_0 refer to the center of the signature and may be obtained by

$$x_0 = \frac{\sum_X\sum_Y xI(x,y)}{\sum_X\sum_Y I(x,y)}, \qquad y_0 = \frac{\sum_X\sum_Y yI(x,y)}{\sum_X\sum_Y I(x,y)}. \tag{4}$$

In this case, r is expressed as follows:

$$r = \frac{\sqrt{(x-x_0)^2 + (y-y_0)^2}}{r_{\max}}, \tag{5}$$

where $r_{\max} = \max\sqrt{(x-x_0)^2 + (y-y_0)^2}$.

In the context of our application, it has been found that considering the first 8-order Zernike moments was sufficient to obtain the best performances. This implies to compute 25 polynomials for each of the pixels within an image. Then, it is necessary to accumulate these values in order to obtain 25 real values corresponding to all the Zernike moments of the image. The module of these moments is then computed and a normalization step is fulfilled in order to scale the obtained values within the -1 to 1 range. These values are then provided to the neural network.

3.2.3. The Neural Classifier. The envisaged classifier is a feed-forward neural network named Multilayer Perceptron (MLP) with one hidden layer and three outputs. The general structure of such a network is depicted in Figure 6:

According to the nature of the network, the value of the output may be computed according to (6).

$$y = g\left(\sum_{j=0}^{M} w_{kj}^{(2)}\, g\left(\sum_{i=0}^{d} w_{ji}^{(1)} x_i\right)\right). \tag{6}$$

In (6), y represents the output; $w_{kj}^{(2)}$ and $w_{ji}^{(1)}$, respectively, represent the weights connecting the output layer and the hidden layer and the weights connecting the hidden layer and the input nodes. M is the number of neurons in the hidden layer and d is the number of inputs. x_i denotes the value of an input node and g is an activation function. In our case, the nonlinear function g has been used:

$$g(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{7}$$

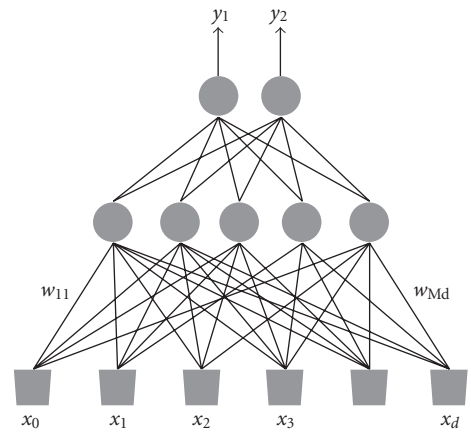

Figure 6: Structure of a 2-layer perceptron.

In the considered application, the output layer is composed of a three neurons (which value ranges between -1 and 1) corresponding to the type of particle to identify. Each output refers to a gamma, proton, and muon particle, respectively. If the value of an output neuron is positive, it may be assumed that the corresponding particle has been identified by the network. In the case that more than one output neuron is activated, the maximum value is taken into account.

The learning phase has been performed off-line on a set of 4500 patterns computed on simulated images as the HESS2 telescope is not yet installed. The simulated images are generated thanks to series of Monte Carlo simulations. These patterns covered all ranges of energies and types of particles. 1500 patterns were considered for each class of particles. A previous study had determined the reliability of the patterns in order to consider the most representative patterns that may be collected by the telescope.

A classical backpropagation algorithm has been programmed off-line in order to get the optimal value of weights. The training have been performed simultaneously on two sets of patterns (learning and testing set). Once the error on the testing phase was minimum, the training was stopped ensuring that the weights had an optimal value.

The size of the input layer was determined according to the type of preprocessing that was envisaged. In the case of a Zernike preprocessing, this number has been set to 25 since it corresponds to the number of outputs furnished by the preprocessing step.

The number of hidden nodes (in the hidden layer) has been evaluated regarding the results obtained on a specific validation set of patterns. This precaution has been handled in order to ensure that the neural network was able to generalize on new data (i.e., it has not learnt explicitly).

3.3. Simulated Performances. The best performances that have been obtained are summarized in Table 1. It corresponds to a trigger with a preprocessing based on the first 25 Zernike moments. Other results concerning different preprocessings have also been described in [10].

Table 1: Performances according to both approaches.

	Gamma	Muon	Proton
Hillas approach	60%	56%	37%
Neural approach	95%	58%	41%

According to Table 1, it may be seen that the neural solution provides significant improvement compared to classical methods in terms of classification. This improvement resides in the fact that a largest dimensionality of the problem has been taken into account. Whereas Hillas processing takes only five parameters into consideration, the number of inputs in the case of a neural preprocessing is set to 25. Moreover, as the Hillas approach only consists in applying strong "cuts" on predefined parameters, the neural approach is more flexible and guarantees nonlinear decision boundaries. It may be assumed that the considered neural network is capable of extracting the relevant information and discriminate between all images, efficiently. The major drawback of the neural approach is its relative complexity in terms of computation and hardware implementation. Although Hillas algorithms may be implemented in software, it is impossible to implement both the neural network and the preprocessing step in the same manner. In this context, dedicated circuits have to be designed in order to be compliant with the strong timing constraints imposed by the entire system. In our case, an L2-decision has to be taken at a rate of 3.5 KHz which corresponds to a timing constraint of 285 microseconds.

4. Hardware Implementation

The complete L2 trigger system is currently being built, making intensive use of the reconfigurable technology. Components such as FPGAs constitute an attractive alternative to classical circuits such as ASICs (Application Specific Integrated Circuits). This type of reconfigurable circuits tends to be more and more efficient in terms of speed and logic resources and is more and more envisaged in deeply constrained applications.

4.1. Hardware Implementation of Zernike Moments. Although very efficient, Zernike moments are known for their computation complexity. Many solutions have been proposed for the fast implementation of the Zernike moments. Some algorithms are based on recursivity [11], reuse of previous parts of the computation [12] or moment generators [13].

Since using a moment generator allows a reduction of the number of operations, we have decided to follow this approach, that is, to compute Zernike moments from accumulation moments.

4.1.1. Zernike Moments via Accumulation Moments. The mechanism of a moment generator [14] can be summarized by the expression of the geometric moments with respect

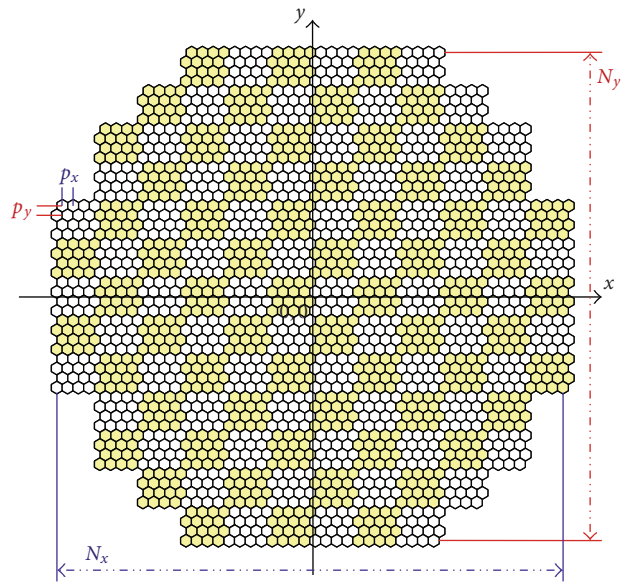

Figure 7: Image topology in the L2-trigger of the HESS Phase-II project.

to the point of coordinates (N_x, N_y) from the accumulation moments:

$$m_{p,q}^{N_x,N_y} = \sum_{x=0}^{N_x}\sum_{y=0}^{N_y}(N_x - x)^p(N_y - y)^q I(x,y) = \sum_{e=0}^{p}\sum_{f=0}^{q} S(p,e)S(q,f)\psi_{e,f} \tag{8}$$

with $\psi_{e,f}$ being the accumulation moments of order (e, f), and $I(x, y)$ being the pixels' values in the image and

$$S = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -1 & 1 & 0 & 0 & 0 & 0 & \\ 1 & -3 & 2 & 0 & 0 & 0 & \\ -1 & 7 & -12 & 6 & 0 & 0 & \\ 1 & -15 & 50 & -60 & 24 & 0 & \\ -1 & 31 & -180 & 390 & -360 & 120 & \\ \vdots & & & & & & \ddots \end{pmatrix}. \tag{9}$$

According to (8), it is important to note that geometric moments may be expressed as a function of accumulation moments. In the context of our application, (10) to (23) demonstrate how to calculate the Zernike moments from the geometric moments and thus from accumulation moments.

Note that, in the particular case of HESS Phase-II, one of the issues is the image topology which consists of a hexagonal grid with empty corners (see Figure 7). Since Zernike moments are continuous, they are particularly suitable for this type of images. The following equations

aim to express the Zernike moments from the accumulation moments in the particular context of HESS Phase-II.

We have seen that Zernike moments may be expressed as follows:

$$Z_{pq} = \frac{p+1}{\pi} \sum_x \sum_y I(x,y) \sum_{k=q,p-k \text{ even}}^{p} B_{pqk} r^k e^{iq\theta}. \tag{10}$$

The expressions of r and $e^{iq\theta}$ can be rewritten as

$$r = \frac{\sqrt{(x-x_0)^2 + (y-y_0)^2}}{r_{\max}}, \tag{11}$$

where (x_0, y_0) are the coordinates of the image center computed as explained in (4):

$$e^{iq\theta} = \frac{((x-x_0) + i(y-y_0))^q}{((x-x_0)^2 + (y-y_0)^2)^{q/2}}. \tag{12}$$

In order to simplify the equations, we note $X = x - x_0$ and $Y = y - y_0$. In this case,

$$\begin{aligned} Z_{pq} &= \frac{p+1}{\pi} \sum_x \sum_y I(x,y) \sum_{k=q,p-k \text{ even}}^{p} \frac{B_{pqk}}{r_{\max}^k} \\ &\quad \times (X^2 + Y^2)^{(k-q)/2} (X + iY)^q. \end{aligned} \tag{13}$$

According to the binomial theorem, the development of a given polynom can be expressed as follows:

$$(a+b)^n = \sum_{m=0}^{n} C_n^m a^{n-m} b^m. \tag{14}$$

It is then possible to modify the following expressions:

$$\begin{aligned} (X^2 + Y^2)^{(k-q)/2} &= \sum_{\xi=0}^{(k-q)/2} C_{(k-q)/2}^{\xi} X^{k-q-2\xi} Y^{2\xi}, \\ (X + iY)^q &= \sum_{\zeta=0}^{q} C_q^{\zeta} i^{\zeta} X^{q-\zeta} Y^{\zeta}. \end{aligned} \tag{15}$$

Thus, (13) can be reformulated as follows:

$$\begin{aligned} Z_{pq} &= \frac{p+1}{\pi} \sum_{k=q,p-k \text{ even}}^{p} \frac{B_{pqk}}{r_{\max}^k} \sum_{\zeta=0}^{q} \sum_{\xi=0}^{(k-q)/2} i^{\zeta} C_q^{\zeta} C_{(k-q)/2}^{\xi} \\ &\quad \times \sum_x \sum_y X^{k-\zeta-2\xi} Y^{2\xi+\zeta} I(x,y) \\ &= \frac{p+1}{\pi} \sum_{k=q,p-k \text{ even}}^{p} (-1)^k \frac{B_{pqk}}{r_{\max}^k} \sum_{\zeta=0}^{q} \sum_{\xi=0}^{(k-q)/2} i^{\zeta} C_q^{\zeta} C_{(k-q)/2}^{\xi} \\ &\quad \times \sum_x \sum_y (x_0 - x)^{k-\zeta-2\xi} (y_0 - y)^{2\xi+\zeta} I(x,y). \end{aligned} \tag{16}$$

The next step consists in considering the last point (N_x, N_y) in the equation of Zernike moments with respect to the center of the image:

$$\begin{aligned} Z_{pq} &= \frac{p+1}{\pi} \sum_{k=q,p-k \text{ even}}^{p} (-1)^k \frac{B_{pqk}}{r_{\max}^k} \sum_{\zeta=0}^{q} \sum_{\xi=0}^{(k-q)/2} i^{\zeta} C_q^{\zeta} C_{(k-q)/2}^{\xi} \\ &\quad \times \sum_x \sum_y (x_0 - N_x + N_x - x)^{k-\zeta-2\xi} \\ &\quad \times (y_0 + N_y - N_y - y)^{2\xi+\zeta} I(x,y) \\ &= \frac{p+1}{\pi} \sum_{k=q,p-k \text{ even}}^{p} (-1)^k \frac{B_{pqk}}{r_{\max}^k} \sum_{\zeta=0}^{q} \sum_{\xi=0}^{(k-q)/2} i^{\zeta} C_q^{\zeta} C_{(k-q)/2}^{\xi} \\ &\quad \times \sum_{a=0}^{k-\zeta-2\xi} C_{k-\zeta-2\xi}^{a} X_c^{k-\zeta-2\xi-a} \sum_{b=0}^{2\xi+\zeta} C_{2\xi+\zeta}^{b} Y_c^{2\xi+\zeta-b} \\ &\quad \times \sum_x \sum_y (N_x - x)^a \left(-N_y - y\right)^b I(x,y), \end{aligned} \tag{17}$$

where $X_c = x_0 - N_x$ and $Y_c = y_0 + N_y$.

Since the coordinates of the pixels in the image are expressed as real numbers, we need to express these coordinates with integers in order to formulate the Zernike moments in function of the geometric moments. As we can see in Figure 7, the even rows have to be distinguished from the odd rows. Therefore, the x coordinate is expressed in two different ways according to the type of row (even or odd row).

x and y may be expressed as

$$\begin{aligned} x &= \begin{cases} (x_d - 0.5)\, p_x + \text{offset}_x, & \text{if } y_d \%2 = 1, \\ (x_d - 1)\, p_x + \text{offset}_x, & \text{if } y_d \%2 = 0, \end{cases} \\ y &= (1 - y_d)\, p_y + \text{offset}_y, \end{aligned} \tag{18}$$

where x_d and y_d are positive integers such as $x_d = 1 \cdots X_d$ and $y_d = 1 \cdots Y_d$ with $X_d = 48$ corresponding to the number of columns, and $Y_d = 52$ corresponding to the number of rows. p_x (resp., p_y) is the distance between two adjacent columns (resp., rows) and offset_x and offset_y correspond to the new position of the origin of the image in the upper left corner.

In the following equations, since the first part of the expressions does not change, the second part is just developed:

$$\begin{aligned}
&\sum_x \sum_y (N_x - x)^a (-N_y - y)^b I(x,y) \\
&= \sum_{x_d=1}^{X_d} \sum_{y_d=1, y_d\%2=1}^{Y_d} \left(N_x - ((x_d - 0.5)p_x + \text{offset}_x)\right)^a \\
&\quad \times \left(-N_y - \left((1 - y_d)p_y + \text{offset}_y\right)\right)^b f(x_d, y_d) \\
&\quad + \sum_{x_d=1}^{X_d} \sum_{y_d=1, y_d\%2=0}^{Y_d} \left(N_x - ((x_d - 1)p_x + \text{offset}_x)\right)^a \\
&\quad \times \left(-N_y - \left((1 - y_d)p_y + \text{offset}_y\right)\right)^b f(x_d, y_d).
\end{aligned} \tag{19}$$

Notice that $N_x = p_x X_d$ and $N_y = p_y Y_d$:

$$\begin{aligned}
&\sum_x \sum_y (N_x - x)^a (-N_y - y)^b I(x,y) \\
&= \sum_{x_d=1}^{X_d} \sum_{y_d=1, y_d\%2=1}^{Y_d} \left((X_d - x_d)p_x + 0.5p_x - \text{offset}_x\right)^a \\
&\quad \times \left((-Y_d + y_d)p_y - p_y - \text{offset}_y\right)^b I(x_d, y_d) \\
&\quad + \sum_{x_d=1}^{X_d} \sum_{y_d=1, y_d\%2=0}^{Y_d} \left((X_d - x_d)p_x + p_x - \text{offset}_x\right)^a \\
&\quad \times \left((-Y_d + y_d)p_y - p_y - \text{offset}_y\right)^b I(x_d, y_d).
\end{aligned} \tag{20}$$

In this case, we propose to figure out the even and odd parts of the image, let $y_d = 2y_{ed}$ if $y_d\%2 = 0$ and $y_d = 2y_{od} - 1$ if $y_d\%2 = 1$. In this case, the sums on y_d will be bounded by $Y_d/2$ and

$$\begin{aligned}
&\sum_x \sum_y (N_x - x)^a (-N_y - y)^b I(x,y) \\
&= \sum_{x_d=1}^{X_d} \sum_{y_{od}=1}^{Y_d/2} \left((X_d - x_d)p_x + 0.5p_x - \text{offset}_x\right)^a \\
&\quad \times \left(-2p_y\left(\frac{Y_d}{2} - y_{od}\right) - 2p_y - \text{offset}_y\right)^b I(x_d, y_{od}) \\
&\quad + \sum_{x_d=1}^{X_d} \sum_{y_{ed}=1}^{Y_d/2} \left((X_d - x_d)p_x + p_x - \text{offset}_x\right)^a \\
&\quad \times \left(-2p_y\left(\frac{Y_d}{2} - y_{ed}\right) - p_y - \text{offset}_y\right)^b I(x_d, y_{ed})
\end{aligned} \tag{21}$$

$$\begin{aligned}
&= \sum_{x_d=1}^{X_d} \sum_{y_{od}=1}^{Y_d/2} I(x_d, y_{od}) \\
&\quad \times \sum_{c=0}^{a} C_a^c (0.5p_x - \text{offset}_x)^{a-c} p_x^c (X_d - x_d)^c \\
&\quad \times \sum_{d=0}^{b} C_b^d \left(-2p_y - \text{offset}_y\right)^{b-d} \left(-2p_y\right)^d \left(\frac{Y_d}{2} - y_{od}\right)^d \\
&\quad + \sum_{x_d=1}^{X_d} \sum_{y_{ed}=1}^{Y_d/2} I(x_d, y_{ed}) \\
&\quad \times \sum_{c=0}^{a} C_a^c (p_x - \text{offset}_x)^{a-c} p_x^c (X_d - x_d)^c \\
&\quad \times \sum_{d=0}^{b} C_b^d \left(-p_y - \text{offset}_y\right)^{b-d} \left(-2p_y\right)^d \left(\frac{Y_d}{2} - y_{ed}\right)^d
\end{aligned} \tag{22}$$

$$\begin{aligned}
&= \sum_{c=0}^{a} \sum_{d=0}^{b} C_a^c M_{\text{odd}}^{a-c} N^c C_b^d O_{\text{odd}}^{b-d} P^d \\
&\quad \times \sum_{x_d=1}^{X_d} \sum_{y_{od}=1}^{Y_d/2} (X_d - x_d)^c \left(\frac{Y_d}{2} - y_{od}\right)^d I(x_d, y_{od}) \\
&\quad + \sum_{c=0}^{a} \sum_{d=0}^{b} C_a^c M_{\text{even}}^{a-c} N^c C_b^d O_{\text{even}}^{b-d} P^d \\
&\quad \times \sum_{x_d=1}^{X_d} \sum_{y_{ed}=1}^{Y_d/2} (X_d - x_d)^c \left(\frac{Y_d}{2} - y_{ed}\right)^d I(x_d, y_{ed}),
\end{aligned} \tag{23}$$

where $M_{\text{odd}} = 0.5p_x - \text{offset}_x$, $M_{\text{even}} = p_x - \text{offset}_x$, $N = p_x$, $O_{\text{odd}} = -2p_y - \text{offset}_y$, $O_{\text{even}} = -p_y - \text{offset}_y$, and $P = -2p_y$.

Equation (23) shows that the Zernike moments can be computed from the geometric moments. If we consider two accumulation grids, the first computes the accumulation moments on the odd lines of the image and the second on the even lines. Since the computation is divided into two different parts, the image should be arranged in two components: the odd component of the image and the even one. Therefore, according to (8), the analogy gives an expression of the Zernike moments which is function of ψ_{odd} (accumulation moments computed from the odd component of the image) and ψ_{even} (accumulation moments computed from the even component of the image) by setting $N_x = X_d$ and $N_y = Y_d/2$:

$$\begin{aligned}
&\sum_{x_d=1}^{X_d} \sum_{y_{od}=1}^{Y_d/2} (X_d - x_d)^c \left(\frac{Y_d}{2} - y_{od}\right)^d I(x_d, y_{od}) \\
&= \sum_{e=0}^{c} \sum_{f=0}^{d} S(c,e) S(d,f) \psi_{\text{odd},e,f}
\end{aligned}$$

$$\sum_{x_d=1}^{X_d}\sum_{y_{ed}=1}^{Y_d/2}(X_d-x_d)^c\left(\frac{Y_d}{2}-y_{ed}\right)^d I(x_d,y_{ed}) = \sum_{e=0}^{c}\sum_{f=0}^{d}S(c,e)S(d,f)\psi_{\text{even},e,f.} \tag{24}$$

By reinjecting (24) in the (23), Zernike moments are reformulated as follows:

$$\begin{aligned}Z_{pq} = {} & \frac{p+1}{\pi}\sum_{k=q,p-k\ \text{even}}^{p}\sum_{\zeta=0}^{q}\sum_{\xi=0}^{(k-q)/2} i^{\zeta}(-1)^k C_q^{\zeta}C_{(k-q)/2}^{\xi}\frac{B_{pqk}}{r_{\max}^k}\\ & \times\sum_{a=0}^{k-\zeta-2\xi}C_{k-\zeta-2\xi}^{a}X_c^{k-\zeta-2\xi-a}\sum_{b=0}^{2\xi+\zeta}C_{2\xi+\zeta}^{b}Y_c^{2\xi+\zeta-b}\\ & \times\left[\sum_{c=0}^{a}\sum_{d=0}^{b}C_a^c M_{\text{odd}}^{a-c}N^c C_b^d O_{\text{odd}}^{b-d}P^d \times\sum_{e=0}^{c}\sum_{f=0}^{d}S(c,e)S(d,f)\psi_{\text{odd},e,f}\right.\\ & \left.+\sum_{c=0}^{a}\sum_{d=0}^{b}C_a^c M_{\text{even}}^{a-c}N^c C_b^d O_{\text{even}}^{b-d}P^d \times\sum_{f=0}^{d}S(c,e)S(d,f)\psi_{\text{even},e,f}\right].\end{aligned} \tag{25}$$

In an analogous way, the coordinates of the center of the image (x_0, y_0) can be computed from the accumulation moments:

$$\begin{aligned}x_0 &= \frac{m_{01}}{m_{00}},\\ y_0 &= \frac{m_{10}}{m_{00}},\end{aligned} \tag{26}$$

where

$$\begin{aligned}m_{00} &= \psi_{\text{odd},0,0}+\psi_{\text{even},0,0}\\ m_{01} &= (-1)\times\sum_{b=0}^{1}C_1^b N_y^{1-b}\\ &\quad\times\left[\sum_{d=0}^{b}C_b^d O_{\text{odd}}^{b-d}P^d\sum_{f=0}^{d}S(d,f)\psi_{\text{odd},0,f}\right.\\ &\quad\left.+\sum_{d=0}^{b}C_b^d O_{\text{even}}^{b-d}P^d\sum_{f=0}^{d}S(d,f)\psi_{\text{even},0,f}\right]\\ &= (-1)\times\left[N_y(\psi_{\text{odd},0,0}+\psi_{\text{even},0,0})\right.\\ &\quad+(O_{\text{odd}}\psi_{\text{odd},0,0}+O_{\text{even}}\psi_{\text{even},0,0}\\ &\quad+P\times S(1,0)(\psi_{\text{odd},0,0}+\psi_{\text{even},0,0})\\ &\quad\left.+P\times S(1,1)(\psi_{\text{odd},0,1}+\psi_{\text{even},0,1}))\right],\\ m_{10} &= (-1)\times\sum_{a=0}^{1}C_1^a(-N_x)^{1-a}\\ &\quad\times\left[\sum_{c=0}^{a}C_a^c O_{\text{odd}}^{a-c}P^c\sum_{e=0}^{c}S(c,e)\psi_{\text{odd},e,0}\right.\\ &\quad\left.+\sum_{c=0}^{a}C_a^c O_{\text{even}}^{a-c}P^c\sum_{e=0}^{c}S(c,e)\psi_{\text{even},e,0}\right]\\ &= (-1)\times\left[(-N_x)(\psi_{\text{odd},0,0}+\psi_{\text{even},0,0})\right.\\ &\quad+(O_{\text{odd}}\psi_{\text{odd},0,0}+O_{\text{even}}\psi_{\text{even},0,0}\\ &\quad+P\times S(1,0)(\psi_{\text{odd},0,0}+\psi_{\text{even},0,0})\\ &\quad\left.+P\times S(1,1)(\psi_{\text{odd},1,0}+\psi_{\text{even},1,0}))\right].\end{aligned} \tag{27}$$

It comes

$$\begin{aligned}x_0 &= -N_y - P\times S(1,0) - \frac{O_{\text{odd}}\psi_{\text{odd},0,0}+O_{\text{even}}\psi_{\text{even},0,0}}{\psi_{\text{odd},0,0}+\psi_{\text{even},0,0}}\\ &\quad-\frac{P\times S(1,1)(\psi_{\text{odd},0,1}+\psi_{\text{even},0,1})}{\psi_{\text{odd},0,0}+\psi_{\text{even},0,0}},\\ y_0 &= N_x - P\times S(1,0) - \frac{O_{\text{odd}}\psi_{\text{odd},0,0}+O_{\text{even}}\psi_{\text{even},0,0}}{\psi_{\text{odd},0,0}+\psi_{\text{even},0,0}}\\ &\quad-\frac{P\times S(1,1)(\psi_{\text{odd},1,0}+\psi_{\text{even},1,0})}{\psi_{\text{odd},0,0}+\psi_{\text{even},0,0}}.\end{aligned} \tag{28}$$

We have developed here an algorithm enabling the computation of Zernike moments based on the moment generator using the accumulation moments. This algorithm has the advantage to be used on images which have particular topologies since their mesh grid is regular or semiregular by the use of a second accumulation grid. The second advantage of this algorithm is its simplicity to be implemented on FPGA, for instance. The base of this algorithm relies on the accumulation moments and is easily computed thanks to a simple accumulation grid.

4.1.2. Architecture Description. To make the exploitation of (25) easier, we need to reorder the terms to get an expression of the Zernike moments such as

$$Z_{pq} = \sum_e\sum_f \Gamma_{\text{odd},e,f}^{p,q}\psi_{\text{odd},e,f} + \Gamma_{\text{even},e,f}^{p,q}\psi_{\text{even},e,f}, \tag{29}$$

where

$$\Gamma^{p,q}_{\text{odd},e,f} = \frac{(p+1)}{\pi}\sum_{k=q}^{p} t_{k,e}t_{k,f}\sum_{\zeta=0}^{q}\sum_{\xi=0}^{(k-q)/2}\alpha^{k}_{e,\zeta,\xi}\beta^{k}_{f,\zeta,j}i^{\zeta} \times (-1)^{k}\frac{B_{pqk}}{r^{k}_{\max}}C^{\zeta}_{q}C^{\xi}_{(k-q)/2} \times \sum_{a=0}^{k-\zeta-2j}\sum_{b=0}^{2\xi+\zeta} t_{a,e}t_{b,f}C^{a}_{k-\zeta-2\xi}C^{b}_{2\xi+\zeta} \times X_c^{k-\zeta-2\xi-a}Y_c^{2\xi+\zeta-b} \times \sum_{c=0}^{a}\sum_{d=0}^{b} C^{c}_{a}C^{d}_{b}M^{a-c}_{\text{odd}}N^{c}O^{b-d}_{\text{odd}}P^{d}S(c,e)S(d,f),$$

$$\Gamma^{p,q}_{\text{even},e,f} = \frac{(p+1)}{\pi}\sum_{k=q}^{p} t_{k,e}t_{k,f}\sum_{\zeta=0}^{q}\sum_{\xi=0}^{(k-q)/2}\alpha^{k}_{e,\zeta,\xi}\beta^{k}_{f,\zeta,\xi} \times i^{\zeta}(-1)^{k}\frac{B_{pqk}}{r^{k}_{\max}}C^{\zeta}_{q}C^{\xi}_{(k-q)/2} \times \sum_{a=0}^{k-\zeta-2\xi}\sum_{b=0}^{2\xi+\zeta} t_{a,e}t_{b,f}C^{a}_{k-\zeta-2\xi}C^{b}_{2\xi+\zeta} \times X_c^{k-\zeta-2\xi-a}Y_c^{2\xi+\zeta-b} \times \sum_{c=0}^{a}\sum_{d=0}^{b} C^{c}_{a}C^{d}_{b}M^{a-c}_{\text{even}}N^{c}O^{b-d}_{\text{even}}P^{d}S(c,e)S(d,f) \quad (30)$$

with

$$t_{g,h} = \begin{cases} 1, & \text{if } h \le g, \\ 0, & \text{else,} \end{cases}$$

$$\alpha^{k}_{e,\zeta,\xi} = \begin{cases} 1, & \text{if } e \le k-\zeta-2\xi, \\ 0, & \text{else,} \end{cases} \quad (31)$$

$$\beta^{k}_{f,\zeta,\xi} = \begin{cases} 1, & \text{if } f \le 2\xi+\zeta, \\ 0, & \text{else.} \end{cases}$$

The general scheme of the architecture of Zernike moments (see Figure 8) can be described as follows. (i) The image is first divided into two parts: the odd component which only contains the odd rows of the images (resp., even). (ii) The accumulation moments are computed in parallel according to two accumulation grids. (iii) On the one hand the accumulation moments of order $(0,0)$, $(0,1)$ and, $(1,0)$ reach the block which computes $X_c^{k-\zeta-2\xi-a}$, $Y_c^{2\xi+\zeta-b}$, and $r^{-k}_{\max}$. On the other hand the accumulation moments are delivered to the Zernike computation block, waiting for the completion of the computations. (iv) As soon as $X_c^{k-\zeta-2\xi-a}$, $Y_c^{2\xi+\zeta-b}$, and $r^{-k}_{\max}$ are computed, the coefficients $\Gamma^{p,q}_{\text{odd},e,f}$ and $\Gamma^{p,q}_{\text{even},e,f}$ can be computed. (v) The coefficients are transmitted to the final computation block in order to evaluate the Zernike moments according to (29). Their module is then computed.

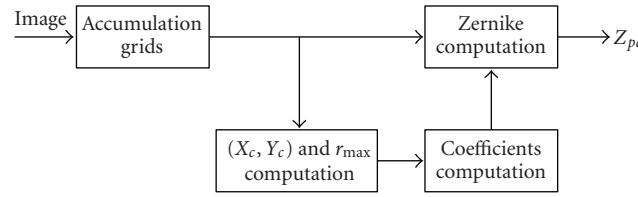

Figure 8: Zernike architecture general scheme.

The scheme of the accumulation grid of width 4 is given in Figure 9, and we can notice that it consists of a simple series of accumulators. They are arranged in a way that the accumulation is first computed on each row via an accumulation row (row of y_m) and then the accumulation is performed on the columns (set of y_{mn}). As soon as a row ends in a given accumulator y_m, the result of this accumulator is furnished to its corresponding first column accumulator, and y_{m0} and y_m are cleared. At the same time all the corresponding column accumulators transmit their accumulation to the next one.

The registers used between the column accumulators are synchronized at the end of each row; so their clock enable depends on the image topology. In our case, corners have been filled with zeros before dividing the image. Therefore, the size of each image's component is $X_d \times Y_d/2$. In this case, the accumulation moment $\psi_{e,f}$ is computed on $X_d \times (Y_d/2 + f) + e$ clock cycles from the moment when the first pixel arrives into the accumulation grid.

One major point of the Zernike moments implementation is the computation of the coefficients. The main issue of this computation relies in the trade-off between the number of coefficients stored in the chip and the number of operations that are useful to compute these coefficients. Table 2 shows the number of operations that are necessary for the computation of the coefficients for Zernike moments until order 8. Configuration 1 corresponds to the case where B_{pqk} (55 values), C^{k}_{p} (45 values), the matrix S (45 values) and the M^{p}_{odd}, the M^{p}_{even}, the N^{p}, the O^{p}_{odd}, the O^{p}_{even}, and the P^{p} ($9 \times 6 = 54$ values), that is, 199 values stored. The second configuration corresponds to a storage of the results dealing with the operations: $(-1)^{k}(B_{pqk}/r^{k}_{\max})C^{\zeta}_{q}C_{(k-q)/2}$, $C^{a}_{k-\zeta-2\xi}C^{b}_{2\xi+\zeta}$, $C^{c}_{a}M^{a-c}_{\text{odd}}N^{c}$, $C^{c}_{a}M^{a-c}_{\text{even}}N^{c}$, $C^{d}_{b}O^{b-d}_{\text{odd}}P^{d}$, $C^{d}_{b}O^{b-d}_{\text{even}}P^{d}$, and $S(c,e)S(d,f)$. If there is no optimisation of the storage of these values, the occupied memory will be huge (4564 values), but by using the redundancy in each group and the centralized storage of the 1 value, the number of stored values may be reduced to 1203. Note that even if the number of values to store has hardly increased, the number of multiplications is divided by two compare to the first configuration.

Figure 10 shows the envisaged computation of the zernike coefficients taking into account the second configuration. The control block deals with the bounds of the sum. T1, T2, T3, T4, T5, T6, and T7 look-up tables correspond, respectively, to $C^{c}_{a}M^{a-c}_{\text{odd}}N^{c}$, $C^{d}_{b}O^{b-d}_{\text{odd}}P^{d}$,

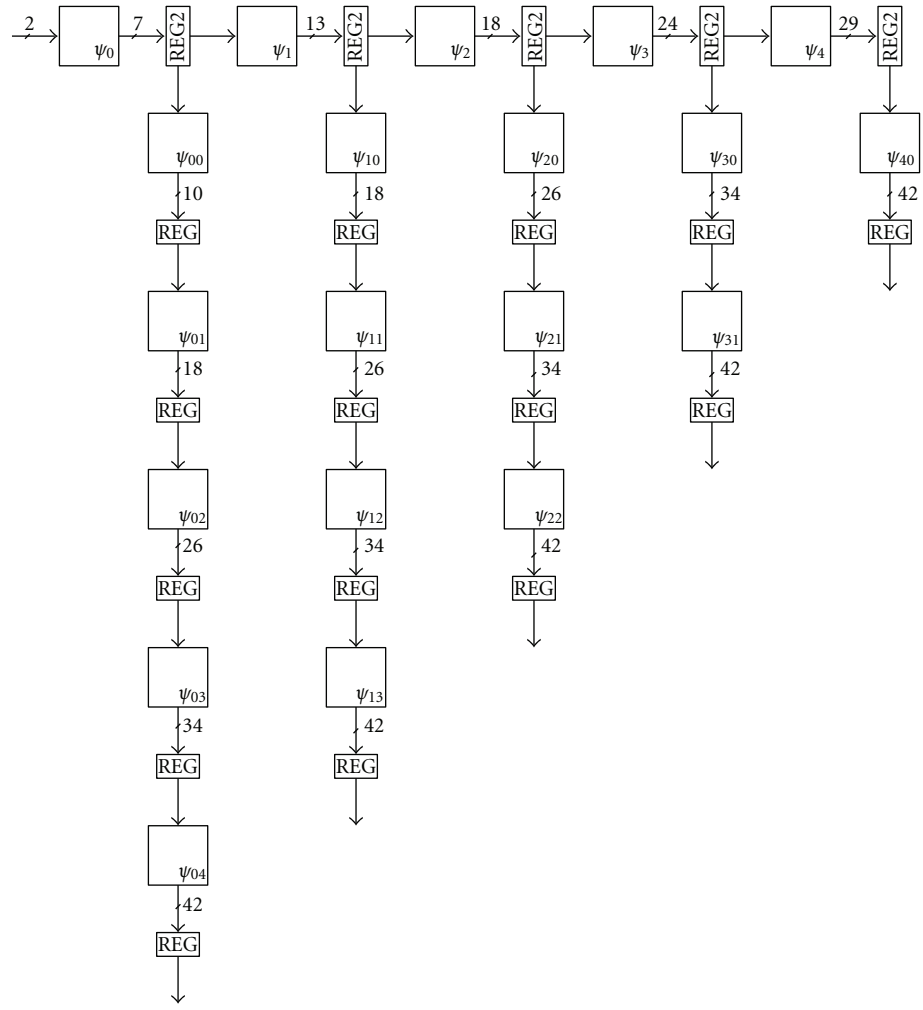

Figure 9: Example of accumulation grid of width 4.

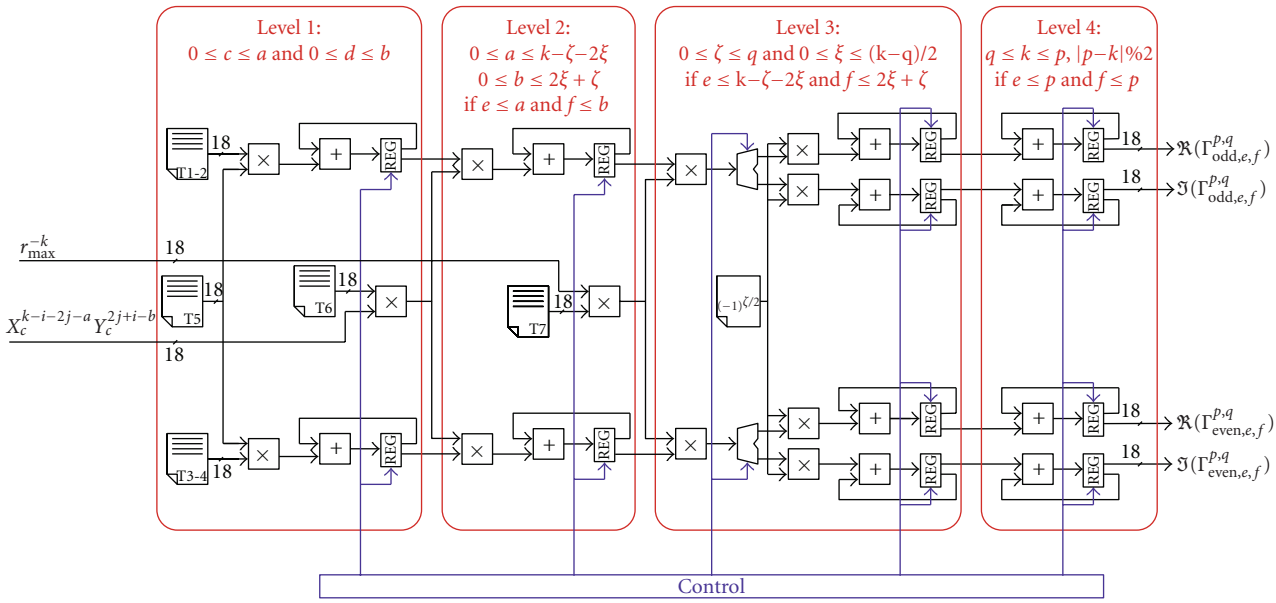

Figure 10: Computation of Zernike coefficients.

Table 2: Number of operations executed to compute the $\Gamma_{e,f}^{p,q}$ coefficients.

p	Nb. accumulations	Nb. Multiplications	
		Configure 1	configure 2
0	2	25	14
1	16	183	102
2	101	1225	672
3	349	5543	3000
4	1311	22987	12266
5	4267	77637	41010
6	13642	241767	126592
7	38860	660481	343560
8	104663	1692910	875720

$C_a^c M_{\text{even}}^{a-c} N^c$, $C_b^d O_{\text{even}}^{b-d} P^d$, $S(c,e)S(d,f)$, $C_{k-\zeta-2\xi}^a C_{2\xi+\zeta}^b$, and $(-1)^k (B_{pqk}/r_{\max}^k) C_q^\zeta C_{(k-q)/2}$. T1-2 (resp., T3-4) means that T1 and T2 (resp., T3 and T4) are first read and then the products between the read values are computed.

A Zernike computation block aims to compute the module of Zernike moments from the accumulation moments that are provided by the grids and from the module that furnishes the coefficients (see Figure 8). This block consists in summing the different coefficients and in computing the module of each moment. In order to reduce the amount of logical resources to provide, the computation of the square root is simplified according to the following approximation:

$$\sqrt{x^2+y^2} \approx \max\left(|x|, |y|, \frac{3}{4}(|x|+|y|)\right). \quad (32)$$

This approximation is often utilized in image processing and does not impact significantly the final results.

4.1.3. FPGA Implementation of Zernike Moment's Computation. In order to compute the Zernike moment from the accumulation, we proposed an original architecture which is presented in Figure 10. This architecture is very regular and simplifies the implementation on an FPGA target. Furthermore, we can notice that the hardware required is simple to design for both the moments' accumulation and moments' computation. In fact, the computations are based on a multiplier and an adder. These constitute the MAC (for Multiply-ACcumulate) operator and are widely available in current FPGA devices. In order to improve performances, MAC operators are integrated in some FPGA devices as a hardwired component like DSP48 in Xilinx Virtex4.

Two implementation approaches are possible in which either hardware or time optimization is considered.

Hardware Optimization. This approach allows to reproduce partially the temporal model of processors. The computations are performed iteratively and coefficients are read from the tables sequentially. The results can be temporarily stored in paged memory rather than registers. In this approach, the total number of iterations is directly proportional to the order of the desired moment and it remains relatively small (some thousands only). Figure 11 depicts one of the two variants of realization: with or without pipelined computation. The pipelined organization allows to increase the calculation frequency of the iterations.

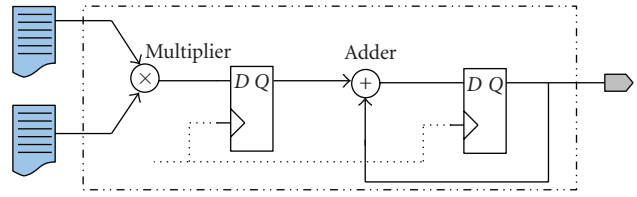

Figure 11: Using MAC operator in data flow architecture.

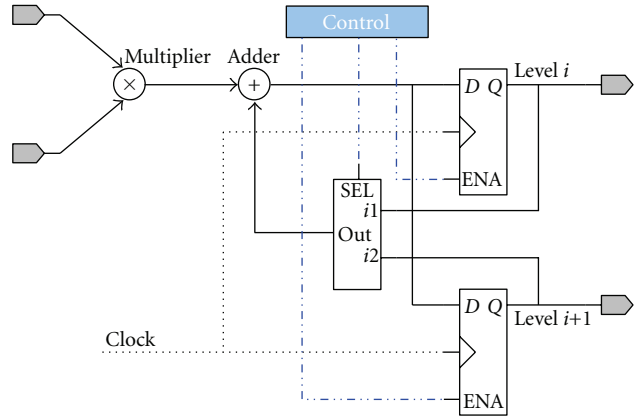

Figure 12: Reduction of the calculation resources by reusing hardwired operators.

Time Optimization. In that case, we consider that the amount of the computation hardware is sufficient. Therefore, the architecture includes all necessary pipelined operators as it is suggested in Figure 10. The intermediate results are stored in registers. This solution offers the possibility of reducing the number of operators by reusing the same hardware resources as shown in Figure 12.

Figure 13 describes the hardware implementation of the Zernike computation block. Its main objective is to generate the different Zernike moments from the accumulation moments calculated with the accumulation grids and from the coefficients computation module. It mainly consists of MAC blocks and of a module destined to compute the square root of the module according to (32). Only 75 slices are required to implement the entire block.

4.2. Hardware Implementation of the Neural Network. The parallel nature of neural networks makes them very suitable for hardware implementation. Several studies have been performed so far allowing complex configurations to be implemented in reconfigurable circuits [15, 16].

The proposed architecture strives to reduce the amount of logic to be utilized. This is mainly due to the fact that the neural network has to be implemented with its associated

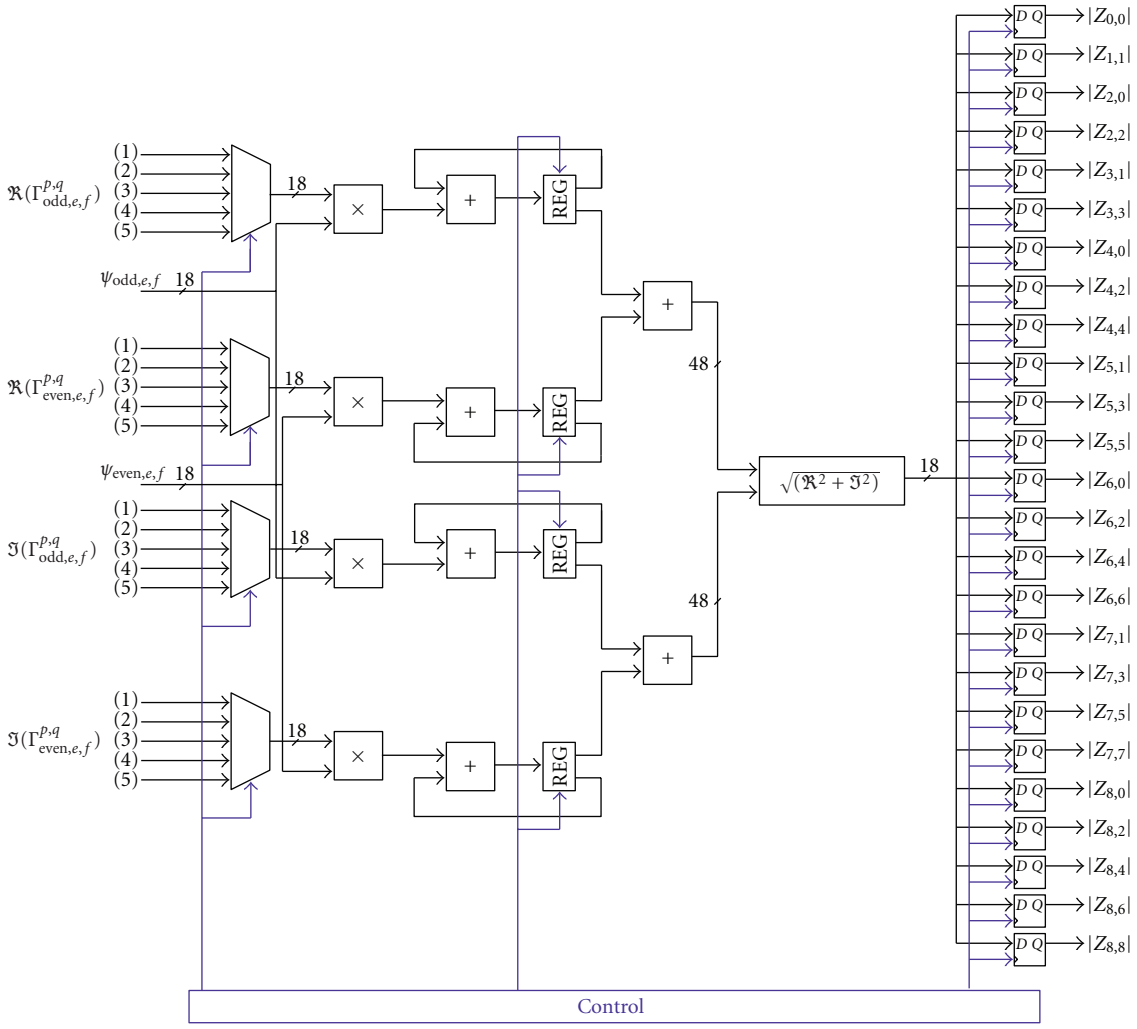

Figure 13: Computation of Zernike moments from the accumulation moments.

complex preprocessing that may require a lot of resources. An example of such architecture is presented in Figure 14.

In this example, the neural architecture is implementing a 5-input MLP with 7 hidden nodes and 3 outputs. These parameters are easily modifiable since the proposed circuit is scalable.

Input data are accepted sequentially and applied to the series of multipliers. A_j corresponds to the jth input of the present state whereas B_j corresponds to the jth of the next set. Data arrive at each clock cycle.

At each clock cycle, at any particular level of adder, apart from the addition operation between the multiplier output and the sum from the previous level, the multiplication operation of the next set of inputs at the adjacent multiplier is also simultaneously performed. The sum, thus, ripples and accumulates through the central adders (48 bits) until it is fed to a barrel shifter that aims to translate the data into a 16-bit address. The obtained sum addresses a sigmoid block memory (SIGMOID0) containing 65536 values of 18 bits.

This block feeds the outputs of the hidden layer sequentially to three MAC units for the output layer calculation. Finally a multiplexer distributes serially the results of the output layer to another sigmoid block memory (SIGMOID1). After a study on data representation, it has been decided to code the incoming data in 18 bits. Weights are stored in ROM (Read-only Memories) containing 256 words of 18 bits. The control of the entire circuit is performed by a simple state machine that aims to organize the sequence of computations and memory management.

The number of multipliers required for the network is $I+O$, where I is the number of inputs and O is the number of outputs. Considering that the number of hidden nodes may be large compared to the number of inputs and the number of outputs, the adopted solution does not affect the number of multipliers which is a great relief. In this context, it is also important to note that the design is very easily scalable to accommodate more hidden, input or output nodes. For example, adding a hidden node does not impact the number of resources but requires an additional cycle of computation. Adding an input may be accommodated by the addition of another ROM, multiplier, and adder set to the series of adders at the centre (part HL of the figure). Moreover, the

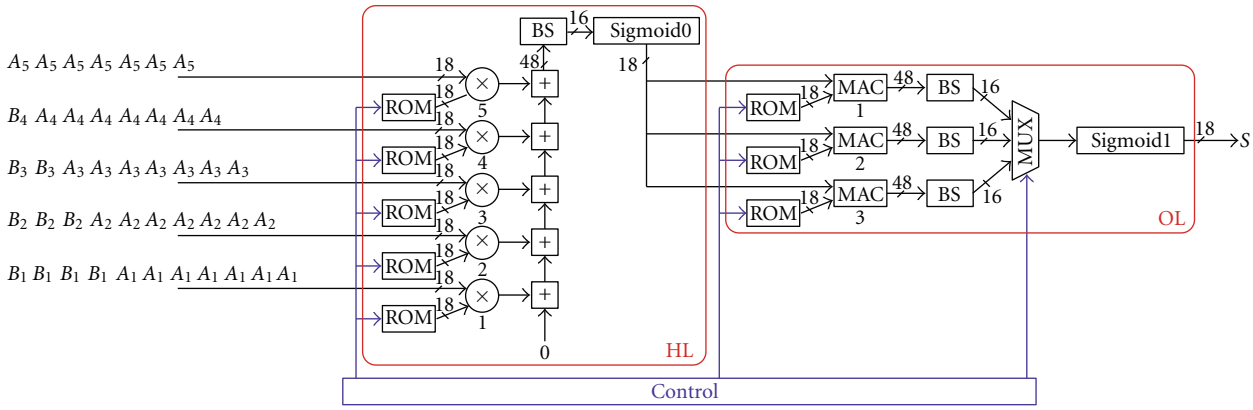

Figure 14: Example of the hardware implementation of a basic neural network.

Table 3: Summary of occupied resources.

Module	Number of used logic slices/ available	Used DSP blocks/ available	Number of used memory bits/ available (Kb)
Accumulation moments	1786/49152	0/96	0 /4320
Zernike moments	75/49152	60/96	21/4320
Neural Network	477/49152	28/96	2808/4320
Total	2338/49152 (4%)	88/96 (92%)	2829 /4320 (65.5%)

addition of an output node can be fulfilled by adding another ROM, MAC unit, and sigmoid block to the part OL of the figure.

Another advantage of the architecture is that a single activation function (sigmoid block) is required to compute the complete hidden layer. This block consists of a Look-up Table (LUT) that stores 65536 values of the function.

In general, the time required to obtain the outputs after the arrival of the first input is fixed to $I + H + 6$, where I is the number of inputs, and H is the number of hidden units. In every cycle, $I + O$ number of multiplications is performed (O is the number of output units).

5. Performances

The complete architecture (preprocessing + neural network) has been implemented in a Xilinx Virtex4 (xc4lx100) FPGA which is the part that has been retained for the trigger implementation. This type of reconfigurable circuit exhibits a lot of dedicated resources such as memory blocks or DSP blocks that allow to compute a MAC very efficiently.

5.1. Resources Requirements. The resources that are required to implement the global L2 trigger are given in Table 3. The accumulation grid is essentially realized with logical resources. No DSP block is utilized at this level. The computation of Zernike moments from the accumulation moments makes intensive use of parallelism. Five computation stages enable to compute 25 Zernike moments very rapidly and make use of 60 DSP blocks.

Concerning the hardware implementation of the neural network, it is important to notice that, independently of the configuration, the amount of used resources is very low. Nevertheless, one may deplore an important usage of memory blocks destined to store the values of the sigmoid functions. This issue may be circumvented in case where hardware resources constitute an issue. A modified activation function $\mathrm{sgn}(x) \times (1 - 2^{-absx})$ could be used [17]. This has a shape quite similar to the sigmoid function and is very easy to implement on hardware with just a number of shifts and adds. This function can be executed with an error less than 4.3%

According to Table 3, it is clear that the entire system fits in an FPGA without consuming to much logic (4%). Moreover, the complete architecture has been devised in order to take full benefit of the intrinsic dedicated resources of the FPGA, that is, DSP and memory blocks.

5.2. Timing. The computation time of the complete trigger is summarized in Table 4. According to this table, it is important to notice that the timing constraints imposed by the HESS system have been met since the mean frequency to take a decision is fixed to 3.5 KHz, that is, 285 microseconds. The global latency time of the proposed L2-trigger is 115.3 microseconds which makes it possible to envisage other improvements.

It is important to note that most of the computation time is monopolized by the computation of Zernike moments from the accumulation moments. This is mainly due to the fact that the number of accumulations to perform is huge (104663 accumulations for an order-8) and that these

Table 4: Timing performances.

Module	Processing time in μs
Accumulation moments	13.5
Zernike moments	101.4
Neural Network	0.4
Total	115.3

computations are performed iteratively. Even if we have decided to parallelize the architecture in five stages, the number of iterations remains high ($\approx$30 000). A current work is performed to optimize the computations in this block for further improvements.

The maximum clock frequency has been estimated at 120 MHz and 366 MHz for the DSP blocks.

6. Conclusion

In this article, we have presented an original solution that may be seen as an intelligent way of triggering data in the HESS Phase-II experiment. The system relies on the utilization of image processing algorithms in order to increase the trigger efficiency. The hardware implementation has represented a challenge because of the relatively strong timing constraints 285 microseconds to process all algorithms. This problem has been circumvented by taking advantage of the nature of the algorithms. All these concepts are implemented making intensive use of FPGA circuits which are interesting for several reasons. First, the current advances in reconfigurable technology make FPGAs an attractive alternative compare to very powerful circuits such as ASICs. Moreover, their relatively small cost permits to rapidly implement a prototype design without major developmental constraints. The reconfigurability also constitutes a major point. It allows to configure the whole system according to the application needs, enabling flexibility and adaptivity. For example, in the context of the HESS project, it may be conceivable to reconfigure the chip according to the surrounding noise or to deal with specific experimental conditions.

References

[1] J. A. Hinton, "The status of the hess project," *New Astronomy Reviews*, vol. 48, no. 5-6, pp. 331–337, 2004.

[2] S. Funk, G. Hermann, J. Hinton, et al., "The trigger system of the Hess telescope array," *Astroparticle Physics*, vol. 22, no. 3-4, pp. 285–296, 2004.

[3] E. Delagnes, Y. Degerli, P. Goret, P. Nayman, F. Toussenel, and P. Vincent, "Sam: a new ghz sampling asic for the Hess-ii front-end electronics," *Nuclear Instruments and Methods in Physics Research*, vol. 567, no. 1, pp. 21–26, 2006.

[4] A. M. Hillas, "Cerenkov Light Images of EAS Produced by Primary Gamma Rays and by Nuclei," in *Proceedings of the 19th International Cosmic Ray Conference (ICRC '85)*, San Diego, Calif, USA, August 1985.

[5] B. Denby, "Neural networks in high energy physics: a ten year perspective," *Computer Physics Communications*, vol. 119, no. 2, pp. 219–231, 1999.

[6] C. Kiesling, B. Denby, J. Fent, et al., "The h1 neural network trigger project," in *Advanced Computing and Analysis Techniques in Physics Research*, vol. 583, pp. 36–44, 2001.

[7] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[8] M. R. Teague, "Image analisis via the general theory of moments," *Journal of the Optical Society of America*, vol. 70, pp. 920–930, 1979.

[9] F. Zernike, "Beugungstheorie des schneidenverfahrens und seiner verbesserten form, der phasenkontrastmethode," *Physica*, vol. 1, pp. 689–704, 1934.

[10] S. Khatchadourian, J.-C. Prévotet, and L. Kessal, "A neural solution for the level 2 trigger in gamma ray astronomy," in *Proceedings of the 11th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT '07)*, Proceedings of Science, Nikhef, Amsterdam, The Netherlands, April 2007.

[11] S. O. Belkasim, M. Ahmadi, and M. Shridhar, "Efficient algorithm for fast computation of zernike moments," *Journal of the Franklin Institute*, vol. 333, pp. 577–581, 1996.

[12] E. C. Kintner, "On the mathematical properties of the zernike polynomials," *Journal of Modern Optics*, vol. 23, pp. 679–680, 1976.

[13] L. Kotoulas and I. Andreadis, "Real-time computation of zernike moments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 6, pp. 801–809, 2005.

[14] M. Hatamian, "A real-time two-diensional moment generating algorithm and its single chip implementation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, pp. 546–553, 1986.

[15] J.-C. Prévotet, B. Denby, P. Garda, B. Granado, and C. Kiesling, "Moving nn triggers to level-1 at lhc rates," *Nuclear Instruments and Methods in Physics Research A*, vol. 502, no. 2-3, pp. 511–512, 2003.

[16] A. R. Omondi and J. C. Rajapakse, *Fpga Implementations of Neural Networks*, Springer, 2006.

[17] M. Skrbek, "Fast neural network implementation," *Neural Network World*, vol. 9, pp. 375–391, 1999.