Gamma-Ray Astronomy

Luca Serfilippi
luca.serfilippi@studio.unibo.it

Riccardo Turra
riccardo.turra@studio.unibo.it

1 Abstract

Gamma-Rays are the most energetic form of electromagnetic radiation in the universe. This kind of phenomena are typically associated with events such as neutron stars, back holes and white dwarfs. To astronomers it is important to detect the transient of the Gamma-Ray source and monitor them during their evolution in order to understand these events. The project Gamma-Ray Astronomy is about finding the coordinates, Euclidian and Spherical, of one or multiple Gamma-Ray sources located in the Skymap of the Crab Nebula. To achieve this, it has been simulated a Skymap (a ".fits file" that consists of a matrix of intensities) and estimated Gamma-Ray source positions through Computer Vision methods. The project has been divided in two parts, one for creating a Dataset from a given virtual environment and one for the detection of source points, which requirements is to be as fast and precise as possible in order to generates science alerts, which are detection that require a follow-up strategy in a realistic scenario (no further developed in this project).

2 DatasetMaker

The environment used to create model and Skymap files is called Cherenkov Telescope Array (CTA) Simulation. Ctools, a library embedded in ctaSimulation, is used to simulate event data.

2.1 Creation of Skymaps

It has been used two functions of the ctool library: ctobssim and ctskymap. By using 'ctools.ctobssim' function is possible to create an xml model with the chosen parameters, which is the base for the creation of the actual skymap. Once the model has been generated, by running 'ctools.ctskymap' function with the previous generated model as input, it gives in output the Skymap.

The source position has been randomized using as constraint a deviation angle of maximum 4.5° from the central position (which correspond to the angulation of where the telescope is pointing in the sky). To achieve this, it has been randomly generated a vector which point to the source position, with a magnitude of maximum 4.5 (named mod), a direction that spans from 0° up to 2π (named $alpha$) and the origin at the centre of the image (Image 1).

The coordinates Ra and Deg are computed as:
$Ra = math.sin(alpha) * mod$
$Deg = math.cos(alpha) * mod$

Either Skymap with and without Instrument Response Function (IRF) subtraction has been generated, with single, double sources, and without sources in order to make the Dataset as heterogeneous as possible.

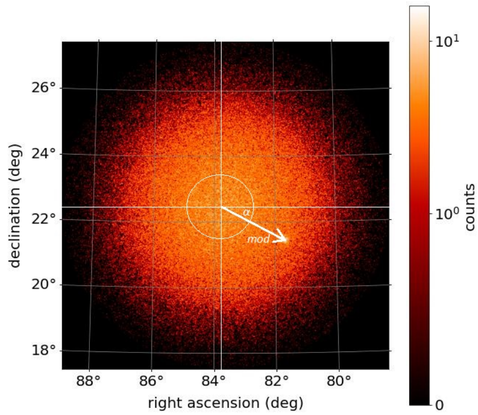
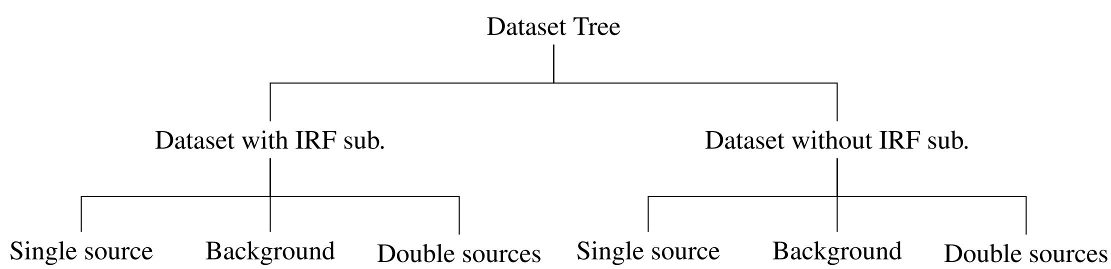


Figure 1: Representation of the generated vector to change the source position in the simulation

2.2 Structure of the Dataset



The Dataset has been created with 500 entries, 250 Skymaps with IRF subtraction and 250 Skymaps without IRF subtraction. Both in IRF and without IRF sub-Dataset, there are 105 single source Skymaps, 105 double sources Skymaps and 40 background only Skymaps.
The Model files are stored in the same way, in order to match the source point position with the Skymaps during the process of extraction of the Database elements.

3 Gamma-Rays Detection

The objective of the detection phase is to detect Gamma-Rays that come up as a spot which intensity is higher than its local background noise. (Image 2)

3.1 Image analysis

The Skymap consists of a matrix of one-channel intensity, with value that spans from negative number (e.g -3 or -4, due to noise) up to high positive number (depending on the position of the Gamma-ray source and the use of IRF subtraction).

3.1.1 Noise

In the generated Skymaps there are both Impulse and Gaussian noise. Due to an uneven sensitivity of the telescope, the intensity of both Source and noise are amplified at the center of the image (Image 3). In the simulation an automated noise cancellation (IRF subtraction) is present, that will reduce the Gaussian noise, but not the salt and pepper. For the sake of robustness, the algorithm should work either on IRF subtracted Skymaps and also on Skymaps without noise cancellation.

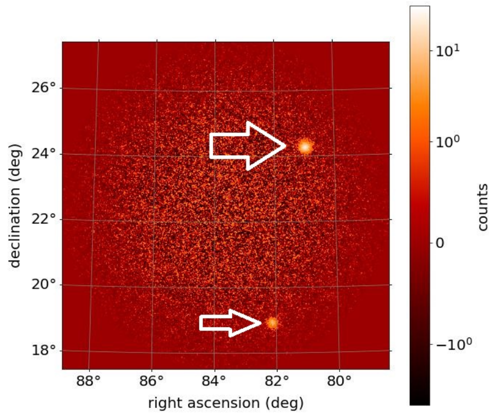


Figure 2: Original data with highlighted sources

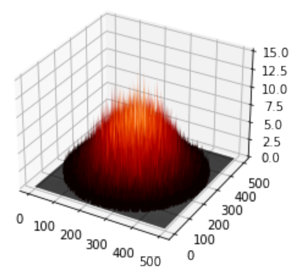


Figure 3: 3D plot of only background noise

3.1.2 Intensity distribution

Looking at the histogram of a Skymap (Image 4), it appears that it has an uneven distribution of intensity, with a high percentage of the intensity shifted on the lower part of the Intensity Value axis and a low percentage of pixel representing the source position and spurious salt and pepper noise spread across higher Intensity value. Beside histogram value distribution, another aspect that is worth pointing out is the separation between noise and sources. It is possible to distinguish between simple-to solve Skymap and hard case. The first definition is attributable to the Skymap which source point are easy to detect in the image due to an higher value of intensity with respect to the background (Image 2). The latter is attributable to the Skymap which source point are not near the centre of the image and are characterized by a lower intensity value in respect to the noise in the central part of the Skymap and thinner separation with the background (Image 9a). These sources are often near the limit value of mod and has a low energy associated.

3.2 Image conditioning

The source has not always the highest intensity value in the Skymap. Hence the best way found to detect a Gamma-Ray source is separating the source from the background as much as possible and then remove the background with a threshold.

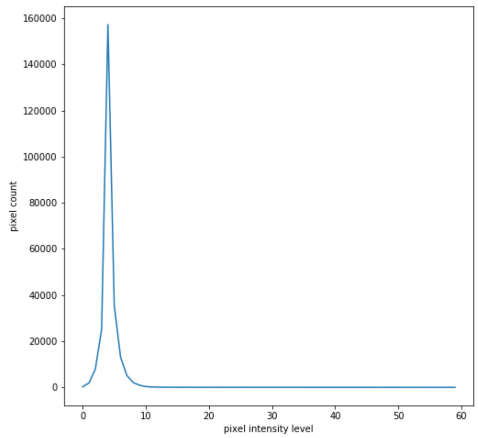


Figure 4: histogram of the Skymap, pixel intensities over 60 where not present, the background distribution is the only visible part

3.2.1 Image format

The first step of image intensity conditioning has been rearrange intensity value of the image by restricting the values of the Skymap in range from 0 to 255 gray-scale integer values. Since the noise can generate also negative values, the strategy followed has been to sum to the image the absolute of the lowest value and then saturate at a value of 255 in case any value exceeds this limit. With this solution, the Skymap retains at best the information.

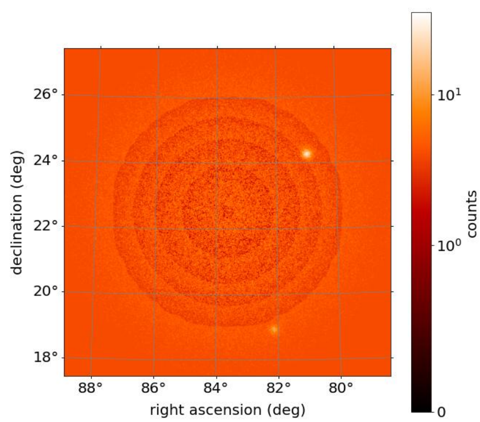
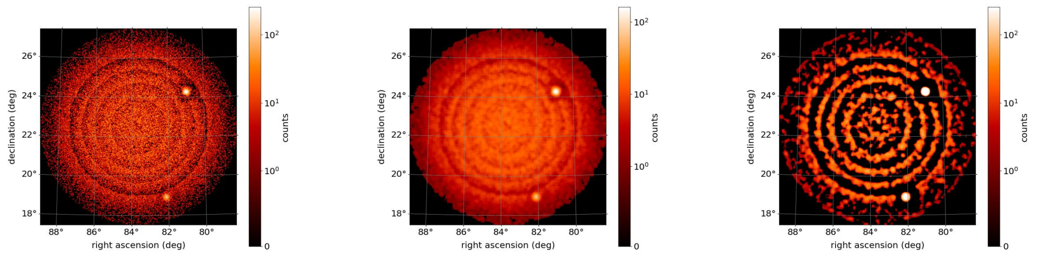


Figure 5: Image formatted in [0,255] integer values

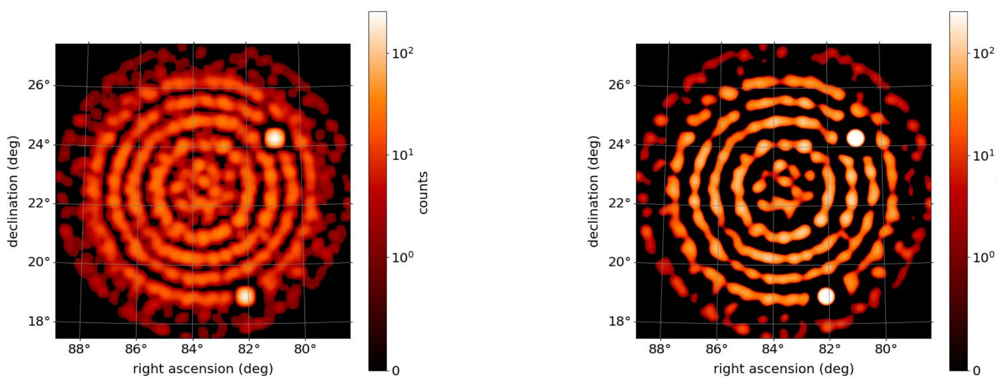
3.2.2 Image filtering

The second step has consisted in bringing out the source from the background. This has been achieved by a series of five convolutions with two types of kernels, a mean filter and a Difference of Means (DoM) between the values of a square core and the outside pixels.

The mean filter has as a parameter only its kernel size, while the DoM has his kernel size, core size k and multiplicative constant c. The two kernels have been alternated in application, starting with the DoM that is supposed to highlight spots that are as big as the core and reduce the intensity of homogeneous regions. Then the smoothing of the mean filter helps organizing noisy pixel into an homogeneous region so that in the next DoM convolution does not emphasize the highlighting of artifacts, avoiding the creation of false source (False Positive). At each smoothing step the source gets bigger so the core dimension of the DoM has been increased (Images 6).



(a) DoM with size= 50,k=0,c = 15 (b) Mean filter with size= 10 (c) DoM with size= 50,k=2,c = 10



(d) Mean filter with size= 15 (e) DoM with size= 50,k=5,c = 5

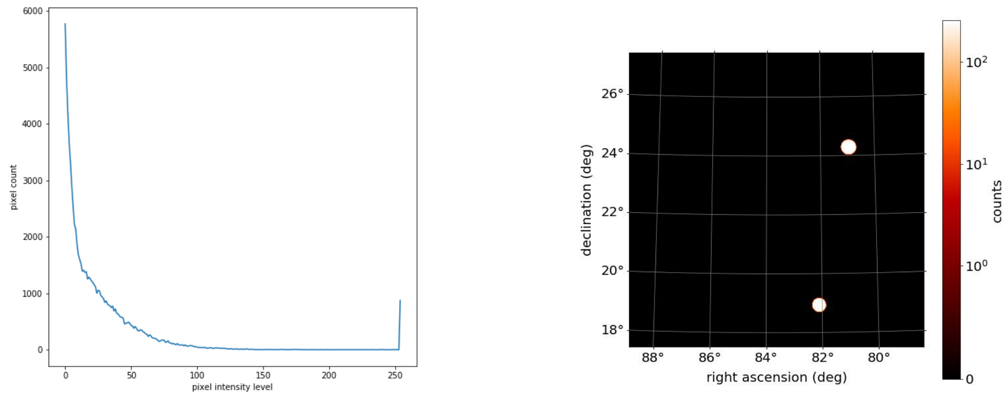
Figure 6: Series of convolutions

The DoM kernel has been used to discriminate a patch of pixels that stands out for its gradient with respect to its neighbours, instead of only its magnitude. If the source is in the outskirts of the image, its magnitude is weaker than the noise in the centre of the Skymap, but not its gradient after a smoothing over a large patch (Image 7).

```
[[-0.0625     -0.0625     -0.0625     -0.0625     -0.0625    ]
 [-0.0625      0.11111111  0.11111111  0.11111111 -0.0625    ]
 [-0.0625      0.11111111  0.11111111  0.11111111 -0.0625    ]
 [-0.0625      0.11111111  0.11111111  0.11111111 -0.0625    ]
 [-0.0625     -0.0625     -0.0625     -0.0625     -0.0625    ]]
```

Figure 7: DoM kernel matrix representation with size=5,k=1,c=1

As a side effect of this process, the intensity of the source is augmented at each iteration of the DoM kernel, with value that might exceed 255. As a result of this, the threshold value gets more robust to small changes in the parameter of the algorithm and also to parameters of Skymaps generation. As result of the filtering, the Gamma-Ray sources are well separated from the noise (Image 8a) so a threshold is applied to binarize the image. This operation has the effect of cutting the noise and keeping only the blobs that contain the sources (Image 8b).



(a) histogram of the Skymap after the convolutions, the source pixels can be seen at 255 intensity

(b) binary result of filtering and thresholding

Figure 8: Example of binarization of a Skymaps

3.3 Blobs detection

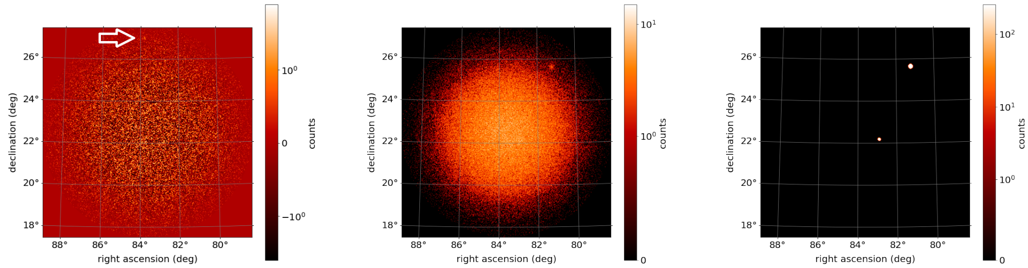
Once the Skymap has been binarized, the last thing left to do in order to retrieve the source position is to detect the blobs and calculate their center of gravity as the source position is considered uniformly spread across a single pixel point. By imposing constraint on the size and circularity of the blobs detected, it is possible to get rid of the artifacts generated during the image filtering, while keeping the source unmodified. Finally, in order to check the correctness of the detection, the center of gravity expressed in pixel was converted into Spherical coordinates with the method of AstroPy library $pixel\_to\_world$ and then measured the distance from the ground-truth data extrapolated from the model file. If the difference is within a distance of two pixels using the check-board distance, then it is considered a catch.

3.4 Parameter tuning

The algorithm contains a few editable parameters such as kernels size, DoM core sizes and their multiplicative scalar to amplify the difference, threshold for the binarization and the minimum circularity of the blob detection. A manual tuning of the parameters has been done in order to keep the algorithm as robust and fast in execution as possible.

4 Results

The algorithm performance has been found as follows: $Accuracy = NR/(NR + NE)$ with NR as Number of Right predictions and NE as Number of Errors Based on the Database mentioned in Structure of the Database paragraph, the accuracy of the algorithm over both Databases (500 images) is 95.2%, with a computational time of about 30ms at image. Looking at the errors, it is possible to detect two types of failures; one occurs when the sources generated have low energy and the are located far from the centre of the Skymap (Image 9a). The other one occurs when the background generated contains noise distributed in a circular shape way near the centre of the Skymap, which, during the filtering process gets enlarged and amplified similarly to what happens with the actual Sources (Images 9). By trying to solve one problem, the other one gets worst, so the solution found tries to minimize the error in both cases, without solving one.



(a) Skymap that contains an hard source to detect

(b) Skymap with noise grouped in background

(c) blobs filtered by the algorithm based on (Image 9b)

Figure 9: Examples of hard to detect Skymaps

5 Alternative solution

5.1 Image Filtering

Due to the nature of the noise which affect the image, an alternative solution was implementing an Median filter followed by a Bilateral filter. By doing this kind of image transformation, it is possible to get rid of the major part of the impulse noise at the beginning and then ease the Gaussian noise taking into account both distance and intensity of the considered pixels. As a consequence of the nature of the bilateral filter, the computational time requested to manipulate the image is far greater with respect to the solution cited in the par. 3.2.2, without major accuracy improvement. For this reason this solution was discarded in favor of the one which exploit custom made kernel.

5.2 Parameter Tuning

An improvement tested was an automatic threshold selection for the binarization of the image. it result after various testing that as a consequence of the uneven distribution of the intensity value, it was impossible for the algorithm to determine two peaks and chose a value between them. Exploiting the use of the custom DoM kernel, a constant threshold value (e.g. 190) results as a valid and robust solution

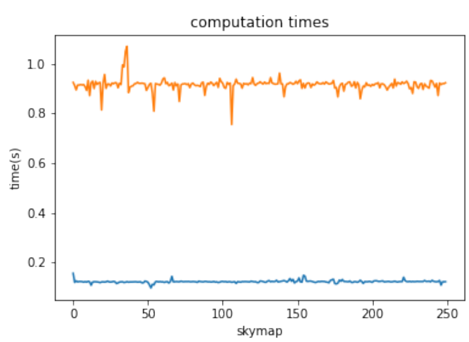


Figure 10: difference in computation times per image, orange referred to alternative solution and blue as the solution reported in par. 3.2.2

6 References

Link to the GitHub project