

# AstroIMP Documentation

Eduardo Bojórquez Martínez<sup>1</sup>

<sup>1</sup>Industrial Engineering Department, University of Sonora,  
Hermosillo, Sonora.

June 8, 2024

## Abstract

*AstroIMP* is a tool for automated pre-processing astronomical images, and manipulating them since the quality of the images is usually low compared to other traditional images. Some functions are contrast enhancement, crop images, reduce noise, reduce *CCD* errors, remove the background, and stay only with the objects of interest to detect objects automatically with some algorithm for image segmentation. This document has the intention to explain how to install and use it.

## 1 Introduction

Since the emergence of Artificial Intelligence, commonly abbreviated as *AI*, aims to solve complex problems smartly. There are various definitions of *AI*, Ben Coppin defines it as: "The use of methods based on the intelligent behavior of humans and other animals to solve complex problems".

One of the areas of study of *AI* is pattern recognition, which we can define as the discipline of *AI* that aims to classify a pattern into several categories or classes; That is, how computers can observe the environment and learn to distinguish between different patterns, to make decisions based on their categories.

Object recognition is the task of identifying an object found in an image or video. One of the most important steps in digital image processing is pre-processing.

For the preprocessing step, everything that could generate noise is eliminated, the image is enhanced to facilitate image analysis, and there are different image enhancement techniques. In astronomical images, extended structures are normally surrounded by point objects (stars), the latter must be eliminated or cut at the maximum values, to prevent them from causing noise in the identification of emission nebulae and allow easier detection. However, when images are very noisy, it can cause the edges of surrounding objects to also be blurred. On some occasions, after carrying out the noise reduction process, they may be distorted, causing the results to be inaccurate or erroneous.

## 2 Software architecture

The architecture is divided into two modules: The first is image manipulation, and the second one is image pre-processing.

The module image manipulation has two functions image contrast enhancement and crop images.

The first functionality image contrast aims to manipulate the data by increasing the dynamic range of intensities in low-contrast images.

The second functionality crop images aim to create multiple sub-images from the original image. These functionalities are described in the subsection software functionalities.

The module pre-processing has three functions, noise reduction, *CCD* error reduction, and background removal.

For the first functionality noise reduction functionality, we did a literature review and found that one of the simplest ways to reduce noise is through the Gaussian and median filter, in several articles they used the combination of both filters to reduce noise Gaussian and eliminate it. of the peaks caused by Poisson noise. In several papers, they applied morphological operations to reduce noise, improve the characteristics of bright objects, and reduce halos. Finally, the anisotropic diffusion filter was used to strengthen the edges of objects and eliminate distant stars in astronomical images. Since the anisotropic diffusion filter has excellent results, we apply it to enhance the edges of point sources and extended objects.

The second functionality is *CCD* error reduction, this process is only applied to images that were obtained from *CCD* number 3 of *IPHAS* since it is the one that presents damage to the *CCD* columns. *CCD* errors are

just columns of pixels that are not working. There are various reasons for a column to be damaged (from electronic to mechanical); some columns are dead (dark), some are bright, and some have a constant added to what they should show. A good *CCD* usually doesn't have more than one pair of them. They are easy to fix when isolated (neighboring columns are interpolated), but nothing can be done when multiple damaged columns are together. We applied a mathematical morphological close operator to reduce *CCD* errors. Once this process is completed, the image is inverted and returned to the original dynamic range, the detailed process is described in the next subsection.

For the third functionality background removal, to remove it the first step is image acquisition, the second step is image normalization with min-max, the third is applying the anisotropic diffusion filter first introduced by Perona-Malik in 1990, the fourth is the normalization with min-max for the algorithm diffuse *PFCM* first proposed by Bezdek in 2005, and is a combination of *PCM* introduced by Krishnapuram y Keller in 1993 and *FCM* proposed by Bezdek in 1981. In this version, the objective function can be obtained through an iterative process in which the function membership, typicality values, and group centers are updated simultaneously for each group. This is a hybridization of *PCM* and *FCM* algorithms that solve the group matching problem in *PCM* and the noise sensitivity of *FCM*.

## 3 Requirements

### 3.1 How to install anaconda and astropy

Anaconda is an open-source distribution that includes packages to process high volumes of information, perform predictive analysis, and scientific computing, is one of the easy ways to perform data science and machine learning on a single machine.

Anaconda distribution is available on the official site for Windows, Mac, and Linux:

<https://www.anaconda.com/download/success>

Download the installer for your operating system then follow the steps:

For Windows, do not install as Administrator unless admin privileges are required:

1. After the download is completed, you need to go to your Downloads

folder and double-click the installer to launch.

2. Then click Next.
3. Read the licensing terms and click I Agree.
4. After that you need to select who will be installed, it is recommended that you install for Just Me, which will install Anaconda Distribution to just the current user account. Only select an install for All Users if you need to install for all users' accounts on the computer (which requires Windows Administrator privileges).
5. Click Next.
6. Select a destination folder to install Anaconda and click Next. Install Anaconda to a directory path that does not contain spaces or unicode characters.
7. Choose whether to add Anaconda to your PATH environment variable or register Anaconda as your default Python. We don't recommend adding Anaconda to your PATH environment variable, since this can interfere with other software. Unless you plan on installing and running multiple versions of Anaconda or multiple versions of Python, accept the default and leave this box checked. Instead, use Anaconda software by opening Anaconda Navigator or the Anaconda Prompt from the Start Menu.
8. Click Install. If you want to watch the packages Anaconda is installing, click Show Details.
9. Click Next.
10. Optional: To learn more about Anaconda's cloud notebook service, go to <https://www.anaconda.com/code-in-the-cloud>. Or click Continue to proceed.
11. After a successful installation you will see the "Thanks for installing Anaconda" dialog box.

For Mac:

1. Download the graphical (.pkg) macOS installer for your system.

2. Double-click the downloaded file and click Continue to start the installation.
3. Answer the prompts on the Introduction, Read Me, and License screens.
4. Anaconda recommends that you choose Install for all users of this computer. As of version 2024.02-1, the default location of the installer is /opt/anaconda3. To install elsewhere, select Install on a specific disk.
5. Click Install.
6. Once the installation is complete, click Continue.
7. Optional: To learn more about Anaconda's cloud notebook service, go to <https://www.anaconda.com/code-in-the-cloud>. Or click Continue to proceed.
8. A successful installation displays the thank you for installing Anaconda Distribution.

Astropy is a collection of software packages written in Python programming language and designed for use in astronomy. The software is a single, free, core package for astronomical utilities due to the increasingly widespread usage of Python by astronomers, and to foster interoperability between various extant Python astronomy packages.

To install Astropy you can use the following command in the Anaconda prompt:

```
conda install -c conda-forge astropy
```

## **3.2 How to install pre-requisites AstroIMP**

### **3.2.1 OpenCV**

The Open Source Computer Vision Library abbreviated as OpenCV, is a library of programming functions mainly for real-time computer vision. In the field of visual computing and image processing, OpenCV has established itself as a fundamental tool. This open-source library offers a wide range of functionalities that allow everything from basic image manipulation to the development of complex facial recognition and object-tracking systems. It was originally developed by Intel, the library is cross-platform and licensed

as free and open-source software under Apache License 2. Starting in 2011, OpenCV features GPU acceleration for real-time operations.

To install the OpenCV library for python, you can do it with the following command:

```
pip install opencv-python
```

### 3.2.2 Imutils

Imutils is a library for Python with a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much easier with OpenCV.

To install the imutils library, you can do it with the following command:

```
pip install imutils
```

## 4 How to use AstroIMP

### 4.1 Specify the input and output directory

Since all the functions are required to give a directory with a set of images, it is necessary to first specify the input and output directory to use the desired function.

To specify the input directory of the images:

```
python principal.py -d «input directory».
```

To specify the output directory of the images:

```
python principal.py -r «output directory»
```

### 4.2 Specify crop images

Given a directory with a set of images and specified desired size to create multiple sub-images from the original image. The tool can generate sub-images with the specified size for the width for each sub-image, the height will be the same size as the specified width. The test images we used have a size of 2048 width and 4096 height in pixels, and the desired size was 512 pixels, it generated 32 sub-images with the size 512 x 512 pixels.

To crop images you can use the argument:

```
python principal.py -d «input directory» -r «output directory» -c
```

## 4.3 Specify preprocess

### 4.3.1 Contrast enhancement

Image enhancement is an area of image processing that consists of a collection of task-dependent methods that accentuate features important for a specific goal, making human or automated analysis more effective. Astronomical images frequently contain many point objects and, at the same time, extended diffuse structures with embedded point objects; Generally, all images are affected by noise due to the acquisition process of *CCD* devices, so noise reduction is one of the necessary steps for the detection of different objects.

For contrast enhancement and background removal, it is required first to specify the argument `-p` or `-preprocess`.

To specify if you are going to preprocess the images:

```
python principal.py -d «input directory» -r «output directory» -p
```

The ZScale algorithm was first introduced in *IRAF* software and updated for Python with the name PyRAF. It is made to avoid the laborious task of computing a whole image histogram by displaying image values close to the median image value. Astronomical images, typically have an extremely peaked histogram matching the continuum in a two-dimensional spectrum or the background sky in direct imaging, which can benefit greatly from this.

Specifies if applies contrast enhancement, using the algorithm `zscale`:

```
python principal.py -d «input directory» -r «output directory» -p -zs
```

Percentile range computes the range of pixel values to use when adjusting the contrast of FIT images using a simple percentile cut. A subsample of the input image data is used for efficiency reasons.

Specifies if applies contrast enhancement, using the algorithm `percentile range`:

```
python principal.py -d «input directory» -r «output directory» -p -pr
```

Arcsinh percentile locates the lower and upper boundaries containing all visible objects using a mapping `arcsinh` that displays the image values.

Specifies if applies contrast enhancement, using the algorithm `arcsin percentile`:

```
python principal.py -d «input directory» -r «output directory» -p -ap
```

### 4.3.2 CCD error reduction and high noise peaks reduction

In the tool we implemented the *CCD* errors reduction, the steps we apply are based on the work of Candeas in 1999 they use morphological operators to enhance the peaks that are contained in the image, and then remove them. The detailed steps that were followed are:

- Invert the image with its original dynamic range. The literature review mentions that it is a good practice in astronomy because the contrast of the image is improved, and it is easier to visualize for the human eye.
- Restore the original image with the inverted one. Candeas carried out this procedure to enhance the inverted image and reduce the optical effects of halos around bright stars.
- Apply the mathematical morphological closure operator. By using this operator, all internal noise present in the region of the object is removed and the background is not affected. This technique also has the property of highlighting the peaks contained in the image. We are interested in those with saturated pixels, and flowering effect, among others. To achieve this, a disk-type structure with a radius  $r = 8$  was used to enhance the inverted image and highlight the high peaks contained in the image.
- Once the operator closure is completed, the image is inverted again, and the result is subtracted from the original image.
- Finally, it is returned to its original dynamic range again.

To specify if you want to reduce CCD errors:

```
python principal.py -d «input directory» -r «output directory» -ccde
```

Due to the dynamic range of both point and extended objects, the types of noise described in the previous sections, and the errors of *CCD*, it is necessary to remove the background from the image to improve object detection extended with low emission. To achieve background elimination, a threshold is determined with the help of the *PFCM* algorithm.



### 4.3.3 Background removal

For background removal it uses the *PFCM* algorithm, translated to python by Djemai in 2018, with this algorithm we generate two groups, the objective is to divide the background of the image and the extended objects or stars that are in the image, with the result obtained, a mask is generated in which the background is assigned the value of 0, and the objects are assigned the value 1. Then a new image is generated by multiplying the values of the generated mask with the original image, in this way, we maintain the original values of the image, and the rest remains with a value of 0 (black).

To specify the background removal using the algorithm PFCM:

```
python principal.py -d «input directory» -r «output directory» -pf
```

## 5 Examples

The following examples were executed from the command prompt, they can be executed with the python command with the name of the main.py file and the arguments specified previously. In each case, a folder was created to store the result obtained, and in two other folders the original fits and tiff images were stored, in each case, it was specified. In the following subsections, examples of the use of the instructions/arguments of the automated tool for the preprocessing of astronomical images are given. All test images were taken from <https://www.iphas.org/>, and are currently in <http://www.star.ucl.ac.uk/IGAPS/>.

### 5.1 Contrast enhancement

#### 5.1.1 ZScale algorithm

For this example, two folders have been created as shown in the figure 1, in which the `imagenes_fits` folder was created within the same directory where the source code is located and the results folder.

Within the `imagenes_fits` folder, 5 images in fits format have been placed, as shown in figure 2.

In the console or command prompt in Windows, we execute the following instruction to use the ZScale algorithm.

Figure 3 shows an example of how to run the tool in the Windows command prompt. First, we accessed the folder where the source code is, and
















 .idea	✓	06/06/2024 02:30 p. m.	Carpeta de archivos	
 __pycache__	✓	21/05/2024 01:30 p. m.	Carpeta de archivos	
 imagenes_fits	✓	06/06/2024 02:23 p. m.	Carpeta de archivos	
 resultados	✓	06/06/2024 06:42 p. m.	Carpeta de archivos	
 __init__.py	✓	26/02/2020 04:18 p. m.	Archivo PY	1 KB
 dice_coef_pfc.py	✓	22/03/2024 03:34 p. m.	Archivo PY	1 KB
 gui.py	✓	06/06/2024 11:35 a. m.	Archivo PY	2 KB
 histogram.py	✓	28/03/2024 10:21 a. m.	Archivo PY	1 KB
 image.py	✓	20/05/2024 05:08 p. m.	Archivo PY	15 KB
 line.py	✓	18/04/2024 01:48 p. m.	Archivo PY	6 KB
 PFCM.py	✓	07/04/2024 10:43 a. m.	Archivo PY	6 KB
 pointarray.py	✓	18/04/2024 01:50 p. m.	Archivo PY	14 KB
 preprocess.py	✓	20/05/2024 06:24 p. m.	Archivo PY	22 KB
 principal.py	✓	06/06/2024 02:44 p. m.	Archivo PY	4 KB
 rename_files.py	✓	20/05/2024 05:53 p. m.	Archivo PY	1 KB

Figure 1: List of files and folders in the source code directory.





 PN_NGC_7048_Ha_1.fits	✓	29/08/2020 08:29 p. m.	Archivo FITS	16,403 KB
 PN_NGC_7354_1_Ha.fits	✓	29/08/2020 08:29 p. m.	Archivo FITS	16,403 KB
 PN_SH_1-89_Ha_4.fits	✓	29/08/2020 08:29 p. m.	Archivo FITS	16,400 KB
 PN_SH_2-71_Ha_2.fits	✓	29/08/2020 08:29 p. m.	Archivo FITS	16,400 KB
 PN_SH_2-188_Ha_1.fits	✓	29/08/2020 08:29 p. m.	Archivo FITS	16,403 KB

Figure 2: List of files in the imagenes\_fits directory.

executed `principal.py`, we provided the input and output directory, in this case, we placed the `imagenes_fits` and `resultados` folders, and because we want to run `ZScale`, we require the `-p` argument followed by `-zs`; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_fits -r resultados -p -zs
```

After executing this command, the results are saved in the results folder.

```

C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_fits -r resultados -p -zs
Preprocess...
PN_NGC_7048_Ha_1.fits
Contrast enhancement with zscale...
PN_NGC_7354_1_Ha.fits
Contrast enhancement with zscale...
PN_SH_1-89_Ha_4.fits
Contrast enhancement with zscale...
PN_SH_2-188_Ha_1.fits
Contrast enhancement with zscale...
PN_SH_2-71_Ha_2.fits
Contrast enhancement with zscale...

```

Figure 3: Executing the tool in the Windows command prompt, in the example the -zs instruction is executed to make use of the ZScale algorithm.

In figure 4, the files in the results directory are shown, the tool will generate a tif image and another with the result obtained after applying the contrast adjustment with ZScale.

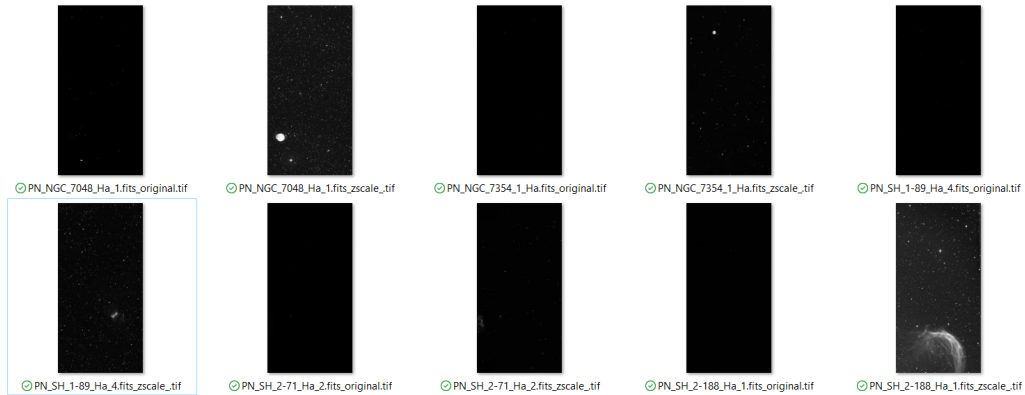


Figure 4: List of files in the folder intended to store the results of the ZScale algorithm.

### 5.1.2 Example of using the percentile range algorithm

In the same folders as the previous example, the command was executed to generate the images using percentile range contrast adjustment. The example of the execution of the algorithm is shown in figure 5. As in the previous example, the folder where the source code is accessed, and principal.py is executed, we provide the input and output directory, in this case, the imagenes\_fits and resultados folders were placed, and Because we want to run

percentile range, we require the -p argument followed by -pr; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_fits -r resultados -p -pr
```

```
C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_fits -r resultados -p -pr
Preprocess...
PN_NGC_7048_Ha_1.fits
Contrast enhancement with percentile range...
PN_NGC_7354_1_Ha.fits
Contrast enhancement with percentile range...
PN_SH_1-89_Ha_4.fits
Contrast enhancement with percentile range...
PN_SH_2-188_Ha_1.fits
Contrast enhancement with percentile range...
PN_SH_2-71_Ha_2.fits
Contrast enhancement with percentile range...
```

Figure 5: Executing the tool in the Windows command prompt, in the example the -pr instruction is executed to make use of percentile range.

After executing this command, the results are saved in the results folder. In figure 6, the files in the results directory are shown, the tool will generate a tif image and another with the result obtained after applying the contrast adjustment with percentile range.

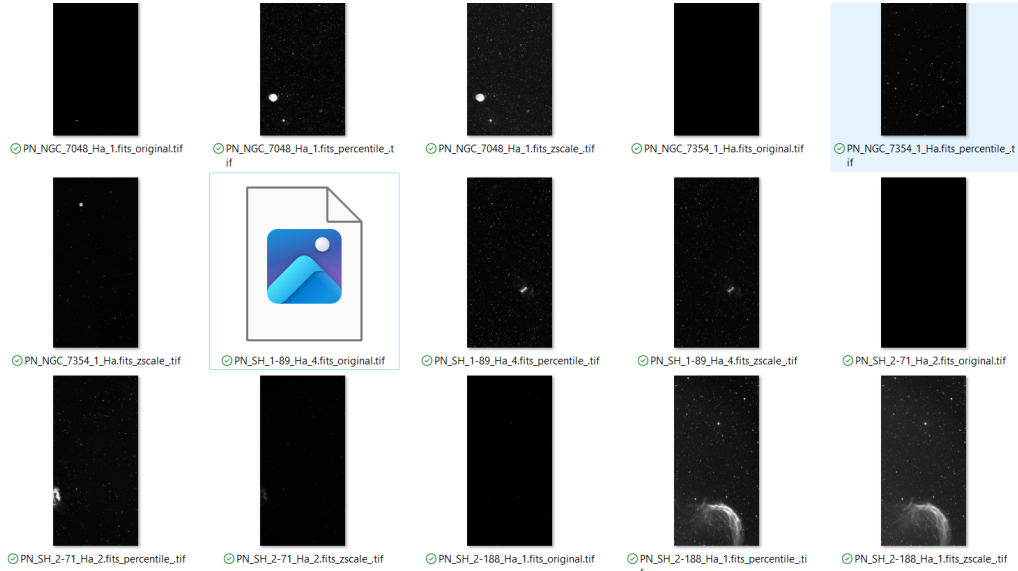
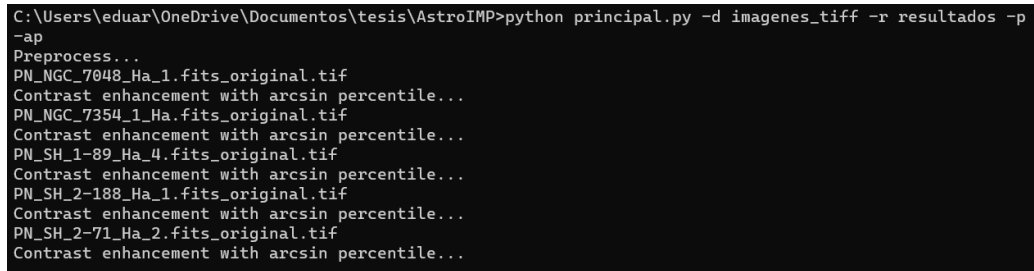


Figure 6: List of files in the folder intended to store the percentile range results.

### 5.1.3 Example of using the arcsinh percentile algorithm

In the same folders as the previous example, the command was executed to generate the images using arcsin percentile contrast adjustment. The example of the execution of the algorithm is shown in figure 7. As in the previous example, the folder where the source code is accessed, and principal.py is executed, we provide the input and output directory, in this case, the images\_tiff and results folders were placed, and because we want to run arcsinh percentile, we require the -p argument followed by -ap; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_fits -r resultados -p -ap
```



```
C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_tiff -r resultados -p -ap
Preprocess...
PN_NGC_7048_Ha_1.fits_original.tif
Contrast enhancement with arcsin percentile...
PN_NGC_7354_1_Ha.fits_original.tif
Contrast enhancement with arcsin percentile...
PN_SH_1-89_Ha_4.fits_original.tif
Contrast enhancement with arcsin percentile...
PN_SH_2-188_Ha_1.fits_original.tif
Contrast enhancement with arcsin percentile...
PN_SH_2-71_Ha_2.fits_original.tif
Contrast enhancement with arcsin percentile...
```

Figure 7: Executing the tool in the Windows command prompt, in the example the -ap instruction is executed to make use of arcsinh percentile.

After executing this command, the results are saved in the results folder. In figure 8, the files in the results directory are shown, the tool will generate a tif image and another with the result obtained after applying the contrast adjustment with arcsinh percentile.

### 5.1.4 Example of using the arcsinh percentile range algorithm

In the same folders as the previous example, the command was executed to generate the images using contrast adjustment arcsin percentile range. The example of the execution of the algorithm is shown in figure 9. As in the previous example, the folder where the source code is accessed, and principal.py is executed, we provide the input and output directory, in this case, the images\_tiff and results folders were placed, and because we want to run arcsinh percentile range, we require the -p argument followed by -apr; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_fits -r resultados -p -apr
```

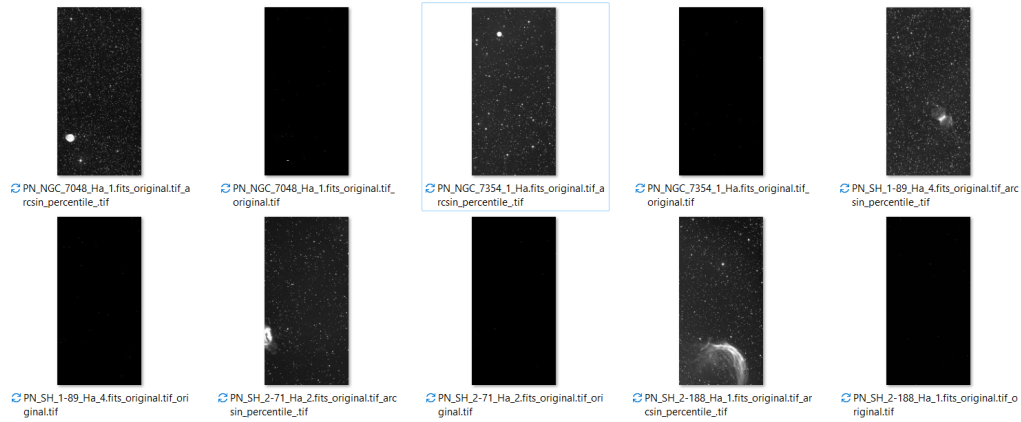


Figure 8: List of files in the folder intended to store the results of the arcsinh percentile algorithm.

```
C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_tiff -r resultados -p
-apr
Preprocess...
PN_NGC_7048_Ha_1.fits_original.tif
Contrast enhancement with arcsin percentile range...
PN_NGC_7354_1_Ha.fits_original.tif
Contrast enhancement with arcsin percentile range...
PN_SH_1-89_Ha_4.fits_original.tif
Contrast enhancement with arcsin percentile range...
PN_SH_2-188_Ha_1.fits_original.tif
Contrast enhancement with arcsin percentile range...
PN_SH_2-71_Ha_2.fits_original.tif
Contrast enhancement with arcsin percentile range...
```

Figure 9: Executing the tool in the Windows command prompt, in the example the -apr instruction is executed to make use of arcsinh percentile range.

After executing this command, the results are saved in the results folder. In figure 10, the files in the results directory are shown, the tool will generate a tif image and another with the result obtained after applying the contrast adjustment with arcsinh percentile range.

## 5.2 Crop images

In this example, an image of size 2049 x 4096 pixels was taken, and two folders images\_tiff and resultados\_recortar were created.

Figure 11 shows the example of executing the crop images functionality. The folder where the source code is is accessed, and the following instruction was executed: `python principal.py`, we provide the input directory with `-d`

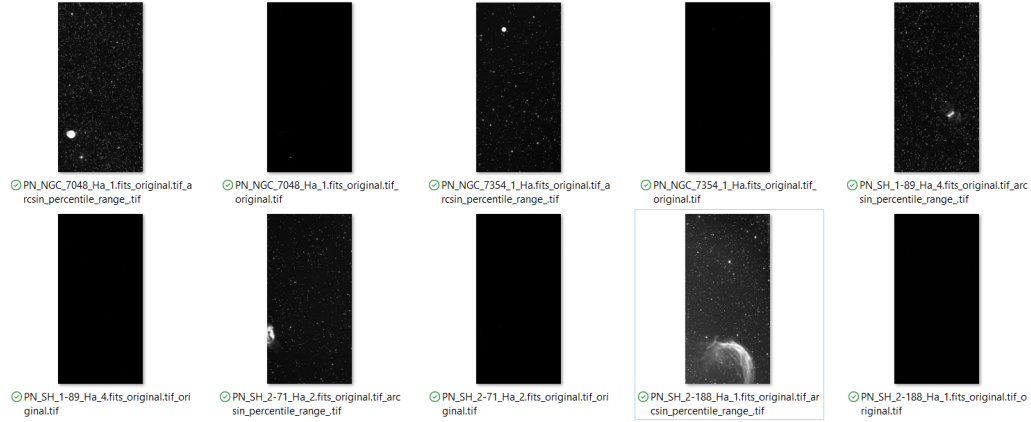


Figure 10: List of files in the folder intended to store the results of the arcsinh percentile range algorithm.

```
C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_tiff -r resultados_recortar -c
Crop images...
```

Figure 11: Executing the tool in the Windows command prompt, in the example the -c command is executed to crop the images.

and output with -r, in this case, the imagenes\_tiff and results folders were specified resultados\_recortar respectively, and because we want to execute the functionality to crop the image, we require the -c argument; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_tiff -r resultados_recortar -c
```

After executing this command, the results are saved in the results folder. In figure 12, the files in the results directory are shown, and the tool will generate several tif images with a size of 512 x 512 pixels.

### 5.3 Background removal with PFCM

For this example, it is required that you first have the image with contrast adjustment, in this case, the percentile range algorithm was executed first, since it is the one that provides the best result, another folder called imagenes\_tiff was created, to store the images to remove the background.

The example of the execution of the algorithm is shown in figure 13. The folder where the source code is is accessed, and the following instruction

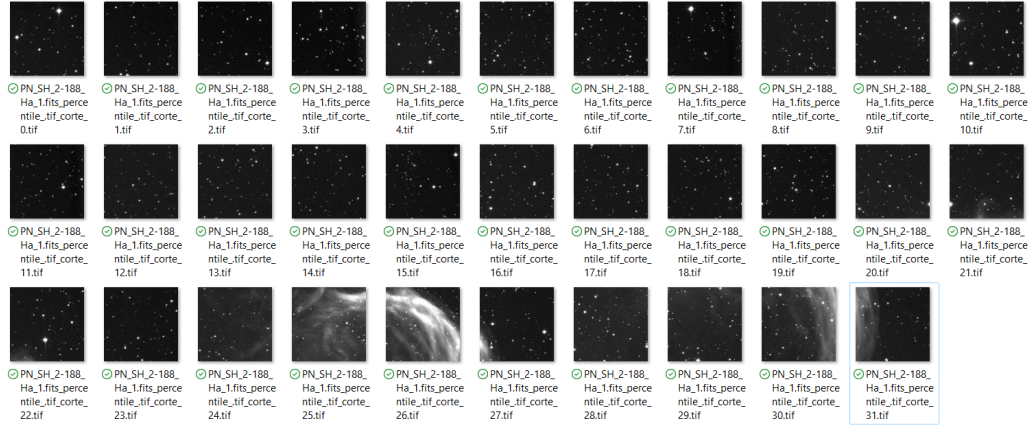


Figure 12: List of files in the folder intended to store the results of the cropping performed on the image, which generates several images of size 512 x 512 pixels.

```
C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_tiff -r resultados -p
-pf
Preprocess...
PN_NGC_7048_Ha_1.fits_percentile_.tif
PN_NGC_7354_1_Ha_1.fits_percentile_.tif
PN_SH_1-89_Ha_4.fits_percentile_.tif
PN_SH_2-188_Ha_1.fits_percentile_.tif
PN_SH_2-71_Ha_2.fits_percentile_.tif
```

Figure 13: Executing the tool in the Windows command prompt, in the example the PFCM algorithm and background removal are executed.

was executed: `python principal.py`, we provide the input directory with `-d` and output with `-r`, in this case, the `imagenes_tiff` and `resultados` folders were specified respectively, and because we want to run the PFCM algorithm and remove the background from the image, we require the `-p` argument followed by `-pf`; Therefore, the instruction would be as follows:

`python principal.py -d imagenes_tiff -r resultados -p -pf`

After executing this command, the results are saved in the results folder. In figure 14, the files in the results directory are shown, the tool will generate a tif image, an image with the result obtained after applying the PFCM algorithm, in said result only a binary image will be shown where Only the background will be shown in black and the objects in white, and said image serves as a mask, to generate another image that eliminates the background and keeps the objects with the original dynamic range.



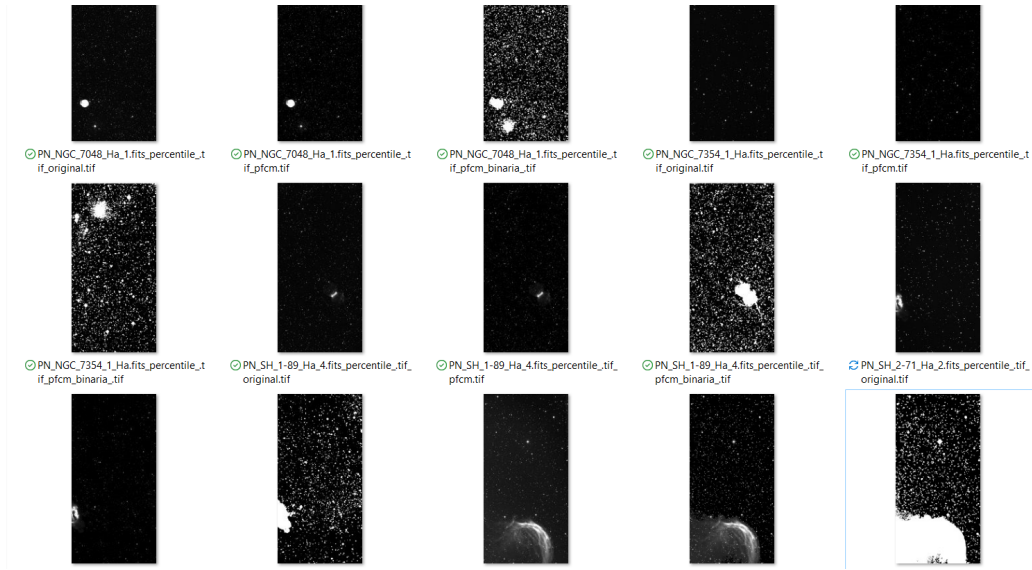


Figure 14: List of files in the folder intended to store the results of the PFCM algorithm and background removal.

## 5.4 CCD error reduction

For this example, the original images in fits format were added to the `imagenes_fits` folder. In this case, the `-ccde` instruction was first executed followed by `-pr` so that at the end of the error reduction a new image with fit would be generated. contrast and the result can be seen. Another folder called `resultados_errores_ccd` was created to store the result of the CCD error reduction run.

Figure 15 shows the example of executing the `-ccde` instruction to reduce CDD errors. The folder where the source code is is accessed, and the following instruction was executed: `python principal.py`, we provide the input directory with `-d` and output with `-r`, in this case, the folders `imagenes_fits` and results were specified respectively, and because we want to execute the close morphological operator to reduce CCD errors, we require the `-ccde` argument followed by `-pr`; Therefore, the instruction would be as follows:

```
python principal.py -d imagenes_fits -r resultados_errores_ccd -ccde -pr
```

After executing this command, the results are saved in the results folder. In figure 16, the files in the results directory are shown, the tool will generate several tif images: The first image is the original, the second image is the

```

C:\Users\eduar\OneDrive\Documentos\tesis\AstroIMP>python principal.py -d imagenes_tiff_errores_ccd -r re
sultados -p -ccde -pr
Preprocess...
HII_LBN_188.69+04.25_Ha_3.fits
HII_LBN_188.69+04.25_Ha_3.fits
HII_NGC_7000_2_1_Ha.fits
HII_RFS_546_Ha.fits
HII_SH_2-241_Ha_3.fits
HII_SH_2_131_Ha_2.fits
Contrast enhancement with percentile range...
HII_NGC_7000_2_1_Ha.fits
HII_LBN_188.69+04.25_Ha_3.fits
HII_NGC_7000_2_1_Ha.fits
HII_RFS_546_Ha.fits
HII_SH_2-241_Ha_3.fits
HII_SH_2_131_Ha_2.fits
Contrast enhancement with percentile range...
HII_RFS_546_Ha.fits
HII_LBN_188.69+04.25_Ha_3.fits
HII_NGC_7000_2_1_Ha.fits
HII_RFS_546_Ha.fits
HII_SH_2-241_Ha_3.fits
HII_SH_2_131_Ha_2.fits
Contrast enhancement with percentile range...
HII_SH_2-241_Ha_3.fits
HII_LBN_188.69+04.25_Ha_3.fits
HII_NGC_7000_2_1_Ha.fits
HII_RFS_546_Ha.fits
HII_SH_2-241_Ha_3.fits
HII_SH_2_131_Ha_2.fits
Contrast enhancement with percentile range...

```

Figure 15: Executing the tool in the Windows command prompt, in the example the `-ccde` instruction is executed to reduce CCD errors.

result obtained after inverting the original image, the third image is the one where the original image is subtracted from the inverted one, the fourth image is after applying the morphological operator close, the fifth image is after inverting the previous result, the sixth image is after subtracting the result with the original image and the last image is the result of returning to the original dynamic range.

## 6 Conclusion

Image processing in astronomy is complex, requiring several steps to be carried out beforehand to ensure that noise, image background, and stars do not interfere with the detection of extended objects. A traditional image segmentation method alone cannot be applied since the quality of the images is not the same as a traditional image.

Given the diffuse nature of the emission nebulae, and the noise produced by the *CCD*, it was best to apply the Perona Malik filter and the fuzzy possibilistic algorithm (*PFCM*) to obtain a threshold and be able to remove

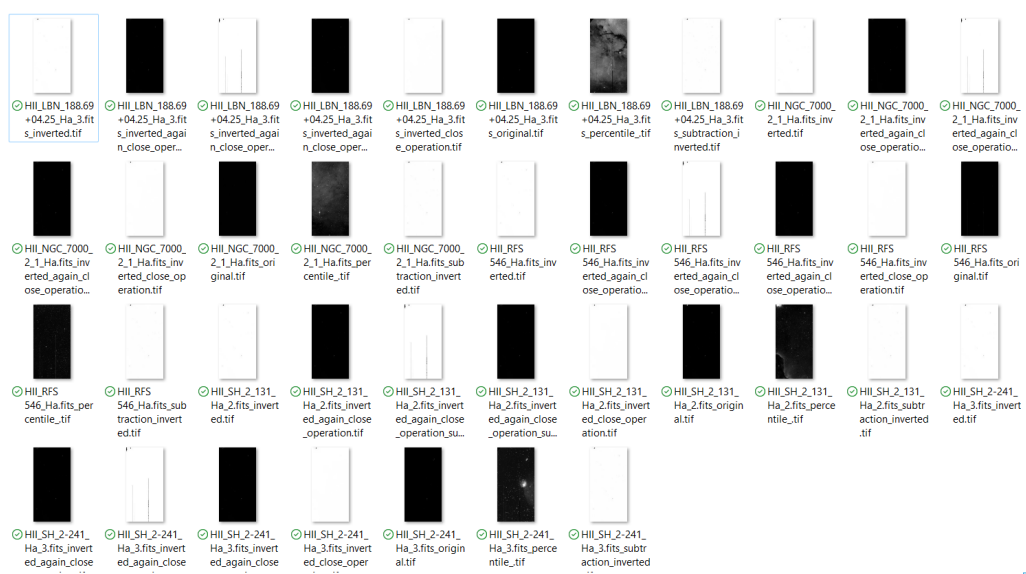


Figure 16: List of files in the folder intended to store the results of the application of morphological operators to reduce CCD errors.

the background of the image, leaving only stars and extended objects.

This tool allows you to eliminate the background of the image, leaving only the objects of interest. This process is essential to be able to apply image segmentation techniques and to ensure that the image does not have noise that could affect the automated detection of extended objects.