COSMIC RAY DETECTION WITH A WEBCAM

Naimish Mani B
President, Stellaria SASTRA.

PRE REQUISITES

OpenCV Processing

1. https://www.youtube.com/watch?v=1XTqE7LFQjI
2. https://www.youtube.com/watch?v=Jvf5y21ZqtQ

Depth First Search

1. https://www.youtube.com/watch?v=PMMc4VsIacU
2. https://www.youtube.com/watch?v=__98uL6wst8

Working with JSON

1. https://www.youtube.com/watch?v=iiADhChRriM
2. https://www.youtube.com/watch?v=aP2KJoTITO0

TABLE OF CONTENTS

01 INTRODUCTION

- Cosmic Rays
- Methods of Detection
- CMOS Detection Working Principle

02 HARDWARE

- Material Required
- Assembly

03 SOFTWARE

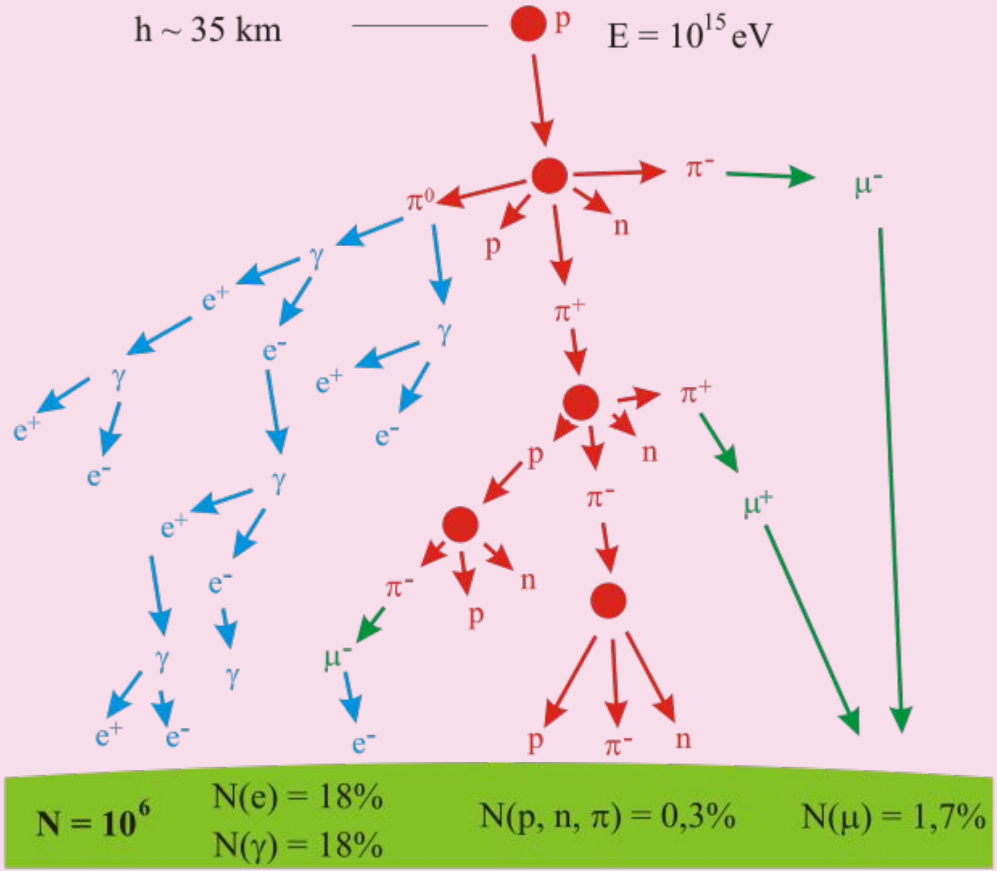
- Calibration Sequence
- Island Calculation
- Storing the Results

04 RESULTS

- Sample Outputs
- Sources of Error
- Future Work

INTRODUCTION

- Cosmic rays are high-energy protons and atomic nuclei that move through space at nearly the speed of light.
- They originate from the Sun, from outside of the solar system in our own galaxy, and from distant galaxies.
- Upon impact with the Earth's atmosphere, they produce showers of secondary particles, some of which reach the surface.



(https://en.wikipedia.org/wiki/File:AirShower.svg)

METHODS OF DETECTION

1. DIRECT

- Direct detection is possible by particle detectors at the ISS, on satellites, or high-altitude balloons.
- There are constraints in weight and size limiting the choices of detectors.

Further Reading :

1. https://books.google.co.in/books?id=yfTBvben3GoC&pg=PR7
2. http://www.ung.si/~sstanic/teaching/APP/P_Ghia_CosmicRayDetection_I.pdf

(https://www.cosmicray.umd.edu/iss-cream/)

METHODS OF DETECTION

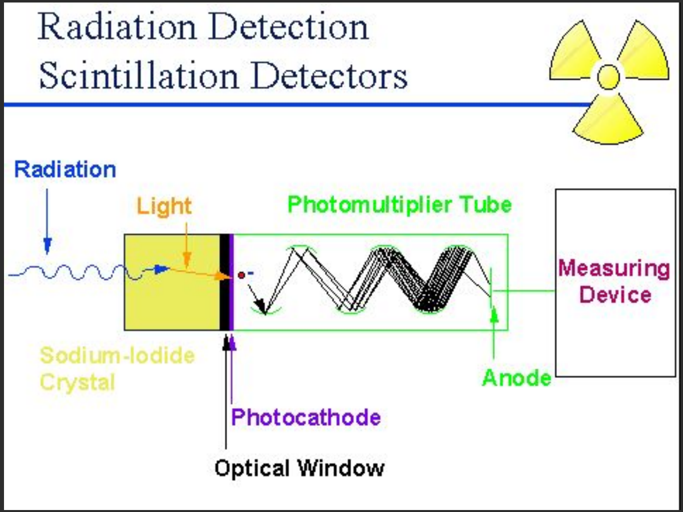
2. INDIRECT

Broadly speaking, can be classified into two categories:

A. **Detection of secondary particles forming extensive air showers (EAS).**
Eg.: Plastic Scintillators, Water/Ice-Cherenkov detectors, Cloud Chambers, CMOS Sensors.

B. **Detection of electromagnetic radiation emitted by EAS in the atmosphere.**
Eg.: Cherenkov telescope, Nitrogen Fluorescence

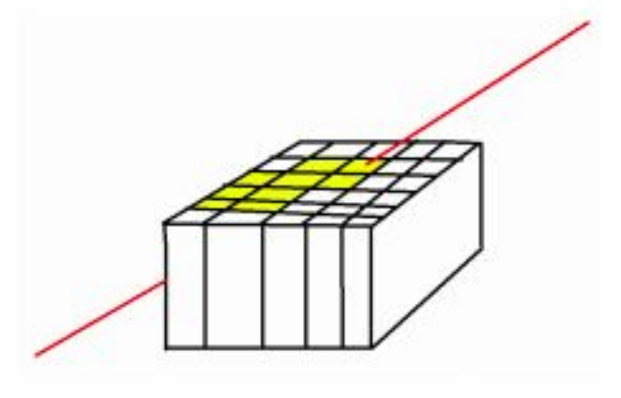
Further Reading:
https://www.fisgeo.unipg.it/~fiandrin/didattica_fisica/cosmic_rays1819/cosmic_ray_lez19_141218_IndirectDetection-2.pdf



(https://www.ntanet.net/how-do-sodium-iodide-scintillation-detectors-work)

Working Principle of CMOS Detection

- Cosmic rays, on collision with the atmosphere, produce secondary particles (pions, muons, etc.).
- These particles, when incident on a CMOS imaging sensor, produce bias.
- Take a CMOS sensor and place it in darkness (i.e., no incident light from the visible spectrum). Then, if it still gets excited, we can assume that this excitation must have been produced due to the collision of a secondary particle with the sensor. (Proof by Exhaustion)
- Note : The proof only holds true when sensor noise is rejected, through calibration.



Here, Red line indicates the path of the cosmic ray. Yellow indicates pixels triggered by the cosmic ray.

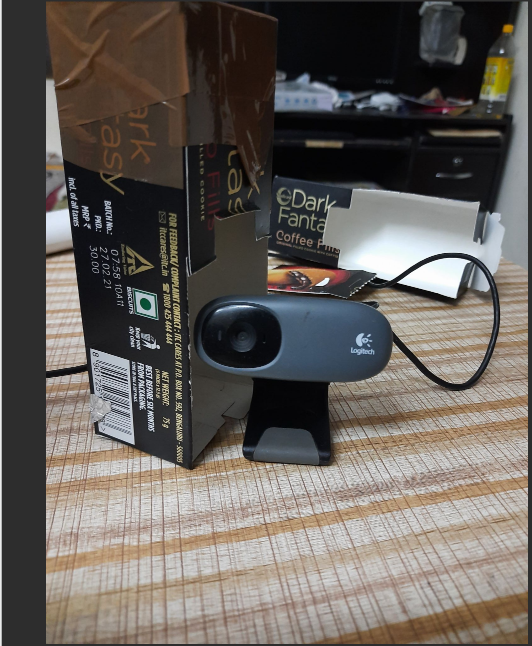
(https://via.library.depaul.edu/cgi/viewcontent.cgi?&httpsredir=1&article=1021&context=ahac)

HARDWARE

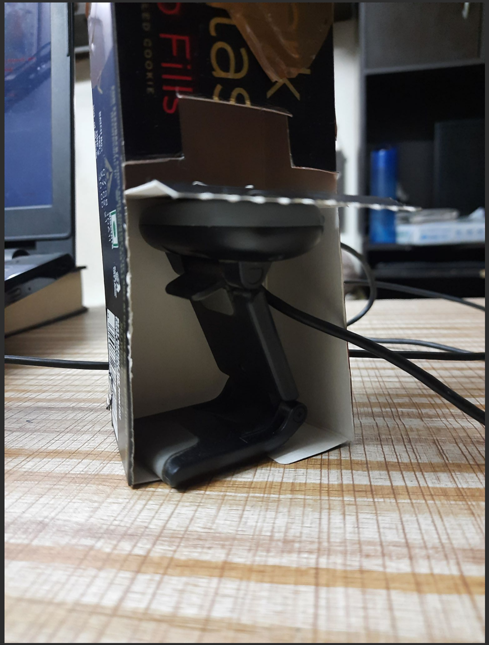
Material Required :

- Cardboard Box (2 x Sunfeast Dark Fantasy Boxes)
- USB Webcam (1 x Logitech C110)
- Tape (As required)
- Scissors

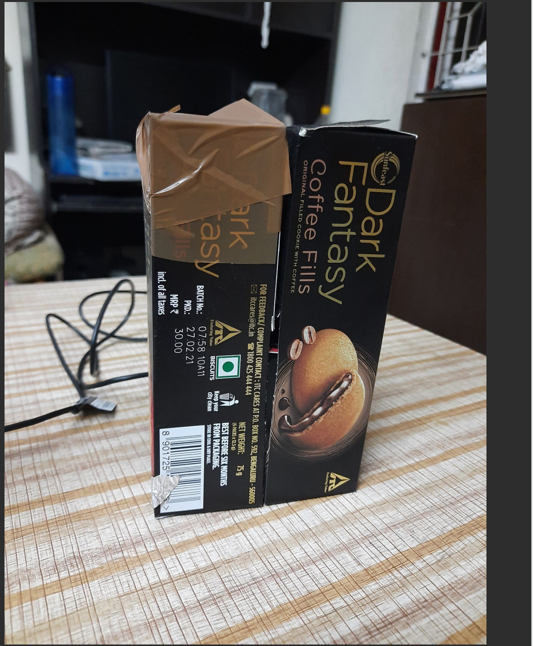
ASSEMBLY



(1)



(2)



(3)

NOTES

- The webcam has an “on” light, which cannot be removed without altering its inner circuitry. Hence, (as seen in image 2) we cover the webcam with another piece of cardboard to prevent the leaking of light.
- Despite this, light still leaked onto the sensor. Hence, I cropped out the section of the image which had the light on it. In retrospect, I could have just used tape to cover the light instead.
- The webcam has the optical element in only one half. Hence, only one of the two boxes had its top wrapped up in tape (as seen in image 3).
- All image frames are converted from RGB to B/W first, before further processing takes place.

SOFTWARE

(Python 3, OpenCV, TinyDB, NumPy, MatPlotLib)

https://github.com/Naimish240/WebCam-Cosmic-Rays

1. CALIBRATION SEQUENCE

Step 1: Average Noise

- Get the video feed from the webcam, and note the noise at each pixel for each frame.
- Add all the noises observed at each pixel across frames, to get the total noise at each pixel. Divide this value by the number of frames to get the average noise at each pixel.

Note :

A secondary emission might hit the sensor during calibration. Hence, we take the average noise across pixel instead of its maximum value during this step, to mitigate this.

Step 2: Filtration

- Display the maximum average pixel noise from the prior step.
- User selects threshold above this maximum average pixel noise for the high-pass filter.

2. ISLAND CALCULATIONS

Step 1: Image Pre Processing

- Converts all pixels from shades of gray to a binary image.
- I.e., if a pixel has a non-zero value, then it is set to ‘1’. Else, ‘0’.

Step 2: Find Islands

- Islands are defined as pixels with value ‘1’ in the sea of ‘0’s.
- The number of islands and their sizes are noted down, through DFS.
- Only islands of size greater than 5 taken to be a “true positive” here, because of hot pixels.

Note :

The island size of ‘5’ was chosen as the cut-off area to prevent one-off very bright hot pixels from giving us false positives. It can be adjusted as required.

3. STORING THE RESULTS

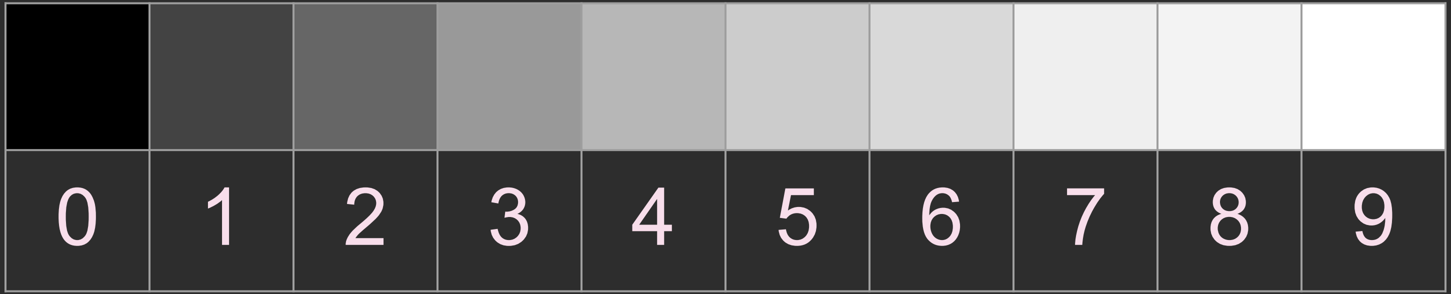
Step 1: Save the Images

- Runs only if island detected.
- Saves the Raw image (before calibration), Processed image (after calibration) and the binary image to disk.

Step 2: Add to DB

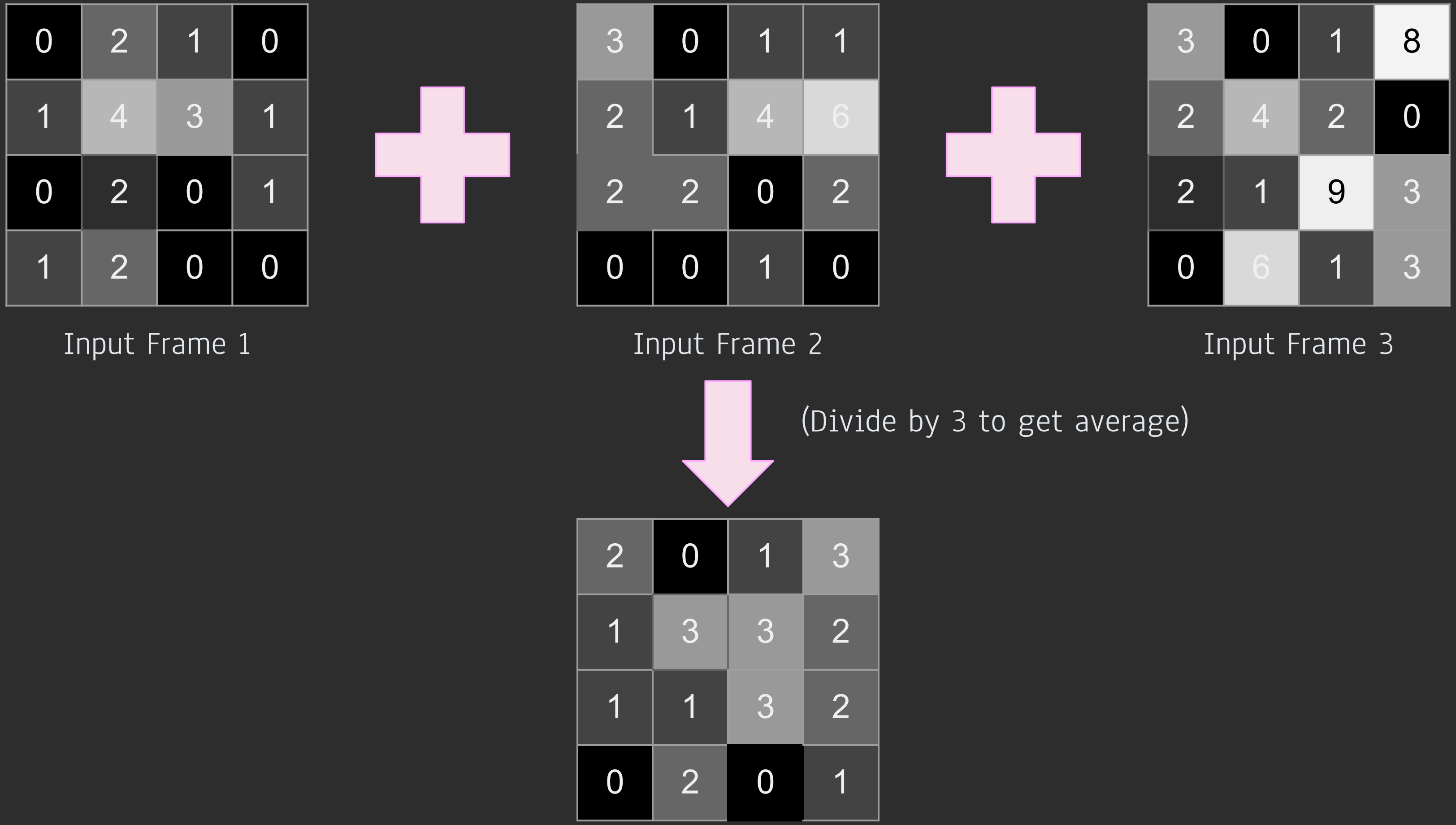
- Runs only after the images have been saved to disk.
- Inserts the information into a NoSQL database (TinyDB) locally.

EXAMPLE CASE



0	1	2	3	4	5	6	7	8	9

(KEY)



0	2	1	0
1	4	3	1
0	2	0	1
1	2	0	0

Input Frame 1

3	0	1	1
2	1	4	6
2	2	0	2
0	0	1	0

Input Frame 2

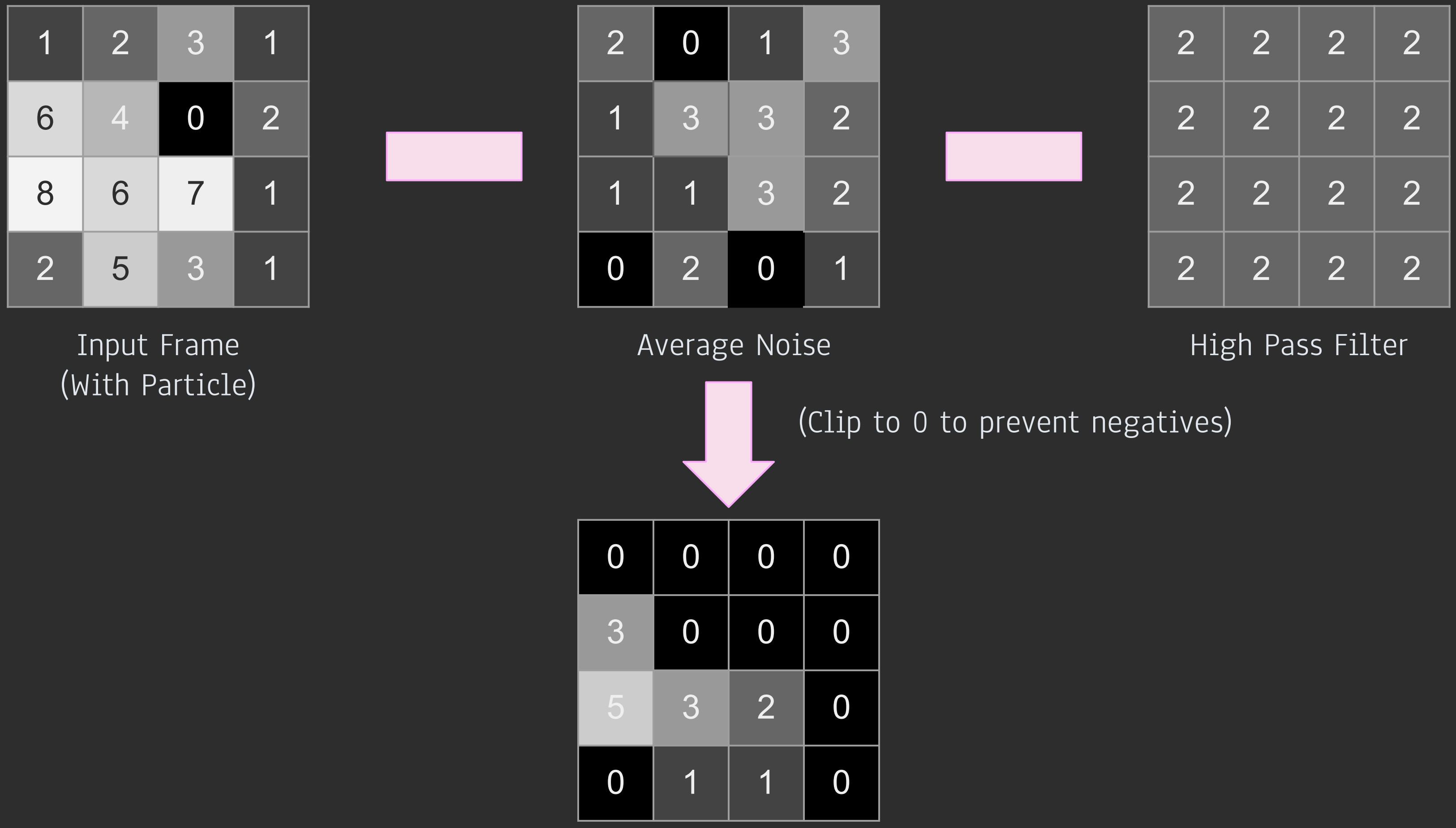
3	0	1	8
2	4	2	0
2	1	9	3
0	6	1	3

Input Frame 3

(Divide by 3 to get average)

2	0	1	3
1	3	3	2
1	1	3	2
0	2	0	1

Average Noise



1	2	3	1
6	4	0	2
8	6	7	1
2	5	3	1

Input Frame
(With Particle)

2	0	1	3
1	3	3	2
1	1	3	2
0	2	0	1

Average Noise

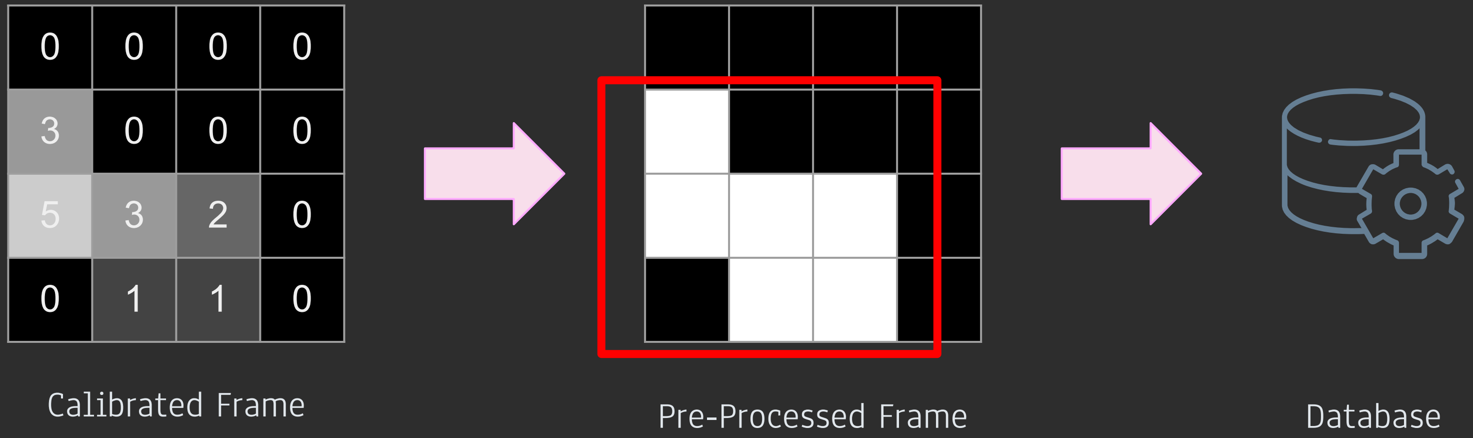
2	2	2	2
2	2	2	2
2	2	2	2
2	2	2	2

High Pass Filter

(Clip to 0 to prevent negatives)

0	0	0	0
3	0	0	0
5	3	2	0
0	1	1	0

Calibrated Frame
]



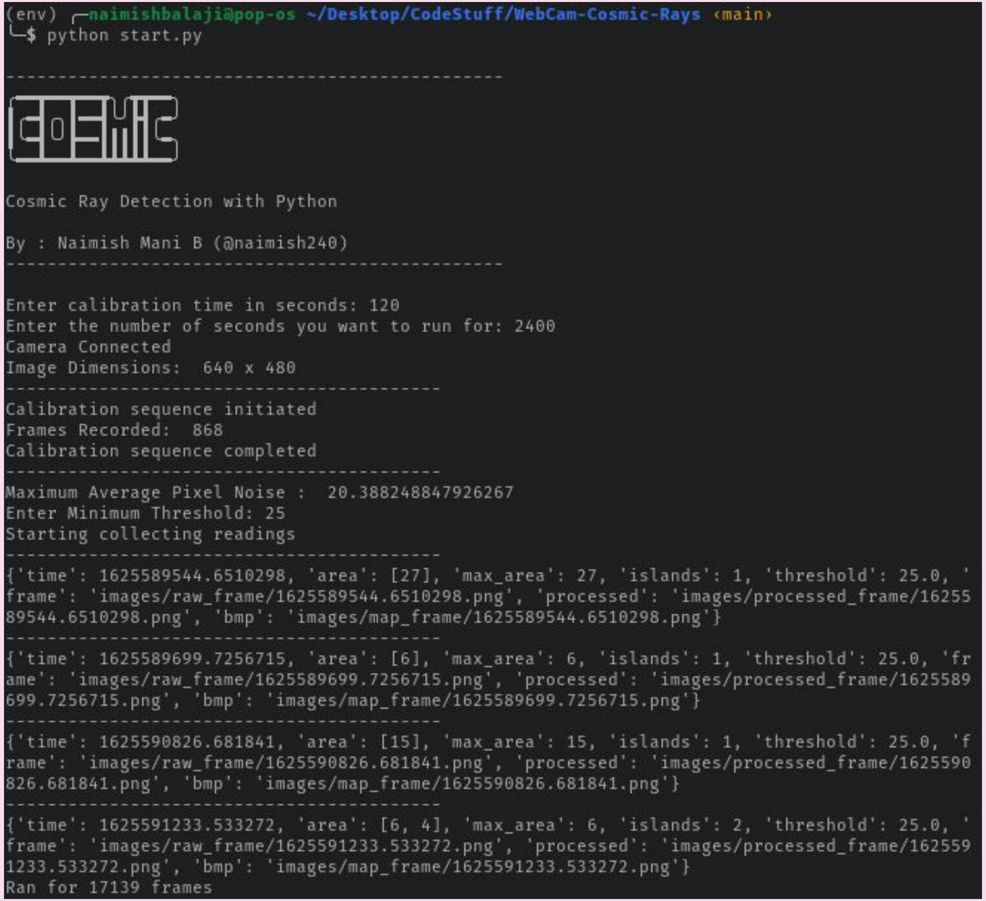
0	0	0	0
3	0	0	0
5	3	2	0
0	1	1	0

Calibrated Frame

Pre-Processed Frame

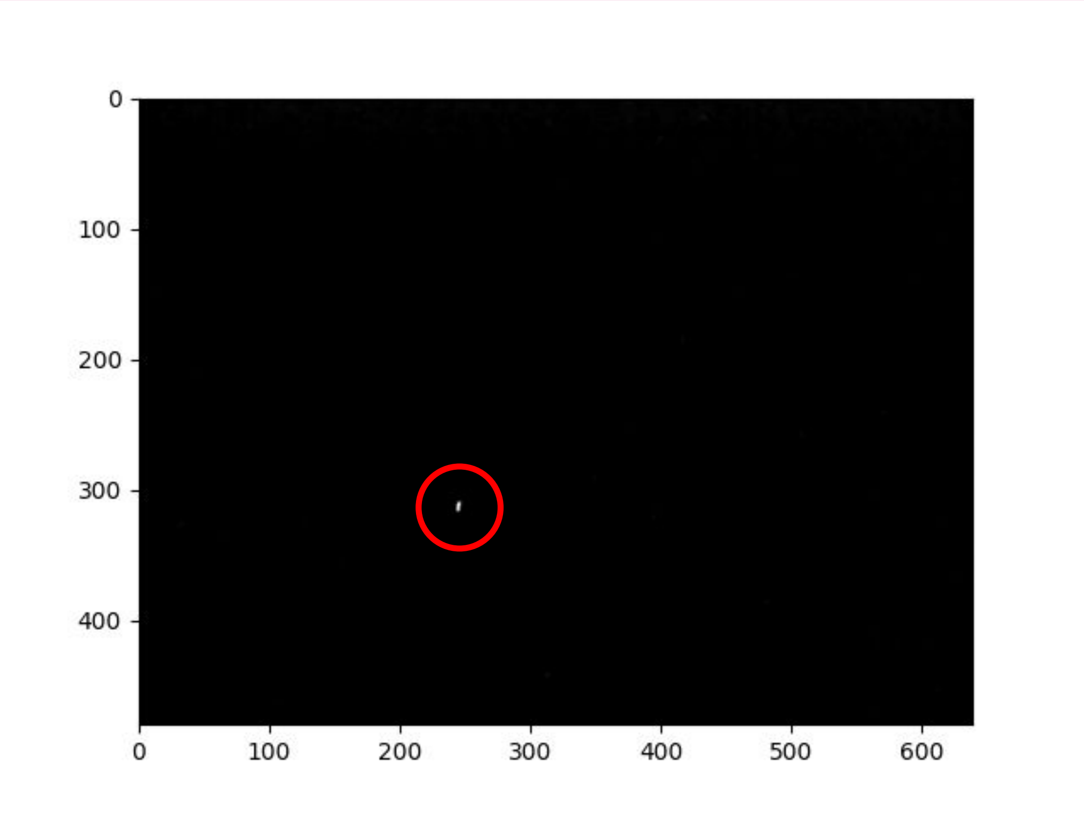
Database

RESULTS



```
(env) ┌─naimishbalaji@pop-os ~/Desktop/CodeStuff/WebCam-Cosmic-Rays ‹main›
└─$ python start.py

------------------------------------------------

Cosmic Ray Detection with Python

By : Naimish Mani B (@naimish240)
------------------------------------------------

Enter calibration time in seconds: 120
Enter the number of seconds you want to run for: 2400
Camera Connected
Image Dimensions:  640 x 480
------------------------------------------
Calibration sequence initiated
Frames Recorded:  868
Calibration sequence completed
------------------------------------------
Maximum Average Pixel Noise :  20.388248847926267
Enter Minimum Threshold: 25
Starting collecting readings
------------------------------------------
{'time': 1625589544.6510298, 'area': [27], 'max_area': 27, 'islands': 1, 'threshold': 25.0, '
frame': 'images/raw_frame/1625589544.6510298.png', 'processed': 'images/processed_frame/16255
89544.6510298.png', 'bmp': 'images/map_frame/1625589544.6510298.png'}
------------------------------------------
{'time': 1625589699.7256715, 'area': [6], 'max_area': 6, 'islands': 1, 'threshold': 25.0, 'fr
ame': 'images/raw_frame/1625589699.7256715.png', 'processed': 'images/processed_frame/1625589
699.7256715.png', 'bmp': 'images/map_frame/1625589699.7256715.png'}
------------------------------------------
{'time': 1625590826.681841, 'area': [15], 'max_area': 15, 'islands': 1, 'threshold': 25.0, 'f
rame': 'images/raw_frame/1625590826.681841.png', 'processed': 'images/processed_frame/1625590
826.681841.png', 'bmp': 'images/map_frame/1625590826.681841.png'}
------------------------------------------
{'time': 1625591233.533272, 'area': [6, 4], 'max_area': 6, 'islands': 2, 'threshold': 25.0, '
frame': 'images/raw_frame/1625591233.533272.png', 'processed': 'images/processed_frame/162559
1233.533272.png', 'bmp': 'images/map_frame/1625591233.533272.png'}
Ran for 17139 frames
```

RESULTS



```
{'time': 1625589544.6510298, 'area': [27], 'max_area': 27, 'islands': 1, 'threshold': 25.0}
```

SOURCES OF ERROR

IMAGE NOISE	HOT PIXELS
● Random variation of brightness in images. ● Produced by the image sensor and circuitry of a digital camera. ● Mitigated through software based high-pass filters.	● Individual pixels which look much brighter than they should. ● Caused by electrical charges that leak into the sensor, when it is hot. ● Mitigated by selecting islands only with area greater than 5px.

FUTURE WORK

CALIBRATION SEQUENCE

- Handle Dead Pixels
- Frame based instead of timer based

APP DEPLOYMENT

- Build Android and iOS apps for edge detection (like CREDO).

SERVER SETUP

- Enable multiple webcams to share results.
- Verify Special Relativity.

THANKS

Do you have any questions?

naimish240@gmail.com

https://www.linkedin.com/in/naimish-balaji-a6182b180/