Iranian Journal of Physics Research, Vol. 25, No. 3, 2025
DOI: 10.47176/ijpr.25.3.71936

Raspberry Pi CMOS image sensor's response to X-rays energy and intensity change

Gholamreza Fardipour Raki[1], Mohsen Khakzad[1], Shehu AbdusSalam[2], Milad Daneshnazar[3]
[1]School of particles and accelerator, Institute for Research in fundamental sciences (IPM), Tehran, Iran
[2] Department of physics, Shahid Beheshti University, Tehran, Iran
[3] Electrical Engineering and Computer Science Department, The University of Texas at Dallas, Richardson, TX, USA

E-mail: fardipour@ipm.ir

(Received 09 July 2024; in final form 17 November 2024)

Abstract
The Raspberry Pi is renowned for its compact design and essential features, coupled with a free Linux operating system, making it an ideal platform for a myriad of scientific and research endeavors. Particularly noteworthy are the official CMOS cameras tailored for Raspberry Pi, offering unique capabilities tailored for image data analysis purposes. The Raspberry Pi's Linux operating system (Raspbian) boasts crucial attributes like binary image output and adjustable shutter speed, rendering it highly conducive for scientific inquiries exploring the impact of particles and rays on CMOS sensors. Indeed, it serves as a valuable tool for conducting X-ray studies. Leveraging Python on Raspberry Pi enables the execution of camera operations, facilitating the generation of RAW image files that capture the effects of X-rays. By capturing RAW images with a shutter speed set to three seconds and conducting subsequent analysis on the RAW image files, one can derive insightful data, including the spot count diagram, the average area of spots diagram, and the average numerical value stored in spots diagram. Through this experimental approach, the distinctive effects of varying energy and intensity levels can be effectively discerned and elucidated.

Keywords: Half Heusler; Phase transition; dielectic function; Semiconductor; Bulk modulus; Band gap

1. Introduction

Raspberry Pi technology is extensively utilized across educational, industrial, and scientific domains for a multitude of applications. It encompasses micro-controllers and ARM architecture-based computers, and comes with a free operating system called Raspberry Pi OS, which is based on Debian-Linux and optimized for the hardware. The OS boasts over 35,000 pre-compiled software packages [1]. Raspberry Pi manufactures camera boards that are compatible with the Sony imx219 (version 2 camera), the Sony imx477, and an older version 1 camera board based on the Omnivision OV5647. These Bayer sensors capture "raw" Bayer images, which have not undergone any processing. The raw pixels are then transmitted back to the system-on-chip in separate image frames [2]. Image sensors integrated into circuits present a promising option for developing low-power, compact detectors with superior spatial resolution compared to conventional detectors. This makes them particularly suitable for ionizing particle detection [3].

In [4], the examination investigated the response of two distinct Commercial Off-the-Shelf CMOS image sensors when used as particle detectors. The sensors were subjected to irradiation using X-ray photons, gamma photons, beta particles, and alpha particles from various sources. An algorithm was employed to compare and assess the amount of charge generated by different particles and the size of the spot recorded on the sensor for classification purposes. Observable effects of these particles on the CMOS image sensor were evident. In [5], the electronic circuit integrates a CMOS sensor with an area of 640x480 pixels for the purpose of detecting ionizing radiation. The sensor was subjected to alpha particles, beta particles, and gamma photons. The results indicate that even after prolonged exposure (168 hours) to irradiation, the sensor maintained full functionality. Additionally, accurate detection of the energy from charged particles and photons was achieved.

The effects of radiation on different types of CMOS image sensors due to X-rays are discussed in [6], indicating that the composition and design of CMOS cells significantly impact their sensitivity to X-rays. Consequently, some CMOS cameras exhibit high sensitivity to X-rays, while others do not.

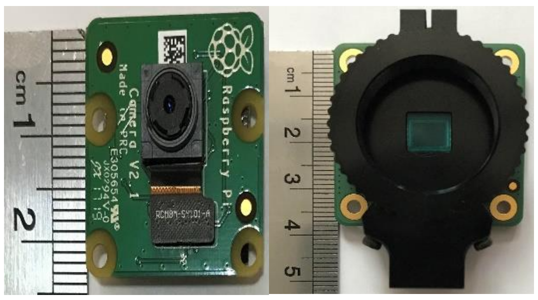
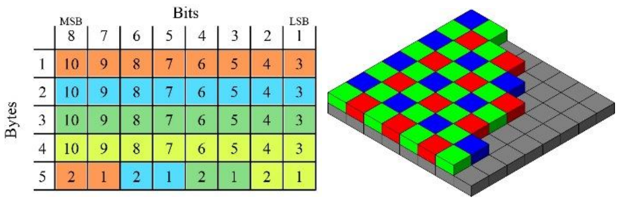


Figure 1. Official Raspberry Pi camera modules: (Left) 5-mega-pixel, (Right) 12.3-mega-pixel.



Bytes	8	7	6	5	4	3	2	1
1	10	9	8	7	6	5	4	3
2	10	9	8	7	6	5	4	3
3	10	9	8	7	6	5	4	3
4	10	9	8	7	6	5	4	3
5	2	1	2	1	2	1	2	1

Figure 2. (Left) Bayer bytes format. (Right) Image CMOS sensor pattern of pixels.

This sensitivity is comparable to the detection of more energetic particles such as gamma photons, electrons, and alpha particles using CMOS technology.

The Raspberry Pi camera is capable of capturing images in low-light environments, making it particularly well-suited for identifying X-ray and gamma ray activity from sources with low activity levels by allowing the image sensor additional time to gather input light or particle impacts.

In this article, we present the results of an experiment assessing the response of the CMOS sensors in Raspberry Pi cameras to changes in X-ray energy and intensity, using the RAW [7] images recorded by the sensors.

2. Raspberry Pi Camera

Several official camera modules are now available for the Raspberry Pi. The initial 5-megapixel model was launched in 2013, followed by an 8-megapixel Camera Module 2 in 2016, and a 12.3-megapixel Camera Module 3 in 2023. These cameras can be best utilized through dedicated applications provided such as libcamera-still or libcamera-vid software packages [8,9]. Figure1 shows two types of camera modules.

A recent software library called libcamera has been developed to provide direct support for advanced camera systems within the Linux operating system. This open-source project is designed to drive the Raspberry Pi's camera system through open-source code on ARM processors, effectively reducing reliance on proprietary code running on the Broadcom GPU, to which users do not have access. The libcamera library offers a C++ interface for applications, providing functionality for configuring the camera and enabling applications to request image frames. These image buffers are stored in system memory and can be directly used by still image encoders (e.g., JPEG) or video encoders (e.g., h.264). The source code for the libcamera-apps is freely available at [10].

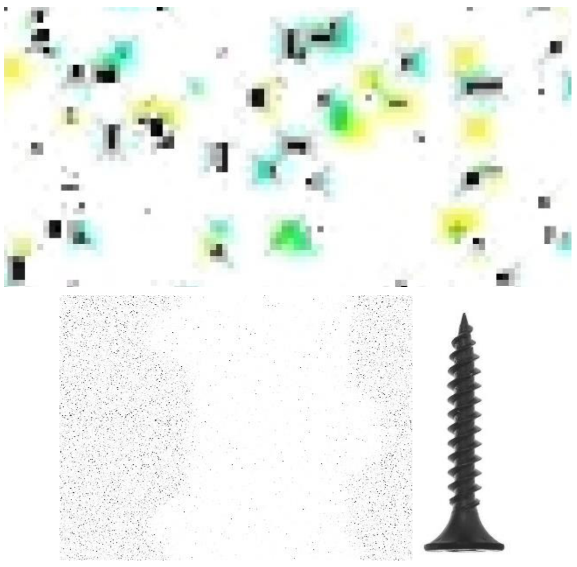


Figure 3. (Left) A small cut of the negative image file took by shutter speed of 3 seconds by 22 keV X-ray with the maximum intensity coefficient equal to 8. (Right) A photo of x-ray shadow of a screw and the screw itself. Both of these two photos were recorded by the 12.3-megapixel camera.

When using a Raspberry Pi OS, the standard libcamera-apps come pre-installed, ensuring that official Raspberry Pi cameras are automatically recognized [1,8].

Another potential library for controlling Raspberry Pi cameras is Picamera2. This Python library offers a convenient way to access the Raspberry Pi camera system, specifically designed for use with cameras connected via flat ribbon cable directly to the connector on the Raspberry Pi itself. Although it primarily supports this type of connection, it also offers limited compatibility with USB cameras.

Picamera2 is built on the open-source libcamera project, which supports intricate camera setups in Linux. Picamera2 is better suited for Raspberry Pi applications compared to libcamera's native bindings. It is specifically designed to optimize the capabilities of the Raspberry Pi's integrated camera and imaging hardware, serving as a replacement for the older PiCamera Python library. While the functionalities offered are largely similar, they are presented in a different manner. Picamera2 provides a more precise and direct insight into the Pi's camera system, streamlining its use for Python applications [9].

The data captured by CMOS image sensors and directly stored in memory can also be accessed by applications as RAW image files or Bayer data. These files contain comprehensive and uncompressed information about the image. A RAW format file preserves the unaltered data from the camera sensor, allowing for greater flexibility in post-processing and image editing. One issue with RAW image files is the absence of a standard file format. Major camera manufacturers each utilize their own unique RAW image format, creating potential challenges for collaboration or file sharing. One widely used universal RAW image file format is .dng, introduced by Adobe in 2003 to establish a standardized RAW file for industry-wide use. While it has not completely resolved the issue, it is now the most widely used RAW file format for both Raspberry Pi camera libraries and other applications [7].

Bayer data is consistently maintained at full resolution, irrespective of the camera's output resolution and any resizing parameter. In the V1 module, this data occupies the final 6,404,096 bytes of the output file, while in the V2 module it occupies the last 10,270,208 bytes. The initial 32,768 bytes represent header data. Bayer data comprises 10-bit values due to its compatibility with the OV5647 and IMX219 sensors used in Pi's camera modules. The 10-bit data is structured into four sets of 8-bit values, and then the least significant 2 bits of these four values are combined into a fifth byte.

To create an image that appears more typical from RAW Bayer data, demosaicing is essential, along with possibly applying some type of color correction [11]. Figure 2 (left) shows the structure of a RAW file data of Bayer image, while figure 2 (right) shows the pattern structure of the CMOS image sensor. Bitmap image files have a format similar to the CMOS pattern. If all pixels in a Bitmap image file are zero and there is no pixel affected, the file size remains fixed based on resolution. Converting a Bayer file to Bitmap can be beneficial because analysis is more straightforward with a fixed pattern. Image file data typically does not change during conversion using software like Windows Paint.

Raspberry Pi cameras lack a physical shutter, making it impossible to prevent light from reaching the CMOS cells. The sensor allows only two operations: resetting or reading a row of cells. By adjusting the delay between resetting and reading a line, we can regulate the exposure time for each frame. To prolong the exposure duration, it is necessary to minimize the number of frames captured per second by using a very low frame rate.

As a result, the maximum exposure time is influenced by the camera's minimum frame rate, which largely depends on how slowly the sensor can read lines. This hardware-level capability relates to factors such as the register size for holding line read-out times. In the Picamera2 module, the exposure-speed parameter indicates the duration of exposure for the most recent processed frame. This value is essentially a multiple of the sensor's line read-out time and is constrained by the camera's frame rate [9].

All libcamera-app allow the user to run the camera with fixed shutter speed. The command below in Raspberry OS terminal would capture a Bayer image with long exposure (shutter open duration time) of 3 seconds [8]:

```
libcamera-still -r -o test.dng --shutter 3000000 --gain 1 --
awbgains 1,1 –immediate
```

In this command:

-r specifies that the output image will be in raw Bayer format.

-o test.dng indicates the output file name and format (in this case, a .dng file).

--shutter 3000000 sets the shutter open duration time to 3 seconds (3000000 microseconds).

--gain 1 and --awbgains 1,1 adjust the gain and white balance gains, respectively.

--immediate initiates the capture immediately after parameter configuration.

Energetic particles and different photon wavelengths affect the CMOS sensor cells differently in terms of sensitivity and lifespan. Adjusting the shutter open time is important for optimizing detection levels during specific experiments.

3. X-ray effect on 5.0- and 12.3-megapixels cameras

We utilized the official Raspberry Pi cameras, including 5-megapixel and 12.3-megapixel models, positioned in front of an X-ray source to capture Bayer image files with a fixed shutter speed of 3 seconds. Additionally, we conducted experiments with varying shutter speeds of 1 ms, 100 ms, 1 s, and 10 s. However, the results obtained with shorter exposure times were less consistent compared to those obtained with a 3-second exposure. Conversely, results obtained with a longer exposure time of 10 seconds did not yield better outcomes due to decay of spots on the CMOS cells, emphasizing the optimal performance achieved with the 3-second exposure duration.

Furthermore, we systematically increased the energy of the X-rays from 0 keV to 22 keV across five increments, while also adjusting the radiation intensity at each step. As anticipated, elevating both the quantity and energy of photons effectively intensified the observable effects recorded in the RAW image files. Consequently, there was a proportional augmentation in the volume of data requiring storage within the file as X-ray effects within the RAW image file increased.

For visual reference, figure 3 presents a negative cut example depicting the effects of X-rays captured by the 12.3-megapixel camera, alongside a photograph showcasing the X-ray shadow of a screw. These visual representations offer insight into the observed phenomena resulting from our experimental setup and parameter variations.

We conducted a Python-based analysis on the RAW image files to detect and enumerate spots affected by X-ray radiation, assigning each spot a unique identifier. These spots represent clusters of adjacent pixels impacted by X-rays. To accurately differentiate X-ray-affected spots from the background and enhance the analysis accuracy, we implemented a thresholding mechanism. This involved setting a minimum pixel value, referred to as a trigger, to classify pixels as affected by X-rays. For the 12.3-megapixel camera, we established a trigger value of 150. Notably, each pixel in the RAW image has a recorded value ranging between 0 and 765.

The spot count, the average area of spots, and the average numerical value stored in spots in the absence of X-rays, as well as at the maximum energy and intensity of the X-rays used in this experiment, are compared in Table 1. In the absence of X-rays, the values in Table 1 represent the background measurement, which is almost canceled by a suitable trigger in the analysis.

The value of uncertainty, or more precisely, the standard deviation between repeated measurements, is shown in Table 1. This uncertainty might be considered background noise; however, suitable analysis and a well-adjusted trigger can mitigate this background noise. Error bars in figure 5 to 7 also illustrate this uncertainty.

Table 1. Results from the analysis of the response of the Raspberry Pi 12.3-megapixel camera to the absence of X-rays and the maximum energy and intensity of X-rays used in this experiment.

	Absence of X-ray		22 keV, 8x intensity X-ray	
	No Trigger	With Trigger	No Trigger	With Trigger
Spot count	33.75 ± 75.71	3.00 ± 0.01	833884.75 ± 7568.44	496010.25 ± 8150.14
The average area of spots	1.25 ± 0.16 pixels	1.00 ± 0.01 pixels	6.11 ± 0.06 pixels	2.84 ± 0.02 pixels
The average numerical value stored in spots	56.90 ± 20.47	276.50 ± 17.21	826.81 ± 14.06	1000.67 ± 8.80

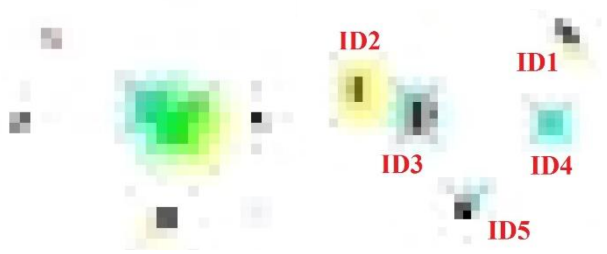


Figure 4. A cut of a negative RAW image file took by the 12.3-megapixel camera. (Left) Some spots with different sizes and values. (Right) Two connected spots (ID2,ID3) distinguished by the analysis.

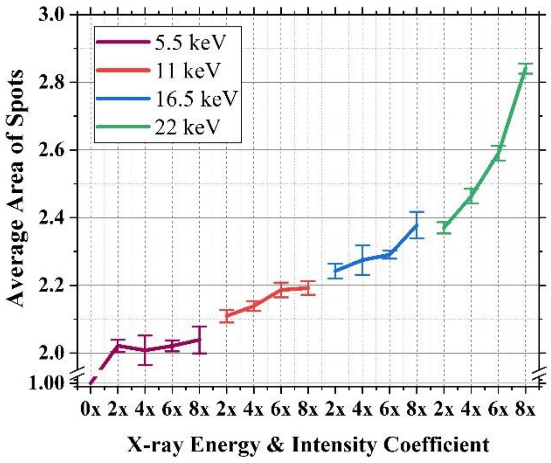


Figure 5. The average area of spot variation interval for each X-ray energy and intensity level.

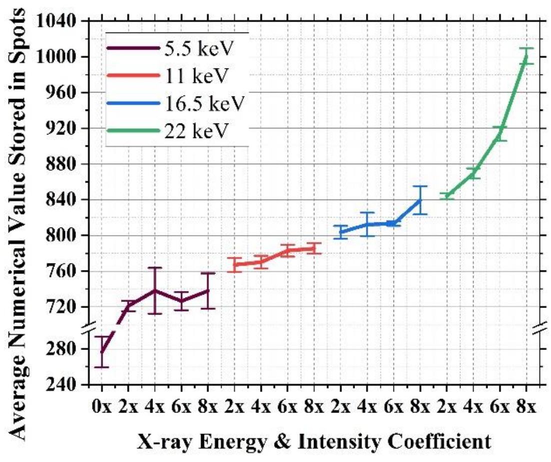


Figure 6. The average numerical value stored in spots for each X-ray energy level.

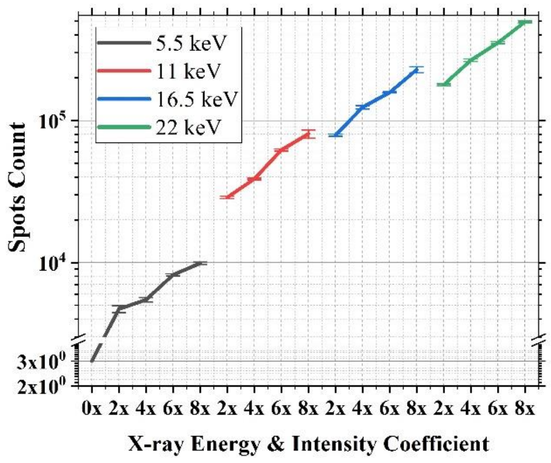


Figure 7. The spot count for each X-ray energy and intensity condition.

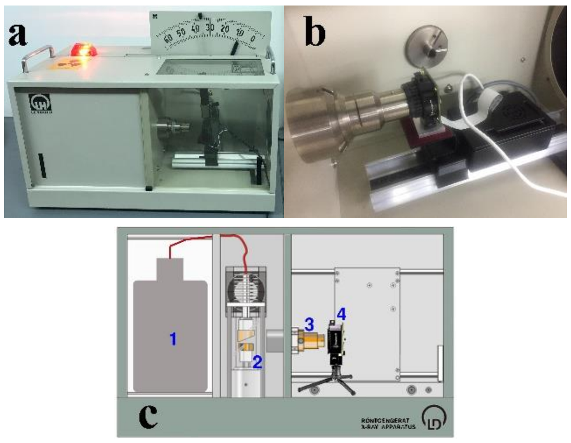


Figure 8. (a) The LEYBOLD X-ray machine. (b) Collimator and camera stand. (c) The experimental setup: 1: High-Voltage, 2: X-ray tube, 3: Collimator, 4: Raspberry Pi CMOS camera.

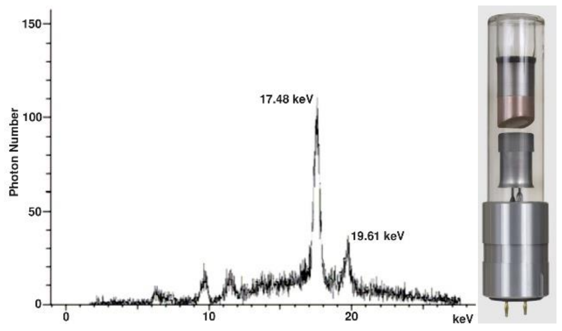


Figure 9. (Left) X-ray spectrum of the LEYBOLD Mo (554 861) molybdenum anode X-ray tube. (Right) Photograph of the X-ray tube [13] [**Error! Bookmark not defined.**].

The variation in each measurement arises from the distribution of X-ray photons' orientation and the strength of each X-ray photon's effect on CMOS pixels. This variation is more pronounced at lower intensities and lower energies.

To ensure proper spot identification (ID), we devised a priority scheme based on the pixel arrangement in the 2D pixel array. Pixels exceeding the trigger value were assigned a group ID. However, if a pixel had neighboring pixels with existing IDs, we prioritized assigning the ID of the top and left neighbors to maintain consistency in spot identification. This approach lets connected spots to receiving multiple IDs, even if they shared common boundaries.

Figure 4 (left) presents a segment of a negative RAW image file captured by the 12.3-megapixel camera, depicting various spots with diverse sizes and intensity values. In figure 4 (right), we illustrate an instance where the analysis effectively distinguished two connected spots, with each spot receiving its unique identifier. This analysis methodology enabled accurate spot detection and identification within the X-ray-affected regions of the images, facilitating comprehensive characterization of the X-ray effects on the CMOS sensor.

By assigning IDs to the pixels meeting the trigger criteria, we construct a 3D array of data, facilitating convenient processing. Each spot is represented by a unique ID in the third axis of the array, enabling spot counting and identification of the average area and the average numerical value stored in spots. While all aspects (spot count and average area of spots and average numerical value stored in spots) tend to increase with higher X-ray energy and intensity, their behavior differs. Consequently, these three aspects serve to differentiate between the effects of energy and intensity changes. In our experiments, we observed that mostly, each different radiation condition yields a unique set of values for spot count, average area and average numerical value stored in spots.

While we meticulously examined the 5-megapixel official camera, we found that the consistency of the 12.3-megapixel camera far exceeded that of its counterpart. Therefore, we solely report the results obtained using the 12.3-megapixel camera. The enhanced consistency of the second camera can be attributed to its higher pixel count and larger CMOS chip size.

The diagram depicted in figure 5 illustrates the variation interval of the average area of spots for each energy and intensity level. It's noteworthy that the trigger level and analysis structure employed in this experiment are specific to our setup. If the same methodology were applied to a different type of CMOS camera or X-ray energy and intensity intervals, the analysis would require modification to suit the new conditions. This underscores the importance of calibration. The average area of spots, and the average numerical value stored in spots are more indicative of the energy level, as changes in spot size and value are more significantly affected by variations in X-ray energy than by intensity, and do not show a very good increase in higher intensities at each energy level, especially in lower energies.

Figure 6 shows the change in the average numerical value stored in spots for each energy level, while figure 7 illustrates the spot count for each combination of energy and intensity conditions.

We explored various aspects, such as the memory size of each RAW image file, the number of affected pixels, the total value recorded in the affected pixels, the maximum-sized spot, the average area of spots, and the average numerical value stored in spots. However, none of these parameters exhibited the same level of consistency as the spot count. Together with the average area of spots, and the average numerical value stored in spots diagrams, this chart serves to distinguish the effect of each energy and intensity level in our experiment.

4. Experimental details

We used the LEYBOLD 55490 X-ray machine [12] to study the effects of X-rays on the Raspberry Pi CMOS cameras. Figure 8 shows a suitable space for placing the Raspberry Pi camera in front of the X-ray beam. To test the effect of X-rays on the Raspberry Pi camera, the collimator provides a beam with a circular cross-section with a diameter of one centimeter. We placed the camera in front of the X-ray beam.

The X-ray tube used in this experiment is the LEYBOLD Mo (554 861) [13]. This X-ray tube is a directly heated hot cathode tube, with a molybdenum anode seated in a copper block to dissipate heat. Technical data for this X-ray tube is provided in Table 2.

Table 2. The LEYBOLD Mo (554 861) X-ray tube technical data [13].

Anode material	Molybdenum K_α = 17.4 keV (71.1 pm), K_β = 19.6 keV (63.1 pm)
Max. anode voltage	35 kV
Max. emission current	1 mA
Size of focal spot approx.	2 mm^2

Figure 9 shows the X-ray spectrum of the LEYBOLD Mo (554 861) molybdenum anode X-ray tube.

To photograph objects like the one shown in figure 6, we positioned the object between the collimator and the camera, minimizing the distance between them.

5. Conclusion

The official 12.3-megapixel Raspberry Pi camera demonstrates strong performance in capturing changes in X-ray energy and intensity. By acquiring RAW images with a 3-second shutter speed and analyzing them as shown in figure 5 and 6 which display the average spots area and the average numerical value stored in spots, respectively and in figure 7's spot count diagram, we could assess the effects of energy and intensity levels in this experiment.

The average spots area and the average numerical value stored in spots are notably more affected by changes in X-ray energy than by intensity. In contrast, the spot count is almost equally sensitive to changes in both X-ray energy and intensity. Lower X-ray energy levels (here, below 11 keV) have minimal impact on CMOS image sensors, even at high intensities, while higher energy levels (here, above 16.5 keV) show a pronounced response to intensity changes. Even at low intensities, higher energy levels (such as 100 keV) can be detected due to this energy level's high efficiency on CMOS image sensors.

Using the presented analysis to distinguish spots, and calculate their area and values with a suitable trigger level, is essential for obtaining meaningful results from the extensive data provided by CMOS image sensors. While the image processing and analysis are complex, the entire procedure from image acquisition to final result takes less than two minutes, performed entirely on a Raspberry Pi 4 with a 12.3-megapixel camera using Python.

As shown in figure 7, the difference in spot count between the absence of X-rays and low-energy, low-intensity X-rays (3 vs. 4,703 spots, respectively) is striking. This method is therefore highly effective in detecting X-rays, particularly at higher energy levels.

We have presented accessible and suitable facilities and methods tailored for young scientists, enabling them to utilize a comprehensive, supported, and integrated package. This package empowers them to employ CMOS technology for detecting X-rays and other energetic particles, and perform analysis using both C and Python within a readily available Linux operating system.

6. Data Availability Statement

The data for this paper is quite large, but the data will be shared in a repository, and the access link will be shared by request from the reader.

7. Acknowledgment

G.F. Raki and M. Khakzad are grateful to the School of Particles and Accelerator at IPM for their financial support, assistance, and interest in this project.

References

1. Raspberry-Pi, Raspberry Pi Documentation, https://www.raspberrypi.com/documentation (2023).
2. Raspberry-Pi, Raspberry Pi Camera Algorithm and Tuning Guide, https://datasheets.raspberrypi.com/camera/raspberry-pi-camera-guide.pdf (2023).
3. Y Degerli, F Guilloux, and F Orsini, Journal of Instrumentation, **9** C05018 (2014).
4. M Pérez, J Lipovetzky, M Sofo Haro, I Sidelnik, J J Blostein, F Alcalde Bessia, and M G Berisso, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, **827** (2016) 171.
5. E Cruz-Zaragoza and I P López, Journal of Physics: Conference Series, **582** (2015) 012047.
6. J Tan, B Büttgen, and A Theuwissen, Proceedings of the 8th International Conference on Advanced Semiconductor Devices and Microsystems (ASDAM 2010) (2010) 279.
7. Expert-Photography, What Is a RAW File? (And How to Open One), https://expertphotography.com/raw-file (2023).
8. LibCamera, LibCamera – A Complex Camera Support Library for Linux, Android, and ChromeOS, https://libcamera.org/docs.html (2023).
9. Raspberry-Pi, The Picamera2 Library – A Libcamera-based Python Library for Raspberry Pi Cameras, https://datasheets.raspberrypi.com/camera/picamera2-manual.pdf (2023).
10. GitHub, Raspberry Pi Libcamera Application Source Code, https://github.com/raspberryPi/libcamera-apps (2023).
11. GitHub, Picamera Release-1.13, Advanced Recipes, https://picamera.readthedocs.io/en/release-1.13/recipes2.html#raw-bayer-data-captures (2023).
12. Leybold Shop, X-ray Apparatus 554800, https://www.leybold-shop.com/x-ray-apparatus-554800.html (2023).
13. Leybold Shop, X-ray Tube Mo 554861, https://www.leybold-shop.com/x-ray-tube-mo-554861.html (2023).