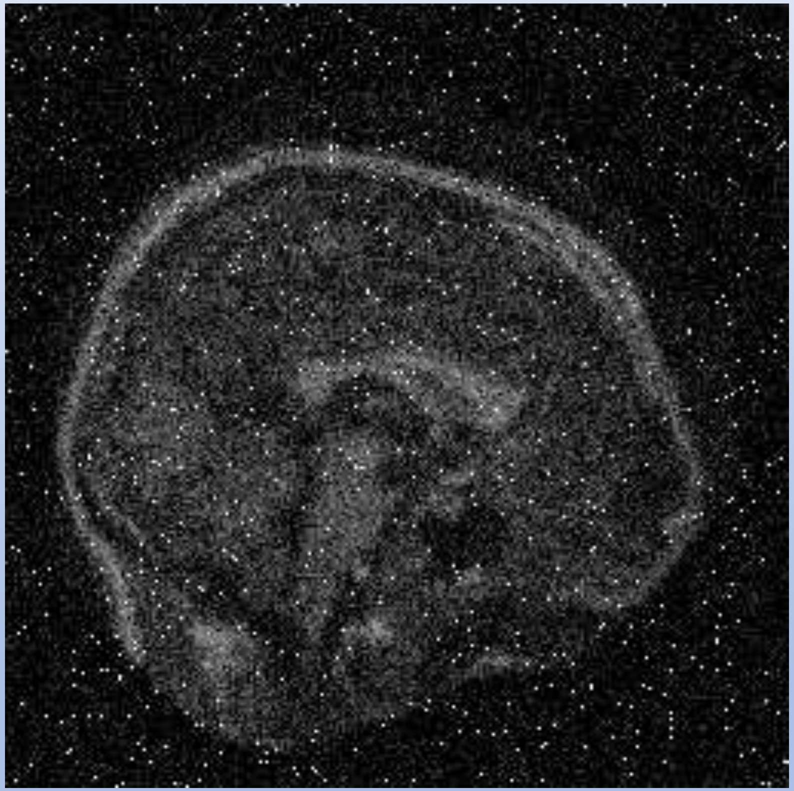
SECV3213– FUNDAMENTAL OF IMAGE PROCESSING

Assignment 2

Image Restoration

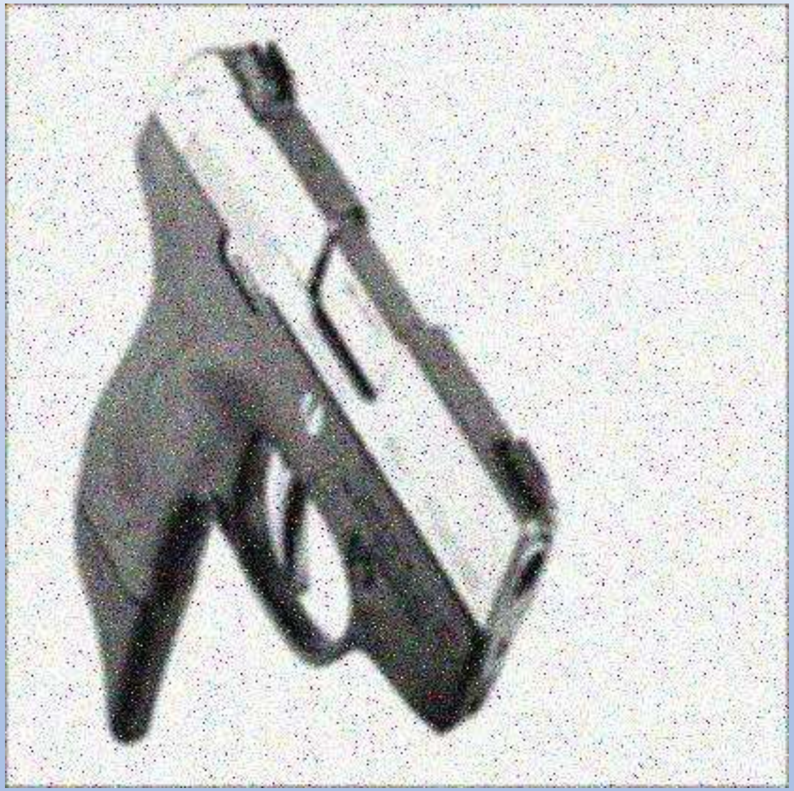
Name	Matric No
Muhammad Akashah Bin Bahar	A21EC0066
Lew Chin Hong	A21EC0044
Yeo Chun Teck	A21EC0148

Problem Analysis (Image A5)



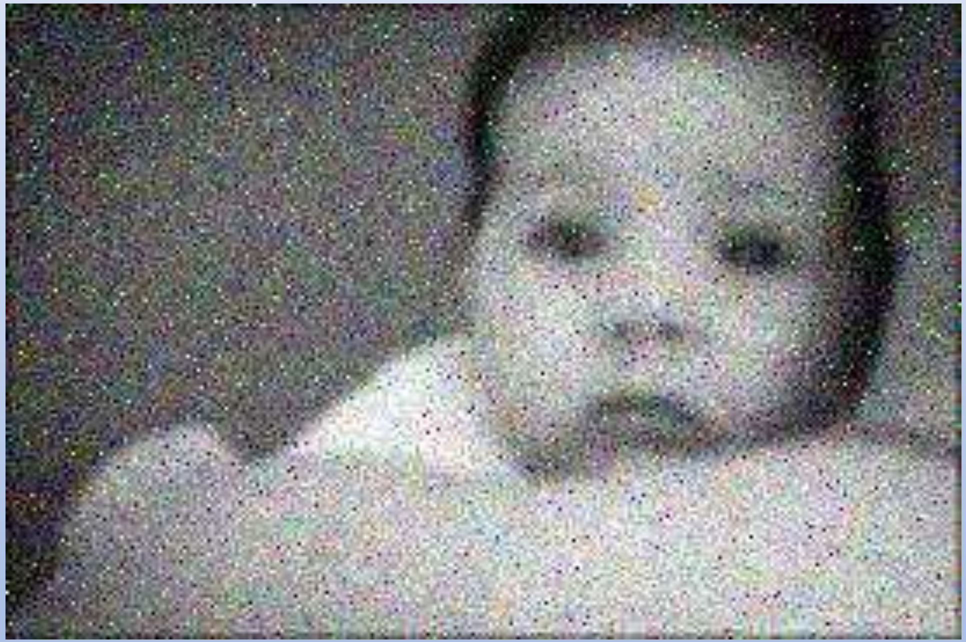
In this image, the main problem that occurred is salt-and-pepper noise, also known as impulse noise. We could notice that random bright (salt) and dark (pepper) pixels scattered throughout the image. This noise has significantly degraded the visual quality of the image. Several factors may lead to this problem, such as transmission issues, image processing errors, and sensor errors.

Problem Analysis (Image B5)



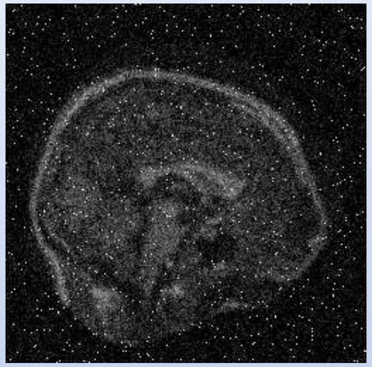
The main issue that occurred in this image is the colour noise. Colour noise is also known as chroma noise. It is a type of unwanted variation in colour that appears as random colour pixels in the image. This noise will affect the visual quality of the images and may lead to errors while parsing the picture. The causes of this type of noise are similar to salt-and-pepper noise, which are transmission issues, sensor errors and another is compression errors.

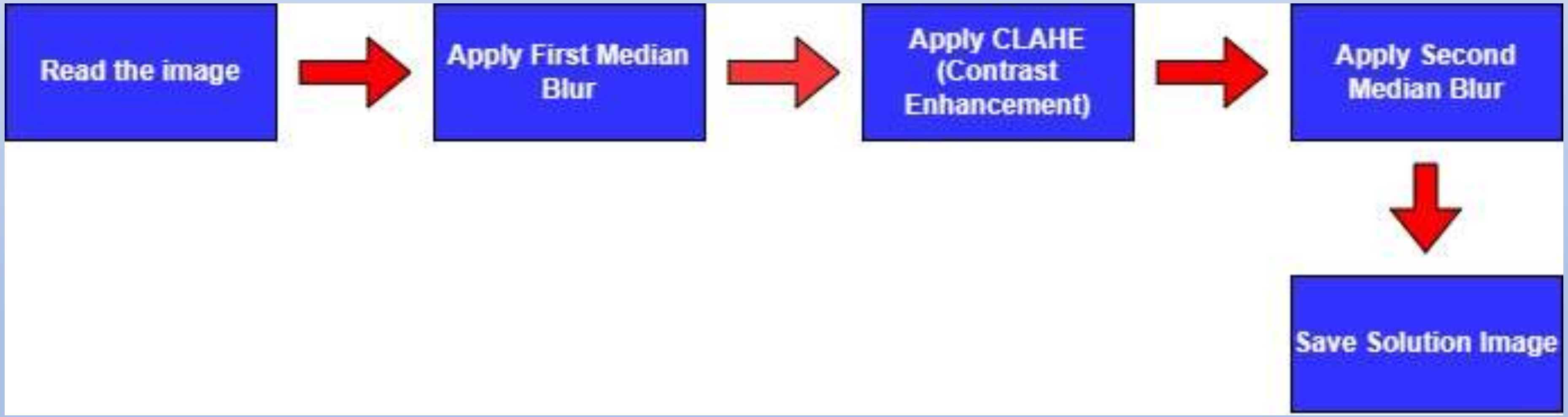
Problem Analysis (Image C5)

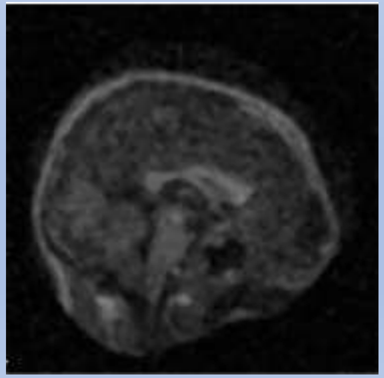


There are two main problems in this image, which are colour noise and image blurriness. As shown in the image above, we can clearly notice that the image is covered with random, scattered colour pixels. Besides, this image has a low visual quality and this caused the details of this image not to be identified easily/

Proposed Solution 1: Process Flow (Example)

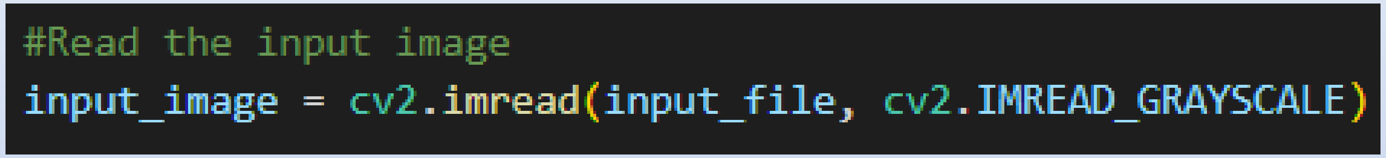






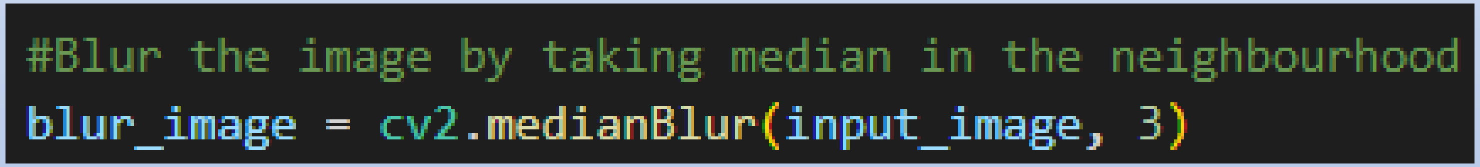
Proposed Solution 1: Explanation

1. Firstly, we will read the input image in grayscale by using cv2.imread



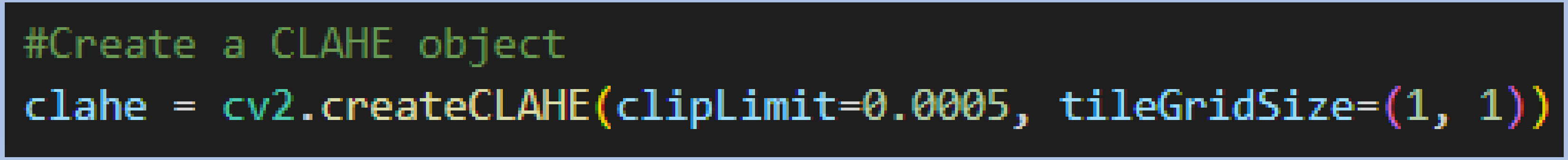
```
#Read the input image
input_image = cv2.imread(input_file, cv2.IMREAD_GRAYSCALE)
```

2. Next, we will apply the median blur by using cv2.medianBlur with a kernel size of 3*3. This step is crucial to reduce noise by smoothing the image



```
#Blur the image by taking median in the neighbourhood
blur_image = cv2.medianBlur(input_image, 3)
```

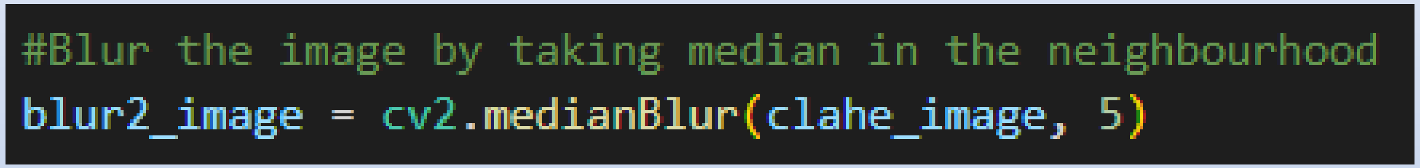
3. Then, we will apply Contrast Limited Adaptive Histogram Equalization (CLAHE) with a clipLimit of 0.0005 and tileGridSize of (1,1). This step is used to enhance local contrast to improve the visibility of details



```
#Create a CLAHE object
clahe = cv2.createCLAHE(clipLimit=0.0005, tileGridSize=(1, 1))
```

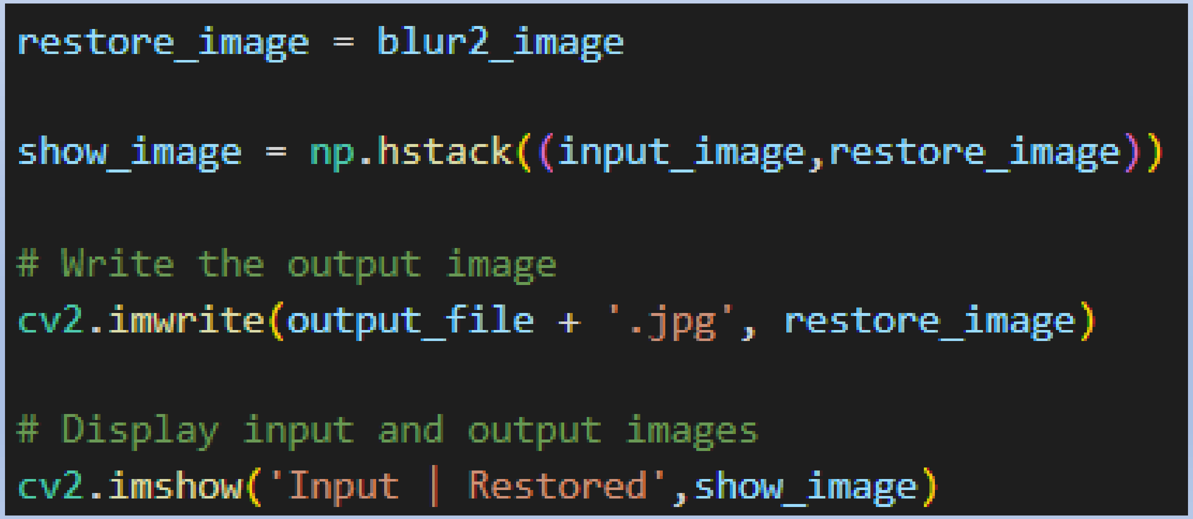
Proposed Solution 1: Explanation (cont.)

4. We perform another median blur on the image after CLAHE is applied. This can help further noise reduction. The kernel size of this median blur process is set to 5*5



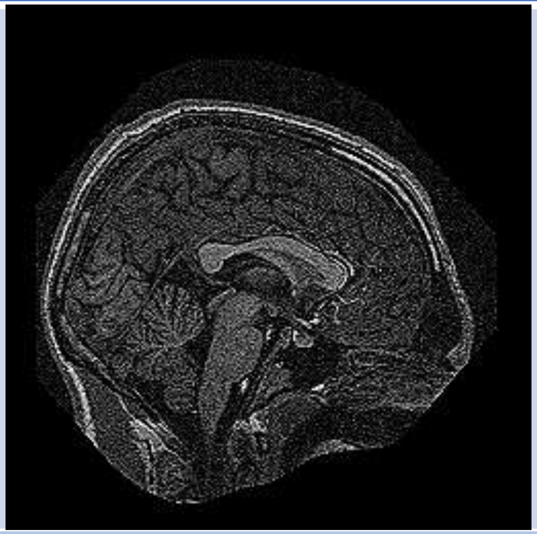
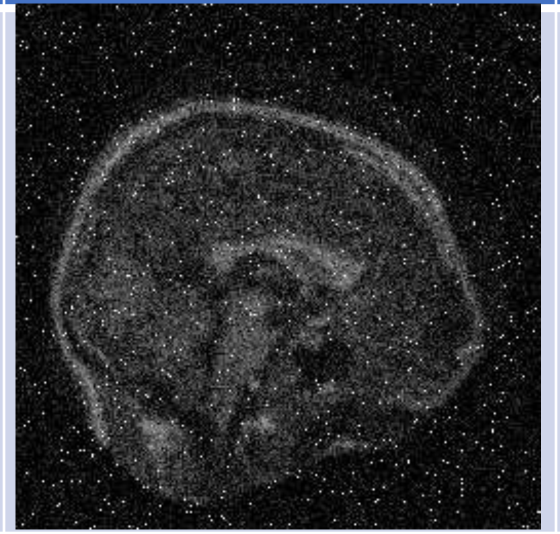
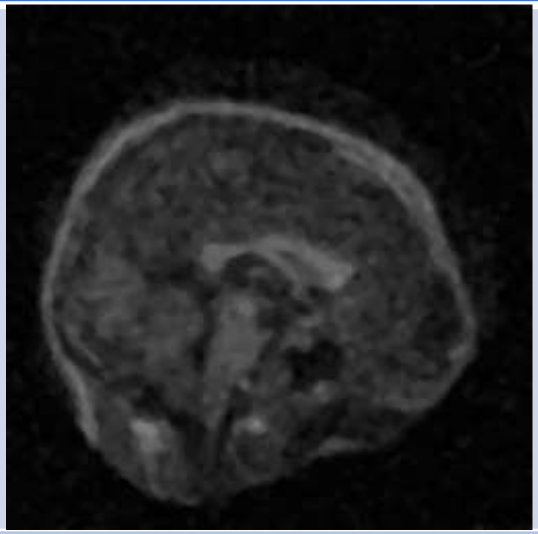
```
#Blur the image by taking median in the neighbourhood
blur2_image = cv2.medianBlur(clahe_image, 5)
```

5. Lastly, we save the final denoised image as restore_image. We also use cv2.write to write out the denoised image and display it together with the input image by using cv2.imshow

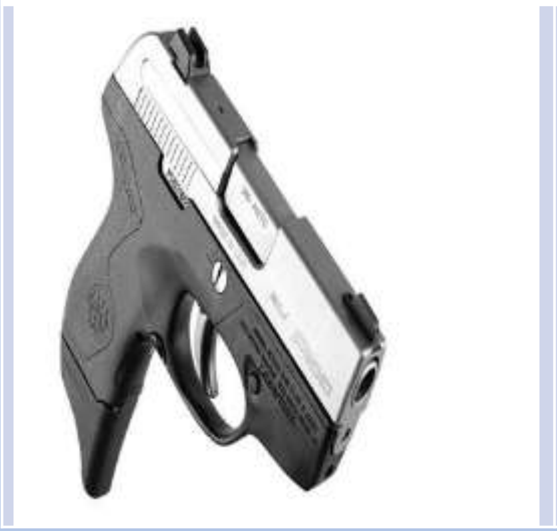
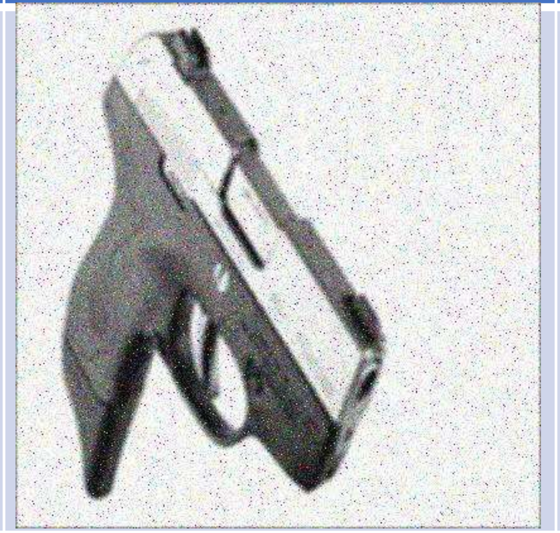
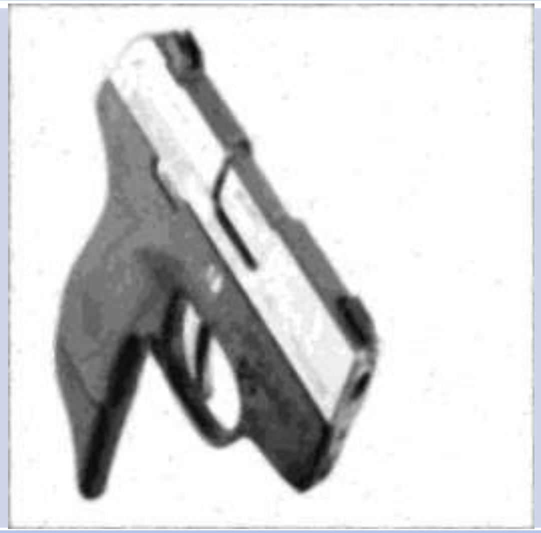


```
restore_image = blur2_image

show_image = np.hstack((input_image,restore_image))

# Write the output image
cv2.imwrite(output_file + '.jpg', restore_image)

# Display input and output images
cv2.imshow('Input | Restored',show_image)
```

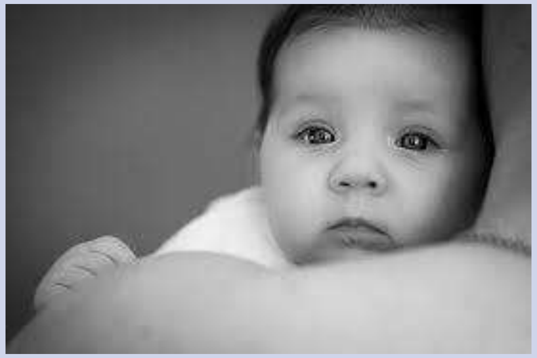
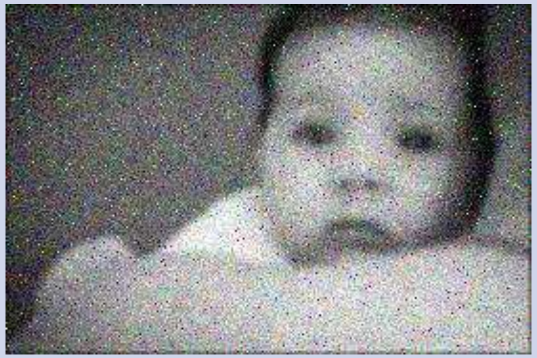
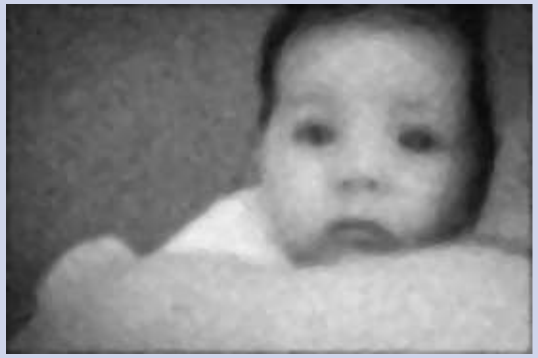
Results of the image A5: proposed solution 1

Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			52.54%

Results of the image B5: proposed solution 1

Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			58.69%

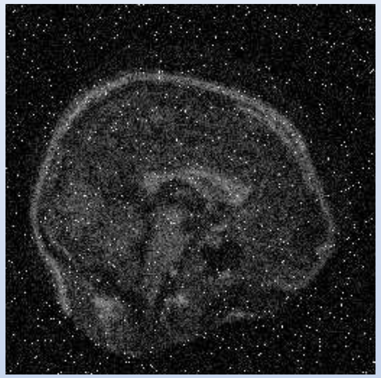
Results of the image C5: proposed solution 1

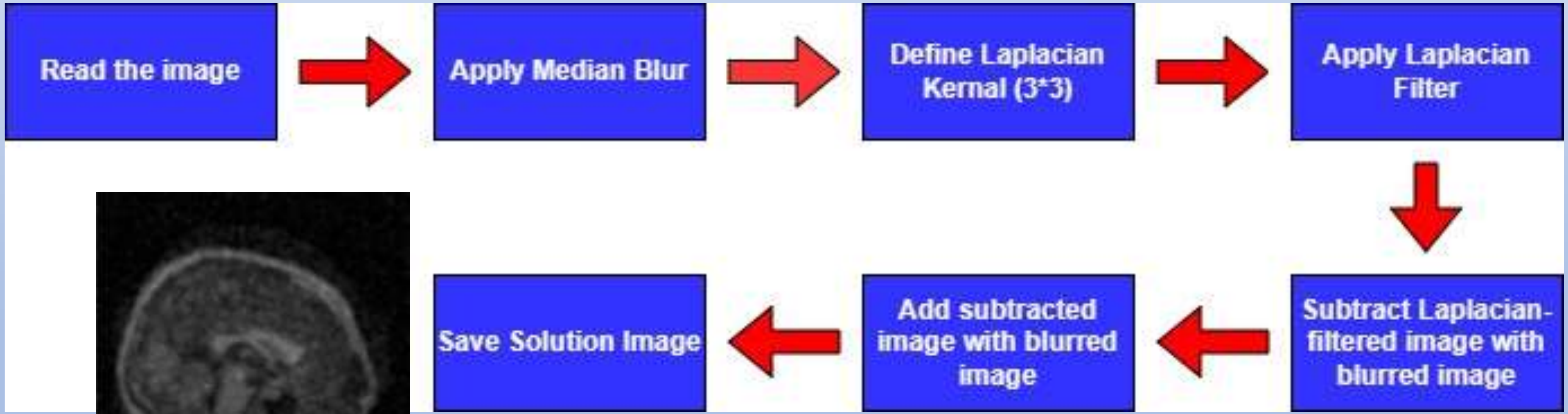
Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			87.93%

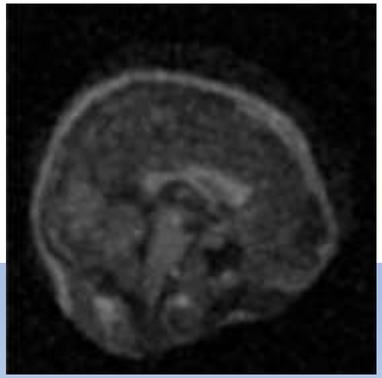
Result Analysis of solution 1

Qualitative	Quantitative
The noise reduction process applied to the images has a positive result. The main problem, which is salt and pepper noise and colour noise has been solve. Also, the contrast of colour in the images has been increased and the grayscale consistency was well-maintained hence the details of the images can be viewed clearer	The similarity percentages obtained were 52.54%, 58.69%, and 87.93% concerning image A5, image B5, and image C5, respectively. Higher similarity percentages indicate better alignment, suggesting that our noise reduction solutions effectively retained important structural details and visual characteristics present in the original images.

Proposed Solution 2: Process Flow

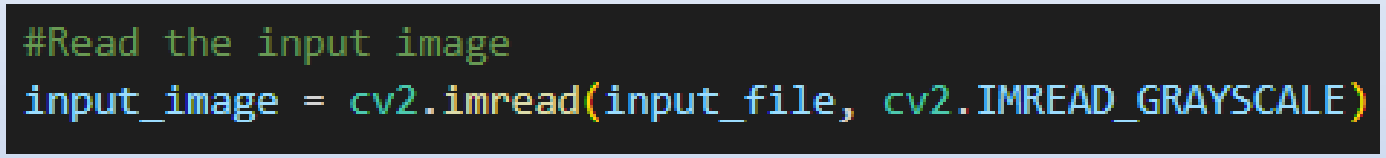



Read the image
Apply Median Blur
Define Laplacian Kernal (3*3)
Apply Laplacian Filter
Subtract Laplacian-filtered image with blurred image
Add subtracted image with blurred image
Save Solution Image



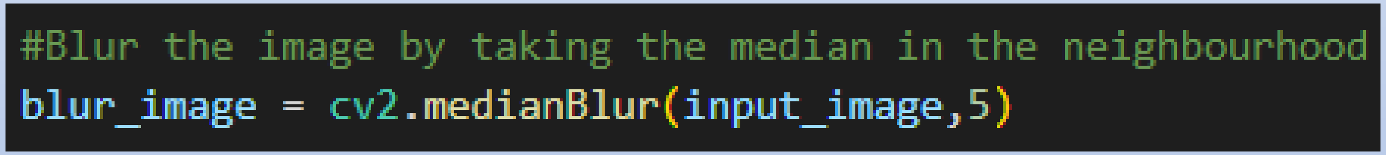
Proposed Solution 2: Explanation

1. Firstly, we will read the input image in grayscale by using cv2.imread



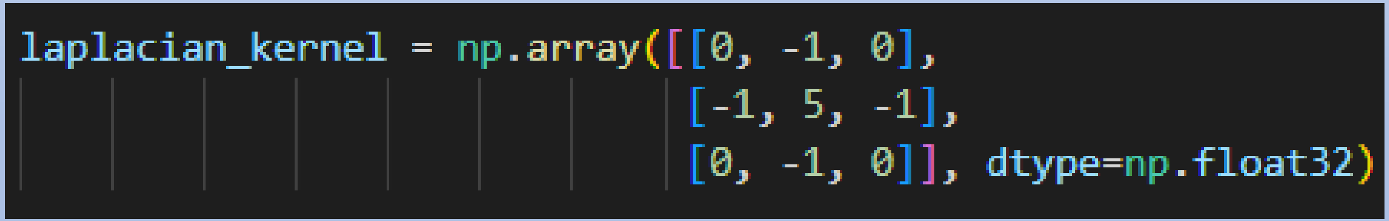
```python
#Read the input image
input_image = cv2.imread(input_file, cv2.IMREAD_GRAYSCALE)
```

2. Next, we will apply the median blur by using cv2.medianBlur with a kernel size of 5*5. Median Blur is a basic approach to noise reduction



```python
#Blur the image by taking the median in the neighbourhood
blur_image = cv2.medianBlur(input_image,5)
```

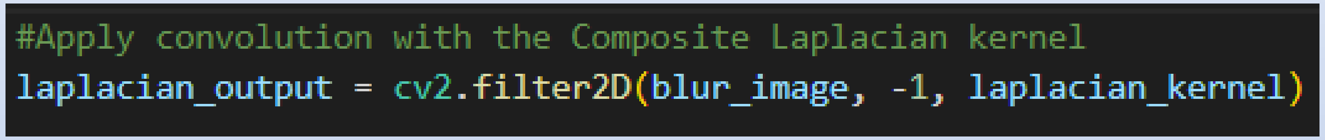
3. Then, we will define a 3*3 Laplacian Kernal



```python
laplacian_kernel = np.array([[0, -1, 0],
                             [-1, 5, -1],
                             [0, -1, 0]], dtype=np.float32)
```

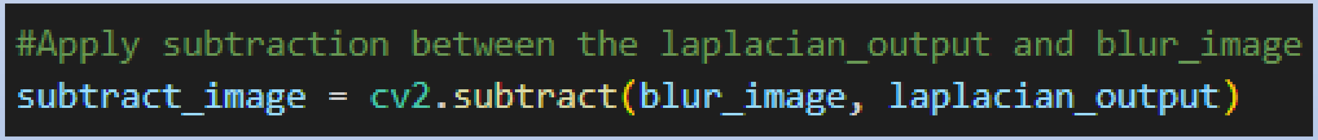
Proposed Solution 2: Explanation (cont.)

4. This kernel is used to convolve with the blurred image just now. We achieve this step by using cv2.filter2D. This will enhance the edges of the image



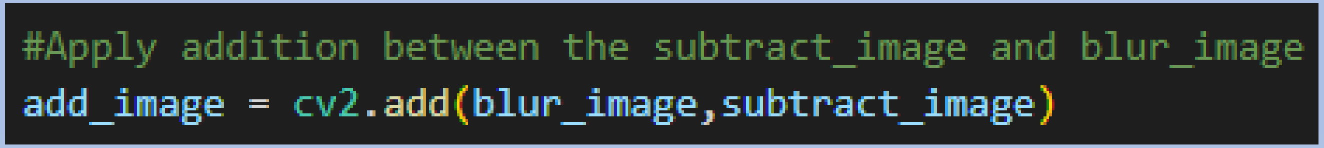
```python
#Apply convolution with the Composite Laplacian kernel
laplacian_output = cv2.filter2D(blur_image, -1, laplacian_kernel)
```

5. Next, the Laplacian-filtered image is subtracted from the blurred image so that the edges in the image can be emphasized



```python
#Apply subtraction between the laplacian_output and blur_image
subtract_image = cv2.subtract(blur_image, laplacian_output)
```

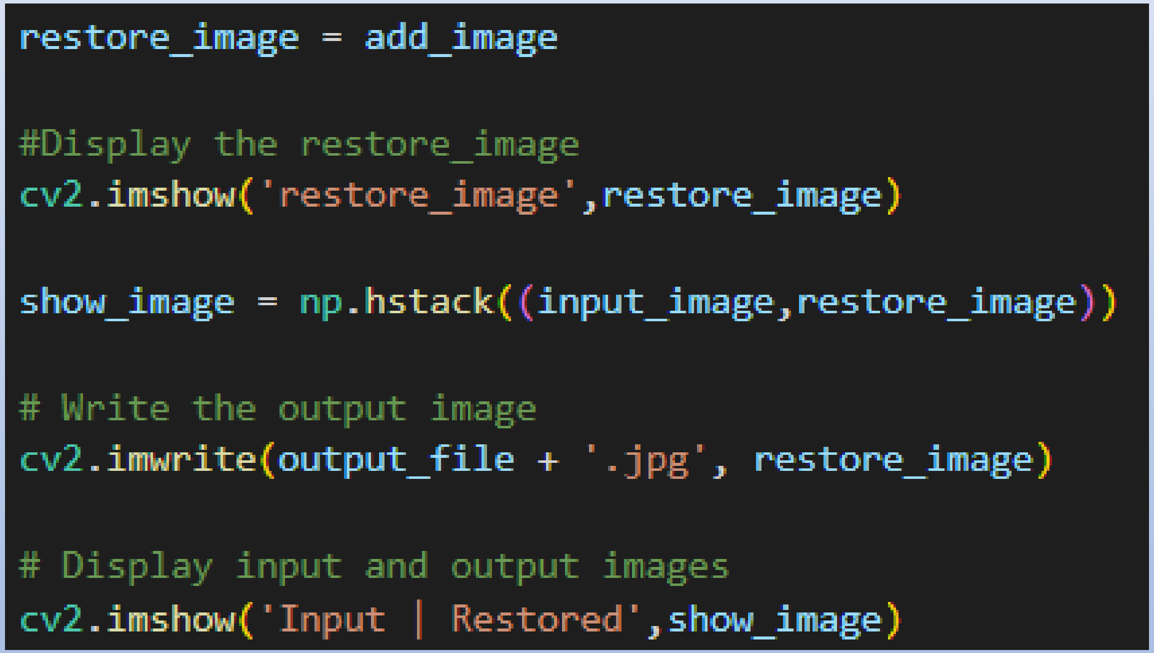
6. The subtracted image is added back to the blurred image so the image can have a further enhancement



```python
#Apply addition between the subtract_image and blur_image
add_image = cv2.add(blur_image,subtract_image)
```

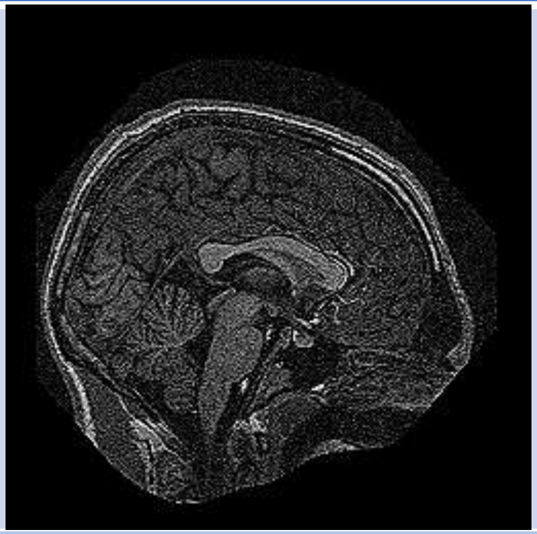
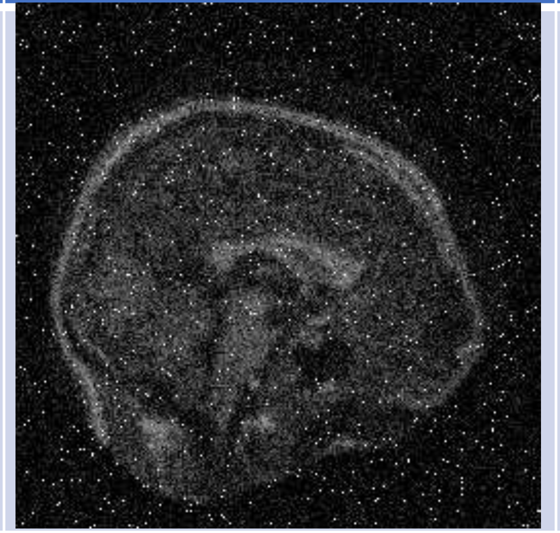
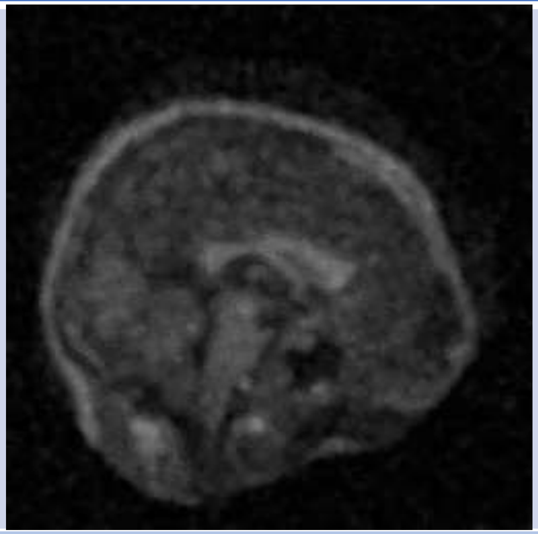
Proposed Solution 2: Explanation (cont.)

7. Lastly, we save the final denoised image as restore_image. We also use cv2.write to write out the denoised image and display it together with the input image by using cv2.imshow

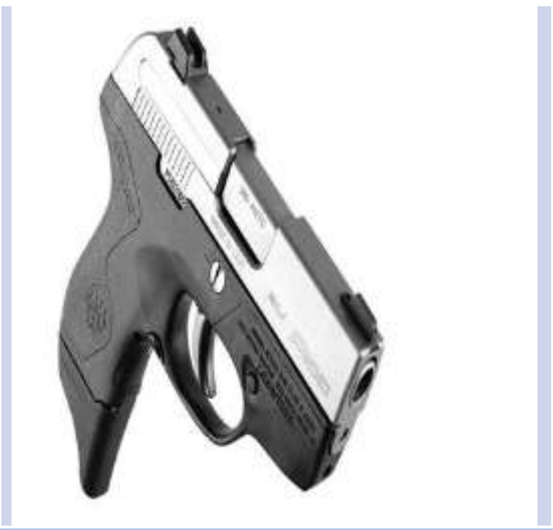
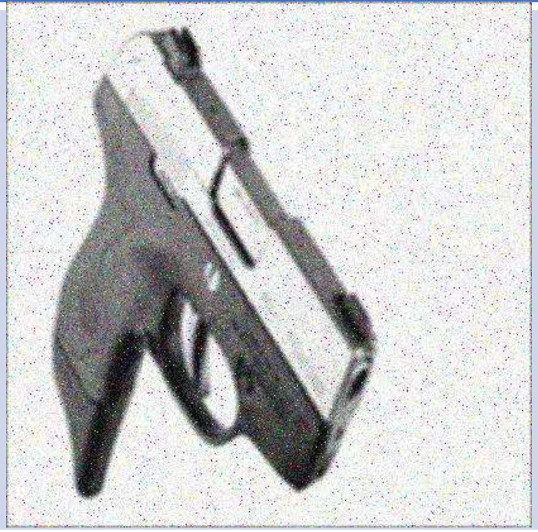
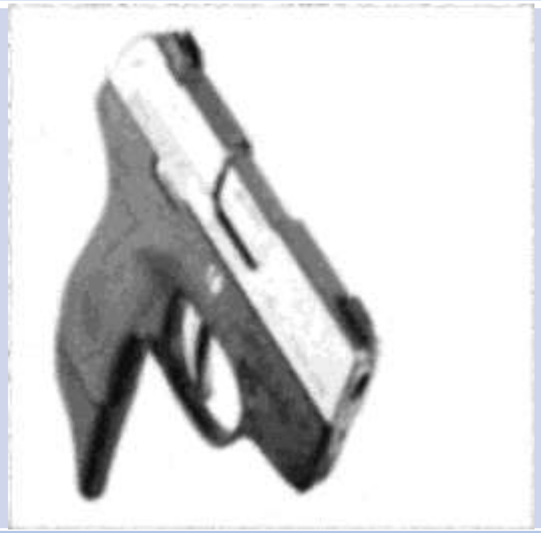


```
restore_image = add_image

#Display the restore_image
cv2.imshow('restore_image',restore_image)

show_image = np.hstack((input_image,restore_image))

# Write the output image
cv2.imwrite(output_file + '.jpg', restore_image)

# Display input and output images
cv2.imshow('Input | Restored',show_image)
```

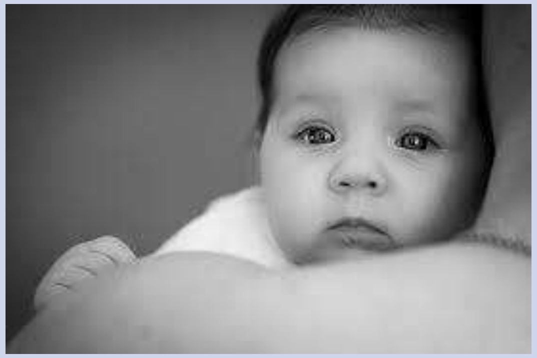
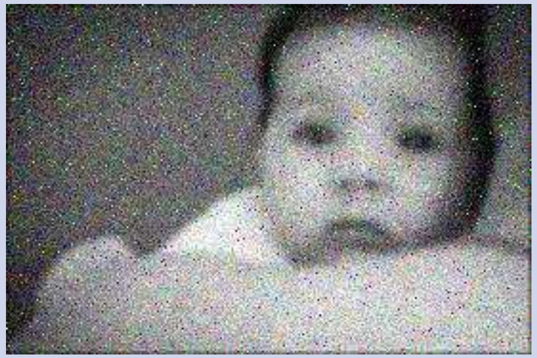
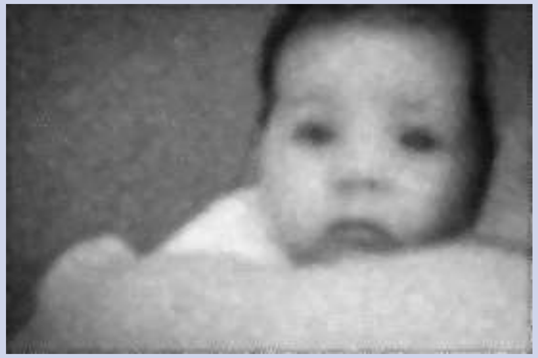
Results of the image A5: proposed solution 2

Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			50.56%

Results of the image B5: proposed solution 2

Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			69.64%

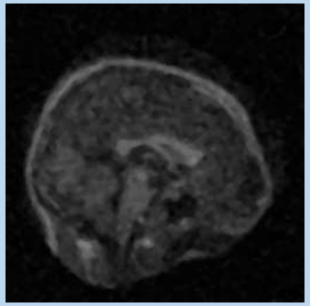
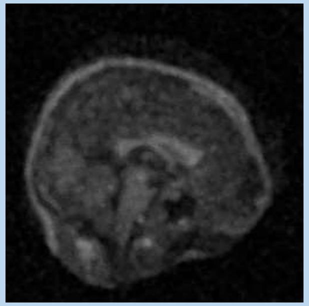
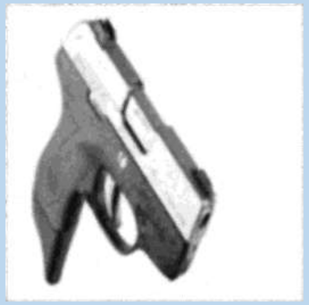
Results of the image C5: proposed solution 2

Reference Image	Noised Image	Restored Image	Pixel Similarity Result
			82.32%

Result Analysis of solution 2

Qualitative	Quantitative
After applying a combination of median filtering and Laplacian sharpening, an output image of visually sharper and cleaner has been produced. Besides, fine details and edges were effectively preserved on each image, and the subjective impression was enhanced clarity.	The obtained similarity percentages were 50.56%, 69.64%, and 82.32% concerning image A5, image B5, and image C5, respectively. The results from the quantitative analysis complement the qualitative assessment, confirming that Solution 2 successfully reduced noise while maintaining image integrity across different noise scenarios.

Analysis of proposed solutions (Table of Comparison)

Reference Image	Solution 1 Output Image	Pixel Similarity Result		Solution 2 Output Image
		Solution 1	Solution 2	
Image1		52.54%	50.56%	
Image2		58.69%	69.64%	

Analysis of proposed solutions (Table of Comparison)

Reference Image	Solution 1 Output Image	Pixel Similarity Result		Solution 2 Output Image
		Solution 1	Solution 2	
ImageC5		87.93%	82.32%	

Table comparison explanation

- Both solutions achieved similarity percentages around 50% for Image A5. This indicates that neither solution performed significantly better in preserving the visual characteristics of Images A5. The similarity percentages are relatively close, suggesting that the effectiveness of both solutions is comparable for this particular image.
- Solution 2 demonstrates a higher similarity percentage (69.64%) for Image B5 compared to Solution 1 (58.69%). This suggests that Solution 2 was more successful in reducing noise and preserving visual details for Image B5. The difference in similarity percentages indicates that Solution 1 may be more suitable for handling the noise characteristics present in Image B5.

Table comparison explanation (cont.)

- Solution 1 achieved a slightly higher similarity percentage (87.93%) for Image C5 compared to Solution 2 (82.32%). While both solutions performed well for Image C5, Solution 1 demonstrated a marginally better alignment with the noise-free original. The similarity percentages indicate that Solution 1 may be more effective in handling the noise pattern in Image C5.
- The similarity percentages highlight the strengths and weaknesses of each solution concerning different images. The performance of the solutions varies depending on the specific characteristics of the noise and image content and we conclude that both solutions have the best effect on the image C5.

Conclusion

- The first solution has incorporated median blurring followed by Contrast Limited Adaptive Histogram Equalization (CLAHE) and lastly a second median blurring. This solution has shown promise in noise reduction and contrast enhancement. The iterative application of median blur and CLAHE contributed to a reduction in noise, particularly in localized areas.
- As for the second solution, we utilized a combination of median filtering and Laplacian sharpening, has demonstrated effectiveness in reducing noise and enhancing image details. The applied median filter successfully smoothed the input image, while the Laplacian filter emphasized edges, resulting in a visually sharper and cleaner output

•

Conclusion (cont.)

- Both solutions employ distinct noise reduction techniques, and their effectiveness depends on the characteristics of the input images and the specific noise patterns present. The choice between the solutions may depend on the nature of the noise, the desired level of detail preservation, and the computational considerations.