Gaussian Noise Removal

Veer Singh

July 31, 2021

1 Introduction

Implementing the algorithm to reduce Gaussian Noise described in [1].

2 Estimating noise standard deviation using Immerkaer's fast method

$$\sigma_{GN} = \sqrt{\frac{\pi}{2}} \frac{1}{6MN} \sum_{i,j=1}^{M,N} |X \circledast F|$$

Where:

- X_{ij} = Current pixel
- M = Width
- N = Height
- $\circledast$ denotes convolution operation
- $F = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} = MASK$

3 Taking Image Segment and Finding the Absolute Difference

We take a 3x3 segment of the grayscale image $\begin{bmatrix} 152 & 149 & 173 \\ 148 & 168 & 137 \\ 181 & 135 & 113 \end{bmatrix}$

The absolute difference between the other pixels and centre pixel(168) is calculated and we get the absolute difference vector with 8 elements = [16, 19, 5, 20, 31, 13, 33, 55]

4 Flagging Pixels

If the absolute difference < Noise Standard Deviation * Smoothing Factor, then we flag that particular pixel.

If Noise Standard Deviation = 17.645, Smoothing Factor = 2, then if the absolute difference value is less than 35.29 (17.645 * 2), then that pixel will be flagged.

Using the absolute difference vector = [16, 19, 5, 20, 31, 13, 33, 55]. All values except the last element are flagged. We get total 7 flagged elements and a vector of the pixel values of only the flagged pixels is created called DA = [152, 149, 173, 148, 137, 181, 135].

5 Applying Changes

Now if the number of flagged pixels is greater than a threshold = (2 * Window width) - 1, then we replace the centre pixel with the mean of the DA vector. If number of flagged pixels is less than the threshold then no changes are applied and we move on to the next image segment.

Here the number of flagged pixels is 7 which is greater than the threshold = 5 ((2*3)-1). So we will calculate the mean of DA which is 153 and replace the centre pixel with this value.

$$\begin{bmatrix} 152 & 149 & 173 \\ 148 & 168 & 137 \\ 181 & 135 & 113 \end{bmatrix} \Rightarrow \begin{bmatrix} 152 & 149 & 173 \\ 148 & 153 & 137 \\ 181 & 135 & 113 \end{bmatrix}$$

6 Applying on the entire image

This 3x3 sliding window is applied on all pixel values.

Consider the following image.

$$\begin{bmatrix} 114 & 85 & 66 & 15 & 24 & 39 & 51 & 3 & 185 & 128 \\ 155 & 87 & 201 & 156 & 43 & 71 & 8 & 163 & 182 & 88 \\ 93 & 87 & 34 & 188 & 203 & 215 & 139 & 37 & 121 & 122 \\ 203 & 213 & 113 & 140 & 69 & 78 & 37 & 24 & 116 & 41 \\ 0 & 95 & 2 & 90 & 168 & 44 & 209 & 227 & 217 & 136 \\ 47 & 98 & 151 & 237 & 100 & 17 & 152 & 94 & 131 & 137 \\ 49 & 208 & 131 & 94 & 27 & 151 & 231 & 177 & 163 & 64 \\ 239 & 177 & 157 & 240 & 129 & 100 & 153 & 16 & 215 & 52 \\ 142 & 64 & 196 & 95 & 181 & 65 & 183 & 123 & 246 & 138 \\ 138 & 213 & 6 & 36 & 86 & 205 & 123 & 6 & 67 & 64 \end{bmatrix}$$

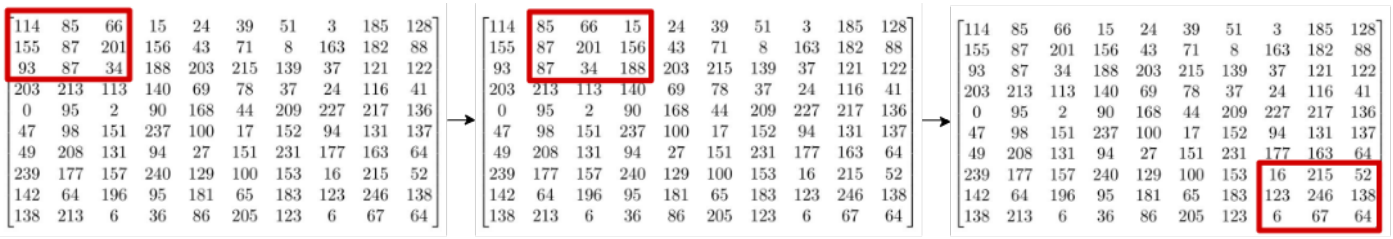

Figure 1: Depiction of Sliding Window

7 Results

Figure 2: Original Image with Gaussian Noise

Figure 3: Applying OpenCV Gaussian Blur with Kernel size 5

Figure 4: Applying Proposed Algorithm with Smoothing Factor of 5

References

[1] V. V.R., P. Vanathi, and P. Kanagasabapathy, "Fast and efficient algorithm to remove gaussian noise in digital images," *IAENG International Journal of Computer Science*, vol. 37, pp. 78–84, 02 2010.