EE569 Digital Image Processing

**HOMEWORK #1**

**HOMEWORK – Introduction to Digital Image Processing**

**Issued: 13/01/2020 Due: 27/01/2020**

# Problem 1: Image Demosaicing and Histogram Manipulation (50%)

1. **Bilnear Demosaicing (10%)**

**Motivation**

The aim is to practice demosaicing method that can transform gray-scaled image to colorful RGB image. The bilinear transformation method is the simplest way to transform the gray-scaled image into the RGB image. The color type of three channels of the pixel depends on the position of processed image and its neighboring pixels. The value varies when the global position of the pixel changes. Hence, if-else argument should be designed to compute pixel value with different position in the image.

The formula show the key computation procedure for each pixel intensity value.

Each pixel intensity value ofcertain coloris estimated by the neighboring pixels with the same color type. The pixel intensity value in three channels is computed with the same procedure because the edge is extended to avoid different solution for intensity value estimation.

**Approach and procedure**

*This is the first problem to be solved. Hence, the basic C++ program method to handle image process problem will be introduced firstly. The procedure for following problem below will not include the explanation again. The basic technique to input image and handle the pixel is similar.*

Before starting computing, memory is allocated to store the initial image and extended image and output image. To prevent different manipulation on the pixels on the edge, the original image is extended with mirror rule. The implementation of mirror rule is to mirror pixel with the first column. For example, the intensity value of pixels in the column at extended edge neighboring the first column of original image has the same value with the first column pixels. In that case, all pixels at the extended edge are mirrored by the first pixel column. The function *extend2DImage()* has finished this function.

and compute the pixel value for different color channel. For different color, there is slightly difference. The computation method for green color is most complex. Before direct computation, whether the row position of pixel should be decided to ensure that neighbor position of other color pixel is right. The type of pixel value is character. Character can be not computed directly.

**Results**

The result is amazing. The obtained image is colorful. The computation cost is low. There are O() complexity for the whole computation procedure. The Figure 1.1.1 shows obtained image computed by the Bilinear algorithm from gray-scaled image. The Figure 1.1.2 shows the original image. With comparison to the original image, the obtained image is more colorful in some area. The grass is more green. In Figure 1.1.3, when the camera is zoomed, the distribution of color pixel intensity value of obtained image is more equal than the distribution of original image.

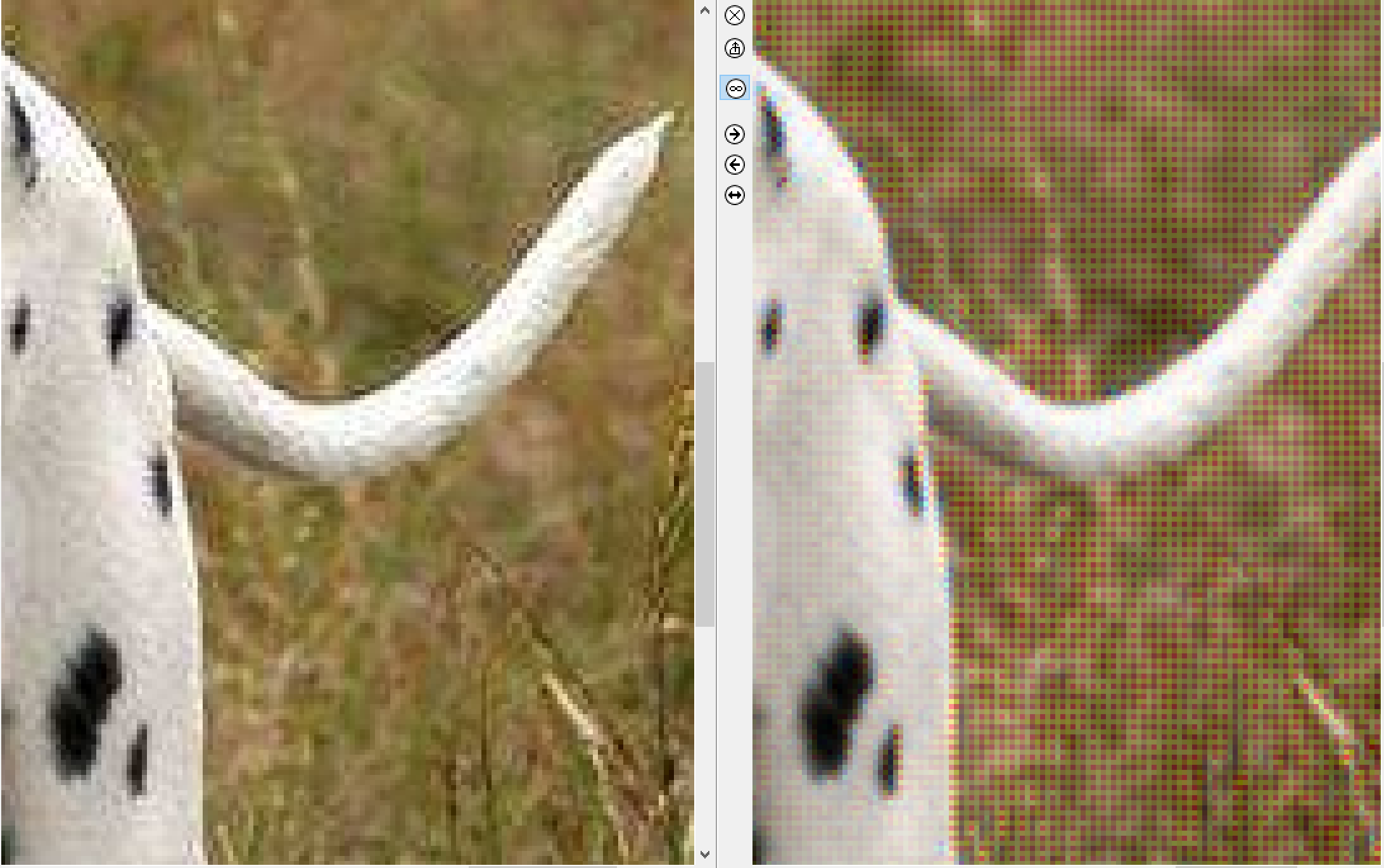
图片包含 小狗, 草, 动物, 哺乳动物

描述已自动生成 图片包含 草, 小狗, 户外, 田野

描述已自动生成

**Figure 1.1.2 original image**

**Figure 1.1.1 obtained image**



**Figure 1.1.3 colorful artifact compared with the original image**

**Discussion**

The computation of the pixels depends on the position of the pixel. The averaging process is safe. There is overflow for the final computed pixel. The maximum pixel value is 255. The addition for C++ is 2 byte operator space. The final result for each pixel will be within 0-255 because the total summation will be divided by 2 or 4, so there is no overflow.

**Answers**

1. In Figure 1.1.1 and Figure 1.1.2, you are shown with the obtained dog picture and the original dog image. Compared with the original image, the obtained image is more colorful. The grass is much greener. More shapes can be viewed from scanning.
2. Problem may be caused by the local average process for computation. The bilinear method ignores the global hue for the image, the color seems to be distributed equally from the obtained image.
3. **Malvar-He-Cutler (MHC) Demosaicing (20%)**

**Motivations**

The MHC method to demosaiced image is to add small computation compliment when we compute color value of three channels for the image.

**Approach and procedure**

1. Judge the primary color of pixel.
2. Compute difference between value of original color and neighbor color.
3. Compute estimated value for other color of the pixel.
4. Loop the above three steps for all pixels and get the image.

The computation procedure for difference value between original color and neighbor color depends on the color category of the pixel, so the specific function is chosen to compute the delta value for the pixel. There may be overflow for the unsigned char data type, so a compliment function needs to revise the computation result for each pixel

**Results**

In Figure 1.2.1, it shows an image with black background.

A brown and white dog standing on top of a grass covered field

Description automatically generated

**Figure 1.2.1 Obtained Image with MHC method**

**A brown and white dog standing in the grass

Description automatically generated**

**Figure 1.2.2 original image**

**Discussion**

Overflow happens when the increment compliment estimation by the neighboring pixels is computed. Hence, the negative value may appear. The possible result can be negative for the pixel value. This cause the overflow for the unsigned char type because the unsigned char value can not be negative. The uncorrelated value causes several peak pixel with pure red, blue that is not correlated to the neighboring pixel value. The solution to eliminate this noise is to combine two methods above. When the negative value is got, average method can be chosen.

**Answers**

**c). Histogram manipulation(10%)**

**Motivation**

The aim of histogram manipulation is change the distribution with respect to pixel value. The foundation method to implement it is to construct a transformation function that maps the pixels for the same value to another pixel value. The problem provides two principle for us to construct transformation array for the pixel value mapping. The method A is to use cumulative probability theory. The probability density for each pixel value should be uniformed so that the probability distribution can be transformed to uniform distribution. [1].

**Approach and procedure**

* 1. compute the number of pixels corresponding to dedicated one gray-scaled value
  2. draw the histogram to describe the distribution with pixels numbers and gray-scaled value
  3. design the transfer function corresponding to the histogram
  4. apply transfer function to transfer current image pixels to expected image pixels
  5. write image result to the file

**Results**

**A close up of a map

Description automatically generated**

**Figure1.3.1 histogram of original Toy.raw image**

**A screenshot of a cell phone

Description automatically generated**

**Figure 1.3.2 histogram of obtained Toy\_a.raw by method A**

A close up of a device

Description automatically generated

**Figure 1.3.3 histogram of obtained Toy\_b.raw by method B**A stuffed animal on a table

Description automatically generatedA large brown teddy bear sitting on a table

Description automatically generated

**Figure 1.3.4 the original toy image**

**Figure 2.3.5 obtained toy image with method A**

**A picture containing coffee, table, indoor, cup

Description automatically generated**

**Figure 1.3.6 obtained Toy\_b.raw image with method B**

**Discussion**

The first try to get RGB image does not consider the boundary problem, so as the figure shows that the boundary of the image is lacked.

**Answers**

# Problem 2: Image Denoising (50%)

1. **Baisc denoising methods (10%)**

**Motivation**

This is the simplest method among the problems to denoise image. Each pixel are obtained by averaging the neighboring pixels with the same weight. This weighs neighboring pixels equally by default. This assumption is defective when there are some discontinuous changes in the image.

**Approach and procedure**

*linear\_filter()*

*aver2DImage()*

*GaussianFilter()*

*compGaussianPixel()*

**Results**

**A person lying on a blanket

Description automatically generatedA group of corn

Description automatically generated**

**Figure 2.1.1 the denoised image with method A and the original image**

**A picture containing person, indoor, sitting

Description automatically generatedA picture containing indoor, person

Description automatically generatedA picture containing bed, indoor, laying

Description automatically generated**

**Figure 2.1.2 the denoised image set by method A**

**A picture containing person, indoor, sitting

Description automatically generatedA picture containing person, indoor, clothing

Description automatically generated**

**Figure 2.1.3 the denoised image set by method B**

**Discussion**

**Answers**

1. **Bilateral Filtering (10%)**

**Motivation**

Bilateral filter compute weight based on Gaussian Probability distribution. It measures the pixels and neighboring pixels by their relative position and intensity pixel value relationship. Both effects on the pixel relationship decides the gaussian coefficients for the pixel intensity value. When the distance between measured pixel and target pixel is large, the weight for the measured pixel will be small. The assumption is that the neighboring pixels has large influence on the target pixel value. This is also defective if there are different shapes or segments in the image.

**Approach and procedure**

More detailed steps are as follows:

1. define the hyper-parameter to estimate the models
2. assign the each neighboring pixel weight value.

*bilateral\_filtering()*

loop all pixels to get the estimated results from *computeBilateralFilteredPixel()*

*computeBilateralFilteredPixel()*

compute pixel intensity value by summing up all gaussian coefficients multiplied by the pixel intensity value and normalizing by summing up all gaussian coefficients themselves

*computeGaussWeight()*

compute

**Results**

The below figures are generated by the parameters in the table 2.2.1.

A picture containing person

Description automatically generatedA picture containing person, indoor

Description automatically generated

Figure 2.2.1 result of experiment one and two

A picture containing person, indoor

Description automatically generatedA picture containing person, indoor

Description automatically generated

**A picture containing person, indoor

Description automatically generatedA picture containing person, indoor

Description automatically generated**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **window size** |  |  | **PSNR** |
| **1** | **4** | **1** | **10** | **17.7505** |
| **2** | **4** | **1** | **20** | **18.3536** |
| **3** | **4** | **1** | **30** | **18.9423** |
| **4** | **4** | **1** | **50** | **19.4444** |
| **5** | **4** | **1** | **100** | **19.5138** |
| **6** | **4** | **1** | **120** | **19.4876** |
| **7** | **4** | **1** | **150** | **19.4552** |
| **8** | **10** | **1** | **100** | **19.5159** |
| **9** | **20** | **1** | **100** | **19.5159** |
| **10** | **10** | **5** | **100** | **19.0669** |
| **11** | **10** | **2** | **100** | **19.3528** |

Table 2.2.1 the debug parameter for bilateral filter

**Discussion**

The hyperparameter to improve the PSNR

**Answers**

1. **Non-Local Means(NLM) Filtering(10%)**

**Motivation**

The basic idea of NLM algorithm is to “build a pointwise estimate of the image where each pixel is obtained as a weighted average of pixels centered at regions that are similar to the region centered at the estimated pixel” [1]. The size of window for the processing pixel to compute the distance from the neighboring pixel and the window to compute the gaussian coefficients is different. The implementation for the algorithm is difficult. The result is not obviously improved.

**Approach and procedure**

Each pixel is computed by weighted averaging neighboring pixels by Euclidean distance metric with the patch.

**Results**

**A picture containing dog, indoor, hot

Description automatically generated**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | window size | patch size | hparm | stdev | psnr | citation |
| 1 | 6 | 4 | 10 | 10 | 19.3719 |  |
| 2 | 6 | 4 | 1 | 10 | 17.6887 |  |
| 3 | 6 | 4 | 20 | 10 | 19.1243 |  |
| 4 | 6 | 4 | 10 | 20 | 19.2585 |  |
| 5 | 6 | 4 | 10 | 50 | 19.093 |  |
| 6 | 6 | 4 | 10 | 25 | 19.2143 |  |
| 7 | 10 | 4 | 10 | 10 | 19.1479 |  |
| 8 | 6 | 4 | 5 | 10 | 18.6428 |  |
| 9 | 6 | 4 | 5 | 25 | 19.3354 |  |
| 10 | 6 | 4 | 5 | 50 | 19.3469 |  |
| 11 | 6 | 4 | 5 | 100 | 19.2152 |  |

**Discussion**

**Answers**

1. **Block matching and 3-D transform filter (10%)**

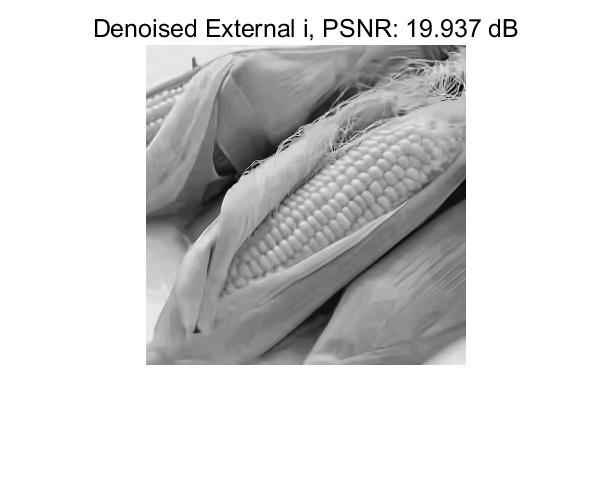
**Motivation**

The BM3D algorithm is the most complex algorithm among the above denoising algorithm. This algorithm can be used in different complex scenario without careful analysis of image noise characteristics before. Hence, the algorithm performs well even if there is no tuning for the variance parameter.

**Approach and procedure**

It is hard to implement the algorithm. The open source matlab code [2] is used to implement algorithm. When the path of image file for input and output is added, all procedure is finished.

**Results**

****

**A picture containing photo

Description automatically generated**

**Discussion**

**Answers**

**1. the algorithm uses many advanced technique to denoise the image. The first step is to group pixel points.**

**2.**

**3.**

**(e). Mixed noises in color image (10%)**

**Motivation**

There is no specific code for this problem. Hence, all parts in this problem is to analyze the above denoising algorithm and compare their advantages and disadvantages. There is no panacea algorithm to denoise the image. The limit of the algorithm comes from its computation procedure. Almost all algorithm mainly focus on the neighboring pixels. In contrast, BM3D algorithm uses the global information to estimate the noisy pixel. The algorithm should be chosen specifically after the input image is carefully analyzed. Analysis is very

**Approach and procedure**

The first step is to analyze the image with eye to decide the type and possible mixed noise that may be included in the image. The algorithm will be

Results

**Discussion**

**Answers**

1. two type of noise is added to the image, impulse noise and uniform noise(Gaussian noise).

2. The impulse noise should be filtered firstly by median filter algorithm. If it is filtered the average filter firstly, the peak pixel value will be distributed equally to all pixels in the window. Hence, the following median filter is useless. It can not detect the peak pixel in the window. The other type of filter for while noise should be used next.

3. The median filter is chosen to eliminate or alleviate the impulse noise.

# Appendix A

1. Compilation environment
2. Coding IDE
3. Github code cloud store

# Reference and Bibliography

|  |  |
| --- | --- |
| [1] | .. Kostadin, 2007. []. Available: https://www.cs.tut.fi/~foi/GCF-BM3D/BM3D\_TIP\_2007.pdf. |
| [2] | Y. Mäkinen.etc, “Image and video denoising by sparse 3D transform-domain collaborative filtering,” 2007. []. Available: http://www.cs.tut.fi/~foi/GCF-BM3D/. |