SGN-16006 Bachelors Laboratory Course in Signal Processing
Lab04 - Image processing

Hung Nguyen - 272585
Khoa Nguyen - 272580

***Abstract*— When images are captured by cameras, the light intensity is obtained by sensors. However, the sensor cannot capture color information, therefore, a color filter array (CFA) is placed on top of it to help with this task. The most common CFA is the Bayer filter, the filter pattern is 50% green, 25% red and 25% blue (figure 1). Each pixel only contains one color value. Therefore, to be able to create a full RGB image from the sensor data, the two missing color values of each pixel must be interpolated from the surrounding pixels. In this lab work, we try to implement three different interpolation methods: Nearest Neighbor, Bilinear and Patterned Pixel Grouping**

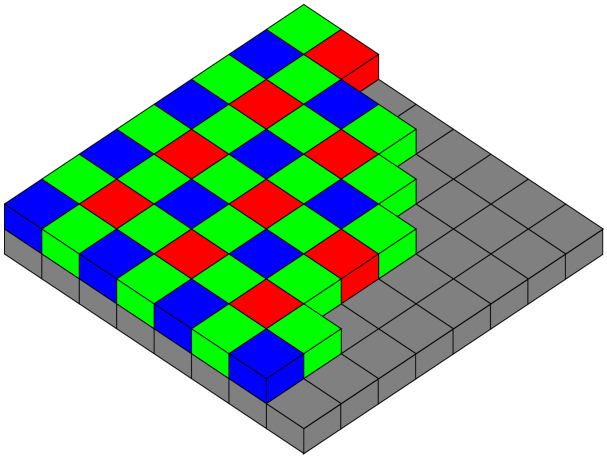

Fig. 1: The Bayer arrangement of color filters on the pixel array of an image sensor [1]

I. INTRODUCTION

With the sensor and the Bayer filter in figure 1, modern cameras can store images in a raw format as in figure 2. The raw image has only one channel which contain color values of Red, Green and Blue with the pattern as in figure 1. In order to achieve a normal image that we often see in our phone or computer, the two missing color values of each pixel need to be interpolated from surrounding pixels.
Building pixel interpolation requires several steps:
Processing the raw image data. Implementing the interpolation methods. Testing the performance of the methods.
All steps are done in python, the code is zipped and submitted together with this report, the implementations of each method are done in separated files which are put and run together in the 'main.py' file.

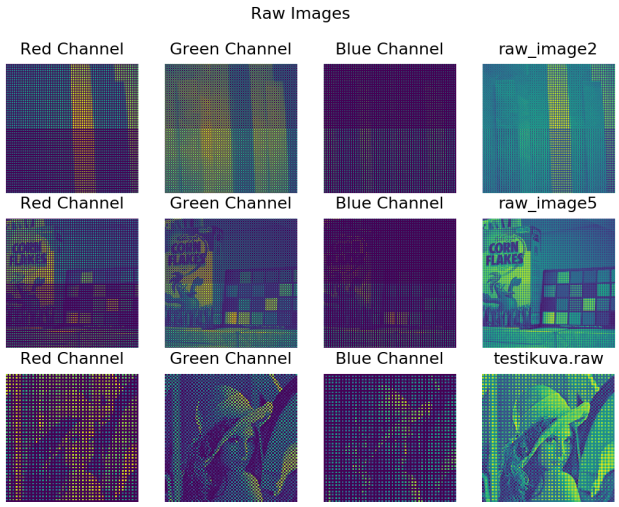

Fig. 2: Raw data from Bayer CFA visualized without interpolation

II. METHODS

A. Nearest neighbor interpolation

B1	G2	B3	G4
G5	R6	G7	R8
B9	G10	B11	G12
G13	R14	G15	R16

Fig. 3: 4x4 Bayer filter matrix for RGB filter array

The nearest neighbor (NN) interpolation is the simplest interpolation method which is very simple to implement and computationally fast. However, the quality of the result is less than other methods.
The missing pixel values are copied from the neighbors. For example, the blue values in the corresponding locations of G2, G5 and R6 in figure 3 are defined by copying them from the pixel B1. The red values for locations B1, G2 and G5 are formed similarly by copying them from R6. The green pixel in B1 is copied from G2 and for pixel R6 the green value is copied from G5. The process is repeated in all 2x2 non-overlapping windows over the image [2]. The results are shown in figure 4.

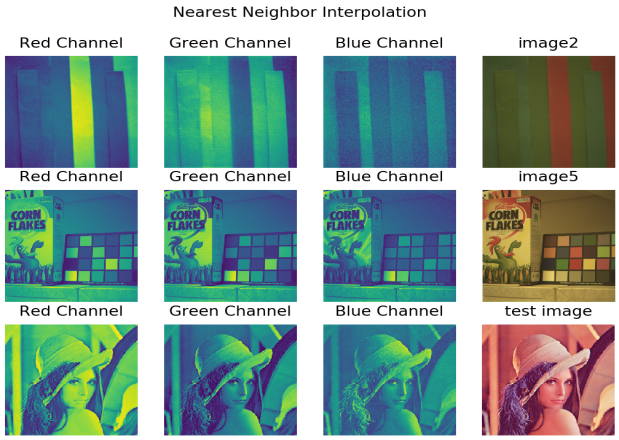

Fig. 4: Results of Nearest Neighbor Interpolation

B. Bilinear interpolation

In bilinear interpolation, pixel values are computed via linear interpolation from surrounding pixels, rather than direct copying. The bilinear interpolation procedure is as follows: in the case of green pixels the value is always the average from surrounding four green pixels. For example G6 = (G2+G5+G7+G10)/4. The blue and red pixels are interpolated in similar manner as an average of the neighbouring two or four pixels, depending on the location within the Bayer matrix. For example, B2 = (B1+B3)/2 and B5 = (B1+B9)/2 are computed as the average of two pixels. The average of four pixels is used to compute, for example, B6 = (B1+B3+B9+B11)/4 [2]. For calculating the border pixels, we pad zeros around the matrix to handle missing values. The results are shown in figure 5.

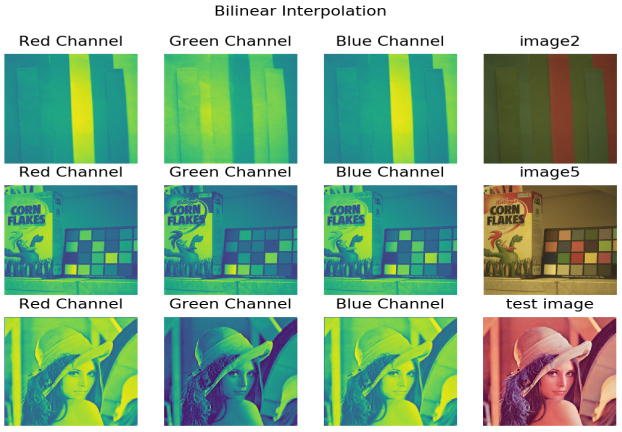

Fig. 5: Results of Bilinear Interpolation

C. Patterned pixel grouping interpolation

The technique implemented in this section is described clearly in `https://sites.google.com/site/chklin/demosaic/`. After implementing and applying the method on 3 raw pictures, the results are plotted together in figure 6.

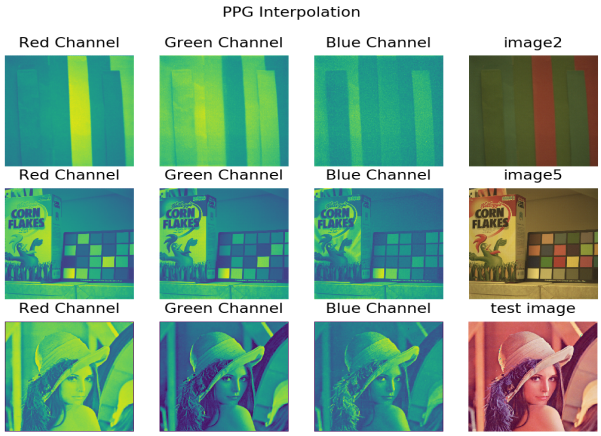

Fig. 6: Results of PPG Interpolation

III. RESULTS AND DISCUSSIONS

A. Visual Comparison

For visual comparison, a magnified area from each of the final RGB images are plotted in figure 7.

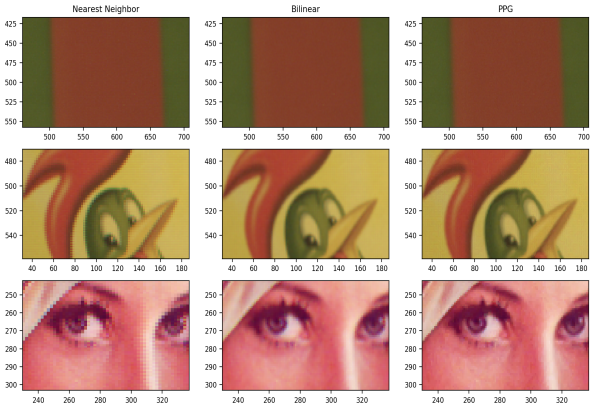

Fig. 7: Magnified Areas for the results of interpolation methods

We observe that the Nearest Neighbor Interpolation gives the worst results while the Bilinear and PPG Interpolation produce somewhat similar images.

B. Performance Comparison

Figure 7, 8 and 9 show the Mean Squared Errors (MSE), Mean Absolute Error (MAE) and time complexities of the 3 interpolation methods, with

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{1}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|(\hat{Y}_i - Y_i)| \tag{2}$$

where the $\hat{Y_i}$ is the i^{th} value of the ground truth and Y_i is the i^{th} value of the result image of the current method.

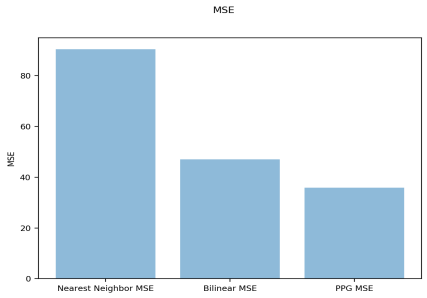

Fig. 8: MSE Comparison

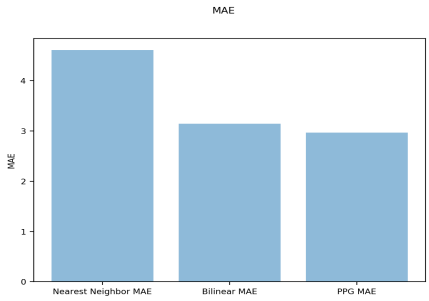

Fig. 9: MAE Comparison

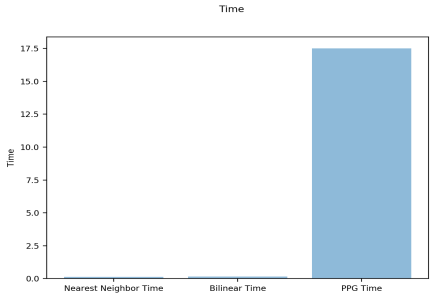

Fig. 10: Time Comparison

For MSE and MAE, the Nearest Neighbor always has the highest error, followed by Bilinear and PPG. The MAE errors for Bilinear and PPG are almost equal to each other. However, PPG consumes much more time than other 2 methods: 17.5s compared to 0.14 (nn) and 0.15 (bilinear).

IV. CONCLUSION

In this lab work, we learned how to read raw image data into matrix format, then implement three interpolation methods to produce the RGB images. After that, we compute the Mean Square Error and Mean Absolute Error to evaluate the performance of our methods. We feel that the lab work is very interesting as we learned to solve a classic problem in image processing. The lab is quite time intensive, and we spent around 20-25 hours in total to understand, implement and then write the report.

REFERENCES

[1] https://en.wikipedia.org/wiki/color_filter_array.
[2] Sgn-16006 - lab 04 instruction.