# **COE 3SK3 Course Project 1:**
# Interpolation for Color Demosaicing of Digital Cameras

Alexis Ng

Feb 14, 2020

## 1 Introduction

The problem presented in this project is regarding to the color demosaicing problem. During demosacing, a considerable amount of information is loss in the reconstruction of the original image from a camera raw CFA data. The goal is to investigate and develop an algorithm that can best estimate the original RGB image from a CFA data.

## 2 Tasks and Requirements

a) Using the MATLAB software, a program has been created to generate Bayer CFA patterns. The algorithm used to do this is creating a for-loop that checks each pixel and assign a colour channel that produces the RGGB pattern. Below shows an example image of a raw Bayer mosaic data.

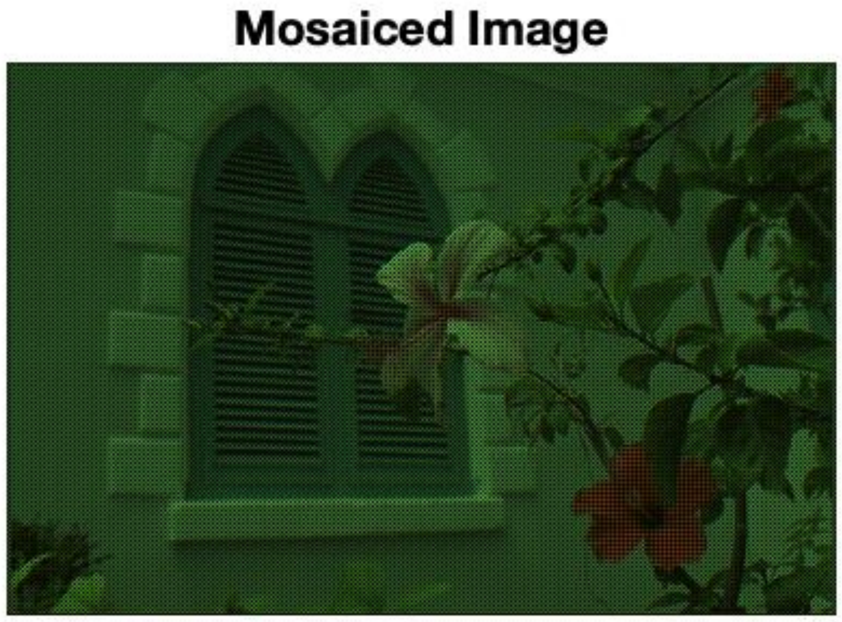

Figure 1

b) The image is reconstructed using the Bayer CFA data from part (a). A few different algorithms have been used to analyze performance.

1) Using a classical method: *Bilinear Interpolation.* The CFA allows one colour channel to be measured and the other two missing channels to be estimated. In bilinear interpolation, the green channel is interpolated by taking the average of the left and right adjacent pixels. For the red and blue channels, the average was calculated using the left and right adjacent pixels for the rows and columns that the channels are on and the average of the 4-diagonal neighbours on the rows/columns that the two channels are not in. The edge cases were calculated using nearest neighbours.

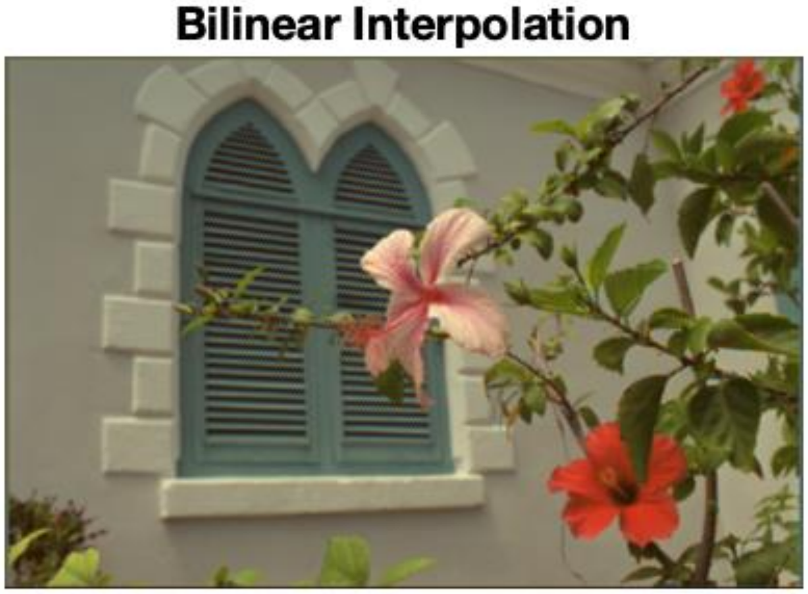

Figure 2

2) A better algorithm to improve the estimate: *The Zhang-Wu Directional LMMSE Image Demosaicking method*. Two papers have been utilized for motivation. [1,2]. This algorithm is outlined as described in the papers:

- The primary difference signals are first estimated in the horizontal direction and vertical direction. In the paper, Zhang and Wu assumes that the primary difference signals are smooth, and that second-order Laplacian interpolation can be used to obtain missing samples:

$$\hat{G}_n = \tfrac{1}{2}(G_{n-1} + G_{n+1}) - \tfrac{1}{4}(R_{n-2} - 2R_n + R_{n+2}),$$
$$\hat{R}_n = \tfrac{1}{2}(R_{n-1} + R_{n+1}) - \tfrac{1}{4}(G_{n-2} - 2G_n + G_{n+2}),$$

  to provide estimate of the difference signal:

$$\hat{\Phi}^h_{g,r} = \begin{cases} \hat{G}_n - R_n & n \text{ is a red pixel,} \\ G_n - \hat{R}_n & n \text{ is a green pixel.} \end{cases}$$

  Convolution is used in the code.

- The estimate is then denoised using LMMSE where the suggested parameters for the window radius M = 4 and using an approximate Gaussian lowpass filter:

$$h(z) = \tfrac{26}{128} + \tfrac{23}{128}(z + z^{-1}) + \tfrac{15}{128}(z^2 + z^{-2}) + \tfrac{9}{128}(z^3 + z^{-3}) + \tfrac{4}{128}(z^4 + z^{-4}).$$

  Convolution was used in the code. It is also noted that the process is the same for the blue row.
- The two denoised directional estimates are then fused to obtain a full resolution green channel by minimizing the estimated mean-square error.
- Finally, the red and blue channels are obtained by interpolating the primary difference signals and subtracting from the green channel where first it is interpolated by their four diagonal neighbours and then averaging their axial neighbours, respectively.

$$\Phi_{g,r}(i,j) = \frac{1}{4}\big(\Phi_{g,r}(i-1,j-1) + \Phi_{g,r}(i+1,j-1) + \Phi_{g,r}(i-1,j+1) + \Phi_{g,r}(i+1,j+1)\big).$$

$$\Phi_{g,r}(i,j) = \frac{1}{4}\big(\Phi_{g,r}(i-1,j) + \Phi_{g,r}(i+1,j) + \Phi_{g,r}(i,j-1) + \Phi_{g,r}(i,j+1)\big).$$

  And the red and blue channels are estimated as:

$$R = G - \Phi_{g,r} \text{ and } B = G - \Phi_{g,b}.$$

  Zhang/Wu and Pascal Getreuer's source codes were also used as an aid to implement the MATLAB code. Noted that the edges were interpolated using linear interpolation. Below shows the result of the MATLAB algorithm:

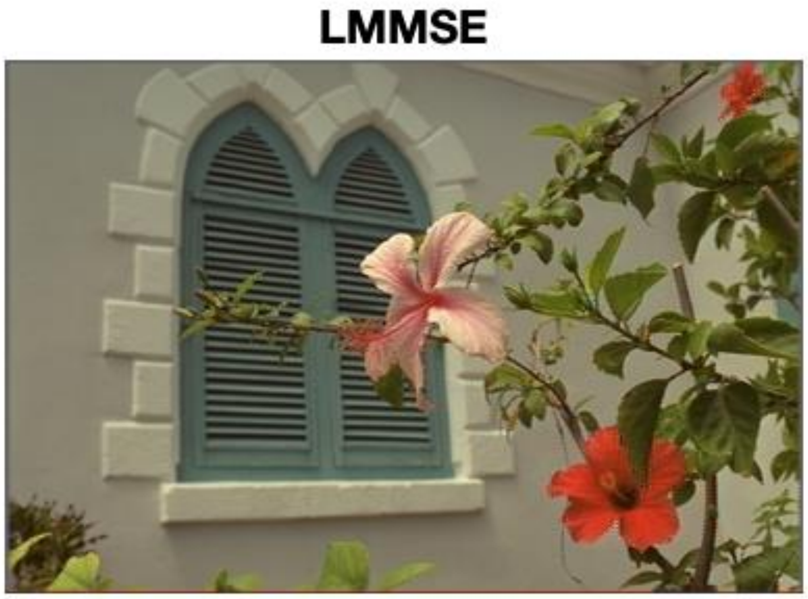

Figure 3

3) Using MATLAB built-in function: “demosaic(image)”

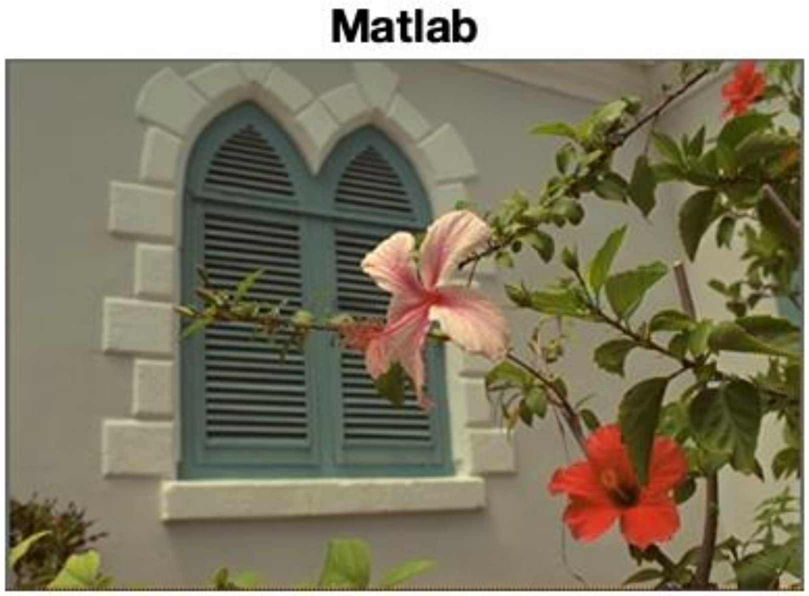

Figure 4

c) Below is a table showing the MSE of the several images of the different algorithms compared to the original image.

| Image | Bilinear | LMMSE | MATLAB |
|---|---|---|---|
| lenna.png | 66.3034 | 34.2319 | 19.5114 |
| kodim05.png | 188.3566 | 85.8719 | 60.2215 |
| kodim07.png | 69.2639 | 36.5267 | 22.9461 |
| kodim21.png | 105.8784 | 34.1551 | 34.2974 |
| kodim23.png | 41.6572 | 28.5493 | 17.6087 |
| lion.png | 101.8609 | 31.1173 | 17.3215 |
| tree.png | 205.6454 | 88.9062 | 78.4965 |

As a result of the calculated MSE, we can see that the LMMSE approach provides a better estimate than the classical approach of bilinear interpolation but not as good as the MATLAB built-in function. And by zooming into the edges of images and the edges of objects within the image, there are still visible artifacts seen from the naked eye. Examples of artifacts are shown in Figure 5 and 6. Artifacts exist in image processing due to the addition of unwanted information or missing necessary information from the representation of the original image. In Figure 5, we can see it is quite noticeable in the bilinear interpolation method, but less in the LMMSE approach and the least in the MATLAB method. This is due to the effectiveness of the how well the missing channels were estimated. An image is well reconstructed based on the algorithm used to estimate the missing colour channels.

Artifact shown in Kodim07

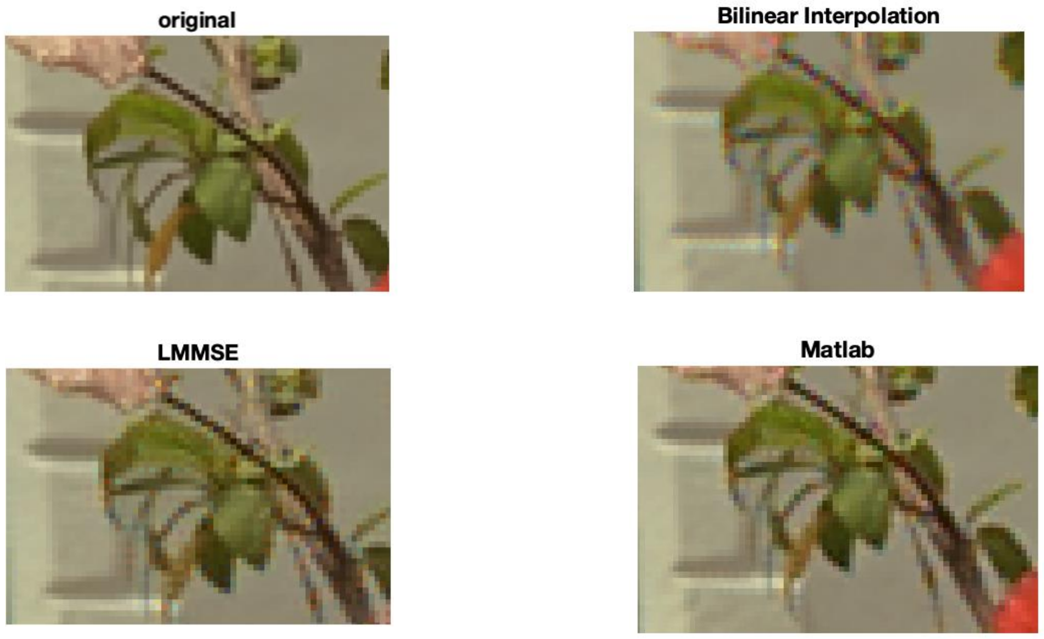

Figure 5

Artifact shown in Kodim23:

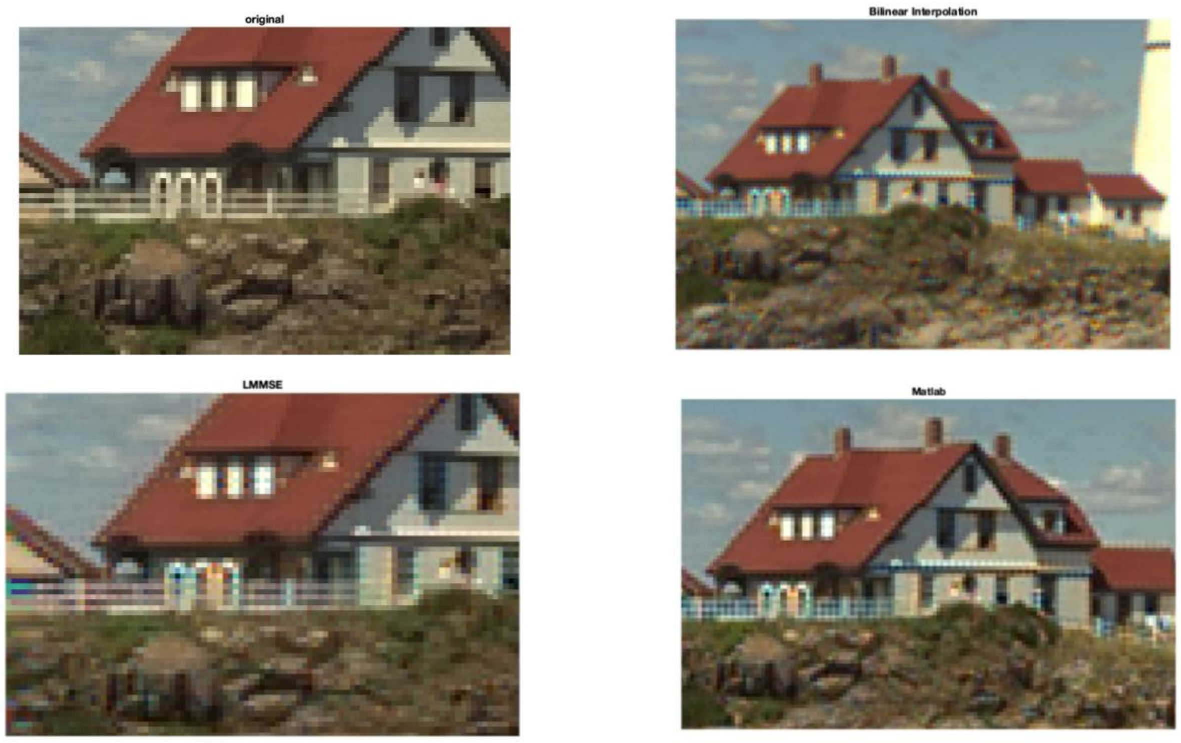

Figure 6

## 3 Image Credits

Kodim07, McMaster Dataset
Kodim21, McMaster Dataset

## 4 References

[1] Lei Zhang and Xiaolin Wu, "Color demosaicking via directional linear minimum mean square-error estimation," in *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2167-2178, Dec. 2005, doi: 10.1109/TIP.2005.857260.

[2] Pascal Getreuer, “Zhang-Wu Directional LMMSE Image Demosaicking,” Image Processing On Line, 1 (2011), pp. 117-126. https://doi.org/10.5201/ipol.2011.g_zwld