BME 590L: Machine Learning in Imaging Final Project

Yuhan Liu, Aoxue Miao, Liangyu Xu
Department of Biomedical Engineering
Duke University

April 29, 2019

Abstract

A camera captures the images using a single channel detector sensor with color filters, followed by a reconstructed process to obtain the three-channel RGB image. The purpose of this project is to develop a sensor measurement pattern for RGB image joined with the reconstruction method. We introduce a convolutional neural network to decide the pattern as well as the reconstructed RGB image along with considerable mean square error compared with the raw RGB images in the data set. The network consists of a physical layer with learnable weights and a reconstruction U-Net. Most cameras used Bayer pattern as sensor measurement layer, while our network produces a DeMU pattern which acts better in specific noise variance range.

1 Introduction

When computers read image data, each pixel consists of three color channels, red, green and blue (RGB). According to this mechanism, modern cameras transform light in the physical world into digital information by sample the light intensity at each pixel with different spectrum frequency. In order to get images with higher spatial, spectral and temporal resolutions, camera sensors contain a pattern combined with reconstruction network to optimize the proportion of each color channel. This technique is known as color multiplexing. Generally, the Bayer pattern is used as sensor measurement in traditional color camera [1]. However, this process will cause low image resolution, because all colors are represented by RGB colors and some of the original full-color components are lost.

The demosaicing process is designed to recover this effect, which is a digital image process method used to recover the incomplete color sampling from the image sensor to the full-color image visualization. There are some demosaicing methods that are widely used, following two basic rules, spatial correlation, and spectral correlation [2]. For example, a method called the edge-adaptive method uses green channel (G plane) as a dominant component in the demosaicing process, making R and B estimation more accurate but may overestimate the error in G plane [3]. Another one is the gradient-based method, first introduced by Hibbard (1995), and it uses the adjacent pixels to interpolate pixel estimation [4]. Other methods like adaptive weighted-edge method [2], which requires more prior neighborhood classification than the normal edge-adaptive method, and local covariance-based method [5], which is built specifically to improve gray image resolution, are also mentioned by Losson (2010) [4].

The cameras have their own built-in firmware to reconstruct the color image, while sometimes it may still get blurry or even aliasing in reconstructed RGB images. In this project, we introduce the convolutional neural network to help us decide the best-trained pattern and recover the three channel RGB image from the sensor measurements based on the previous sensor pattern according to Ronneberger (2015) [6] and Chakrabarti (2016) [7]. This network focuses on image multiplexing pattern as well as the image reconstruction network. We name this network as demosaicing U-Net (DeMU), which consists of a physical layer and a reconstruction U-Net. We compare this DeMU pattern with the traditional Bayer pattern and a recently introduced CFZ pattern [8] to evaluate the performance.

2 Methodology

2.1 Architecture

2.1.1 Physical layer

After importing the three-channel image data, we need to pass them through a physical layer. The architecture of the physical layer is shown in Figure 1, where the fourth channel is the sum of the RGB channels. By introducing a temperature parameter, α, we can train the weights for each pixel through a softmax layer. Then a 4x4 pattern selecting for four channels can be generated and applied to the original image to get the sensor measurement image.

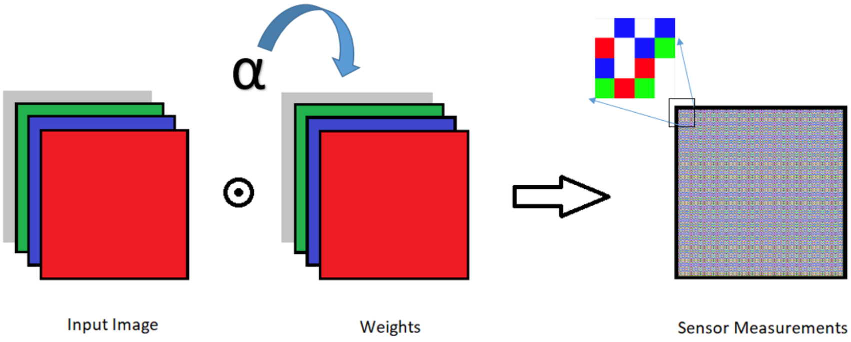

Figure 1: Temperature parameter α is increased across iterations. Sensor measurement is the result of the inner product of input image and weights.

2.1.2 Reconstruction U-Net

U-Net, also known as fully convolutional network, consists of a contracting path and an expansive path [6], as shown in Figure 2. The contracting path has two 3x3 unpadded convolutions each combined with a ReLU operation and a 2x2 max pooling. By changing the filter number, we are able to double the number of channels at each contracting step. The expansive path consists of 2x2 convolution, a cropping concatenation from the corresponding step in the contracting path, and two 3x3 convolutions with the activation function of ReLU. The number of channels can be halved during the up-convolution process. The horizontal cropping recovers the loss of border pixels in each step. After images process through the reconstruction U-Net, the prediction is compared to the ground truth image to get a loss. The training step will be discussed in the next section.

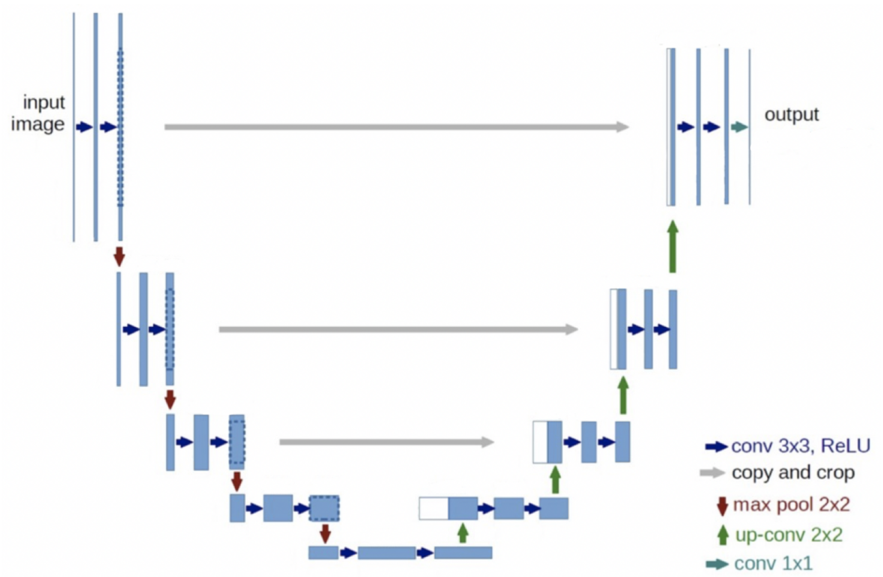

Figure 2: Architecture of the U-Net

2.2 Training

2.2.1 Temperature variable

A temperature variable, α, is utilized in the training process in order to optimize the weights from decimals between 0 and 1 to integers. α is multiplied with the weight, and then the softmax of the product is taken as the final weight. Because α grows larger during each iteration, the softmax result of the product gets closer and closer to 0 and 1. Introducing this temperature variable prevents the mask to be determined when the network is not well trained.

Temperature variable α is determined by

$$\alpha = 1 + (\gamma t)^2$$

, where $\widehat{\text{is}}$ the number of iteration, and γ is a changeable hyperparameter in each training [7]. We use the validation set to choose the best γ which leads to a lower loss. Note that when γ is large enough, the weight will soon reach integer values and remain the same throughout the following training iterations.

2.2.2 Optimization

Adam optimization is implemented in the training network. This algorithm is an advanced type of stochastic optimization that uses the first-order gradient and the second moments of the gradient [9]. This algorithm also has the advantages of high compactness and low memory occupation, so that is efficient in training the desired network.

2.2.3 Loss

Pixel-wise Mean Squared Error (MSE) is used as the loss of the network.

$$Loss = \frac{1}{N}\sum^{N}(\hat{y} - y)^2$$

, where $\hat{y}$ is the predicted value at each pixel, and y is the value of the ground truth. We calculate the MSE for all of the three channels and aim to minimize it. This loss is also taken as the standard to evaluate the networks and weights.

3 Experiment

In our experiment, the sensor pattern is connected with the network one to one. That is to say, if we change the setting of the input training images, like the data set of the images or the variance of the noise, the trained pattern in the physical layer can turn into a totally different array along with the weights in the reconstruction network. It is convenient for us to just evaluate one pattern and it is also impractical to have a different pattern in one camera. So we fix the data set as Funt et al. HDR Dataset [10] and the variance as 0.01 to get the result out in this paper.

The data set that we choose is the Funt et al. HDR Dataset captured with a Nikon D700 digital still camera. Every 9 images in this data set have the same scene but 1 EV (exposure value) difference exposure, which works like data augmentation and can make our training more robust with the illumination. The raw images are then preprocessed to create 16-bit Portable Network Graphics (PNG) format images with the size of 1422x2140x3 which are lossless compressed. As we have hardware limitation, we resize the image into 256x256x3. We use 580 images to train the network, 65 images to validate for hyperparameters and 90 images to test the performance of the network. We then add the Gaussian noise on all of these images with a variance of 0.01.

The input channels of the image we choose are red, green, blue and the sum of these RGB measurements to simulate the unfiltered channel in the camera. Then we apply the normalization on these 4 channels.

The color filter pattern that we use here is a 4x4x4 box that can pick a value in one channel out of four channels. Then this pattern is replicated to the entire image so that we can get the mosaic image in just one channel, which is like the raw image of the color filter camera. When network learning this sensor measurement, we add a scalar soft-max factor α to converge the values on the measurement channels to pick one out of four. We use a quadratic function to express $\alpha_t = 1 + (\gamma t^2)$, where the hyperparameter γ is validated as $2 * 10^{-5}$.

We train our physical layer and the reconstructed layer at the same time with the randomly initialized pattern for 0.4 million iterations. The result is shown in Figure 3, which is named by us as DeMU pattern. Then this pattern is fixed with the changing variance of the noise.

We also compare two patterns with the DeMU pattern. One is the most popular used Bayer pattern, which is 50% green, 25% red and 25% blue. It has twice the green channel because human eyes are more sensitive to this color wavelength. The other one is the CFZ pattern proposed by Chakrabarti (2014) [8]. It utilizes the unfiltered channel as 75% part of the pattern and the rest part is exactly the same as Bayer. We train the reconstructed network for these two patterns the same way as we experiment on the DeMU pattern to compare the performance.

4 Discussion

4.1 The evaluation on the trained sensor pattern

The training process for the output image pattern can be seen as Figure 3, which is trained with the joined reconstruction network. The beginning pattern mixes the values in four channels so that the colors are not pure red, green and blue. But as the temperature parameter α increases, the scaling of the values in each channel becomes larger and larger before soft-max layer picks one value among the four channels. The resulting pattern is called DeMU. We evaluate this pattern by the MSE between the predicted images by the network and the ground truth images in the data set. The MSE for the test data set is 0.0066 which is acceptable.

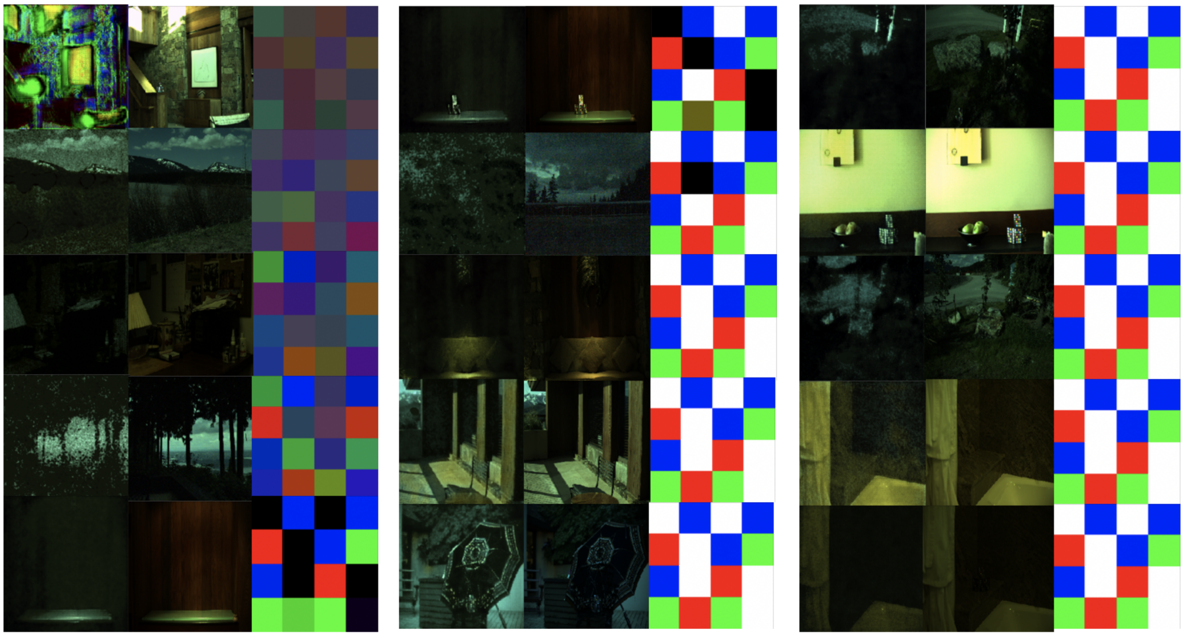

Figure 3: For each set of three images in a row, the first one is the reconstructed image from the network. The second one is the ground truth RGB image. The last one is the corresponding trained sensor pattern. The order of these images are from up to down and then from left to right.

4.2 Compare the DeMU pattern with the Bayer and CFZ

We evaluate the DeMU pattern with the existing Bayer pattern and CFZ pattern with some different noises, and the variances of which are 0.005, 0.01, 0.02 and 0.05. We find out that the DeMU pattern performs the best when the variance of the noise is quite low.

The DeMU pattern does not achieve comparable performance when the variance is greater than 0.02, where the CFZ pattern can have a better MSE. The reason might be that we train this pattern under the variance of 0.01, so that it would perform really reliable around 0.01 variance. Instead, CFZ uses 75% part of the pattern as the sum of three channels, which collects more information than others. Thus it would perform relatively better at higher noise variance.

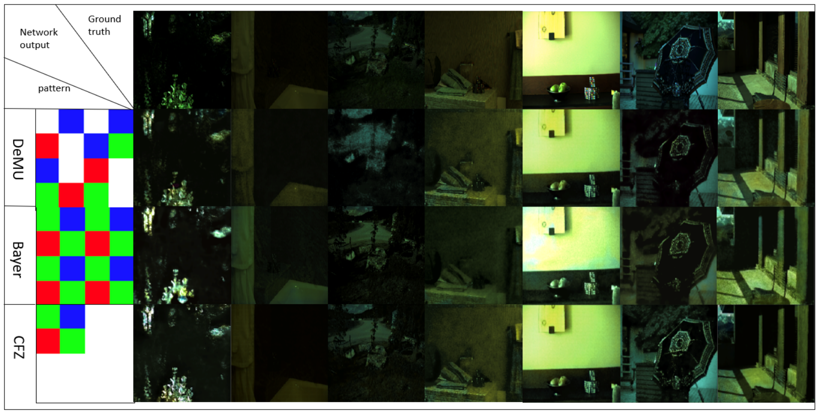

Figure 4: The test output of the reconstructed image from these three sensor patterns.

Table 1: Mean square error between DeMU, Bayer and CFZ

MSE $Pattern$ Var_{Noise}	DeMU	Bayer	CFZ
0.005	**0.0027**	0.0032	0.0032
0.01	**0.0018**	0.0068	0.0023
0.02	0.0044	0.0039	**0.0030**
0.05	0.0047	0.0050	**0.0035**

5 Conclusion

In conclusion, demosaicing can be achieved by training sensor measurement pattern and is dependent on the noise variance. For noise variance under 0.02, DeMU pattern performs the best, while for higher noise variance, CFZ is a better choice. Thus when it comes to reconstructing RGB image, we can first estimate the noise variance and then train the DeMU based on prior information. In this way, we can get the best reconstruction of the RGB images.

If we desire a even higher image quality, we may also introduce data augmentation before applying U-Net, so that the performance of the network can be more robust. In this case, we suggest that the network may have higher accuracy no matter the variety of the noise variance. Moreover, besides the Gaussian noise mentioned in this paper, we can also try some other types of noise to evaluate the DeMU network, for example, the Laplacian noise and the Salt and Pepper noise.

Acknowledgments

Thanks to Instructor Roarke Horstmeyer and teaching assistants Ouwen Huang and Kevin Zhou for their help.

References

[1] Jeon, Jong Ju, Hyun Jun Shin, and Il Kyu Eom. “Estimation of Bayer CFA pattern configuration based on singular value decomposition." *EURASIP Journal on Image and Video Processing* 2017.1 (2017): 47.

[2] Kimmel, R., Sep. 1999. Demosaicing : image reconstruction from color CCD samples. *IEEE Transactions on Image Processing* 8 (9), 1221–1228.

[3] Tsai, C.-Y., Song, K.-T., Sep. 2007. A new edge-adaptive demosaicing algorithm for color filter arrays. *Image and Vision Computing* 25 (9), 1495–1508.

[4] Losson, Olivier, Ludovic Macaire, and Yanqin Yang. “Comparison of color demosaicing methods." *Advances in Imaging and Electron Physics*. Vol. 162. Elsevier, 2010. 173-265.

[5] Li, X., Orchard, M. T., Oct. 2001. New edge-directed interpolation. *IEEE Transactions on Image Processing* 10 (10), 1521–1527.

[6] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*. Springer, Cham, 2015.

[7] Chakrabarti, Ayan. “Learning sensor multiplexing design through back-propagation." *Advances in Neural Information Processing Systems*. 2016.

[8] A. Chakrabarti, W. T. Freeman, and T. Zickler. Rethinking color cameras. *In Proc. ICCP*, 2014.

[9] Kingma, Diederik P., and Jimmy Ba. “Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980(2014).

[10] Funt, B. “Funt Et Al. HDR Dataset.” Funt Et Al. HDR Dataset, 2010, `www.cs.sfu.ca/~colour/data/funt_hdr/`.