1st Assignment 776, computer vision

Biao Jia, 730115162

1 BAYER PATTERN

1.1 CODE

Bayer_pattern.m : input is a raw bayer-pattern image, output is the reconstructed image.

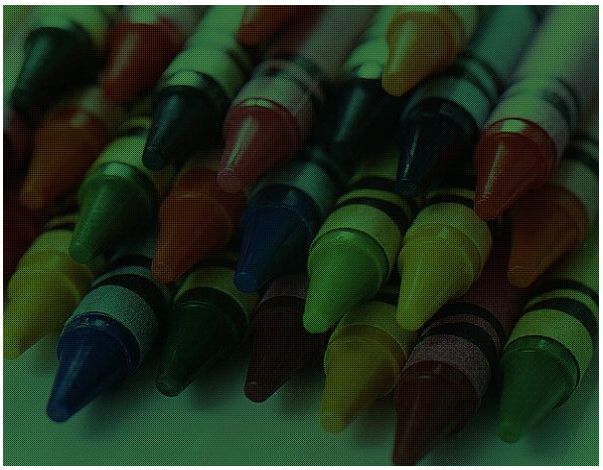
Main_assignment.m : run directly with the picture in the folder.

1.2 EXPERIMENT

Firstly, separate all the different channels into different matrixes. The red channel using the 2d-mask [1, 3, …, end; 1, 3, …, end], green using the 2d-mask [2, 4, …, end; 1, 3, …, end] + [1, 3, …, end; 2, 4, …, end], blue using the 2d-mask [2, 4, …, end; 2, 4, …, end].

After that, using the average values of the around to fill the missing part of each color map.

To check the result, simply add this image together to visualize, which have more green color.



To interpolate to fill the missing pixels, use the 2d filter, for the red channel: [0.25,0.5,0.25; 0.5,1,0.5; 0.25,0.5,0.25], for the blue channel, the same with red one. For the green channel: [0,0.25,0; 0.25,1,0.25; 0, 0.25, 0].

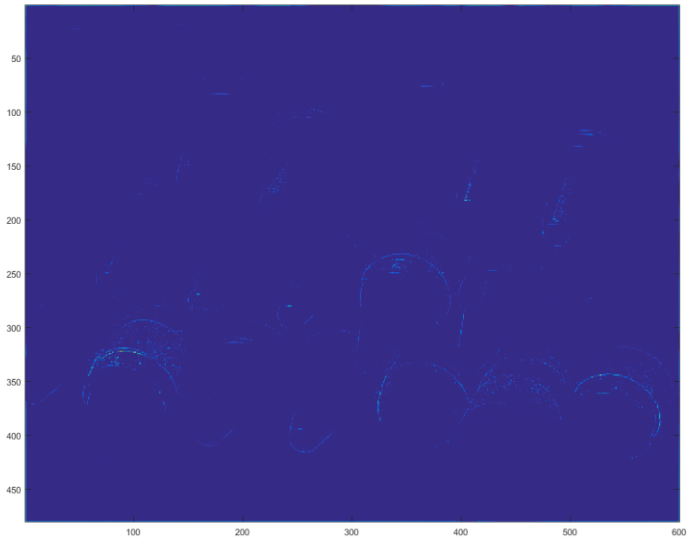
Then apply matlab's imfilter method to the image to get the interpolation's result. Compute the squared difference between the reconstructed image and the original one, and use matlab's imagesc to show. The details like the edges of the original image is lost by the procedure.

After that, I compute the max/mean difference of each channel between the original image and the reconstructed image, the result is listed as follows.

max_diff = [193;155;161]

mean_diff = [1.876375000000000;0.997458333333333;1.940954861111111]

The mean error of the green channel is least, for the sampling rate of the green the is highest.



2 IMAGE ALIGNMENT

2.1 CODE

Main_assignment.m: run directly to show result.

Align_image.m: input is the vertically arranged images. Output is the aligned RGB image.

Align_channels.m: input is two gray image to align, and a shift range. First image is keep still and the second is to be shifted. Output is aligned image and a shift vector.

2.2 EXPERIMENT

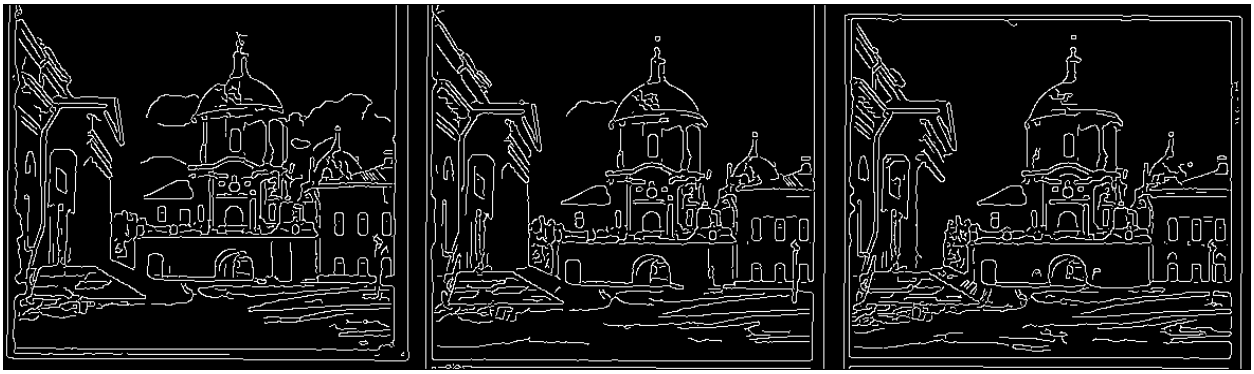
The algorithm is designed by the several processes as follows:

1. Divide the different channels by the average height.
2. Set one channel still, shift other two channels to align it. The range of the movement is set, and a measurement of the match should be set.
3. Stack three channels to form a final photo.

As for the measurement part, I use the edge detection to reduce both the space and time consuming. The measurement is as follows:

1. Extract the edge of both channels. I use Canny edge detection method.
2. logical add two edge image, compute the total amount of the result logical image.
3. The result represents the similarity between the two images.

For the details are kept by the edge detection process, so this method can be used to measure the alignment. The figure shows the edge of the different channel. Though each channel is different, there are still lots of details in common.



3 BONUS PROBLEM

3.1 CODE

Main_assignment.m: run directly to show result.

Align_image.m: input is the vertically arranged images. Output is the aligned RGB image.

Align_channels.m: input is two gray image to align, and a shift range. First image is keep still and the second is to be shifted. Output is aligned image and a shift vector.

3.2 EXPERIMENT

For the high resolution image, it's hard to search all the possible shift. So I follows the instructions to make a pyramid of the different channels. Firstly, I find the shift vector of the top of the pyramid. And then apply that vector to the next layer of the pyramid, and so on.

To build a pyramid of the image, I apply the matlab's imresize method. The 'nearest' method works fine and more fast than the method based on Gaussian filtering.