CSCI435/CSCI935
Computer Vision: Algorithms and Systems

Spring 2017

Assignment One (20%)

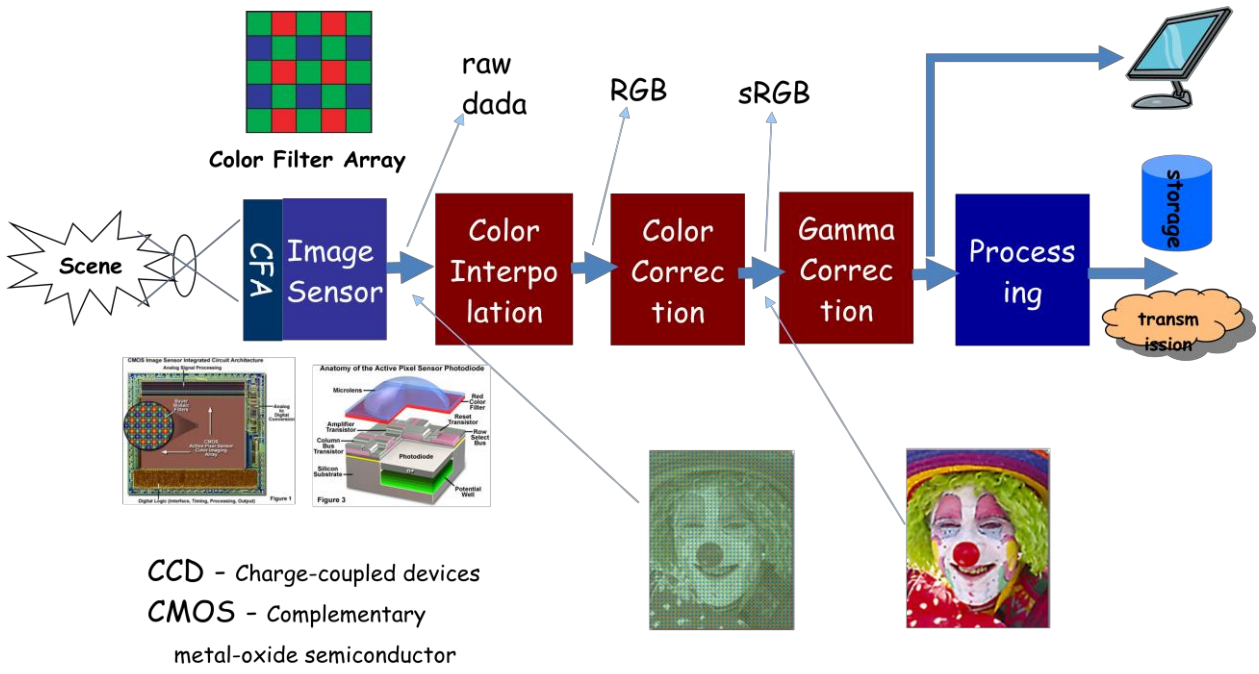
Due Date: 11:59pm 27 August 2017

Objectives

- Getting familiar with OpenCV 3.2.0
- Reading, processing and displaying images using OpenCV 3.2.0
- Understanding the principles of single sensor digital cameras

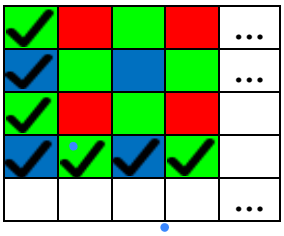
Task One (15%)

Images captured by semiconductor sensors have to be processed before they are passed to later stages in computer vision systems. In this assignment, you are required to **develop a C/C++ program that implements the color image processing chain converting raw image data captured by a CMOS image sensor into true color RGB images.** The chain consists of three components: **color interpolation, color correction and gamma correction** as shown below**.**



You are provided with two bmp files `test1.bmp` and `test2.bmp`, which contain the raw images (Bayer Pattern Color Filter Array data) directly captured by a CMOS image sensor. The

images have resolution 640x480 pixels. The Bayer pattern used in the CMOS sensor is shown below:



Color Interpolation - You are required to implement the **bilinear color interpolation algorithm** to produce an RGB image with the ***same width and height*** as the input raw image. Therefore, you need to take into account boundary conditions, as pixels around the image boundary may not have neighboring pixels on one or two sides. *Try to find the most efficient and simple solution to interpolate boundary pixels*.

Color Correction - Color Correction is a matrix operation. You should use the following matrix that was optimized for this CMOS image sensor:

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1.18 & -0.05 & -0.13 \\ -0.24 & 1.29 & -0.05 \\ -0.18 & -0.44 & 1.62 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Matrix multiplication can result in output values <0 or >255. Thus, you need to check the values and clip them to 0 or 255 if needed.

Gamma Correction - Gamma correction should be implemented using a look-up table. The table entries must be calculated using $\gamma = 0.5$ and implement the transfer function to achieve Gamma correction. You need to find the expression how to calculate and fill the table.

The program should display the raw image (data), color interpolated image, color corrected image and the gamma corrected image in a single window as illustrated below.

Raw Data	**Color Interpolated Image**
Color Corrected Image	**Gamma Corrected Image**

Requirements on coding

1. The program should be named as "**colorChain**" and shall take a BMP image file as the input image, e.g. `colorChain bmpfile`
2. No other third-party libraries should be used in the program except OpenCV 3.2.0. The code has to be in C/C++.
3. The code should be modularized with detail comments AND all source code should be placed in a single file "`colorChain.cpp`" or "`colorChain.c`".

Marking Scheme

1. Zero marks may be graded if your code cannot be compiled.
2. Program structure, comments and usability (3%)
3. Display of the raw image (3%)
4. Generation and display of the color interpolated image (3%)
5. Generation and display of the color corrected image (3%)
6. Generation and display of the gamma corrected image (3%)

Task Two (5%)

You also need to produce a **report of no more than one page** (`report.txt`) which covers the following

1. A brief description of your implementation of the color processing chain on how to deal with boundary pixels, your observations and discussions on the quality of the generated images at each stage.
2. Comparison of the color-corrected and gamma-corrected images generated by running your program on `test3.bmp`, discussion on the differences from `test3.jpg` and possible causes of the differences. Notice that the Bayer raw data file, `test3.bmp`, is simulated from the image `test3.jpg` by sampling the RGB data with the same CFA Bayer pattern as shown above.
3. Comments and discussion on the spectral characteristics of the lighting sources under which images "`light01.jpg`" and "`light02.jpg`" were taken. The images were taken by the same camera with the exactly same settings. You may use sunlight as a reference for comparison.

Submission

Zip the **SOURCE** file and `report.txt` to ***your_login_name.*zip**. The zip file has to be submitted in Moodle

IMPORTANT:

a) ***DO NOT*** *include and submit any object files and images in the zip file. Your submission may not be accepted if you do so.*

b) Submission through email ***WILL NOT*** be accepted

NOTES:

1. SUBMIT AS EARLY AS POSSIBLE. YOU CAN RESUBMIT LATER IF NECESSARY. THE ONE THAT WILL BE MARKED IS ONLY THE LATEST SUBMISSION.
2. SUBMISSION BY EMAIL OR HARDCOPY IS NOT ACCEPTABLE. YOU HAVE TO USE SUBMIT COMMAND TO SUBMIT YOUR WORK.
3. DO NOT FORWARD THE ACKNOWLEDGEMENT FROM THE SERVER THAT YOU WILL RECEIVE AFTER SUBMITTING YOUR ASSIGNMENT TO ANY OTHER EMAIL ACCOUNT, OTHER THAN UNIVERSITY'S EMAIL ACCOUNT. THIS ACKNOWLEDGEMENT WILL BE NEEDED IF YOU WANT TO QUERY ABOUT YOUR ASSIGNMENT. WITHOUT THE ACKNOWLEDGEMENT FROM THE SERVER, YOUR QUERY WILL NOT BE REPLIED.
