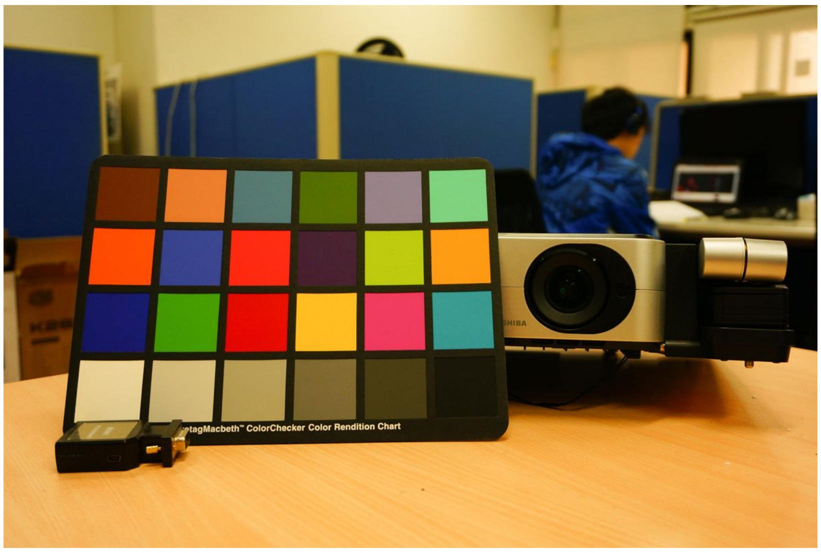
DIP Homework #2

White Balance

In this task, we want to perform white balance on the picture and expect the color checker in the photo could be as similar to real color as possible.



"Pure" White Balance

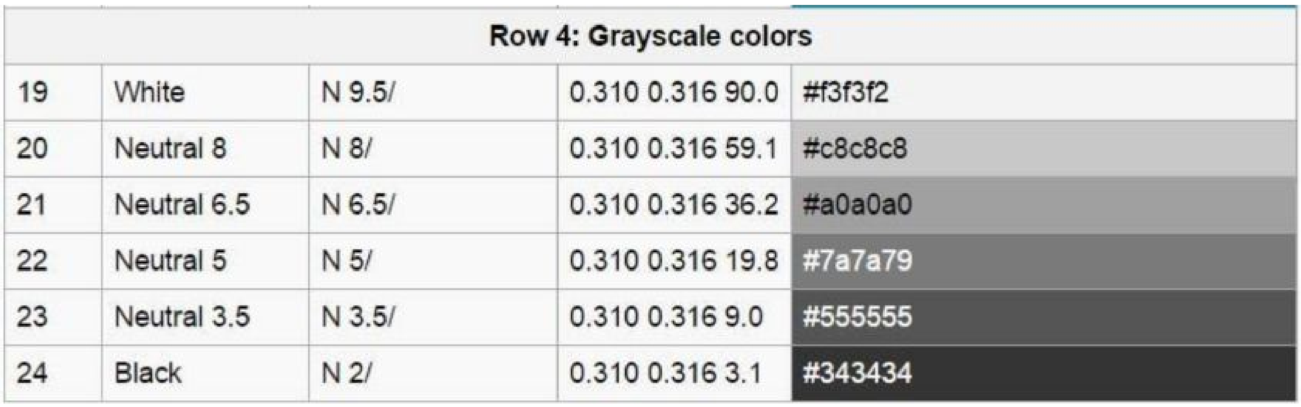
When I see this task at the first time, I come up an idea that just let white value in this picture transform to its real value. I also see the same method to operate white balance in wiki page. Therefore, I start to finish the task with this method.

The algorithm of this job is described here. First of all, choose a color space. There are lots of color space that can represent the color value. I use **sRGB** color space in this homework, but I think other color space, i.e CIEXYZ or CIExyY, can use similar method to achieve same performance. Our input file is bmp file, so we can just read the bmp file and get the **sRGB** of each pixel.

After I get all value of each pixel, I perform the following transformation on every pixel to accomplish "Pure" white balance.

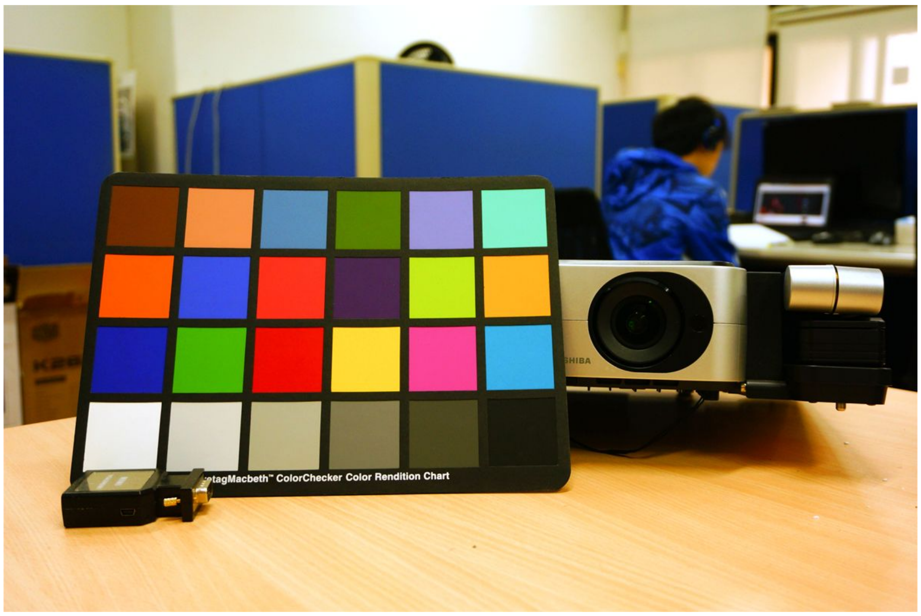
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 255/R'_w & 0 & 0 \\ 0 & 255/G'_w & 0 \\ 0 & 0 & 255/B'_w \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$ (ref: wiki)

where R'_w is red value of the white point in original photo, G'_w is green value of the white point in original photo and B'_w is blue value of the white point in original photo. The (255, 255, 255) is the white value after white balance, but I will use (243, 243, 242) which is white RGB value on color checker. The following table shows the value of different color in different color space. The fifth column means sRGB color space.



Row 4: Grayscale colors				
19	White	N 9.5/	0.310 0.316 90.0	#f3f3f2
20	Neutral 8	N 8/	0.310 0.316 59.1	#c8c8c8
21	Neutral 6.5	N 6.5/	0.310 0.316 36.2	#a0a0a0
22	Neutral 5	N 5/	0.310 0.316 19.8	#7a7a79
23	Neutral 3.5	N 3.5/	0.310 0.316 9.0	#555555
24	Black	N 2/	0.310 0.316 3.1	#343434

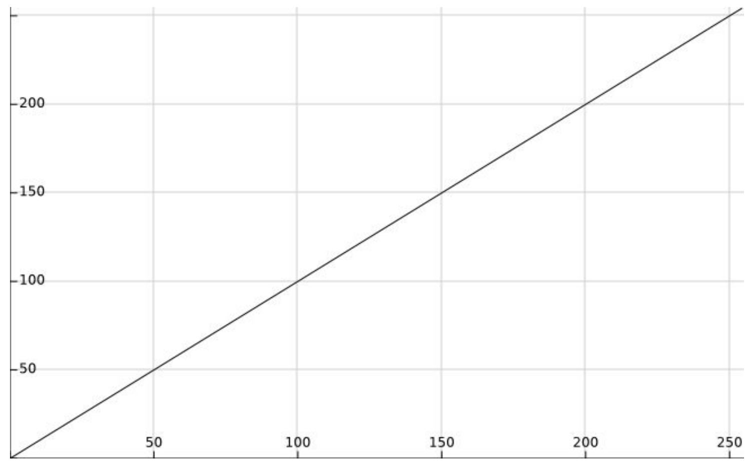
After operating this transformation on each pixel, I get this result:



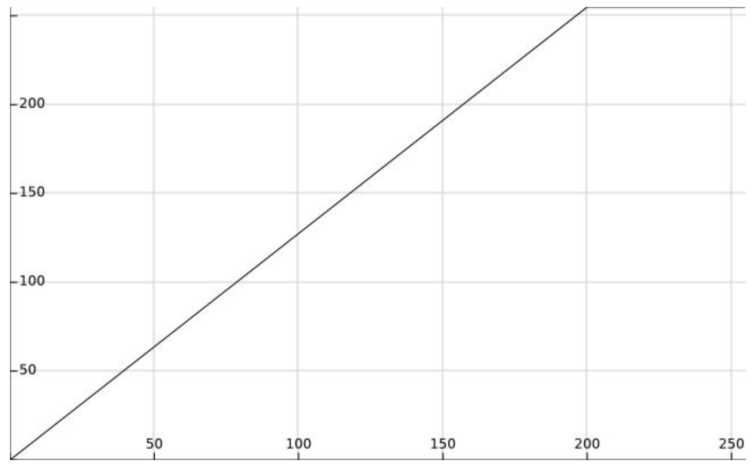
It looks good if we don't observe it carefully. However, if we look closely, we could find a lot of problems on the color checker. Many colors on the color checker are wrong, especially in grayscale colors. Only white and black colors look good in grayscale colors, other colors are totally awful. In order to address this problem, we need a more complex color transformation.

The Problem of "Pure" White Balance

In this part, I would discuss what "pure" white balance do on the RGB value. Initially, we have a linear function that map every value to itself (i.e. 1 map to 1, 255 map to 255). In the other word, this transform do nothing. The following plot shows this function:



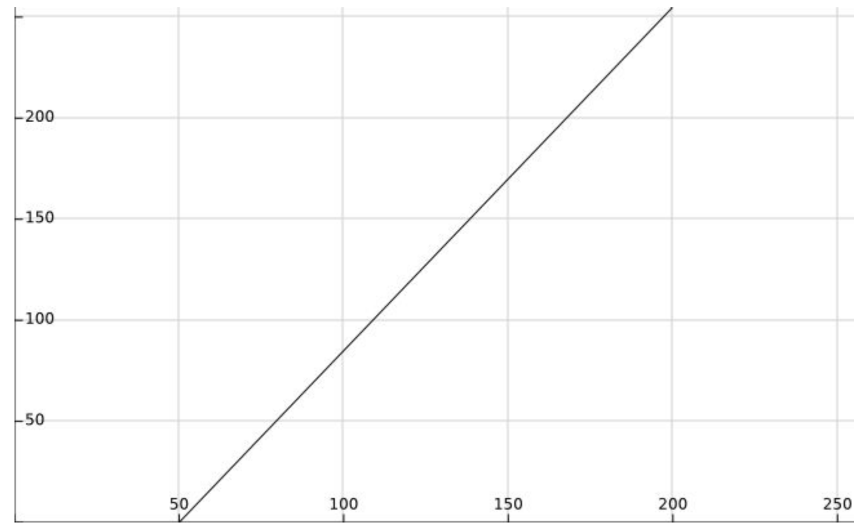
Then we perform "pure" white balance, it will do the following transformation. In this plot, I suppose the white value in original photo is 200 and I want to map 200 to 255 after transformation. This transformation is like:



There is a serious issue in this transformation, it only considers the white point value and just maps the white value in original photo into expected value. In this transformation, the only value we could make sure to be correct is the white value. The other values are depend on the ratio between themselves and white value. If the ratio is not correct, the result would be wrong too.

Figure 1: Two point white balance

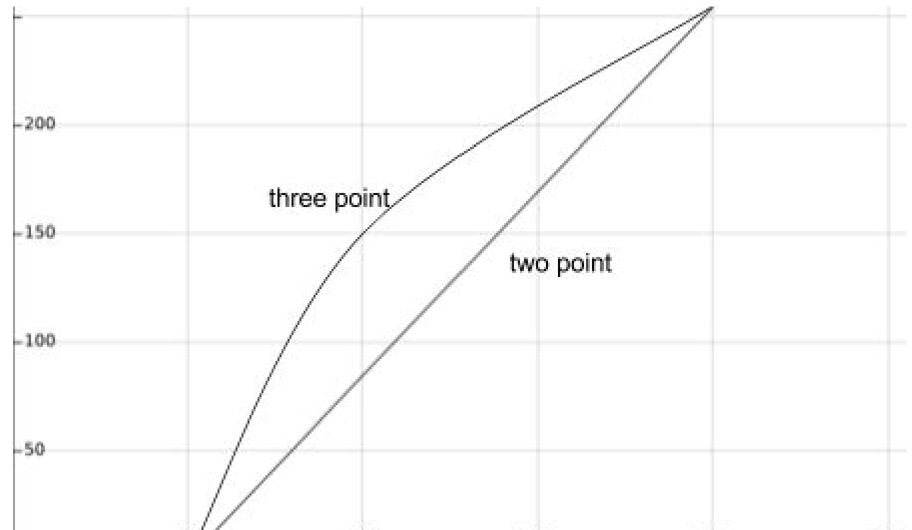
In order to fix the problem in "Pure" white balance, I came up with an idea that use two point to finish this job. I use white point and black point to operate the color transformation. I choose these two points since the value of black point stands for lowest value and the value of white point stands for highest value in the picture typically.



This type of transformation could take care of the ratio between color in original photo and white/black value. However, the same issue still exists. Only white and black color is correct in our control, the other colors are depend on the ratio between themselves and white/black.

Solution: Three point white balance

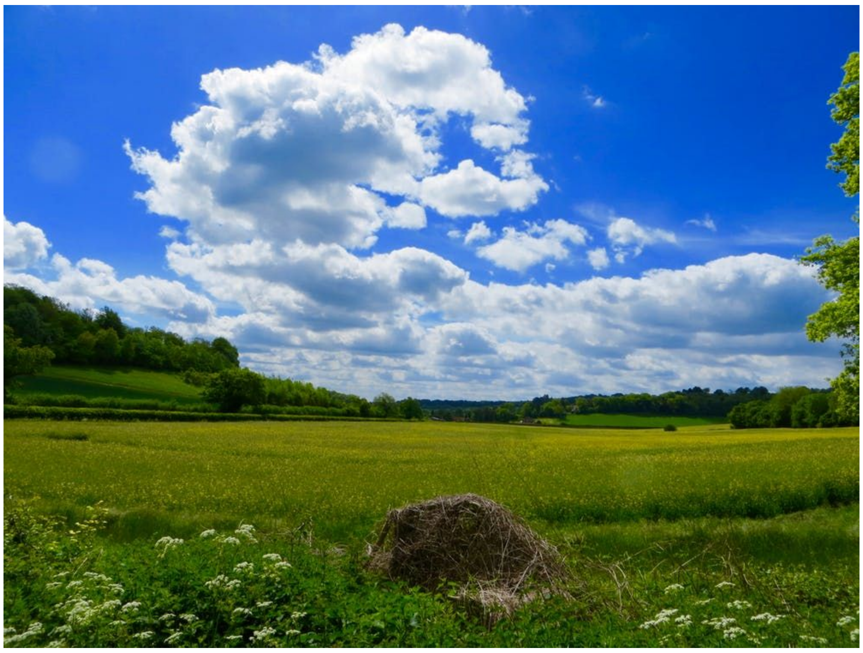
There is a tradeoff in color balance. If we want a perfect solution, we need to take all colors in the picture in consideration and map all colors to their real RGB values. This operation is unrealistic and impossible. Therefore, I would take three points in consideration in order to be realistic and get better performance than previous result. In this step, I will map white, black and gray point to their real values, so this transformation is like the following figure.



Because I use three point to fit the curve, the curve becomes a quadratic function which make the mapping function more smooth. The result is also better than "pure" white balance.

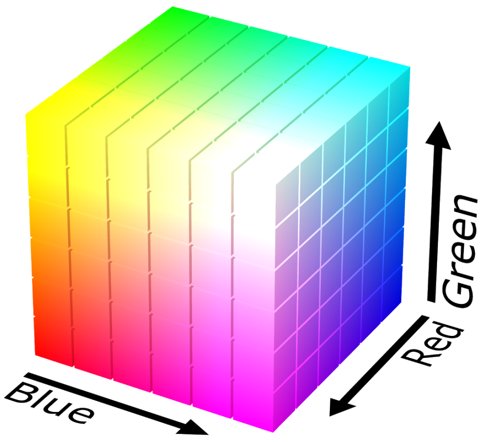
Color Transformation

In this task, I need to change the image below into autumn style. First of all, I analyze some features in this image. The picture is builded up by blue sky, white cloud, large green grassland and tree and brown hay in the middle of grassland.



Then, I check the RGB values of these components. I find I could split them into two clusters by their blue values. One cluster is consisted of white cloud and blue sky, and the other cluster is consisted of grassland, tree and hay. Then, I follow the hint to make the brightness of cloud and sky lower by multiply 0.85 on their RGB values.

For the second cluster, I need to transfer their style to "autumn". For example, map the green grassland into a little bit red grassland. I'm inspired by the figure below. If I change the red and green value in a pixel and fix the blue value, I could change from green line to red line. This way can keep the color of hay because red and green values of the hay are very similar.



Then I get this result: