# Color Correction Using Color Checkers

Senthilkumaran V
{vsk.amcs@psgtech.ac.in}

Department of Applied Mathematics and Computational Sciences, PSG College of Technology, Coimbatore – 641 004[1]

**Abstract.** In this work, we present a simple but a different approach to correct colors of digital photographs. Pictures taken on digital cameras do not portray the actual colors of the photo that a naked eye can see. This is because of the surroundings and the lighting conditions a photo is captured in. The problem of colors is solved here using an external component called a color checker. The algorithm takes 2 inputs and gives a color corrected output which helps photographers and a few other areas of work where pictures play an important role. The method we propose has been tuned and tested on various image data.The paper deals with image processing and machine learning for color calibration and to detect the colorchecker on the target image and uses Python programming language to get the output.

**Keywords:** Color Correction, Macbeth Colorchecker, Xrite, DotProduct, EigenValues, Color Pixel Values, WhiteBalance, ColorCast.

## 1 Introduction

Photographs are a story that we fail to put in words. A picture holds a million emotions. A camera in which the picture is taken is light sensitive [1]. The changes in colors of images are dependent not only on the surface properties of the objects in the frame, but also depends mainly on the lighting conditions which are like the angle the object is placed, the illuminantcolors etc., and on the characteristics of the digital camera we use. Human vision brilliantly captures the true colors but imaging devices do not do so because they cannot easily adapt to the spectral responses to cope up with different conditions and as a result, the digitised image creates an undesirable shift in the entire color range. The great spread of digital cameras, be it Dslr's or a simple phone camera have brought in a new era for digital images depicting a huge variety of subjects. In this frame of reference, we have developed an algorithm that makes it possible to recognize and remove a superimposed color in a digital image. The algorithm is structured in two main parts:computing the color correction matrix and subtracting or adding it to the captured image. Few other methods to this problem and the advantages of this method are discussed in this paper. Future developments are also illustrated.

Color analysis deals with a major problem which is controlling the lighting condition while clicking a picture.A particular color in the target frame appears to be measured as a different color for pictures taken under the same environment. Fields like healthcare, space research, food industries and other color-managed areas apply color calibration on a daily basis.This work deals with a physical object that helps correct colors called the colorchecker. The ColorCheckerColor Rendition Chart, this very well known chart with an array of 4 x 6 color patches, is an icon of the imaging industry. It was formally presented in a 1976 article by

ICCAP 2021, December 07-08, Chennai, India

DOI 10.4108/eai.7-12-2021.2314537

C. S. McCamy and his colleagues from the Macbeth Company, a Division of Kollmorgen Corporation at the time [4].

## 2. Background Study

The methods used in this paper revolve around colors, lighting and the algorithm is all about matrices and eigenvalues.

### 2.1 The XRiteColor Checker

The XriteColor Checker is a device (color rendition chart) used in industries to solve the problem of color correction. It is originally known as the Macbeth color checker. It is a color calibration target of a rectangular framed cardboard arrangement of 24 squares (6x4) of painted samples/colors/shades. This color checker will take care of the white balance too. We have a neutral grey scale for white balance ( it corrects the light and make it as white as possible).Each of the 24 color patches represents the colors of natural objects, such as sky blue, flesh tones and leaf green and each patch reflects light just like its real world counterpart [1]. Each square is individually colored using a solid tone to produce pure, flat, rich color without dots or mixed tints. Figure 1 shows the XRiteColor Checker.


Figure 1. XriteColor Checker

### 2.2 White Balance

Whenever we do a photoshoot, we expect the pictures to come out realistically, mostly similar to how we see it in person. For faithful reproduction, we adjust or remove unrealistic color casts, so that objects which appear white in person are rendered white in our photo is known as the White balance. Relative warmth or coolness of white light is referred to as color temperature. The camera's white balance has to take into account the "color temperature" of a light source.Cameras can create unsightly blue, orange, or even green color casts because our eyes are excellent at judging what's white under different light sources, but digital cameras often have great difficulty with auto white balance (AWB). A white balance

setting in a professional camera is a numerical value (Kelvin temperature). In order to improve our photos under a wider range of lighting conditions, understanding digital white balance plays a major role thereby avoiding these color casts [7].

## 3. Related Works

There are many other methods that can solve the problem of color correction which is discussed in this section. They have their own advantages and disadvantages over the algorithm discussed in this paper, which depends on the necessity to use photographs and digitise images.

### 3.1 Adobe Lightroom

We take a raw picture of the target and the reference XriteColorChecker and then upload it to the adobeLightroom. We then convert the RAW file to DNG using a DNG Converter. Then open the NG in the colorchecker calibration app and we save it as a profile. Importing that profile into lightroom is the major step and we then apply it to all the pictures. We can also check white balance if needed and at the end we copy the profile to all the related pictures that needs to be color corrected [6]. This is indeed a good way to correct colors and it produces the best results, but this depends on a few external software which makes us propose this method described in our paper.

### 3.2 Pantone Card

Pantone Card is similar to an XriteColor Checker which works with our phone cameras to measure and match colored objects, materials, and surfaces to their Pantone Color equivalents. Placing the color correction card over the shade that needs to be matched is the first step. Then we open Pantone's smartphone app on our mobile divide and take a snap of the card.The app is designed such a way that it automatically performs color matching and then returns the most similar shades [5]. This can be used to correct colorstoo the same way we use the color checker. Figure 2shows the Pantone card.

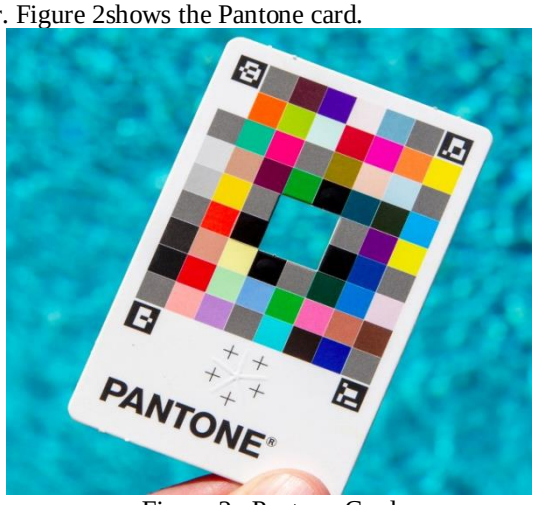


Figure 2. Pantone Card

### 3.3 Supervised Learning for Color Calibration

The image has a set of known color regions like a color checker and we need to find the actual colors of the image. The actual colors of the 24 squares can be extracted from the image using Python's OpenCV. We take it as input, x. We know the colors of these 24 squares from the Macbeth ColorChecker chart. We take it as output y. Using the extracted color of as input(X) and actual colors of the known color region as output(Y), we need to find the transformation relationship (function) between X and Y.Using the transformation relationship, we can calibrate the color of the unknown regions in the image. This can be taken as a machine learning's regression problem that used regression analysis because the values are continuous. This method uses multivariate regression for further results [3].

## 4. Dataset

In order to color correct images and produce best results, the algorithm needs 2 inputs, one image is the target image with the color checker, and the other is the ideal colors checker image, provided both the inputs are under the same lighting conditions. We can compare both the data using the pixel values stored as csv format and then correct the color of the output image. This method has been tried on a variety of objects and has been trained accordingly. The RGB values of a color checker under ideal conditions are shown in Figure 3.

| ColorChecker 2005 | | xyY (CIE D50) | | | L*a*b* (CIE D50) | | | Adobe | | | Apple | | | ProPhoto | | | sRGB | | | sRGB (GMB) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. | Color name | x | y | Y | L* | a* | b* | R' | G' | B' | R' | G' | B' | R' | G' | B' | R' | G' | B' | R' | G' | B' |
| 0 | illuminant | 0,3457 | 0,3585 | 100 | 100 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 1 | dark skin | 0,4316 | 0,3777 | 10,08 | 37,99 | 13,56 | 14,06 | 107 | 82 | 70 | 94 | 63 | 51 | 81 | 67 | 54 | 116 | 81 | 67 | 115 | 82 | 68 |
| 2 | light skin | 0,4197 | 0,3744 | 34,95 | 65,71 | 18,13 | 17,81 | 184 | 146 | 129 | 183 | 128 | 109 | 159 | 135 | 114 | 199 | 147 | 129 | 194 | 150 | 130 |
| 3 | blue sky | 0,2760 | 0,3016 | 18,36 | 49,93 | -4,88 | -21,93 | 101 | 122 | 153 | 74 | 103 | 139 | 94 | 102 | 133 | 91 | 122 | 156 | 98 | 122 | 157 |
| 4 | foliage | 0,3703 | 0,4499 | 13,25 | 43,14 | -13,10 | 21,91 | 95 | 107 | 69 | 73 | 89 | 48 | 75 | 86 | 55 | 90 | 108 | 64 | 87 | 108 | 67 |
| 5 | blue flower | 0,2999 | 0,2856 | 23,04 | 55,11 | 8,84 | -25,40 | 128 | 127 | 173 | 110 | 108 | 162 | 118 | 111 | 154 | 130 | 128 | 176 | 133 | 128 | 177 |
| 6 | bluish green | 0,2848 | 0,3911 | 41,78 | 70,72 | -33,40 | -0,20 | 129 | 188 | 171 | 84 | 178 | 155 | 127 | 168 | 157 | 92 | 190 | 172 | 103 | 189 | 170 |
| 7 | orange | 0,5295 | 0,4055 | 31,18 | 62,66 | 36,07 | 57,10 | 201 | 123 | 56 | 211 | 102 | 30 | 167 | 118 | 54 | 224 | 124 | 47 | 214 | 126 | 44 |
| 8 | purplish blue | 0,2305 | 0,2106 | 11,26 | 40,02 | 10,41 | -45,96 | 77 | 92 | 166 | 52 | 71 | 156 | 79 | 74 | 145 | 68 | 91 | 170 | 80 | 91 | 166 |
| 9 | moderate red | 0,5012 | 0,3273 | 19,38 | 51,12 | 48,24 | 16,25 | 174 | 83 | 97 | 180 | 59 | 79 | 141 | 83 | 80 | 198 | 82 | 97 | 193 | 90 | 99 |
| 10 | purple | 0,3319 | 0,2482 | 6,37 | 30,33 | 22,98 | -21,59 | 86 | 61 | 104 | 73 | 42 | 88 | 68 | 49 | 82 | 94 | 58 | 106 | 94 | 60 | 108 |
| 11 | yellow green | 0,3984 | 0,5008 | 44,46 | 72,53 | -23,71 | 57,26 | 167 | 188 | 75 | 145 | 177 | 39 | 144 | 170 | 74 | 159 | 189 | 63 | 157 | 188 | 64 |
| 12 | orange yellow | 0,4957 | 0,4427 | 43,57 | 71,94 | 19,36 | 67,86 | 213 | 160 | 55 | 220 | 143 | 19 | 181 | 152 | 60 | 230 | 162 | 39 | 224 | 163 | 46 |
| 13 | blue | 0,2018 | 0,1692 | 5,75 | 28,78 | 14,18 | -50,30 | 49 | 65 | 143 | 26 | 47 | 131 | 57 | 50 | 120 | 35 | 63 | 147 | 56 | 61 | 150 |
| 14 | green | 0,3253 | 0,5032 | 23,18 | 55,26 | -38,34 | 31,37 | 99 | 148 | 80 | 60 | 133 | 54 | 85 | 123 | 69 | 67 | 149 | 74 | 70 | 148 | 73 |
| 15 | red | 0,5686 | 0,3303 | 12,57 | 42,10 | 53,38 | 28,19 | 155 | 52 | 59 | 159 | 29 | 43 | 120 | 59 | 46 | 180 | 49 | 57 | 175 | 54 | 60 |
| 16 | yellow | 0,4697 | 0,4734 | 59,81 | 81,73 | 4,04 | 79,82 | 227 | 197 | 52 | 232 | 187 | 0 | 199 | 188 | 66 | 238 | 198 | 20 | 231 | 199 | 31 |
| 17 | magenta | 0,4159 | 0,2688 | 20,09 | 51,94 | 49,99 | -14,57 | 169 | 85 | 147 | 174 | 60 | 134 | 143 | 85 | 127 | 193 | 84 | 151 | 187 | 86 | 149 |
| 18 | cyan | 0,2131 | 0,3023 | 19,30 | 51,04 | -28,63 | -28,64 | 61 | 135 | 167 | 0 | 118 | 154 | 78 | 111 | 148 | 0 | 136 | 170 | 8 | 133 | 161 |
| 19 | white 9.5 (.05 D) | 0,3469 | 0,3608 | 91,31 | 96,54 | -0,43 | 1,19 | 245 | 245 | 242 | 242 | 243 | 239 | 242 | 243 | 240 | 245 | 245 | 243 | 243 | 243 | 242 |
| 20 | neutral 8 (.23 D) | 0,3440 | 0,3584 | 58,94 | 81,26 | -0,64 | -0,34 | 200 | 201 | 201 | 189 | 191 | 191 | 189 | 190 | 191 | 200 | 202 | 202 | 200 | 200 | 200 |
| 21 | neutral 6.5 (.44 D) | 0,3432 | 0,3581 | 36,32 | 66,77 | -0,73 | -0,50 | 160 | 161 | 162 | 144 | 146 | 146 | 145 | 146 | 146 | 161 | 163 | 163 | 160 | 160 | 160 |
| 22 | neutral 5 (.70 D) | 0,3446 | 0,3579 | 19,15 | 50,87 | -0,15 | -0,27 | 120 | 120 | 121 | 101 | 102 | 102 | 102 | 102 | 102 | 121 | 121 | 122 | 122 | 122 | 121 |
| 23 | neutral 3.5 (1.05 D) | 0,3401 | 0,3548 | 8,83 | 35,66 | -0,42 | -1,23 | 84 | 85 | 86 | 65 | 66 | 68 | 66 | 66 | 68 | 82 | 84 | 86 | 85 | 85 | 85 |
| 24 | black 2 (1.5 D) | 0,3406 | 0,3537 | 3,11 | 20,46 | -0,08 | -0,97 | 52 | 53 | 54 | 37 | 37 | 38 | 37 | 37 | 38 | 49 | 49 | 51 | 52 | 52 | 52 |

Figure 3. Color coordinates of an ideal color checker

## 5. Methodology

Different input images are passed through the algorithm written to produce a color corrected output. The algorithm goes through different phases which majorly includes the detection of the color checker and the matrix calculation. The algorithm takes help of image

processing where we have numerical values called “pixels” and they represent an image.These matrix entries are the pixel values which forms a whole image (Figure 4).The algorithm is as follows:

**Input-1:** Picture of the target image with the colorchecker.

**Input-2:** Picture of the colorchecker alone as a reference under the same lighting conditions.

**Step 1**: Take a picture of the target image along with the colorchecker on any digital lens

**Step 2** : Take a picture of the color checker alone under the same surroundings and the same lighting conditions.

**Step 3** : An algorithm to detect the coordinates of the color checker is devised.

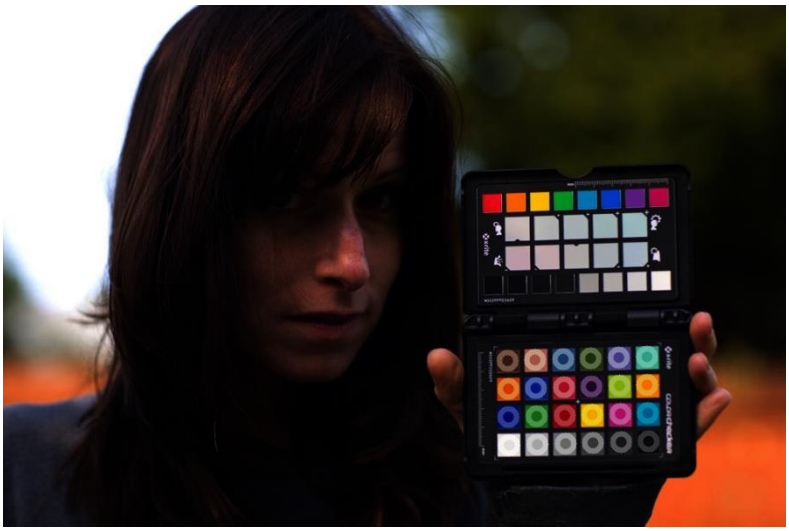

Figure 4. ColorChecker Detection

**Step 4**: Find the color coordinates (or) the r,g,b values of all the 24 colors

**Step 5:** Repeat step 4 and 5 for input-2 also

**Step 6:** Store the pixel values (color coordinates) of both the inputs in a csv file

**Step 7:** Compute the dot product of the 2 pixel valued matrices.

**Step 8:** Correct input - 1 with the color corrected matrix.

**Step 9:** Convert the pixel valued matrix to a corrected image.

**Output:** Color Corrected target image.

The workflow of this paper and the algorithm is depicted as a flowchart in Figure 5.

### 5.1 Color Checker Detection

The target image (input-1) contains the color checker. In order to get the color pixels of each color in the checker, we need to find their coordinates. So we devise an algorithm to find the x, y coordinates and then compute the RGB values at that point. This method will return ‘found’ when it finds a checker on the image and it will return an image overlaid with circles on each patch of the color. The outer circle is the "reference" value, the inner circle is

the average value from the actual image. The coordinates as output is stored as a csv file and its first 24 lines contains the x, y values of each color locations and the average values. [2]

**x, y, r, g, b** (csv format)

In addition to this, the size of the color square of the checker is also stored in the last two lines. The alignment of the color squares are in order of the typical MacBethcolor checker ("dark skin" top left, "black" bottom right).

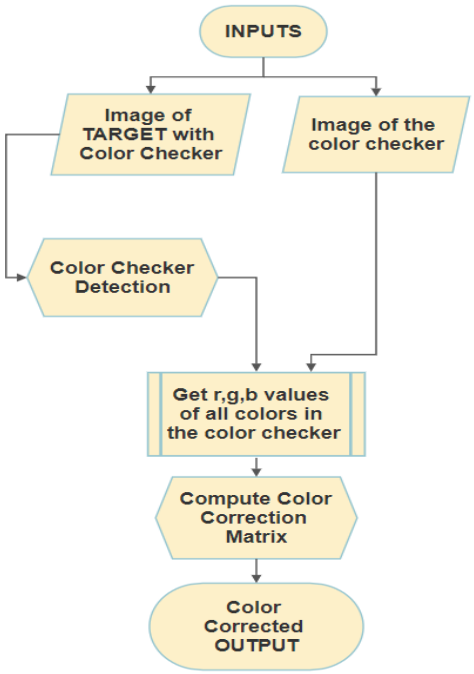


Figure 5. Work-Flow

## 6. Results And Analysis

The aim of this paper is to bring out and reproduce the original colors of an image as seen from a human eye. We compute Color Correction Matrix (CCM) 'A'. We can describe it as a 4x3 matrix A which approximate the following equation:

"Let P be a referencecolor checker matrix (24 x 3) and C be a color checker matrix to correct (24 x 3)."

$$P = [C\ 1]\ A$$

Results observed here in Figure 9 and 10 shows a prodigious difference in the input and output. Figure 6 shows the snippet of the code written in python where the dot product is computed and the color correction matrix is printed. Figure 7 shows the type of variable the CCM is and how the matrix is calculated. Figure 8 shows the format in which the CCM is stored. Figure9 and 10 are calculated and trained on different inputs and that is why the numerical values of the CCM vary.

```
source_xyz_hm = np.append(source_xyz, np.ones((24, 1)), axis=1)
ccm = np.linalg.pinv(source_xyz_hm).dot(reference_xyz)

print('CCM:')
```

Figure 6. Code Snippet

```
PS D:\tri3d\webglshift\color_correction\colorcorrectionmatrix> python computeCCM.py data/xrite_photo_actual.csv data/xrite_overlay_rendered.csv ccm_overlay.csv
>>
CCM:
[[  1.05951799  -0.30504165   0.51257974]
 [  0.09436276   1.38234374  -0.54956426]
 [  0.04144909   0.0771739    1.36439536]
 [-13.17272893  11.80768251  26.5275649 ]]
PS D:\tri3d\webglshift\color_correction\colorcorrectionmatrix>
```

Figure 7. Color Correction Matrix

| | A | B | C | D |
|---|---|---|---|---|
| 1 | 1.597667 | -0.30919 | 0.933348 | |
| 2 | -0.53326 | 1.30948 | -1.67546 | |
| 3 | -0.04434 | -0.15581 | 2.182076 | |
| 4 | -23.1037 | -24.7029 | -29.2919 | |
| 5 | | | | |

Figure 8. CSV format of the matrix

| REFERENCE | FORCED INPUT | COLOR CORRECTED |
|---|---|---|
|  |  |  |
|  |  |  |
|  |  |  |

Figure 9 Result

Figure 10 Color Calibration

## 7. Conclusion And Future Work

ColorCheckers, be it any color targets can be captured by digital cameras, lens and other color input bias, and the performing image affair can be compared to the original map, or to reference measures, to check the degree to which image accession reduplication systems and processes compare the mortal visual systems. It also can be used to color correct one print with the map in it (that may have a special color cast, for case thanks to a lighting achromatism difference) to a different" reference" print with the map in it. Because of its wide vacuity and use, its careful design, and its thickness, and because comprehensive spectrophotometric measures are available, the ColorChecker has also been used in academic exploration into motifs similar as spectral imaging.

This work model gives a decent color corrected output taking in account of a few conditions, like lighting, camera angles and more. The method described in this paper uses an external, physical object - a colorchecker which is pretty expensive in the market. So in the future, we can optimise a few steps and also try to create a fully automated way to correct colors instead of depending on a materialistic thing. We should try to reproduce the original colors or try to print the colors rather than buying a whole new product. Also the colorcheck detection algorithm is pretty technical and we can work on improving it. Other methods such as using a pantone card, or using the method of histogram matching or using other unsupervised or deep learning algorithms can be taken into account to yield best results.

## References


[1] Gasparini, F., &Schettini, R. (2003, September). Color correction for digital photographs. In 12th International Conference on Image Analysis and Processing, 2003. Proceedings. (pp. 646-651). IEEE.

[2] Fernández, P. D. M., Peña, F. A. G., Ren, T. I., & Leandro, J. J. (2019). Fast and robust multiple colorchecker detection using deep convolutional neural networks. Image and Vision Computing, 81, 15-24.

[3] Rizzi, A., Gatta, C., & Marini, D. (2003). A new algorithm for unsupervised global and local color correction. Pattern Recognition Letters, 24(11), 1663-1677.

[4] Varghese, D., Wanat, R., &Mantiuk, R. K. (2014). Colorimetric calibration of high dynamic range images with a ColorChecker chart. Proceedings of the HDRi.

[5] Ciocca, G., Marini, D., Rizzi, A., Schettini, R., &Zuffi, S. (2003). Retinexpreprocessing of uncalibrated images for color based image retrieval. Journal of Electronic Imaging, 12(1), 161-172.

[6] Marini, D., &Rizzi, A. (2000). A computational approach to color adaptation effects. Image and Vision Computing, 18(13), 1005-1014.

[7] Barnard, K., Cardei, V., &Funt, B. (2002). A comparison of computational color constancy algorithms. I: Methodology and experiments with synthesized data. IEEE transactions on Image Processing, 11(9), 972-984.