MATLAB CCM Coefficient Generation Script User Guide

This document describes the use of several MathWorks® MATLAB® scripts and demonstrates how they were used to generate color correction coefficients for the Xilinx Color Correction Matrix IP core (CCM). The color corrections are designed to correct an image for four different illumination types: Daylight, Cool White Fluorescent, Hot Fluorescent and Incandescent light and produce an image that has consistent colors for each illumination. The scripts and their use as described in this document are specific to the Xilinx Zynq-7000 All Programmable SoC Video and Imaging Kit (ZVIK) 1080P60 Camera Image Processing Reference Design.

The purpose of this guide is to help the user understand running the MATLAB scripts in this specific design and circumstances. Example input images are provided and this guide walks the user through running the scripts on the example images. This guide is not a tutorial on MATLAB, color correction, or any other training material. The user should be able to apply these scripts in generating coefficients for other applications and image sensors. These scripts are not supported by Xilinx, they are merely provided as-is to assist the user.

Requirements: MathWorks® MATLAB® version 2012.A was used to develop and run these scripts. It has not been tested against any other versions. There is nothing fundamental about these scripts tying them to this version, so a user is likely to be able to get them to work in another version.

Unzip the ZVIK_MATLAB.zip file in a known location on your computer. The following files are included in the Zip file:

1) anneal.m
2) camera_calibration_master_script.m
3) ccm_coeffs_offsets.mat
4) color_patch_averages.mat
5) get_averages.m
6) input_registration.m
7) This user guide
8) There are also four images that were used to generate the color corrections and will be used in this example:
    a. DAY.bmp (daylight image)
    b. CWF.bmp (cool white fluorescent image)
    c. U30.bmp (hot fluorescent image)
    d. INC.bmp (incandescent image)

The basic process is described as follows:

1) A X-Rite ColorChecker® 24 Patch Classic target was placed in a X-Rite Macbeth Judge® II Light Box (www.xrite.com). Note: in order for the scripts to work correctly, the 24 patch target must be placed flat on a gray background. This is required for the de-warping and shading corrections that are included in the MATLAB scripts. Bitmap images were collected using the ZVIK 1080P60 Camera Image Processing Reference Design Graphical User Interface (GUI) for the four different illumination settings (Daylight, Cool White Fluorescent, U30, and Incandescent sources) with the Color Correction Matrix set to “Bypass” (no color corrections applied). The auto exposure must be turned on. Please refer to the included example images.
2) The Camera_calibration_master_script.m executes the following operations:

a. De-warp the image and correct for irregular illumination, such as lens-shading and light drop-off.
b. Determine the average values for the 24 patches at each illumination setting.
c. Generate color correction coefficients for the CCM that result in the best match for the known target values for the 24 patches. Repeat for each of the three remaining example images.

3) The calculated CCM coefficients are now available for the user to program into the IP core, and calculated color corrected output images are displayed for the user.

Detailed Step-by-Step Instructions

This example describes running the scripts using the example images. The input file names are hard coded in the scripts to work with the filenames as shown above. They can be changed in the script by the user if desired.

1) Run MATLAB and change directory to the location with the ZVIK_MATLAB.zip files contents. Run the camera_ calibration_master_script.m file. This will start the input registration script.

2) **Input Registration and De-warping/illumination correction** This script de-warps the image and attempts to get rid of any lens shading and light drop off at the edges of the image. In this script, the user will indicate registration points around the ColorChecker chart image and the corresponding registration point on a diagram of the chart. Refer to Figure 1 below for the following description.

   The four panes of the window on the screen are:

   a) Lower Left: This window shows the full size input image. The blue square box drawn over the image indicates the area of the image that is scaled up in the upper left window.
   b) Upper left: The scaled up portion of the full size image. The location of the window is controlled by the familiar scroll sliders on the bottom and right edge of this image. This is where the user first indicates the location of the registration points on the actual image.
   c) Lower Right: this window shows a diagram of the expected target shape with the location of all the registration points the user must locate in this process.
   d) Upper right: A scaled up portion of the diagram in c) above. The location of the window is controlled by the familiar scroll sliders on the bottom and right edge of this image. This is where the user indicates the location of each corresponding registration point on the image.

   Important note: It is required that for the following instructions, the user:

   a) **Mark all points in pairs: Mark the first point on the actual color chart image in the upper left window. Mark the matching point in the diagram of the chart in the upper right window. Repeat for subsequent pairs of registration points**

b) **Mark all 20 points in exactly the order shown in Figure 3**

Using the mouse, adjust the sliders on the upper left image to display the upper left corner of the Chart. Adjust the sliders on the upper right image to display the same area of the diagram as shown in the lower right pane. Move the mouse pointer to the white crosshair target in the upper left corner of the upper left image and left click the mouse. The system will place a marker at this location with the number "1" as seen in Figure 2a. Next, use the mouse to select the corresponding location in the upper right image. The image will label this image with a corresponding "1" as shown in figure 2b. Repeat the sequence in the upper left window for location "2" as shown in Figure 2a by moving the mouse to a location between the first and second targets on the top edge in the upper left pane in figure 1. Note: there are only 4 targets on the ColorChecker Chart, so the user needs to make a best estimate of the other locations on the perimeter of the 6x4 color-patch matrix. The locations should be evenly spaced between the targets and at the same distance above/below/left or right of the target edge as the first target. Place the second mark at location "2" in the location shown in Figure 2b. Complete marking the second pair of registration points by placing a mark at the corresponding location in the upper right window.

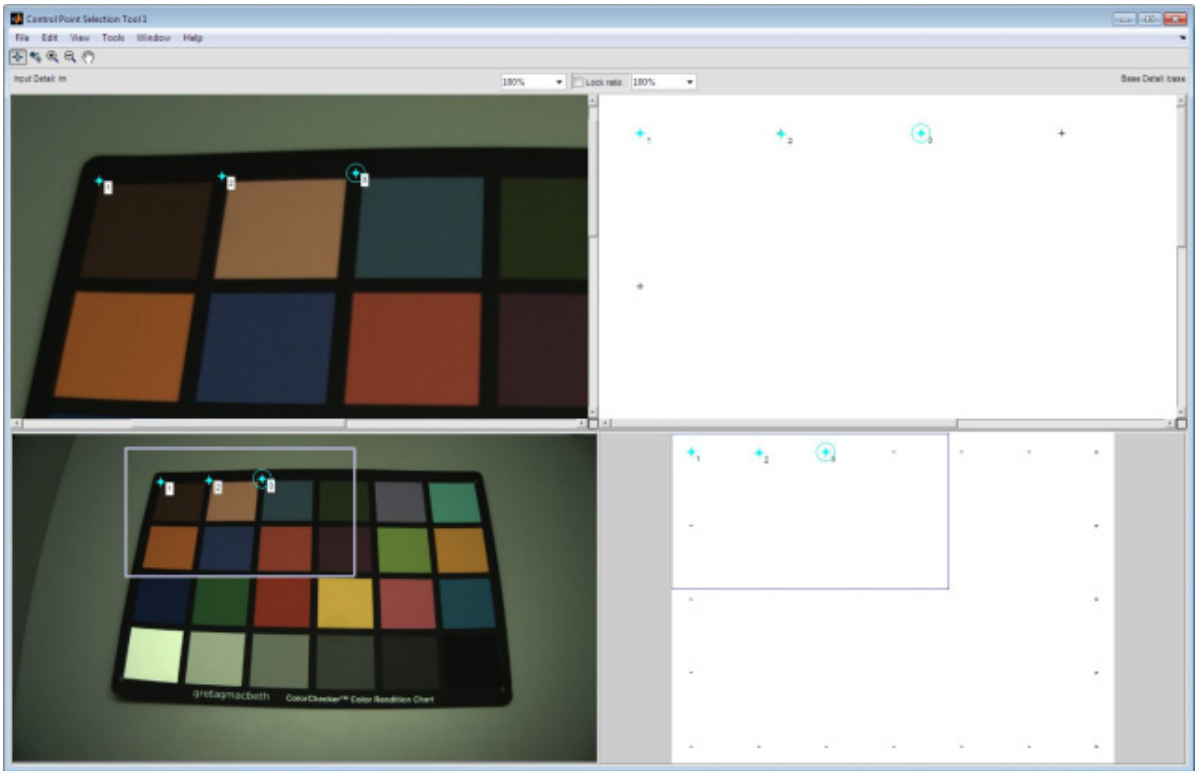
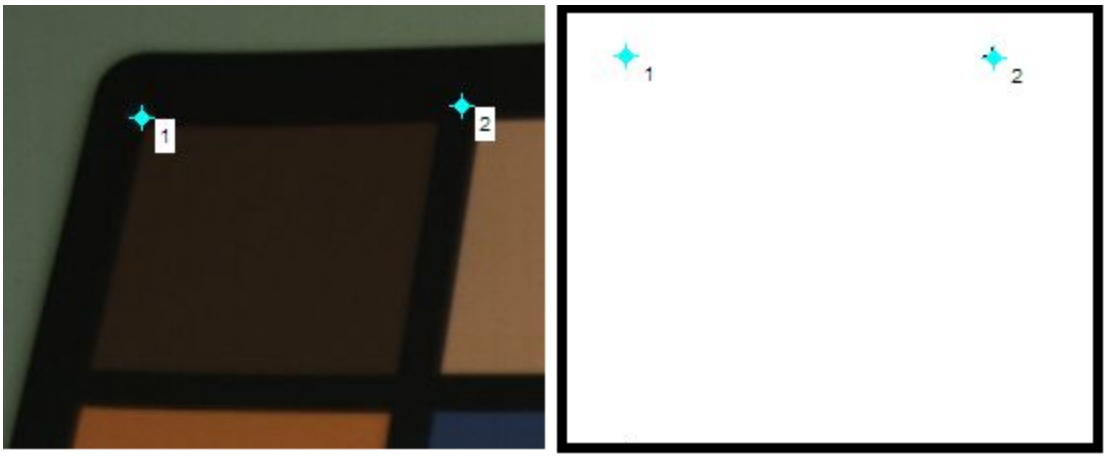


Figure 1) Input registration window



a) Left upper window b) Right upper window

Figure 2) Input Registration window detail

Continue selecting registration pairs as described above. **Important note:** in order for the registration script to work, you must start in the upper left corner and follow the points from left to right and top to bottom until all of the targets are located as shown in figure 3. The completed screen should look like the images shown in Figure 4 and 5. If at any point you make a mistake, close the window, select the MATLAB command window and if the script is still running type CTRL-C to stop the script from processing, clear all variables with the "clear all" command and start over.

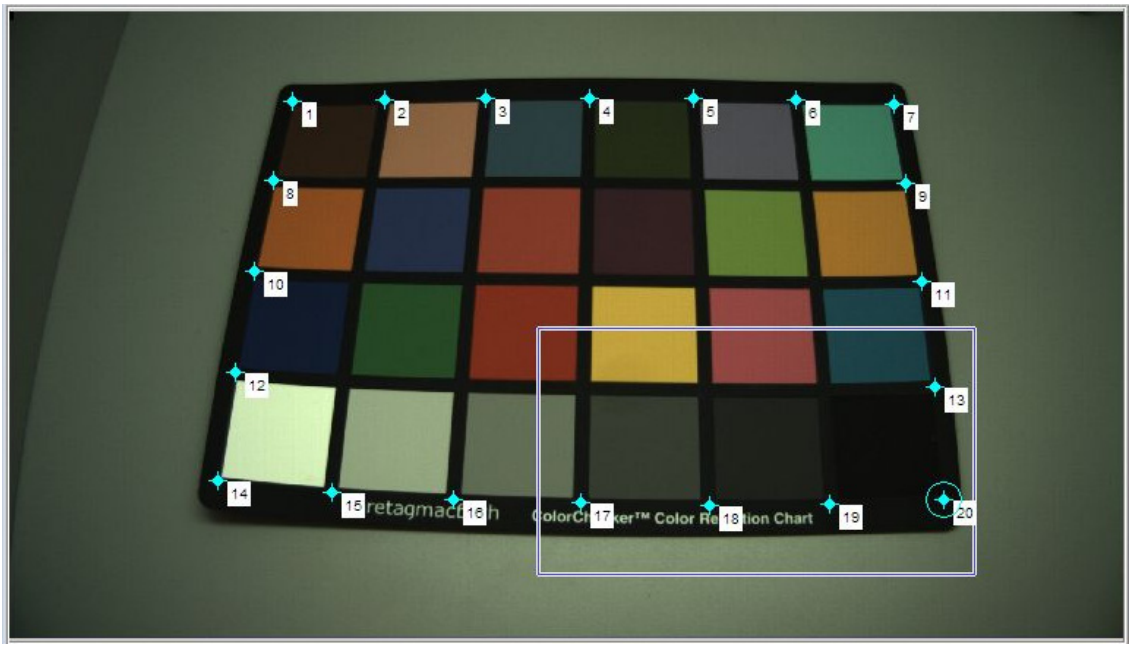


Figure 3) Lower left window, pair order detail

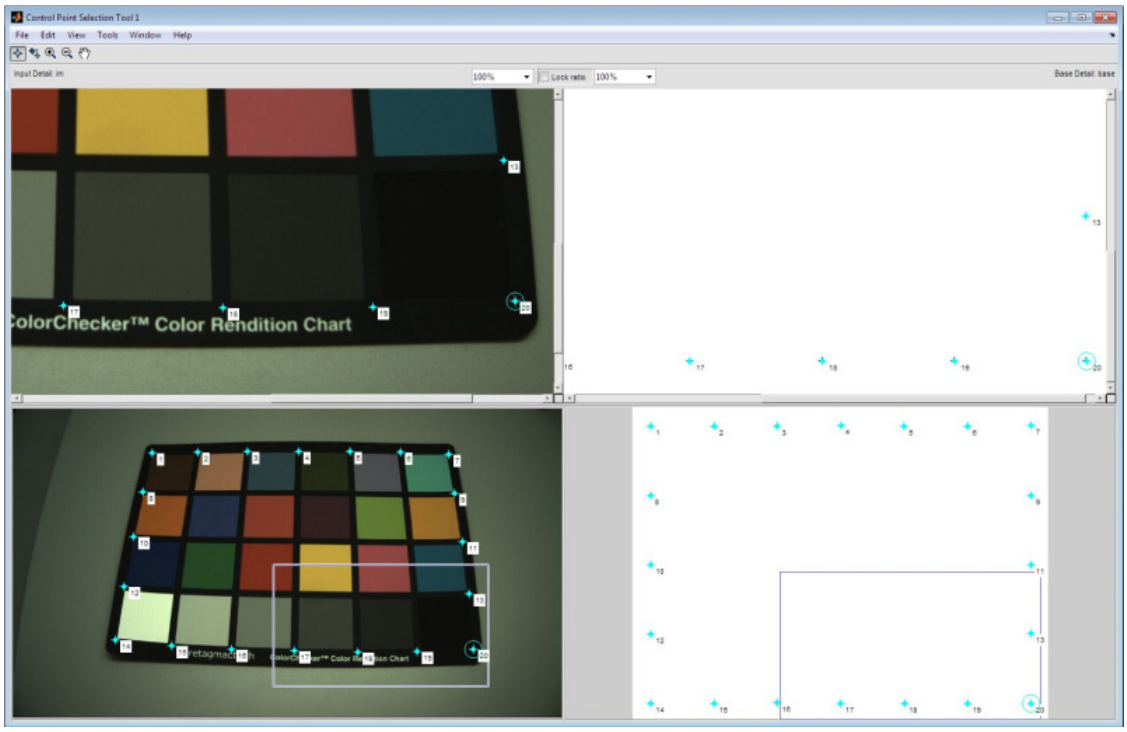


Figure 4) Completed input registration window

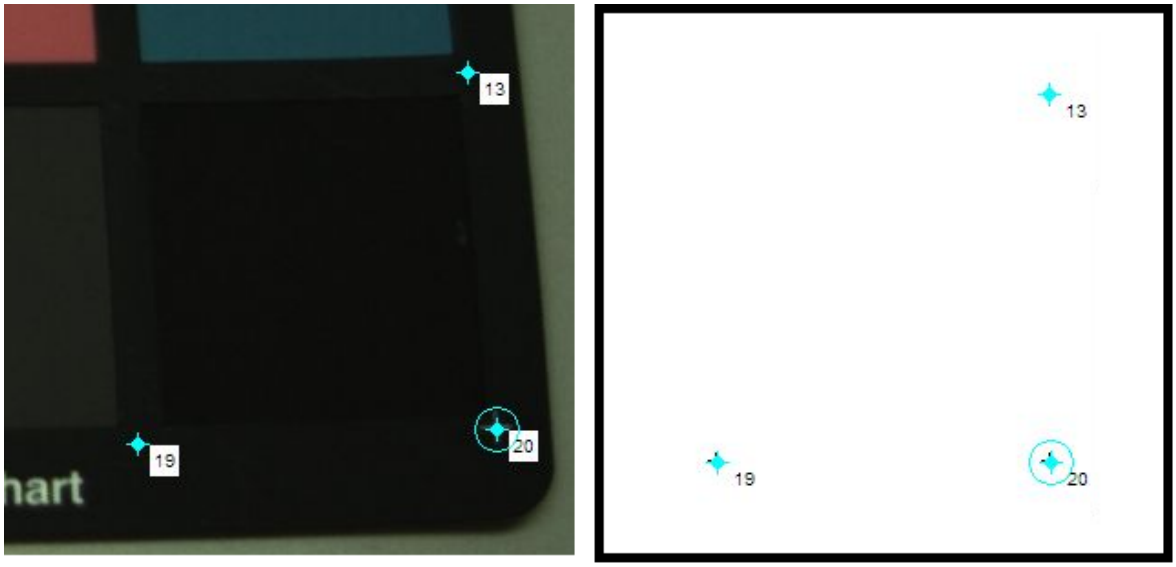


Figure 5) Completed input registration window detail

To complete the input registration process, select File -> Close Control Point Selection Tool and MATLAB will create files which have the same name as the original input files with dewarp_deshade appended to the name.

The algorithm picks gray levels about 50 pixels away from the edges of the chart to assess the gray intensity and adjust the image perspective and intensity accordingly. The 50 pixels distance allows for some shadows that may be cast around the edge of the chart. You may notice that

the fixed pattern noise is amplified in the resulting images, but that will be averaged out in the next step.

3) **Average Values of Color Patches** The next image that is presented to the user is a graphical tool that allows the user to indicate an area of each color patch that will be used to generate average values for the 24 color patches.

Important note: For the following instructions, it is required that the user mark all 24 color patches in order from left to right and from top to bottom as shown in Figure 6.

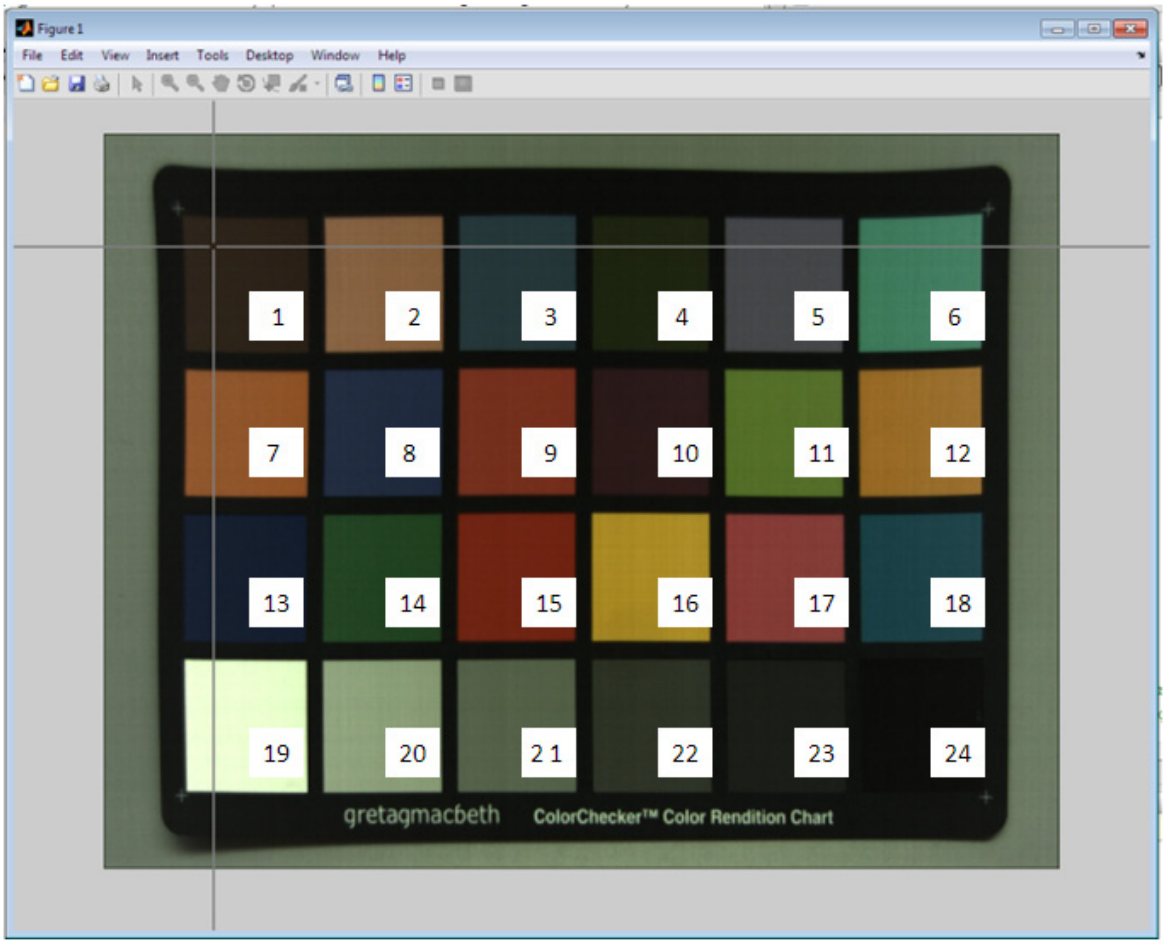


Figure 6) Color Patch Average selection order

Using a mouse, move the crosshairs to an area near the upper left corner of the upper left color patch (brown in this case) as shown in Figure 6. Click the left mouse button. MATLAB will draw a square on the color patch with numbers indicating the average RGB values for the area contained in the square as shown in Figure 7. Repeat this process for each color patch working from left to right and top to bottom until all squares are completed.

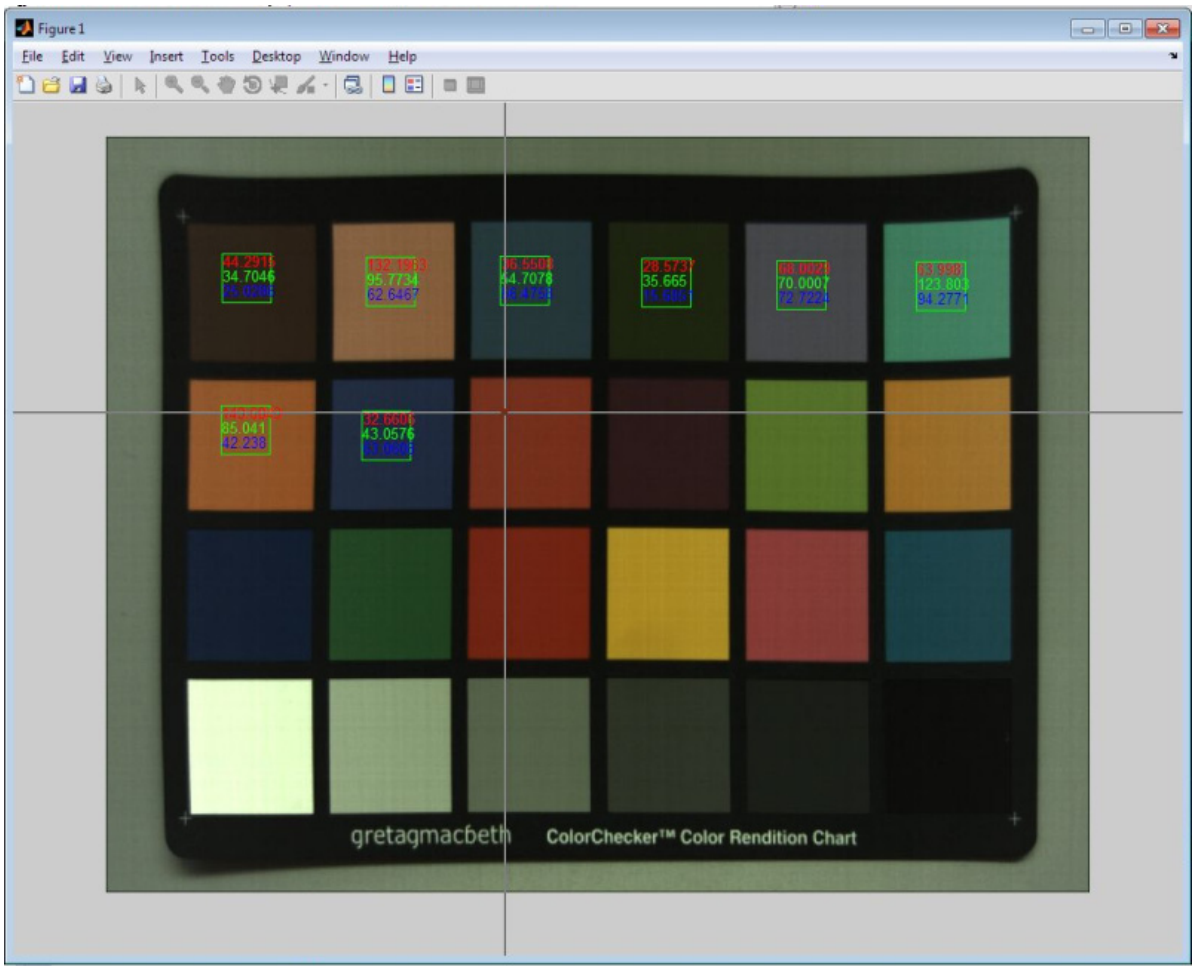


Figure 7) Color patch average values

Four new files are created with the name appended to xxxx_s.png. These are lower resolution versions of the input images. A new file is also created: colorpatch_averages.mat . This data file contains the average measured RGB value for each color patch.

The script next calculates the non-gamma corrected RGB values for each color patch target from the known gamma corrected RGB values that are supplied with the ColorChecker Chart. The target values are already stored in the master script at the top of the file. These could be changed to support other values by the user as desired.

Selection of which color patches to be included in the CCM coefficient optimization process is contained in the variable “range” in the script. The color patches are numbered 1-24 from right to left and top to bottom. The weight of a particular color patch can be doubled by including the index number again in the range variable. It is acceptable to skip patches. Numbers can be included multiple times. In this example, we have included all patches 1-24 and triple weighted patches 21 and 22. There must be at least 4 color patches included in the optimization that are non-white/gray/black, otherwise the optimization may yield gray results.

4) **Determine Color Correction Coefficients** The script now determines coefficients for the Xilinx Color Correction Matrix IP core that will result in an image with the least error distributed over the selected color-patches when compared to the desired target values described in the previous section. As shown in Figure 8, the script opens a window and displays the Original Source image and the Current Best Approximation image with the current coefficients applied. The image changes as the algorithm iterates to optimize the results.

Figure 8) Original Source and Current Best Approximation window

After the first image completes, processing commences on the next image with different illumination in the original window. In addition, a new window opens and the resulting color corrected image is displayed along with space for the three additional future results. Eventually, all 4 color corrected images are displayed side by side as shown in Figure 9.

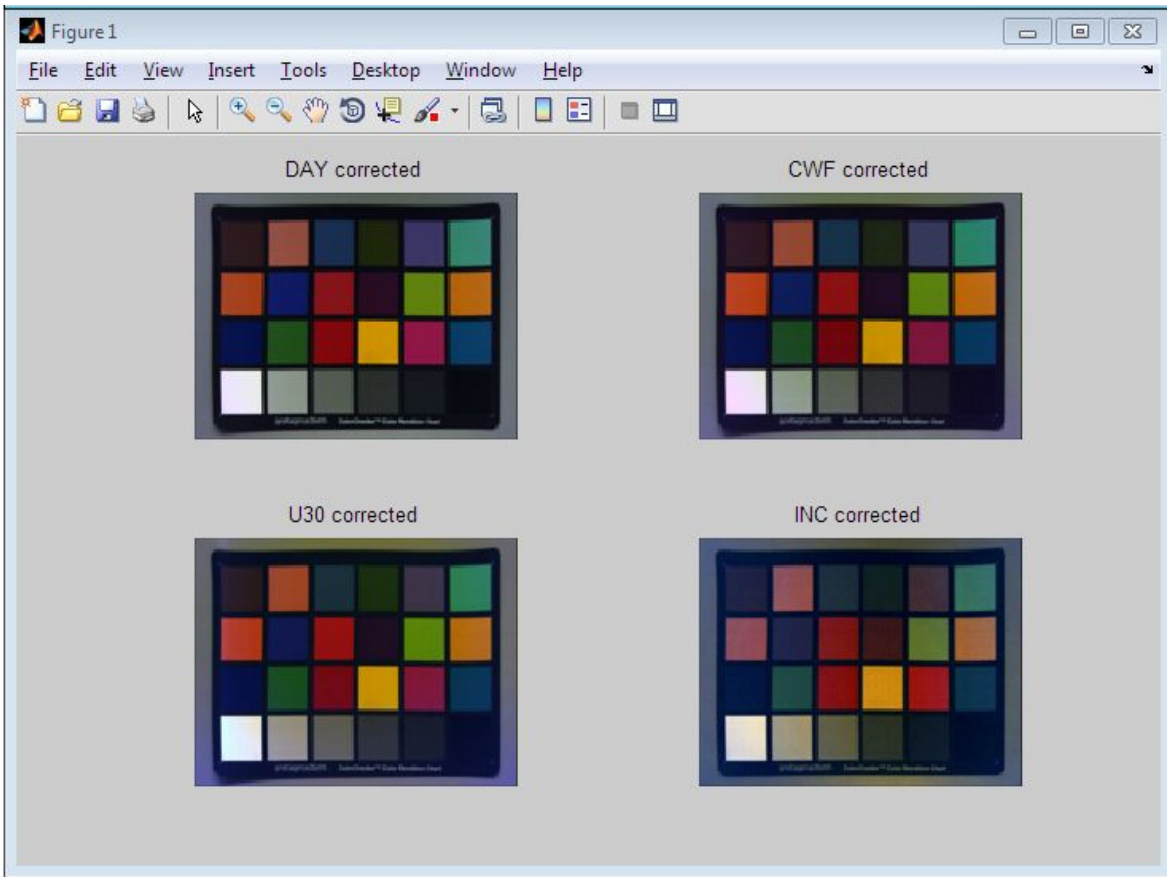


Figure 9) Results for the four different color corrections

The script will save the resulting images as filenames Xxxx_dewarp_deshade_sres.png. These files are low resolution versions of what the color correction results should provide for the same input conditions. The gamma correction is applied by the monitor so that the displayed image should look very similar to the original ColorChecker Chart.

The script writes a new MATLAB file: ccm_coeffs_offset.mat. This file contains the CCM coefficients and offsets. Matrix 1 corresponds to the first file input (called DAY in this example). Matrix 2 corresponds to the second (CWF in this example), etc.

Finally, type the following commands in the MATLAB Command Window to display the color correction coefficient values for the 4 different color corrections:

```
load ccm_coeffs_offset.mat
CCM{:}
```

This displays all of the coefficients for the user. The data displayed in columns 1, 2 and 3 represent the 3x3 CCM coefficient matrix. The final column is the offset matrix.

The camera_calibration_master_script also prints out quantized, integer coefficients suitable to program the CCM core in a camera calibration system.