Table of Contents

Calculate normalized RGB values 1
Lab Step 4 - Calculate normalized Y values 2
Graph RGB Versus Grey Y values 2
Linearize Camera RGB Output and Plot 3
Camera RGB Visualization 4
Derive 3x3 Transformation Matrix 6
Calculate CIELab and DeltaEAB of ColorChecker for Estimated XYZs 7
Visualize using chromatic adaptation 7

Calculate normalized RGB values

Using Readings from Jenee's 15" 2010 matte Macbook Pro

```
%Manual entry of RGB values taken from photoshop
cam_RGBs = [101,71,57;216,166,143;98,127,169;82,108,63;148,140,197;...

 120,219,199;216,120,44;57,73,160;214,88,100;82,42,95;169,213,74;...
    242,184,61;29,33,124;55,140,66;188,42,57;245,229,63;206,75,153;...
    31,129,166;233,238,232;194,200,196;151,153,152;105,109,108;...
    62,63,65;31,32,36]';

%Normalize RGB values
cam_RGBs = cam_RGBs./255;
fprintf('cam_RGBs = \n');
disp(cam_RGBs);

%Grab final 'row' of RGB values for colorchecer chart
%Sort from black to white
cam_gray_rgbs = cam_RGBs(:,19:24);
cam_gray_rgbs = fliplr(cam_gray_rgbs);
fprintf('cam_gray_rgbs = \n');
disp(cam_gray_rgbs);
```

```
cam_RGBs =
  Columns 1 through 7

    0.3961    0.8471    0.3843    0.3216    0.5804    0.4706    0.8471
    0.2784    0.6510    0.4980    0.4235    0.5490    0.8588    0.4706
    0.2235    0.5608    0.6627    0.2471    0.7725    0.7804    0.1725

  Columns 8 through 14

    0.2235    0.8392    0.3216    0.6627    0.9490    0.1137    0.2157
    0.2863    0.3451    0.1647    0.8353    0.7216    0.1294    0.5490
    0.6275    0.3922    0.3725    0.2902    0.2392    0.4863    0.2588

  Columns 15 through 21
```

```
    0.7373    0.9608    0.8078    0.1216    0.9137    0.7608    0.5922
    0.1647    0.8980    0.2941    0.5059    0.9333    0.7843    0.6000
    0.2235    0.2471    0.6000    0.6510    0.9098    0.7686    0.5961

  Columns 22 through 24

    0.4118    0.2431    0.1216
    0.4275    0.2471    0.1255
    0.4235    0.2549    0.1412

cam_gray_rgbs =
    0.1216    0.2431    0.4118    0.5922    0.7608    0.9137
    0.1255    0.2471    0.4275    0.6000    0.7843    0.9333
    0.1412    0.2549    0.4235    0.5961    0.7686    0.9098
```

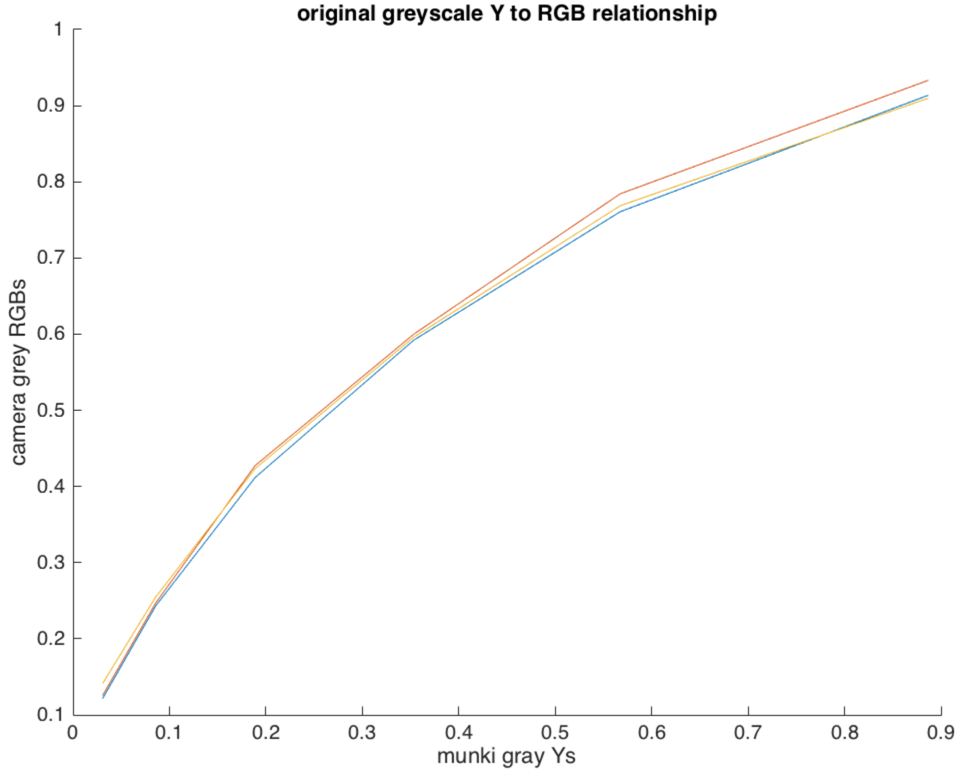
Lab Step 4 - Calculate normalized Y values

```
%Calculate normalized Y values for patches 19-24
munki_values = importdata('munki_CC_XYZs_Labs.txt');
munki_gray_Ys = fliplr(munki_values(19:24,3)'./100);

fprintf('munki_gray_Ys = \n');
disp(munki_gray_Ys);
```

```
munki_gray_Ys =
    0.0307    0.0858    0.1889    0.3534    0.5674    0.8868
```

Graph RGB Versus Grey Y values

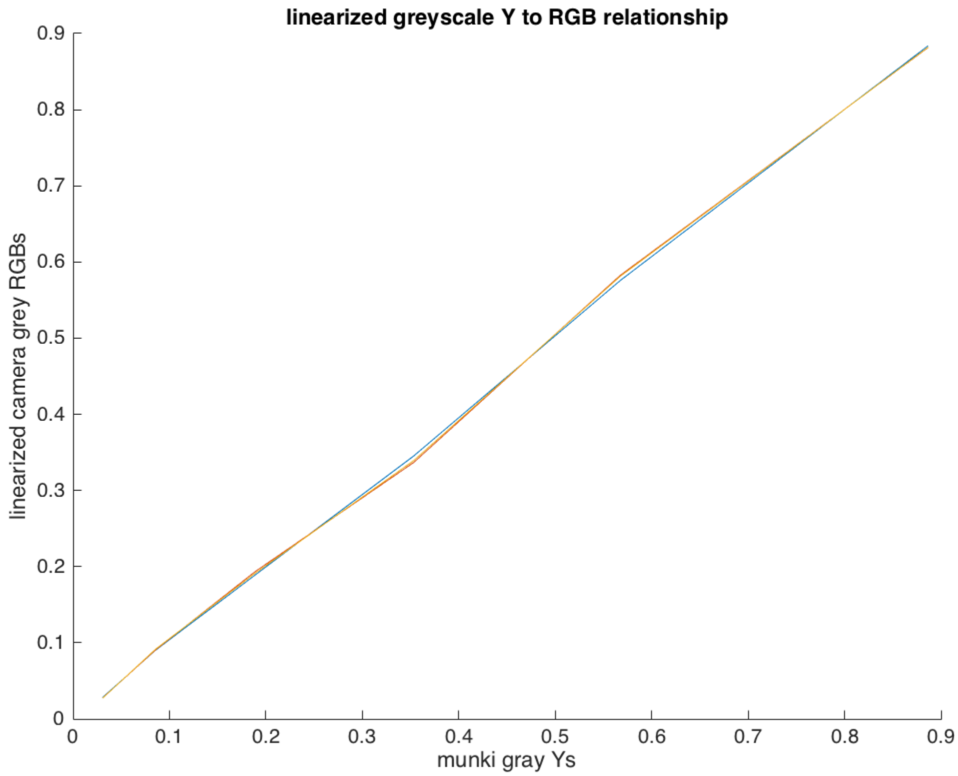
```
clf;
hold on;
plot(munki_gray_Ys, cam_gray_rgbs(1,:));
plot(munki_gray_Ys, cam_gray_rgbs(2,:));
plot(munki_gray_Ys, cam_gray_rgbs(3,:));
title('original greyscale Y to RGB relationship');
xlabel('munki gray Ys');
ylabel('camera grey RGBs');
hold off;
```



Linearize Camera RGB Output and Plot

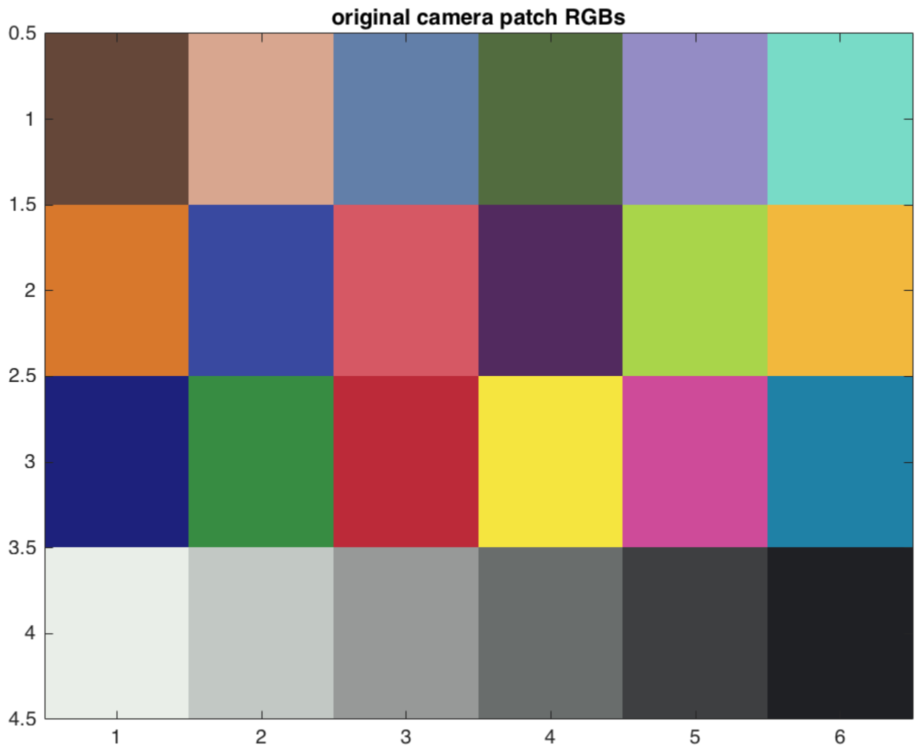
```
r = 1; g = 2; b = 3;
% fit low-order polynomial functions between the camera gray rgbs
% and the munki gray Ys
cam_polys(r,:)=polyfit(cam_gray_rgbs(r,:),munki_gray_Ys,3);
cam_polys(g,:)=polyfit(cam_gray_rgbs(g,:),munki_gray_Ys,3);
cam_polys(b,:)=polyfit(cam_gray_rgbs(b,:),munki_gray_Ys,3);

% use the functions to linearize the camera RGB data
cam_rgbs_lin(r,:) = polyval(cam_polys(r,:),cam_RGBs(r,:));
cam_rgbs_lin(g,:) = polyval(cam_polys(g,:),cam_RGBs(g,:));
cam_rgbs_lin(b,:) = polyval(cam_polys(b,:),cam_RGBs(b,:));

% clip out of range values
cam_rgbs_lin(cam_rgbs_lin<0) = 0;
cam_rgbs_lin(cam_rgbs_lin>1) = 1;

% Fetch gray values of linearized cam RGBs
cam_rgbs_gray_lin = fliplr(cam_rgbs_lin(:,19:24));

clf;
hold on;
plot(munki_gray_Ys, cam_rgbs_gray_lin(1,:));
plot(munki_gray_Ys, cam_rgbs_gray_lin(2,:));
```

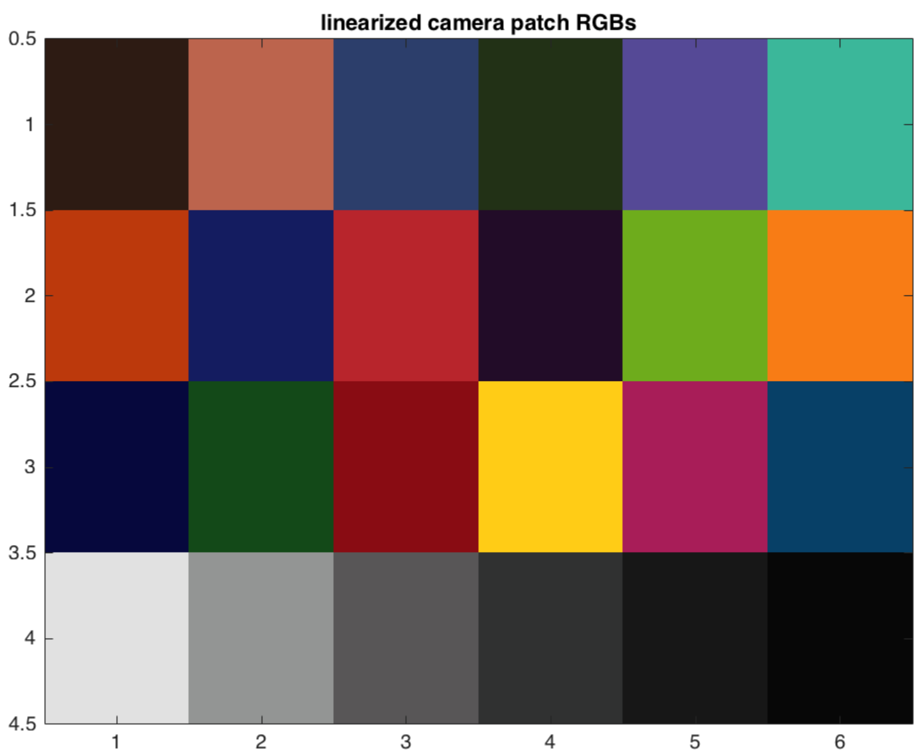
```
plot(munki_gray_Ys, cam_rgbs_gray_lin(3,:));
title('linearized greyscale Y to RGB relationship');
xlabel('munki gray Ys');
ylabel('linearized camera grey RGBs');
hold off;
```



Camera RGB Visualization

```
% visualize the original camera RGBs
pix = permute(cam_RGBs, [3 2 1]);
pix = reshape(pix, [6 4 3]);
pix = imrotate(pix, -90);
pix = flipdim(pix,2);
figure;
image(pix);
title('original camera patch RGBs');

% visualize the linearized camera RGBs
pix = permute(cam_rgbs_lin, [3 2 1]);
pix = reshape(pix, [6 4 3]);
pix = imrotate(pix, -90);
pix = flipdim(pix,2);
figure;
image(pix);
title('linearized camera patch RGBs');
```


original camera patch RGBs
0.5
1
1.5
2
2.5
3
3.5
4
4.5
1
2
3
4
5
6


linearized camera patch RGBs
0.5
1
1.5
2
2.5
3
3.5
4
4.5
1
2
3
4
5
6

Derive 3x3 Transformation Matrix

```
% use the linearized camera rgbs and munki-measured XYZs
% to derive an rgb to XYZ transformation matrix
% pinv finds the matrix that provides the a least squares
% linear solution for relating the rgbs and XYZs
munki_data = importdata('munki_CC_XYZs_Labs.txt');
munki_XYZs = munki_data(:,2:4)';
cam_matrix = munki_XYZs * pinv(cam_rgbs_lin);

fprintf('cam_matrix = \n');
disp(cam_matrix);

% estimate the CC XYZs from the linearized camera rgbs using
% the camera model matrix
cam_XYZs = cam_matrix * cam_rgbs_lin;

fprintf('cam_XYZs = \n');
disp(cam_XYZs);
```

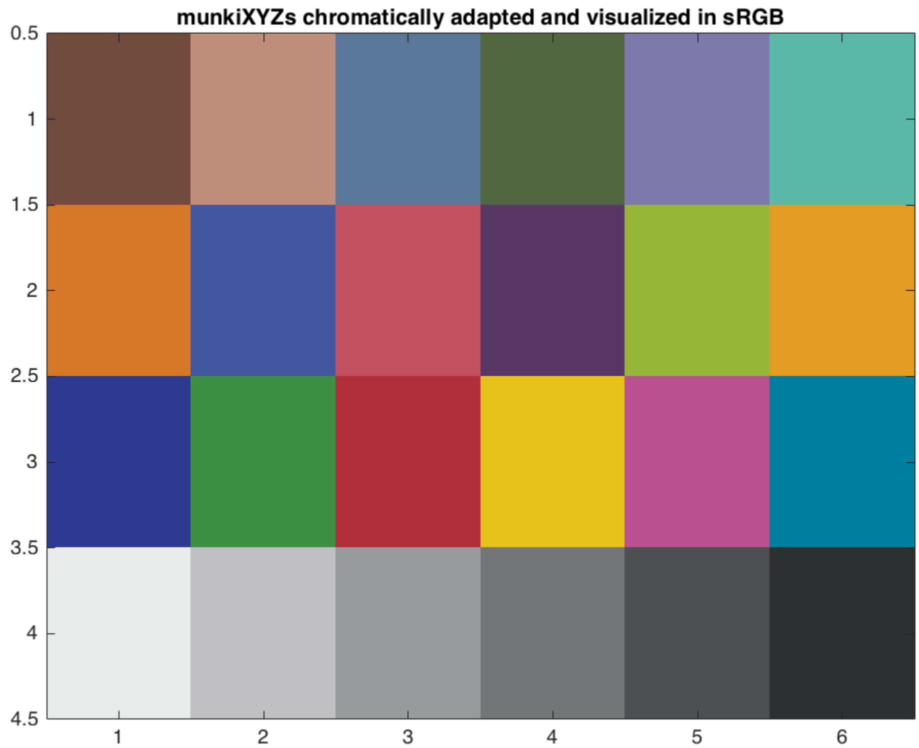
```
cam_matrix =
   37.6725   28.3424   17.6664
   18.4793   54.9651   13.5832
    0.4202    1.7343   70.9941

cam_XYZs =
  Columns 1 through 7

   11.0731   44.2390   20.7674   11.9588   31.0797   39.7690   34.8852
   10.1954   39.3439   22.2907   14.1347   29.9476   51.9954   26.5213
    5.5525   22.5015   30.2462    6.5960   42.3935   44.1033    3.9504

  Columns 8 through 14

   12.8033   34.2098    9.1787   37.2383   51.8260    5.9432   12.6876
   12.6901   23.4996    7.2726   46.4579   45.7617    5.2992   18.4954
   26.8451   12.6750   11.3985    9.2373    7.1541   16.9077    7.1969

  Columns 15 through 21

   22.9340   61.9260   34.1731   15.3098   73.8783   48.4625   28.5559
   13.6066   63.7064   23.1884   19.7956   76.7818   50.5468   29.5096
    5.6017    8.0188   24.8715   29.1177   64.5216   42.5128   24.8204

  Columns 22 through 24

   15.9825    7.6036    2.3267
   16.7258    7.9118    2.4045
   13.9976    6.7093    1.9934
```

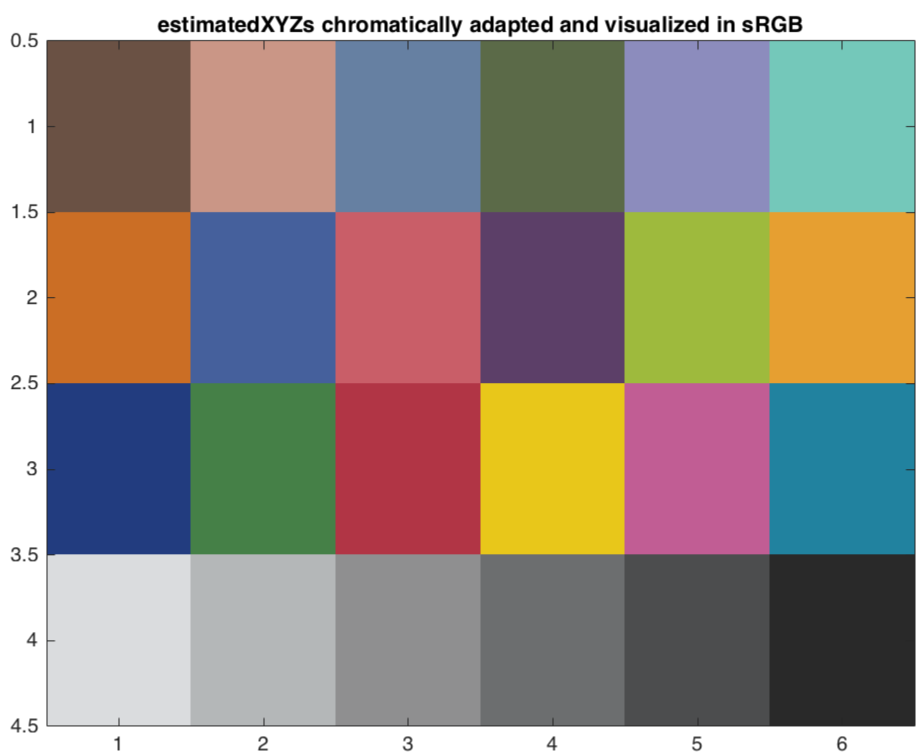
Calculate CIELab and DeltaEAB of ColorChecker for Estimated XYZs

```
cie = loadCIEData();
XYZn_D50 = ref2XYZ(cie.illE,cie.cmf2deg,cie.illD50);
RGB_Labs = XYZ2Lab(cam_XYZs, XYZn_D50);

ColorMunki_CieLabs = munki_data(:, 5:7)';

deltas = deltaEab(RGB_Labs, ColorMunki_CieLabs);
```

Visualize using chromatic adaptation

```
XYZ_D50 = ref2XYZ(cie.illE,cie.cmf2deg,cie.illD50)';
XYZ_D65 = ref2XYZ(cie.illE,cie.cmf2deg,cie.illD65)';

% visualize ColorMunki XYZs in sRGB

munki_XYZs_D65 = catBradford(munki_XYZs, XYZ_D50', XYZ_D65');
munki_XYZs_sRGBs = XYZ2sRGB(munki_XYZs_D65);
pix = reshape(munki_XYZs_sRGBs', [6 4 3]);
pix = uint8(pix*255);
pix = imrotate(pix, -90);
pix = flipdim(pix,2);
figure;
image(pix);
title('munkiXYZs chromatically adapted and visualized in sRGB');

% visualize camera-estimated XYZs in sRGB

cam_XYZs_D65 = catBradford(cam_XYZs, XYZ_D50', XYZ_D65');
cam_XYZs_sRGBs = XYZ2sRGB(cam_XYZs_D65);
pix = reshape(cam_XYZs_sRGBs', [6 4 3]);
pix = uint8(pix*255);
pix = imrotate(pix, -90);
pix = flipdim(pix,2);
figure;
image(pix);
title('estimatedXYZs chromatically adapted and visualized in sRGB');
```


munkiXYZs chromatically adapted and visualized in sRGB
0.5
1
1.5
2
2.5
3
3.5
4
4.5
1
2
3
4
5
6


estimatedXYZs chromatically adapted and visualized in sRGB
0.5
1
1.5
2
2.5
3
3.5
4
4.5
1
2
3
4
5
6

Published with MATLAB® R2015b