James Murphy
CS381

Problem 1 :

Take the built in matlab functions for rgb and ycbcr conversions

```
%Take ycbcr of both Expedition and Kalapatther %
ycbcr_Kalapatthar = rgb2ycbcr(everest_kalapatthar);
ycbcr_Expedition = rgb2ycbcr(everest_expedition);

%Take rgb of both Expedition and Kalapatther %
rgb_Kalapatthar = ycbcr2rgb(ycbcr_Kalapatthar);
rgb_Expedition = ycbcr2rgb(ycbcr_Expedition);
```

RGB -> YCBCR

Take the the Y Cb Cr values * R G B respectively, and then add the matrix values * RGB and then convert it back into an image and return it.

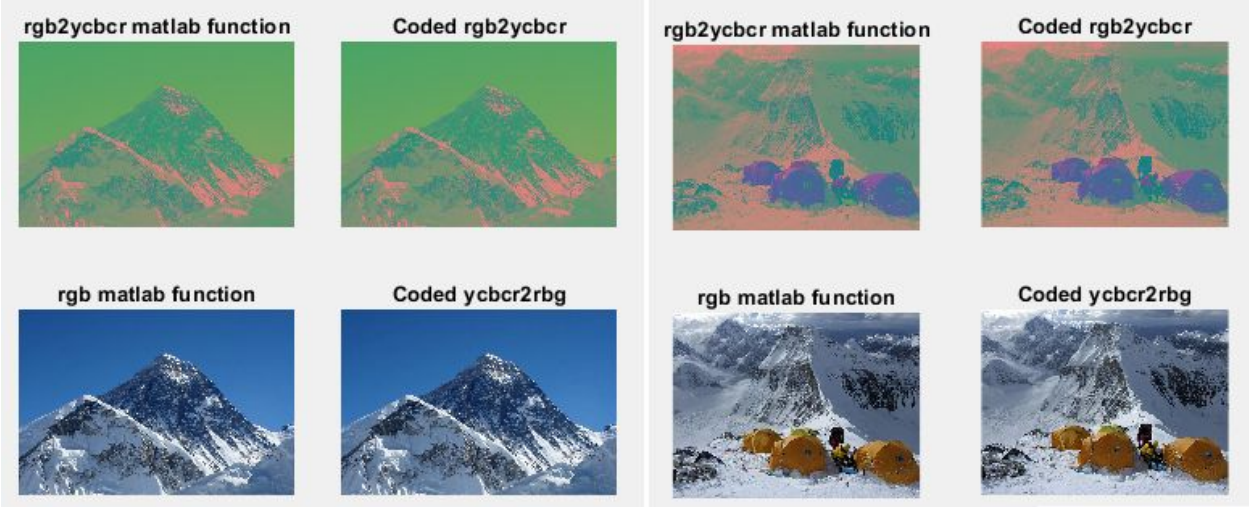
```
function image_out = rgb_ycbcr(image_in)
    %Convert the image into a double to be able to do computation
    temp_image = double(image_in);
    %Set the range of RGB values from [0, 1]
    temp_image = temp_image/255;
    %Take out and store each of the RGB values
    R = temp_image(:,:,1);
    G = temp_image(:,:,2);
    B = temp_image(:,:,3);
    % Get the results from the formula (Y, Cb, Cr)
    % With y = 16, Cb = 128, Cr = 128 from hw assignment * RGB values
    % in the matrix
    Y = 16 + 65.481*R + 128.553*G + 24.966*B;
    Cb = 128 - 37.797*R - 74.203*G + 112.000*B;
    Cr = 128 + 112.000*R - 93.786*G - 18.214*B;
    %Put the image back together by the Y CB Cr values
    image_out = uint8(cat(3,Y,Cb,Cr));
end
```

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix} + \begin{pmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Part 2 : Take the Y, Cb, Cr values and subtract them by the given vector. Then multiply each by the number in the formula in the code to convert it back

```
%Convert the image into a double to be able to do computation
temp_image = double(image_in);
%Take out and store each of the RGB values
Y = temp_image(:,:,1);
Cb = temp_image(:,:,2);
Cr = temp_image(:,:,3);
% Get the results from the formula (Y, Cb, Cr)
% With y = 16, Cb = 128, Cr = 128 from hw assignment * RGB values
% in the matrix
Y = Y - 16;
Cb = Cb - 128;
Cr = Cr - 128;
R = 0.0046 * Y + 0.0000 * Cb + 0.0063 * Cr;
G = 0.0046 * Y - 0.0015 * Cb - 0.0032 * Cr;
B = 0.0046 * Y + 0.0079 * Cb + 0.0000 * Cr;
%Put the image back together by the Y CB Cr values
image_out = cat(3,R,G,B) * 255;
image_out = uint8(image_out);
```

Output :



Problem 2 :

The solution to this problem was simple. For the double exposure, get the 3x3 matrix values of each image.

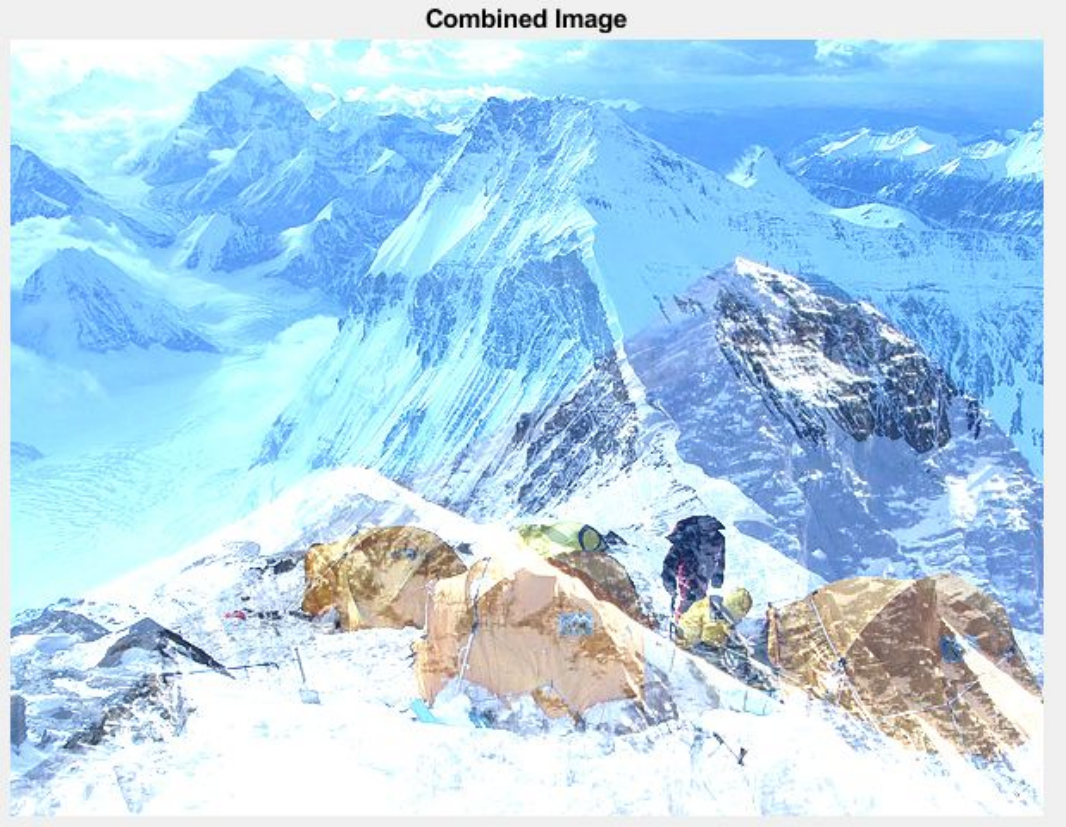
```
%Get the matrix dimensions of kalapatthar
[row_kal, col_kal, num_kal] = size(kalapatthar);
[row_exe, col_exe, num2_exe] = size(expedition);
row = min(row_kal, row_exe);
col = min(col_kal, col_exe);
```

Then, in a for loop combine the coordinates at each m, n pair (row / column) to create the double exposure and place it on the figure

```
% m is the rows in the outer loop
% n is the columns
% iterate over every (m, n) pair and add kalapatthar to expedition
for m = 1 : row
    for n = 1 : col
        combine_imgs(m, n, 1) = kalapatthar(m, n, 1)+ expedition(m, n, 1);
        combine_imgs(m, n, 2) = kalapatthar(m, n, 2)+ expedition(m, n, 2);
        combine_imgs(m, n, 3) = kalapatthar(m, n, 3)+ expedition(m,n,3);
    end
end
```

.

Output :



Problem 3 :

Part - Grayscale Output : Gray scale is calculated by turning the RGB photo by multiply .299 * Red + .587 * Green + .114 * Blue.

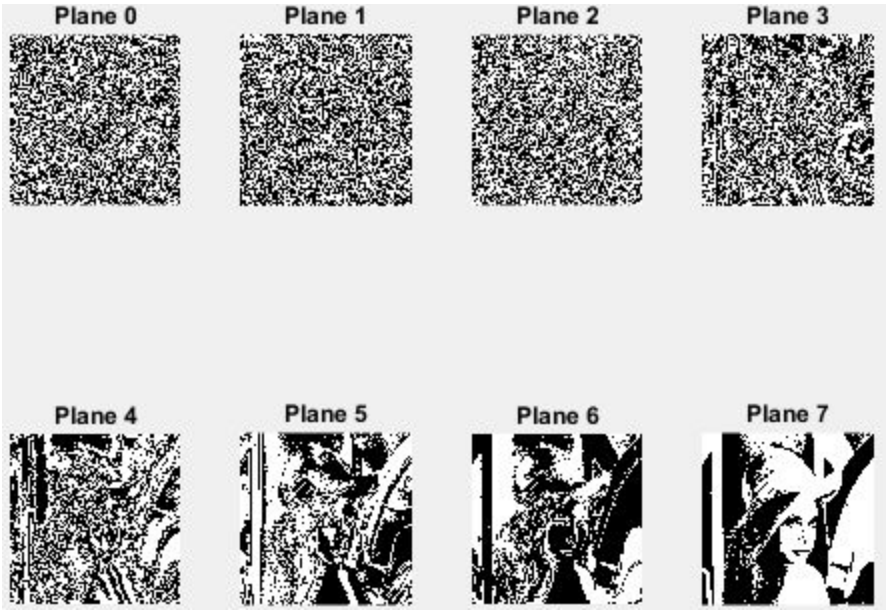
Next the result is raised to 2^n, (2^1 … 2^8, in this example.) Next the floor is taken to turn it from a float into an integer and then the result mod 2.

```
%Converting the image to grayscale.
    for row = 1:num_row
        for col = 1:num_col
            gray_scale_lenna(row, col) = 0.299*Lenna(row, col, 1) +0.587
                * Lenna(row, col, 2)+0.144 * Lenna(row, col, 3);
        end
    end

    % Take the floor of the grayschole / 2^n and mod it and store it
    % Into each 'plane'

    bit_plane_1 = mod(floor(gray_scale_lenna/(2^0)),2);
    bit_plane_2 = mod(floor(gray_scale_lenna/(2^1)),2);
    bit_plane_3 = mod(floor(gray_scale_lenna/(2^2)),2);
    bit_plane_4 = mod(floor(gray_scale_lenna/(2^3)),2);
    bit_plane_5 = mod(floor(gray_scale_lenna/(2^4)),2);
    bit_plane_6 = mod(floor(gray_scale_lenna/(2^5)),2);
    bit_plane_7 = mod(floor(gray_scale_lenna/(2^6)),2);
    bit_plane_8 = mod(floor(gray_scale_lenna/(2^7)),2);
```

Output of Gray Scale :

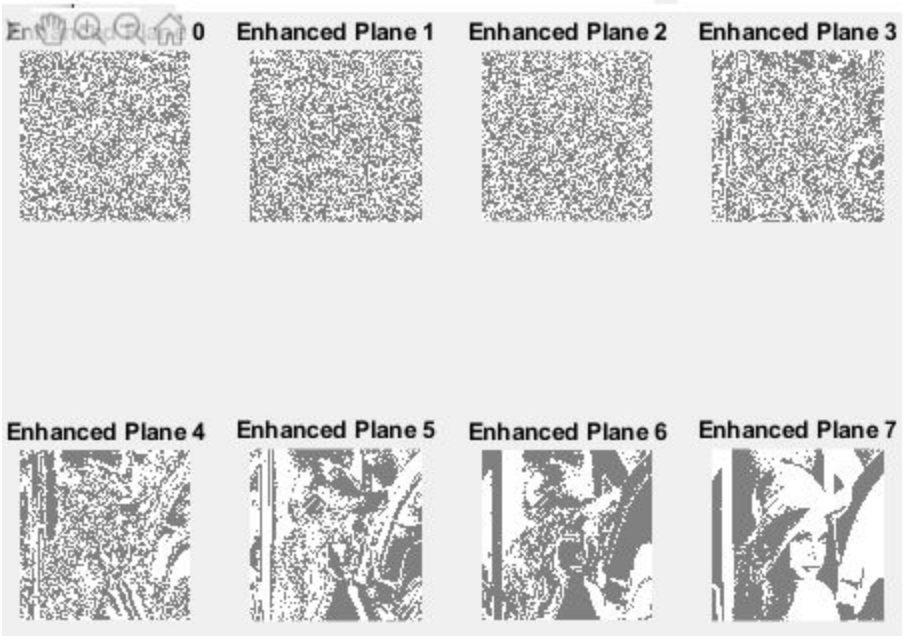


Part 2 :

Using imadjust, it took the previous gray bitplane that was calculated and changes the contrast. By the parameters entered.

```
%use imadjust to adjsut the intenstiy values of the photo
enhanced_gray_1 = imadjust(bit_plane_1, [0 .1], [0.5 1]);
enhanced_gray_2 = imadjust(bit_plane_2, [0 .1], [0.5 1]);
enhanced_gray_3 = imadjust(bit_plane_3, [0 .1], [0.5 1]);
enhanced_gray_4 = imadjust(bit_plane_4, [0 .1], [0.5 1]);
enhanced_gray_5 = imadjust(bit_plane_5, [0 .1], [0.5 1]);
enhanced_gray_6 = imadjust(bit_plane_6, [0 .1], [0.5 1]);
enhanced_gray_7 = imadjust(bit_plane_7, [0 .1], [0.5 1]);
enhanced_gray_8 = imadjust(bit_plane_8, [0 .1], [0.5 1]);
```

Output of Image Enhancement :


Enhanced Plane 0
Enhanced Plane 1
Enhanced Plane 2
Enhanced Plane 3
Enhanced Plane 4
Enhanced Plane 5
Enhanced Plane 6
Enhanced Plane 7