# Task 1

## b) RGB to Indexed Image

The rgb2ind function can be used for converting an RGB image L to an indexed image based on a specified number n of colours in the colormap/lookup table using the *minimum variance quantization* method, which divides the colour cube into smaller bins of different shapes and sizes, depending on the image’s colour distribution. The following command is run for the conversion of image L to an indexed version L\_indexed:

[L\_indexed, colormap] = rgb2ind(L, n);

The command above uses *dithering***,** which means that the colours of neighbouring pixels are changed in order to get an average colour that equals the original RGB colour. However, the image might seem fuzzy, and when looking closely, there might be a blend of other colours/shades. An image without dithering requires less number of colour and might improve the spatial resolution of the image. The following command is run for the conversion of image L to an indexed version L\_indexed, without dithering:

[L\_indexed, colormap] = rgb2ind(L, n, ‘no dither’);

A better way of doing this is the *k-means clustering* method, which aims at identifying *k* number of centroids and allocating all data to the nearest centroid, using the squared *Euclidean distance* metric, shown in Eq. 1.1.

Eq. 1.1

The following command can be used for the k-means clustering method:

[idx, cmap] = kmeans(X, n);

Fig. 1.1 shows the effect of converting an RGB image (peppers.bmp) to an indexed version with 8 colours using the minimum variance quantization method with and without dithering and using the k-means algorithm.

Another method of image quantization is using the *Median-Cut Algorithm*, where each data set is cut at the median of the dimension (*R*, *G* or *B*) with the most extended range and keeps doing this (iteration) for the specified number of entries for the lookup table. The equation for calculating the median of dimension *R* is given in Eq. 1.2, where *n* equals the length of the dimension.

indexed ‘dithering’, n = 8

original



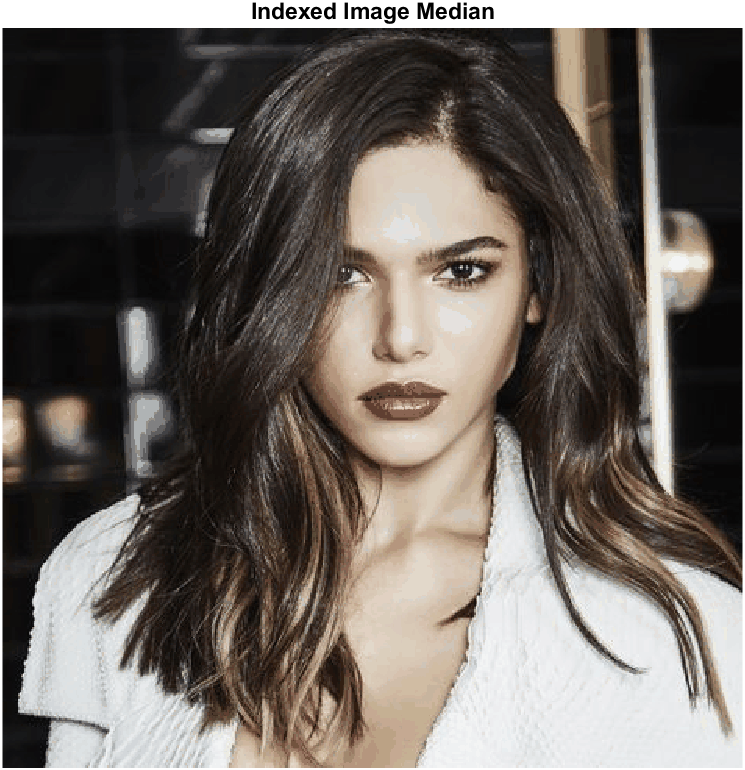
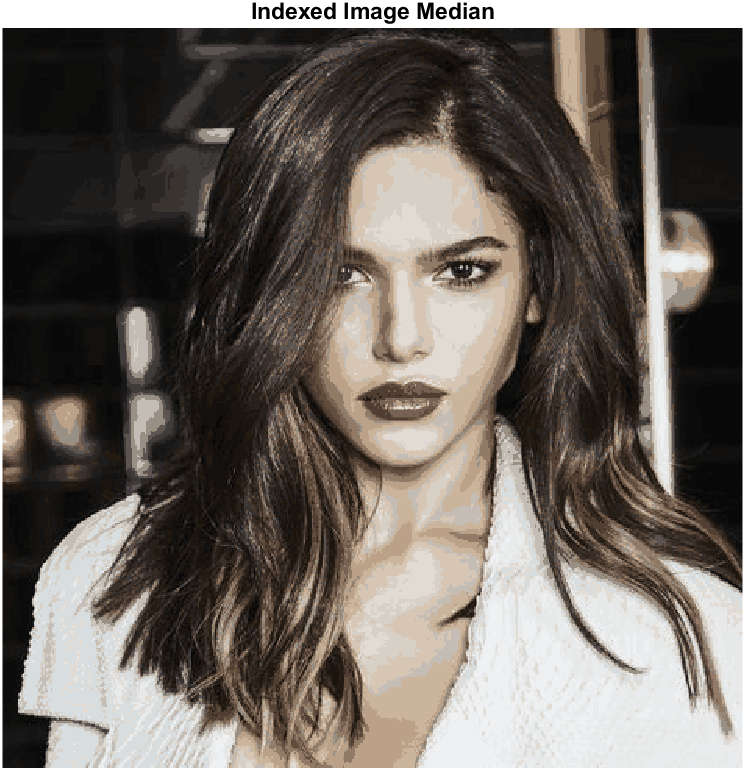
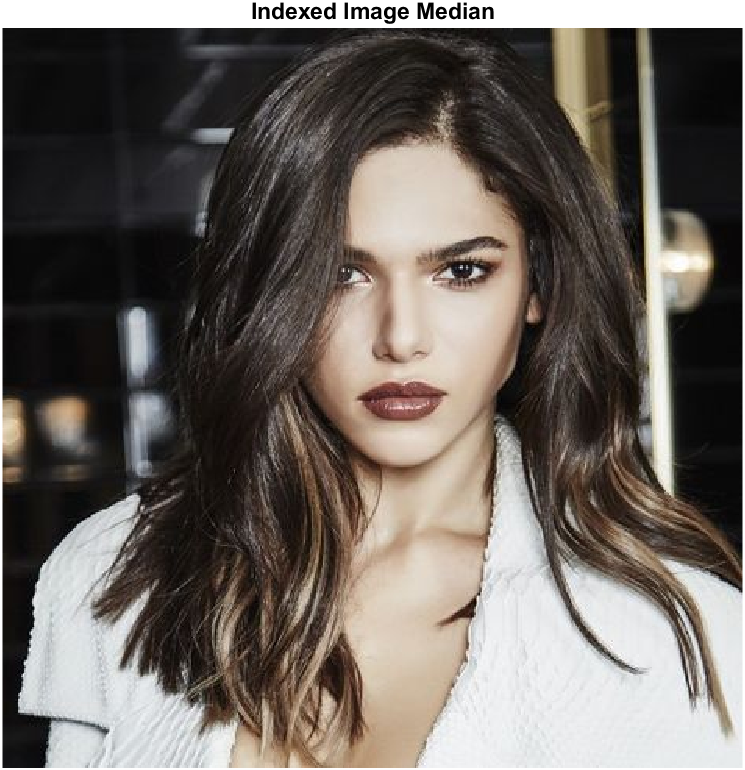
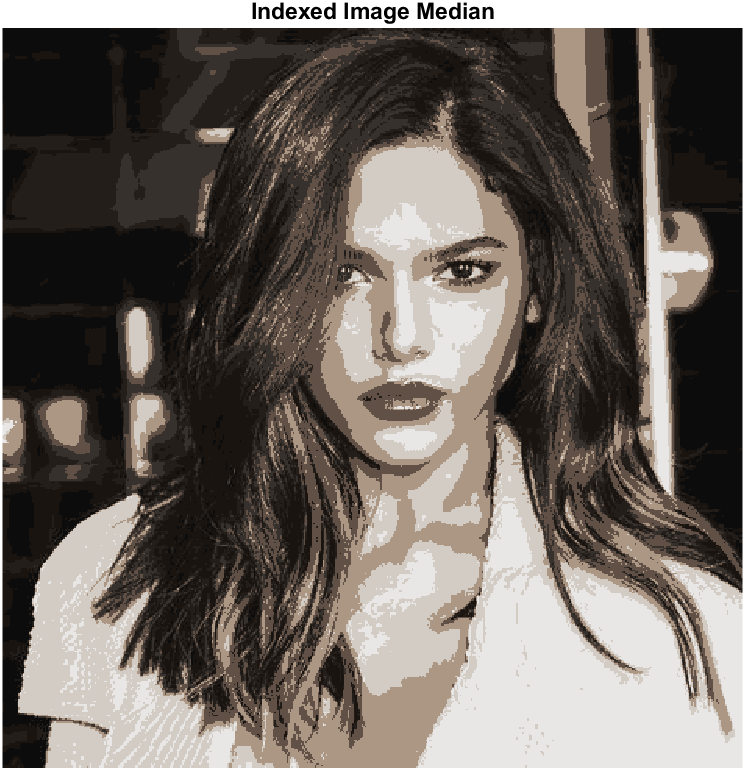
indexed (k-means), n = 8

indexed ‘no dithering’, n = 8

**Figure 1.1**: Effects of three methods of image colour quantization with n = 8 colours in the lookup table of the indexed image.

Eq. 1.2

Fig. 1.2 shows the results after performing colour quantization using the median-cut technique on an image (girl.jpg) and reducing the number of colours to 8, 32 and 128 colours, in MATLAB.



original

indexed (median-cut), 8 colours

indexed (median-cut), 32 colours

indexed (median-cut), 128 colours

**Figure 1.2**: Effects of image colour quantization with 8, 32 and 128 colours using the median-cut algorithm.

## Source Code

Source code of task1.m is added here. Visit MATLAB code for functions.

**task1.m**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Title: Convert RGB to indexed

% Author: Samir Habibi

% Rev. Date: 22/11/2020

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear; % Delete all variables.

close all; % Close all windows.

clc; % Clear command window.

% Ask user for file by presenting options with menu() command.

% Choice between 'peppers.bmp' or user to select own file.

fileChoice = menu('File', 'Girl', 'Peppers', 'Choose own');

% Use switch() to read file based on user's choice (fileChoice).

switch (fileChoice)

case 1 % If user selects 'Peppers'.

filename = ('girl.jpg');

L = imread(filename);

case 2 % If user selects 'Choose own'.

filename = ('peppers.bmp');

L = imread(filename);

case 3 % If user selects 'Choose own'.

filename = uigetfile('');

L = imread(filename);

end % End the switch-statement after obtaining image.

% Get R, G and B value

R = L(:,:,1);

G = L(:,:,2);

B = L(:,:,3);

% Put all data in one matrix

X = double([R(:), G(:), B(:)]);

% Call function to get number of entires for the colormap/lookup table.

numberOfColours = getNumberOfColours();

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% The function rgb2ind() is used to store a new indexed image

% and to get the specified maximum number of colours.

% The line of code below performs the minimum variance quantization

% technique for the colormap, if the image requires less colours than

% specified for 'n' in the input argument, the output image will contain

% less colours than 'n', but still all the colours of the input image.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Convert image 'L' with dithering.

[L\_indexed, colormap] = rgb2ind(L, numberOfColours);

% Convert image 'L' without dithering.

[L\_indexed\_no\_dither, ~] = rgb2ind(L, numberOfColours, 'nodither');

% Display the RGB values for each indexed colour of the colormap in

% the Command Window.

disp(colormap);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% k-means clustering is another method for quantization.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Returns cluster locations in C.

[idx, cmap] = kmeans(X, numberOfColours);

% Make sure idx and L are of the same size.

idx = reshape(idx,size(L,1),size(L,2));

% Normalize, values must be between [0 1].

cmap = cmap/255;

% Convert the label matrix idx with colormap.

L\_indexedCluster = label2rgb(idx, cmap);

% The functions MC(), MCR() and checkcolours() are obtained through Github

% from author: cmanso. Available at:

% https://github.com/cmanso/MedianCut\_Matlab.

L\_medianCut = MC(L, numberOfColours);

figure;

% Plot original image.

subplot(2, 3, 1);

imshow(L);

title('Original Image');

% Plot indexed image after using default rgb2ind().

subplot(2, 3, 2);

imshow(L\_indexed, colormap);

title('Indexed Image (rgb2ind)');

% Plot indexed image after using rgb2ind() without dithering.

subplot(2, 3, 3);

imshow(L\_indexed\_no\_dither, colormap);

title('Indexed Image (rgb2ind) Without Dithering');

% Plot indexed image after using kmeans().

subplot(2, 3, 4);

imshow(L\_indexedCluster)

title('Indexed Image (K-Means)');

% Plot indexed image after performing the median-cut technique

subplot(2, 3, 5);

imshow(L\_medianCut);

title('Indexed Image (Median-Cut)');

## Bibliography

Department of Electrical & Computer Engineering, n.d. *Color*. [pdf] University of Maryland. Available at: <http://classweb.ece.umd.edu/enee408g.S2007/image/readings/408S03\_Image\_read2e.pdf> [Accessed 1 November 2020].

Github, 2017. *cmanso/MedianCut\_Matlab*. [online] Available at: <https://github.com/cmanso/MedianCut\_Matlab> [Accessed 2 December 2020].

MathWorks, 2020. *kmeans*. [online] Available at: <https://nl.mathworks.com/help/stats/kmeans.html?searchHighlight=kmeans&s\_tid=srchtitle#bueftl4-1> [Accessed 2 December 2020].

MathWorks, 2020. *label2rgb*. [online] Available at: <https://nl.mathworks.com/help/images/ref/label2rgb.html> [Accessed 2 December 2020].

MathWorks, 2020. *Reduce the Number of Colors in an Image*. [online] Available at: <https://nl.mathworks.com/help/images/reduce-the-number-of-colors-in-an-image.html> [Accessed 2 November 2020].

PCMag, n.d. *indexed color*. [online] Available at: <https://www.pcmag.com/encyclopedia/term/indexed-color> [Accessed 1 November 2020].

Wikipedia, 2020. *Median cut*. [online] Available at: <https://en.wikipedia.org/wiki/Median\_cut#:~:text=Median%20cut%20is%20an%20algorithm,point%20along%20the%20longest%20dimension.&text=For%20example%2C%20to%20reduce%20a,match%20the%20original%20data%20well.> [Accessed 2 December 2020].