COMPUTER VISION

Practice 2

Introduction to Matlab Image Processing

06/10/2021

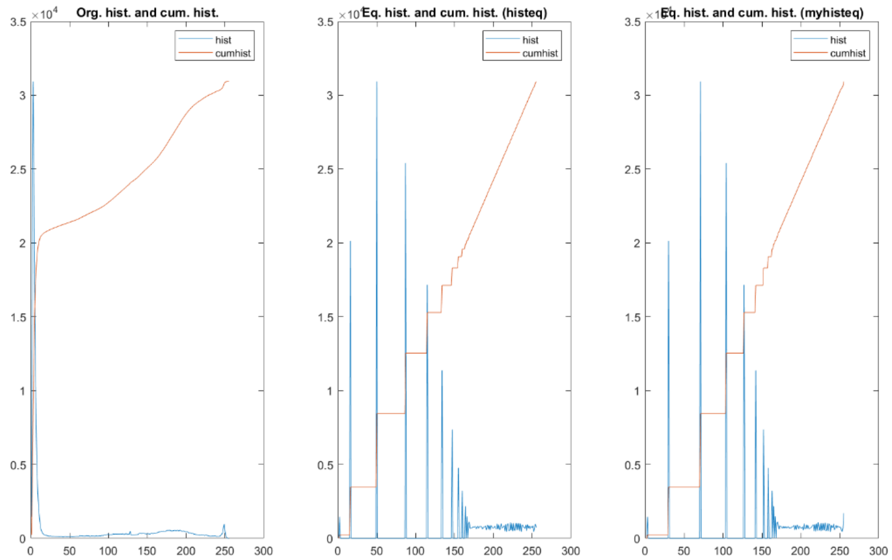
Michal Szabó

Aleix Granero

EXERCISE 1

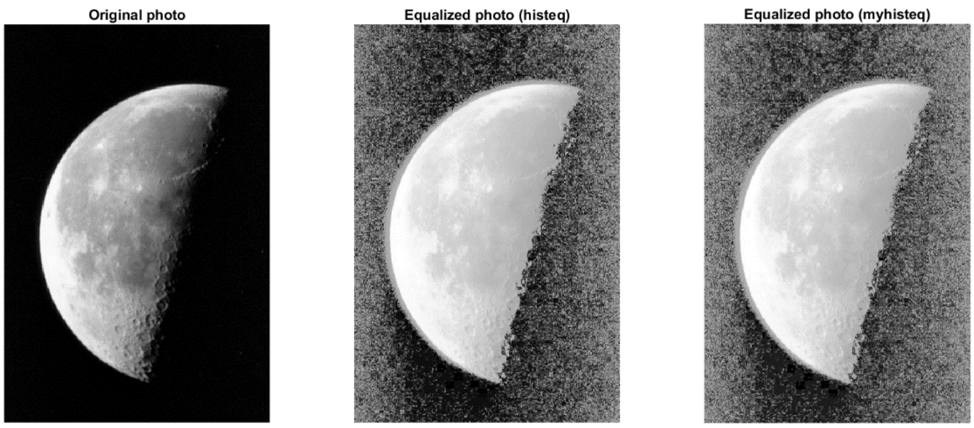
We have created a function in a separately Matlab document that counts the occurrence of each pixel value (*imhist*), calculates the cumulative distribution (*cumsum*) and calculates the cumulative distribution probability. Then, depending on the number of bits, the coefficient K is defined, and this allows us to calculate the value of each output pixels.

In the following graphs we can see a comparison between histogram and cumulative histogram of input and output images, using *histeq* Matlab function and our function:



Trying with different numbers of bins we have realized that decreasing these number, the histogram peak values increase.

Finally, in the following image we can see the results of applying *histeq* function and our function:

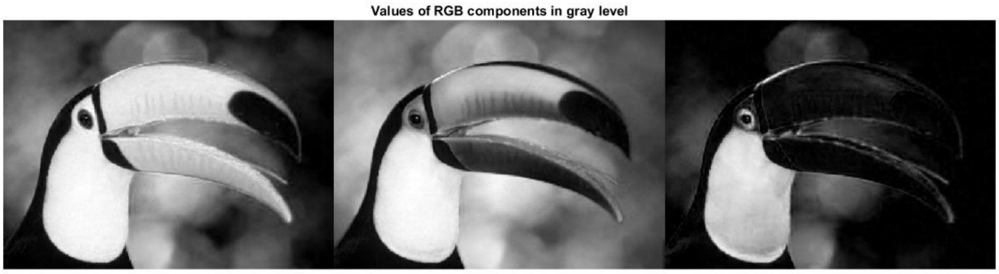


As we can see, these pictures are not the same but they are very similar. With decreasing number of bins we see bigger difference between them. The difference is hidden in the algorithms. Matlab's algorithm is better optimized than ours.

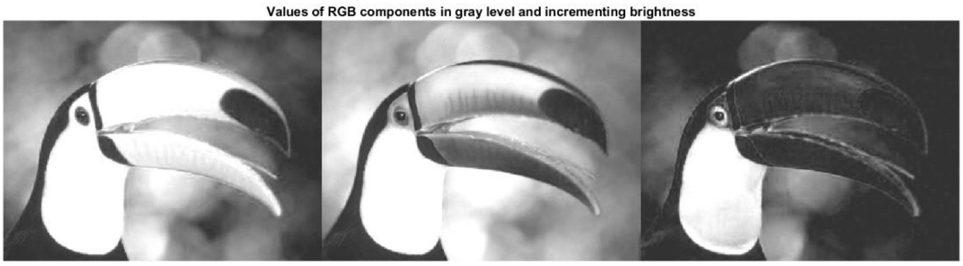
EXERCISE 2

a)

First of all, we have split the image in its RGB components using *imsplit*:

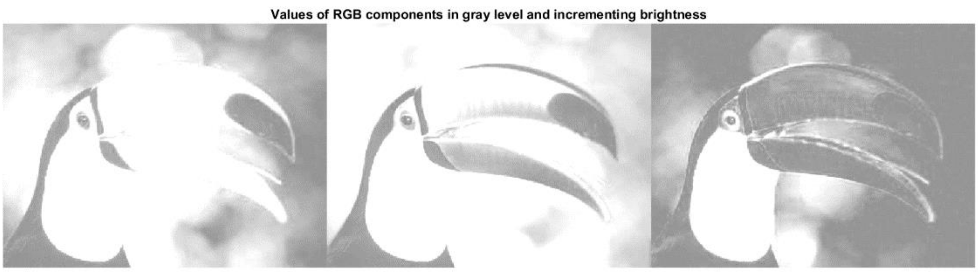


Then, we have added brightness to each image separately, adding a certain offset to each pixel (in this case the offset is 50):



Finally, we have combined the resulting images in a truecolor image again:

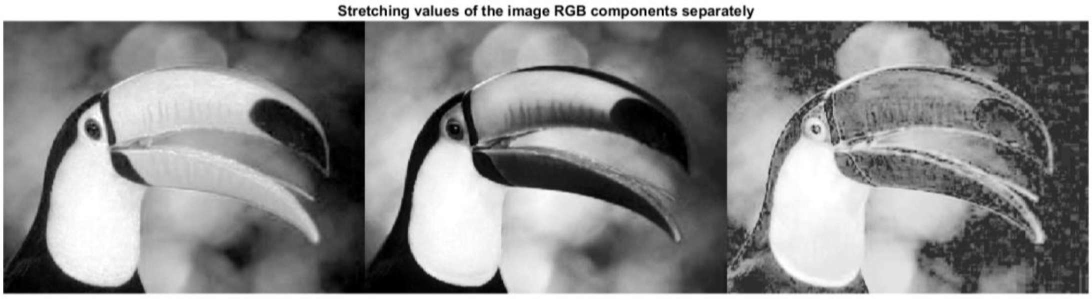
When we increment the brightness, the pictures of the original colours become more and more lighter, and for example with an offset of 150 we obtain the following images:



b)

First, we have tried to increment the contrast of the image by stretching the values of its RGB components separately:

Stretching values of the image RGB components separately



And, combining the result, we have obtained the following image:

Original photo

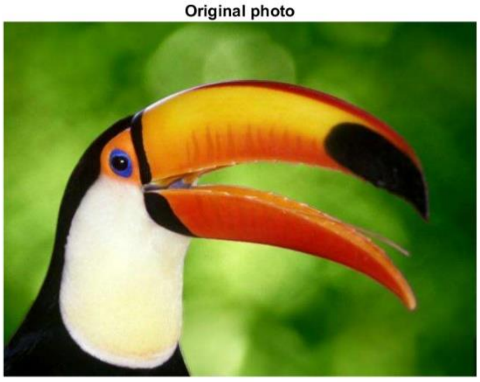
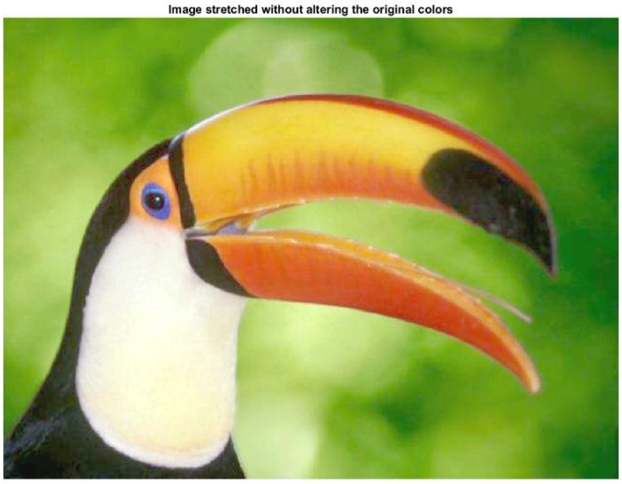


Image after stretching the values of its RGB components

As we can see, the goal of increasing contrast has not been achieved. In the case of the red and green pictures there is hardly any difference but the blue picture seems distorted and many pixels have gained brightness. This fact seems to cause the final image to acquire a strange blue colour.

Finally, we have tried to stretch the image without altering the original colours, but the expected result has not been obtained either.

Image stretched without altering the original colors

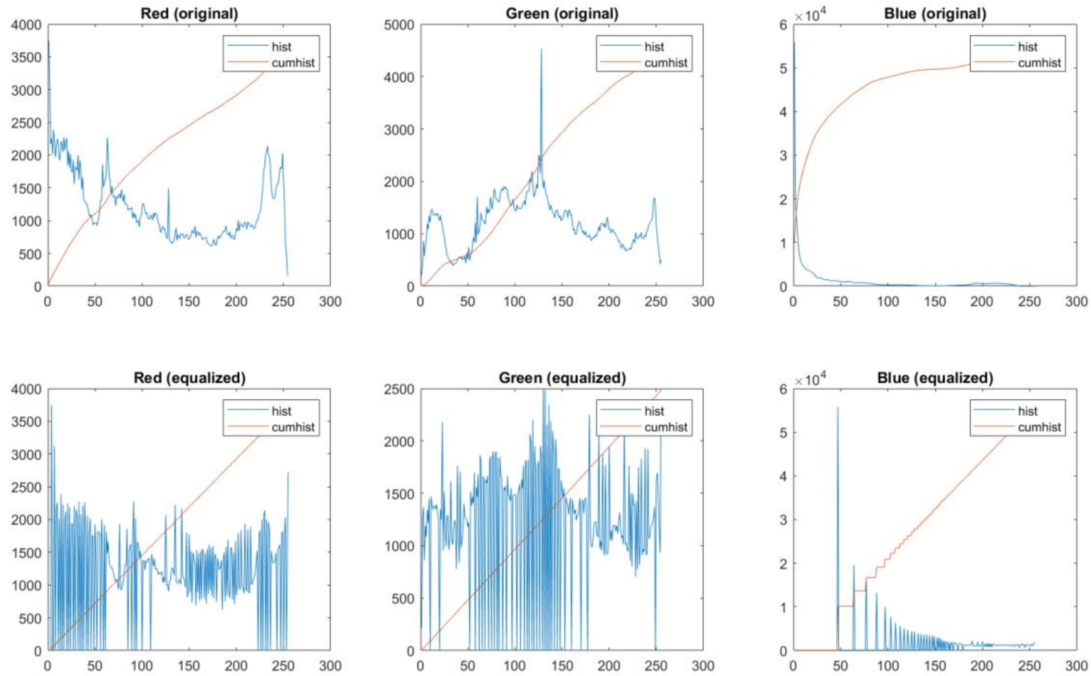


c)

Applying our equalization function to each RGB components, we obtain the following results:

Stretching values of the image RGB components separately using myhisteq

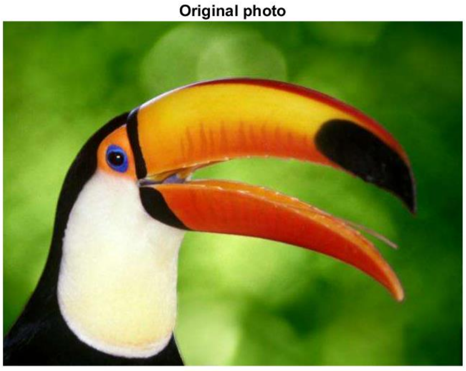
And in the next graphs, we can se a comparison between inputs and otputs histograms per color channel:



d)

Finally, we have combined the resulting images in a truecolor image again:

Original photo



Combining eqialized components

However, as we can see colours are different and image doesn't look real, acquiring a strange blue colour. This happens because histogram equalization is a non-linear process and involves intensity values of the image, not the colour components.

e)

To perform the histogram equialization of a color image we have changed color space using *rgb2hsv*, and we have split the HSV image in its components H,S and V using *imsplit*. Then, we have equalized the VALUE (v) component of the image and we have established this new component in the HSV image. Finally, we have changed the image back to RGB color space and we have obtained the following results:

Before Histogram Equalization

After Histogram Equalization