Image Analysis

Lab 1: Image Resizing and Color Enhancement

Student: TRAN Gia Quoc Bao, ASI

Professor: HENG UY Chhayarith

Date: 07 February 2020

Introduction: The objective of this lab work is to understand how to resample an image with filtering, how to manipulate the contrast and colors for enhancing the aspect of an image, and how to work with histograms.

Preparation before the lab:

I. Resampling:

We wish to reduce an image's size by half. The possible methods include:

- Keeping 1 of 2 consecutive rows and columns: so instead of having rows 1, 2, 3, …, H, we have 1, 3, 5, 7,…H or H-1 (H is the image's height), and the same for columns. The result is that the information of the deleted rows and columns are lost completely. This is also called subsampling or downsampling.
- Choosing 1 pixel out of 4 in a 2x2 zone and keeping it while deleting the other 3. Then we move to other 2x2 zones and continue doing this. This is called box (nearest neighbor) resampling.

We consider the method of subsampling. If we applied this to the Babara image, we would get **moiré patterns**. The reason is that in the image we already have some patterns, e.g on the woman's clothes, and we overlap this with a similar, displaced one. To improve this, we need to pre-filter the image in both directions with a filter (in printers we use descreen filters) to reduce the patterns in the image. According to Shannon, to avoid aliasing, the sampling frequency (fixed in this case) needs to be larger than twice the maximum frequency contained, so we need to filter away the large frequency patterns in the image, and that means we need low-pass filters.

II. Color spaces:

We are changing from the RGB basis to another one. Both the initial and the final color space have a dimension of 3, but the basis of the first one is RGB and the second one is L, C1, and C2. In the new basis, Luminance refers to brightness and the 2 Chrominances refer to colors. We can thus make changes to the saturation and hue by changing the Chrominances, and of course we can change the image's brightness. We have to do this change of basis:

$$\begin{bmatrix} IL \\ IC1 \\ IC2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} IR \\ IG \\ IB \end{bmatrix} = P \begin{bmatrix} IR \\ IG \\ IB \end{bmatrix}$$

So to reconstruct the old one we use the inverse matrix:

$$\begin{bmatrix} IR \\ IG \\ IB \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix}^{-1} \begin{bmatrix} IL \\ IC1 \\ IC2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix}^{T} \begin{bmatrix} IL \\ IC1 \\ IC2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{-2}{\sqrt{6}} & 0 \end{bmatrix} \begin{bmatrix} IL \\ IC1 \\ IC2 \end{bmatrix}$$

Because the matrix P is orthogonal: its inverse equals its transpose.

III. Histograms:

In the context of real imaging sensors, acquisitions are provided as quantization of physical parameters, such as luminance, which may have wildly different dynamics in the probability distribution of intensity values. Additionally, we may focus on just considering a subset of the intensity range.

To estimate these probability distributions with a frequentist approach, we use a **Histogram**, which is a graphical representation of intensity by pixel. It plots the number of pixels for each value of intensity. Clusters of intensities are grouped by the number of pixels with that intensity. For example, with an 8-bit image, we have 2^8 = 256 different intensity values so it would have 256 columns. Each column indicates the number of pixels with that intensity value. So this approach is statistical (frequentist).

For a specific image, the distribution may not be equally spread so for visualization we can stretch this distribution. These methods of redistributing the intensities can be linear (histogram contrast) or nonlinear (histogram equalization).

The first way is the **histogram contrast**. It is actually mapping, so it is like “zooming” or “re-scaling” the zone of intensities we have to fit the entire range of intensities. For example, with an 8-bit image, if the minimum value we have is 8, it would go to 0. And if the maximum value is 185 it would be scaled to 255. The look of the distribution should not change as only its range does.

The second way is using **histogram equalization**. First, we calculate the “frequency” of each intensity, which is the number of pixels with that intensity divided by the total number of pixels. Second, we calculate the cumulative distribution (add them together, will end at 1). Then, we spread this across the values by multiply the cumulative distribution with the “number of values – 1”. We then round it to get the new intensities and sum the number of pixels with the same intensities. In other words, we are flattening the histogram instead of rescaling it, so the distribution would change. That was for grayscale images. For color images, we apply the same procedure on each color of the basis. The problem is for RGB we risk changing the relative intensity distribution of each color, so before that, we would need to change from RGB to a new basis like HSI.

About **advantages and disadvantages**, for linear methods, we do not risk changing the relative distribution of different colors. Another advantage of the linear method is that there exists a 1-1 mapping between the original histogram and the stretched one, so we can indeed restore the original image. This is not possible for histogram equalization. Another convenience is that this guarantees that the whole range is filled. However, the linear method is vulnerable to very sparsely spread distribution patterns. Like, if we only have 0 and 255, this is useless. We cannot have anything in between. But if we used nonlinear methods like histogram equalization, we will get the values in between. The risk is, as said, the change of relative distribution across colors.

Computer session:

I. Resampling:

1. Downsampling:

As stated above, downsampling or subsampling is a method to reduce the size of an image. A grayscale image is actually a matrix of pixel values, with rows and columns. The principle here is to keep only one between 2 rows, and the same for columns. I applied this method to the Babara image:

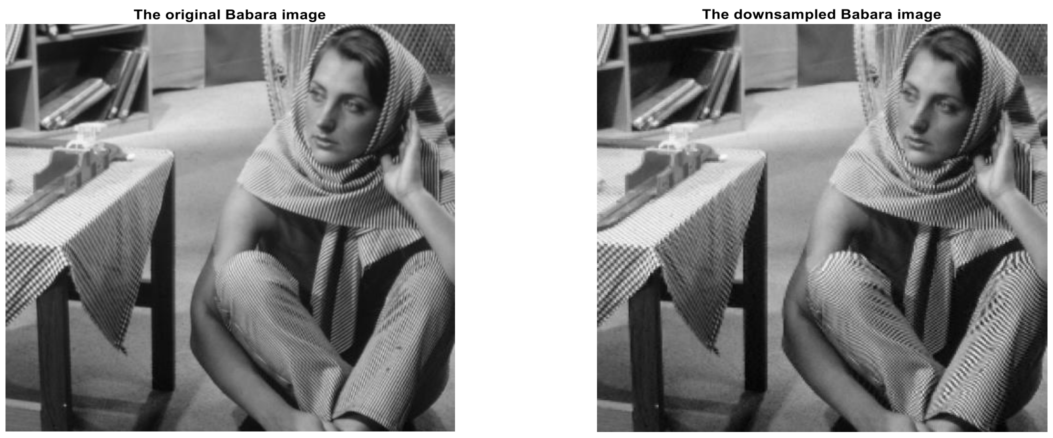


Figure 1. Downsampling the original Babara image

We see that the moiré patterns appeared on the cloth as I said in the preparation part. And I pointed out that a low-pass filter is necessary here, to keep only the low frequencies which satisfy the Shannon condition. The filter that I used has the following Fourier transformation to allow the elimination of high frequencies:

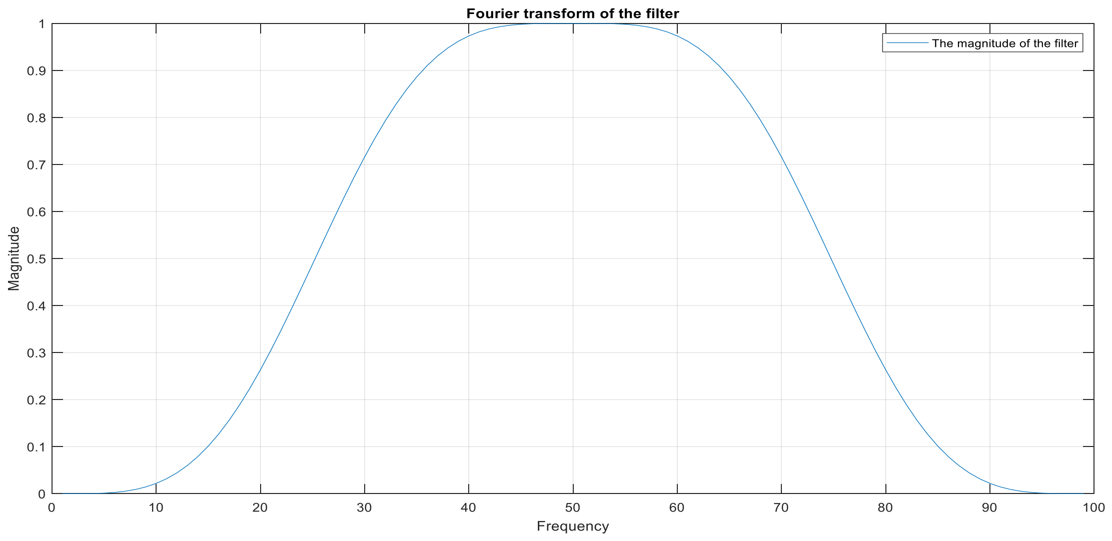


Figure 2. Fourier transform of the filter

This is the result of downsampling the filtered image, displayed alongside the previous one for comparison:

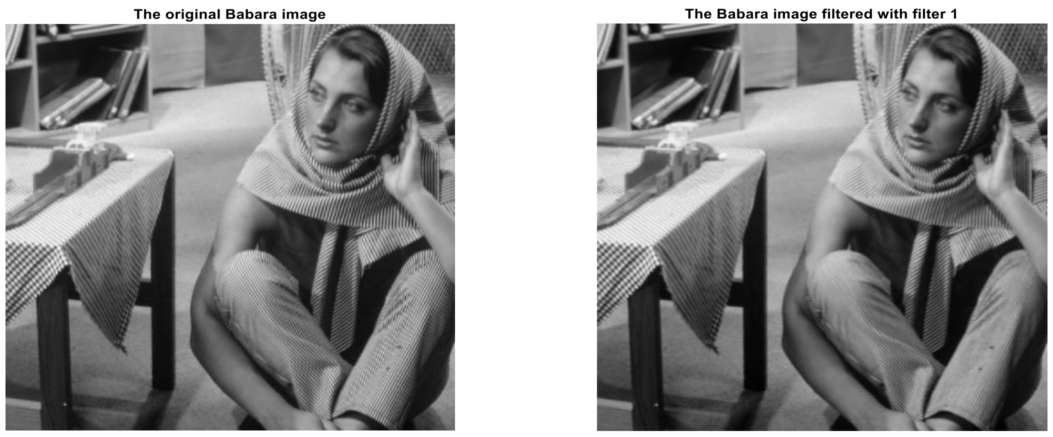

The original Babara image
The Babara image filtered with filter 1

Figure 3. Downsampling the filtered Babara image

We see that the patterns in the cloth have been removed effectively thanks to the filter (look at the woman's left leg to see that it now has much fewer patterns). Now, we can apply downsampling and expect that the moiré patterns should not appear this time:

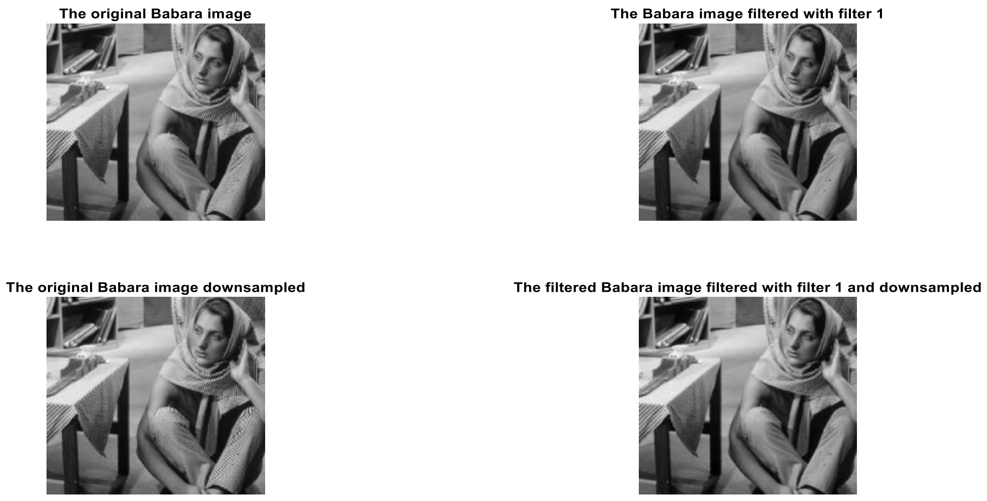

The original Babara image
The Babara image filtered with filter 1
The original Babara image downsampled
The filtered Babara image filtered with filter 1 and downsampled

Figure 4. Comparing the results obtained with downsampling

As expected, the downsampled image has almost no moiré pattern, especially in the left leg of the woman. We have confirmed the effectiveness of using the low-pass filter to eliminate high frequencies. The lesson here is that for images where there is already some pattern, the use of filter-like methods like downsampling will cause some unwanted effect and we can fix this with some filters.

2. Upsampling:

Contrary to downsampling, upsampling increases an image's size by inserting zeros in between the existing rows and corners. This can also cause unwanted imaging effects as below:

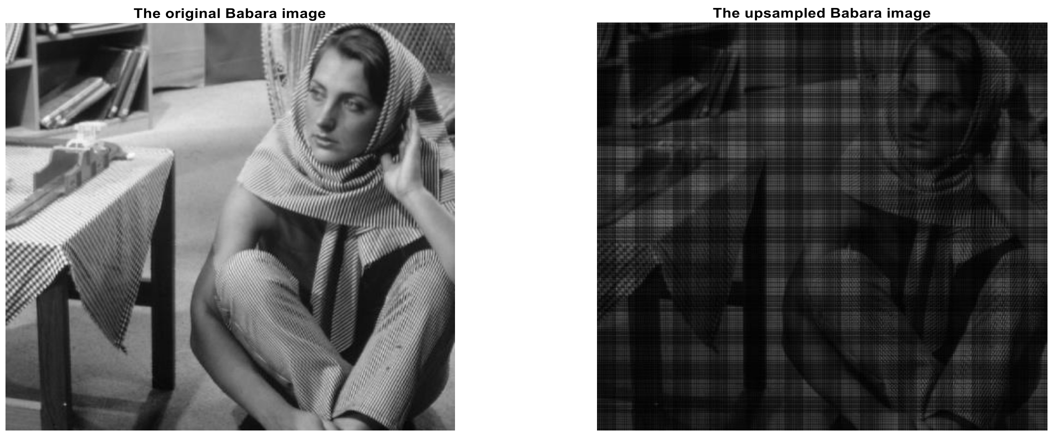


Figure 5. Upsampling the Babara image

To fix this, we can use a filter whose frequency response is twice of that of the previous one, after upsampling the image. Because the image is now twice as large, with new zeros entered, we need to use **interpolation**, which means to smooth out the discontinuities with a low-pass filter. The filter is twice as the previous one:

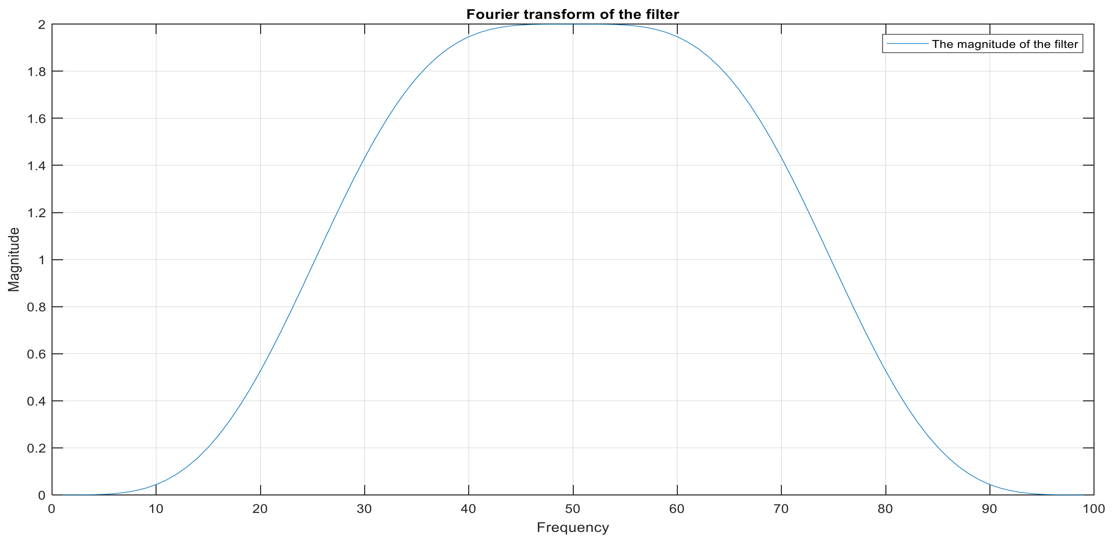


Figure 6. Fourier transform of the 2nd filter

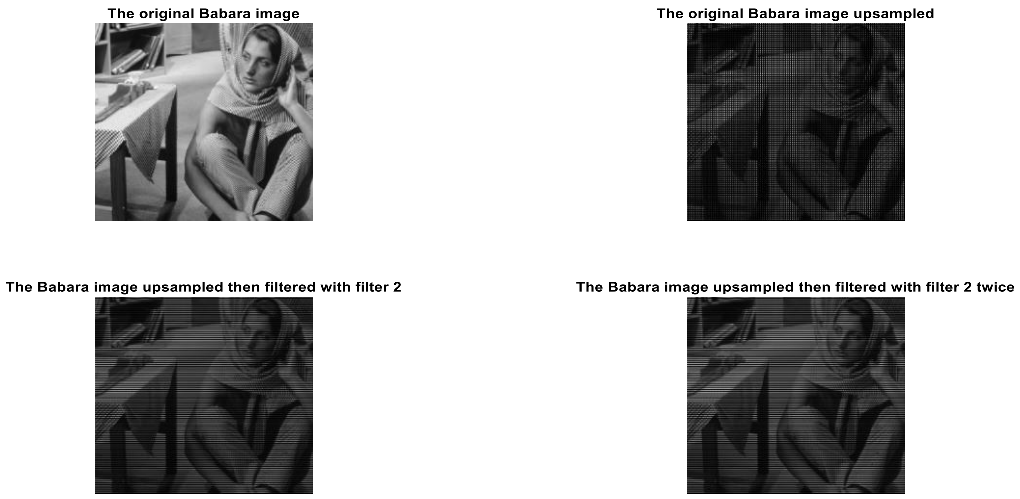

The original Babara image
The original Babara image upsampled
The Babara image upsampled then filtered with filter 2
The Babara image upsampled then filtered with filter 2 twice

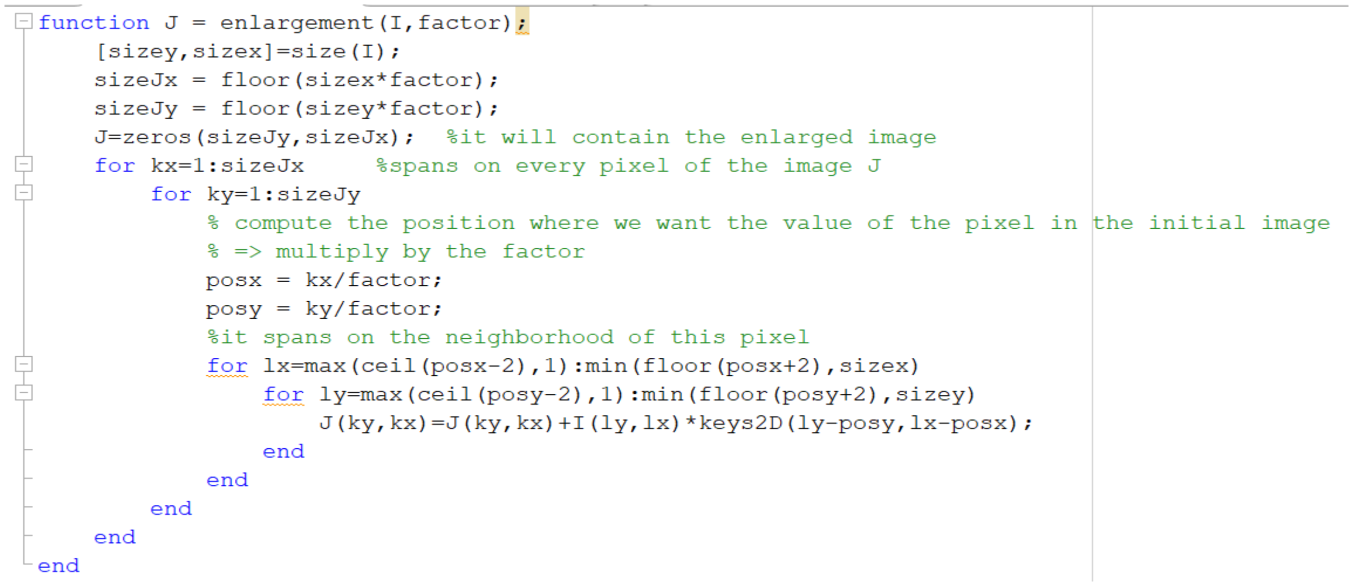
Figure 7. Comparing the results obtained with upsampling

We see clearly that the interpolation process helped distributing the intensity values more evenly across the image, so the discontinuities caused by the zero values have been reduced. I also tried to apply the filter a second time and the result was just a bit better, so the filter we chose is good enough to use just once.

So far we have learned two opposite processes: downsampling and upsampling, to decrease and increase an image's size. We also understood the two methods that go with these, that are **filtering** and **interpolation**.

3. Do it as a function:

After understanding the given code to rotate an image, I made the following code to change its size by factor that we want, which can be any positive number:



```
function J = enlargement(I,factor);
    [sizey,sizex]=size(I);
    sizeJx = floor(sizex*factor);
    sizeJy = floor(sizey*factor);
    J=zeros(sizeJy,sizeJx);  %it will contain the enlarged image
    for kx=1:sizeJx      %spans on every pixel of the image J
        for ky=1:sizeJy
            % compute the position where we want the value of the pixel in the initial image
            % => multiply by the factor
            posx = kx/factor;
            posy = ky/factor;
            %it spans on the neighborhood of this pixel
            for lx=max(ceil(posx-2),1):min(floor(posx+2),sizex)
                for ly=max(ceil(posy-2),1):min(floor(posy+2),sizey)
                    J(ky,kx)=J(ky,kx)+I(ly,lx)*keys2D(ly-posy,lx-posx);
                end
            end
        end
    end
end
```

Figure 8. The code MATLAB to change an image's size

In short, I changed the part of the function where we compute the position where we want the value of the pixel in the initial image. I then compared the result with the MATLAB built-in command *imresize*, by trying to increase the image's size by 30%:

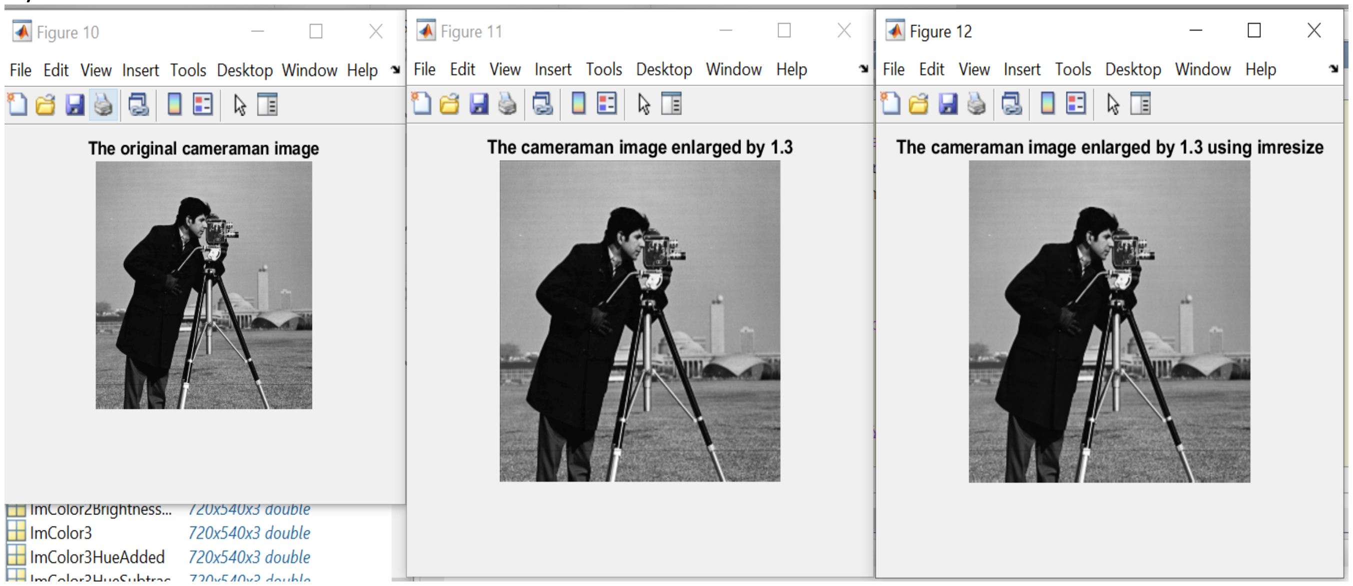


Figure 10
File Edit View Insert Tools Desktop Window Help
The original cameraman image
Figure 11
The cameraman image enlarged by 1.3
Figure 12
The cameraman image enlarged by 1.3 using imresize
ImColor2Brightness... 720x540x3 double
ImColor3 720x540x3 double
ImColor3HueAdded 720x540x3 double

Figure 9. Enlarging the Cameraman image and results comparison

We see that the function was correct as it did exactly what the *imresize* command did. Then, I tested with decreasing the image's size by 30%:

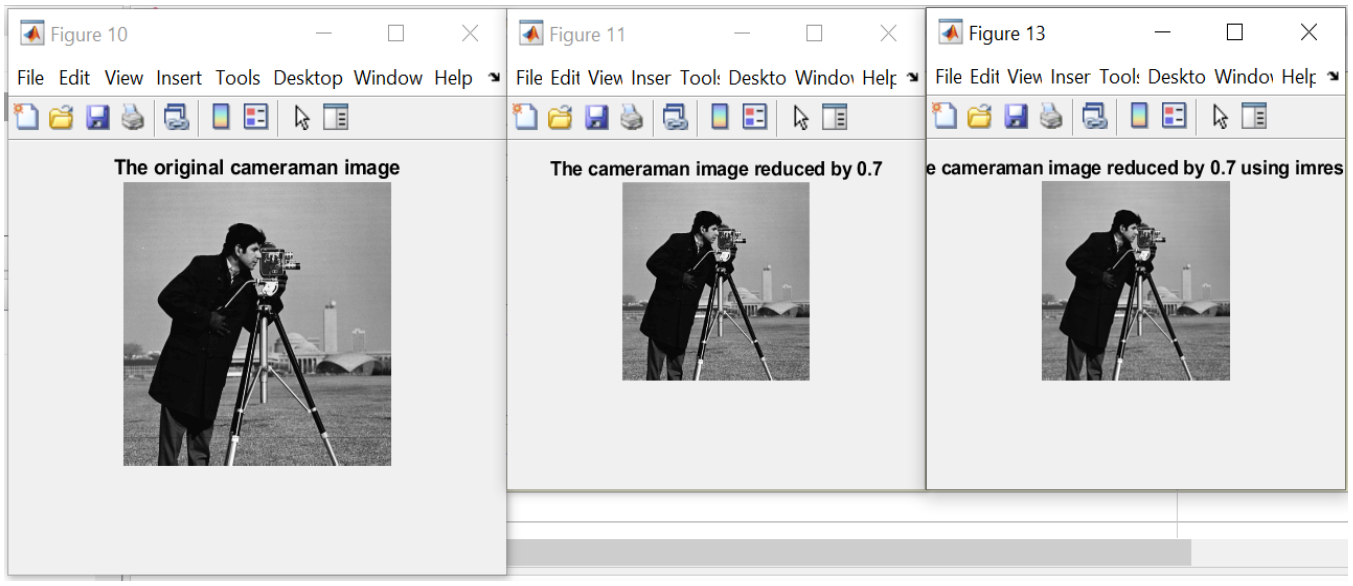


Figure 10
File Edit View Insert Tools Desktop Window Help
The original cameraman image
Figure 11
The cameraman image reduced by 0.7
Figure 13
e cameraman image reduced by 0.7 using imres

Figure 10. Reducing the Cameraman image and results comparison

This time we also got the same thing. This means that the function has been tested to be correct.

To compare this with the 2 previous approach I would say that this one is more helpful as we can change the image's size by any value we want instead of just by integer factors. Also this allows us to combine enlarging and reducing an image's size in one function.

II. Color spaces:

1. Saturation:

We now want to change the basis of an image from RGB which refers to the combination of the 3 main colors, to the luminance-chrominance basis. The last 2 elements about chrominance make the image's saturation and hue, while the first one deals with its brightness. Saturation deals with an image's color aspect. With a change of basis as said in the preparation part, I computed the 3 new basis elements. The image's saturation and hue are defined as:

$$\begin{cases} saturation = \sqrt{IC1^2 + IC2^2} \\ hue \ = \ atan2(IC1, IC2) \end{cases}$$

I then doubled and halved the saturation of this image (in the middle) and compared them:

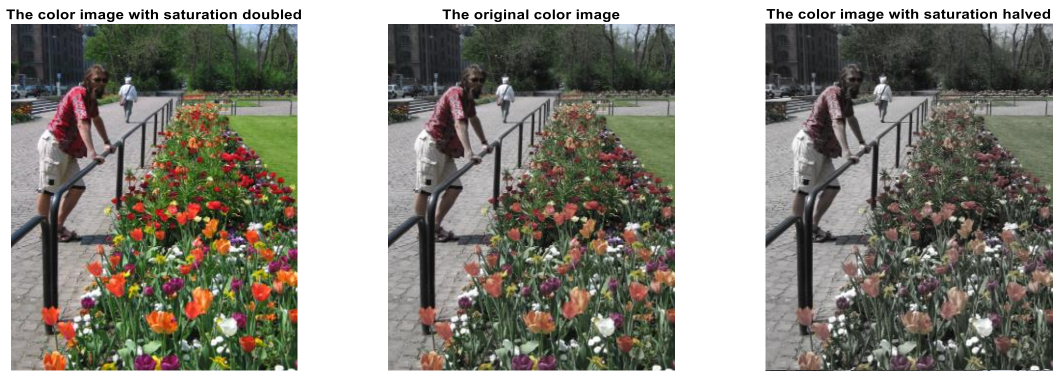

The color image with saturation doubled
The original color image
The color image with saturation halved

Figure 11. Doubling and halving an image's saturation

We see that as saturation only deals with the colors and not the brightness, the image's brightness stays the same. But to the left with a high value of saturation, the image appears more lively. So by changing saturation we can improve an image's visual aspect.

With a basis of RGB, we cannot do this. We need to switch to the new basis of luminance and chrominances, perform the changes in saturation and/or hue and/or brightness, then switch back to the original one using the inverse matrix operation I proposed in the preparation part.

Next, we would like to test some changes in the image's brightness. I did this to the following image by doubling and halving the luminance matrix:

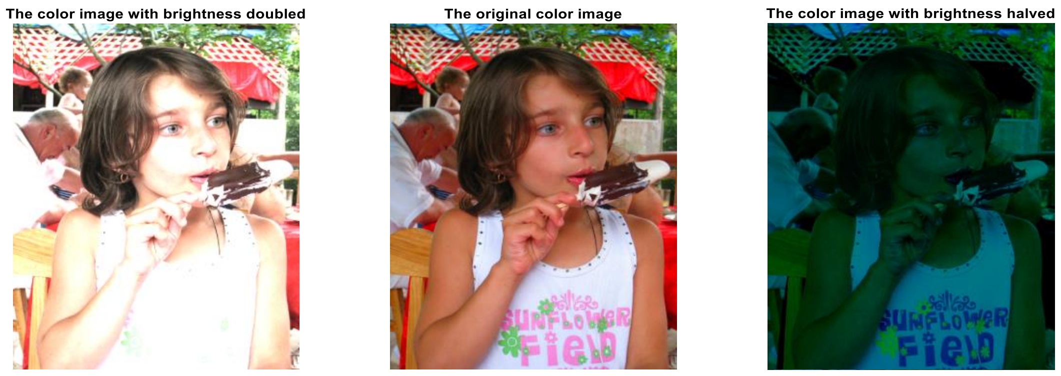

The color image with brightness doubled
The original color image
The color image with brightness halved

Figure 12. Doubling and halving an image's brightness

The result we have is very clear: the image became brighter and darker, but the saturation did not change. We still notice changes in colors (the words on the boy's shirt) because the luminance affects all the 3 main colors (see the P matrix).

2. Hue:

Hue is another parameter with which we can add some artistic effects to an image. Defined as atan2 of IC1 and IC2, hue can change the color of an image like in the following diagram:

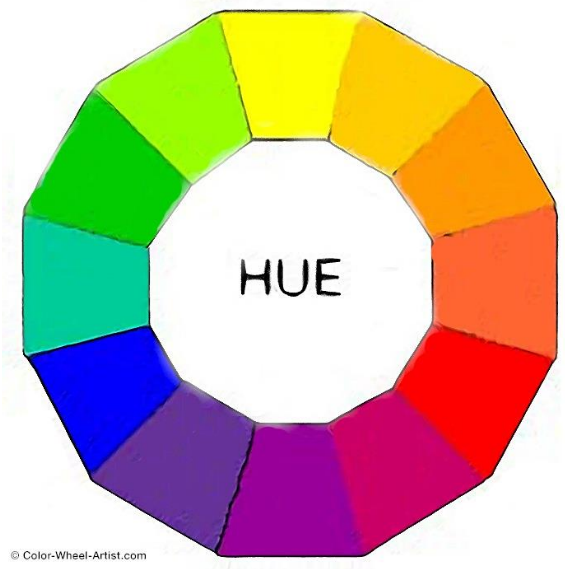


Figure 13. Relationship between hue and color

The image we have has mainly the red color, so an increase or decrease by 0.5 rad or about 28 degrees so move it to the forward to the yellow and backward to the purple zones, respectively. I applied those changes by adding and subtracting 0.5 from the image's hue, then changed back to the RGB basis:

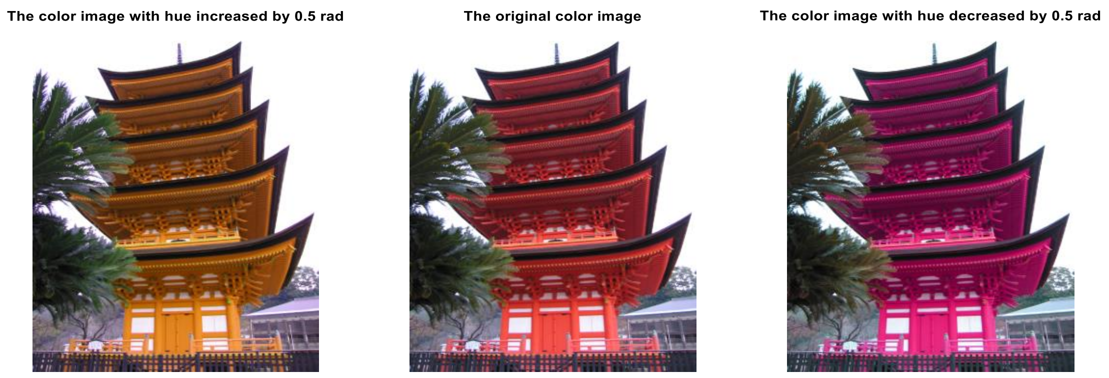

The color image with hue increased by 0.5 rad
The original color image
The color image with hue decreased by 0.5 rad

Figure 14. Increasing and decreasing an image's hue by 0.5 rad

Looking at the results above, we see that the color of the pagoda changed as expected.

III. Histograms:

1. Contrast stretching:

By definition, histogram refers to the statistical representation of a image's intensities by the number of pixels with that the corresponding intensities. I loaded this image of the Earth taken from outer space, then I cropped one part of it that is covered by the cloud. There appears to be some land but it was very difficult to see:

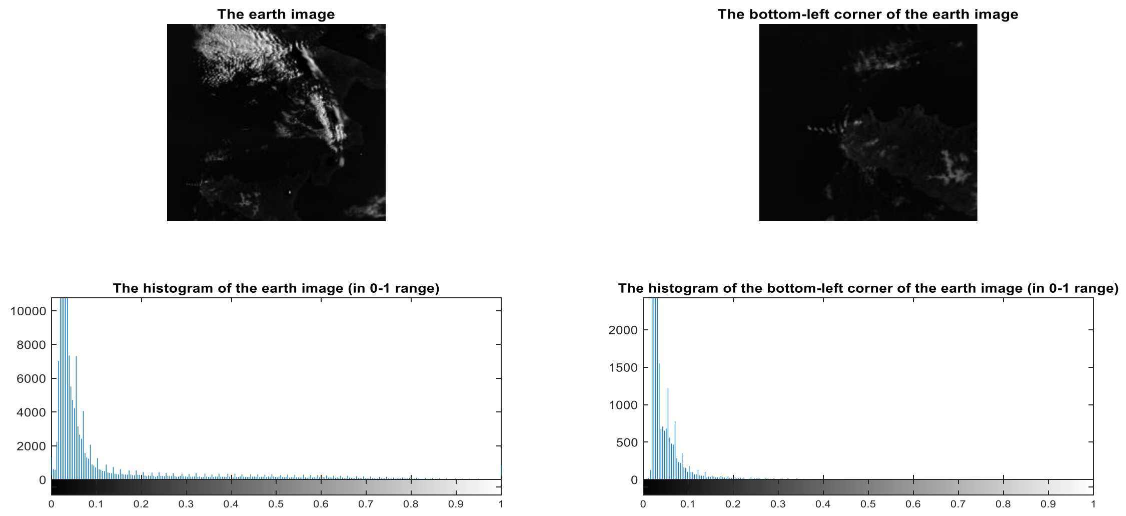


Figure 15. The histogram of the original and cropped photos

The reason why it is difficult to spot the different objects in the image is due to the very low contrast: There is hardly any noticeable difference in the intensity range which is quite narrow. Looking at the histograms (drawn in 0-1 range), we see that almost all pixels in the image has low values of intensities and that there were none with high intensity values. So the image appears dark (low contrast). There can be different methods both linear and nonlinear to stretch this range of values (see the preparation part).

I tried 3 different approaches to improve the contrast of this image. The first was **contrast stretching** which is a linear transformation as describe earlier. I noticed that the range of values is mainly from 0.02 to around 0.1 in the 0-1 scale, so I stretched this zone into 0-1 (0-255 in a 255 range) using the available function:

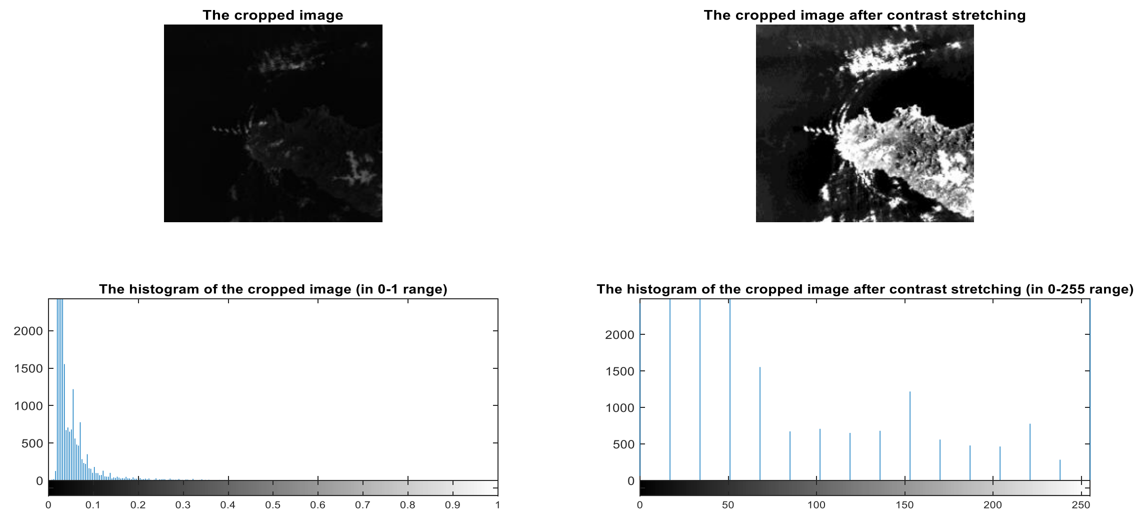


Figure 16. Contrast stretching the image's histogram

We see that with a stretched histogram, now we have a lot of intensity values spread across the full range. The image now has high contrast and we can now see the land part which was originally invisible. The important thing is to choose the correct range to stretch into the full range.

The second method I tried was **clipping**, which was generally a combination of over-stretching (stretching beyond the admissible range) and saturating the values out of scale. This allowed for greater contrast but if we overdo it, we will have a great deal of the high intensities, so we can no longer see the different zones in the land:

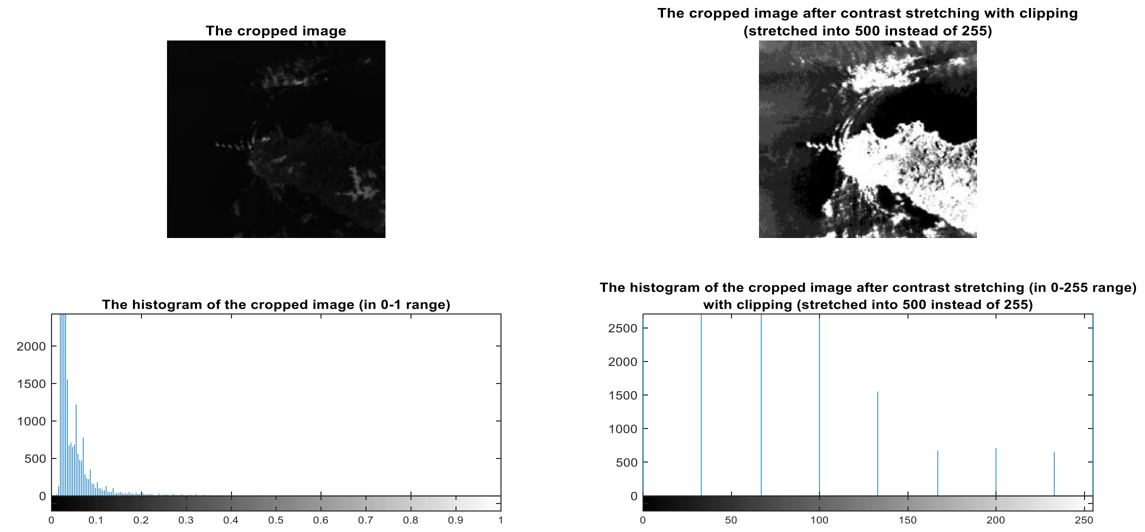


Figure 17. Contrast stretching with clipping

After some trials with different ranges, I finally got this balanced result when clipping in the zone 0-500 (of course the maximum intensity is still 255 in the 0-255 range). We can see clearly the continent and its lands. The important thing is to choose the range to clip into. It must not be too small (not very good contrast) or too big (too white to see anything).

The third method I tried was **thresholding**: to make the intensities below a given value go to 0 (min), and above it go to 1.

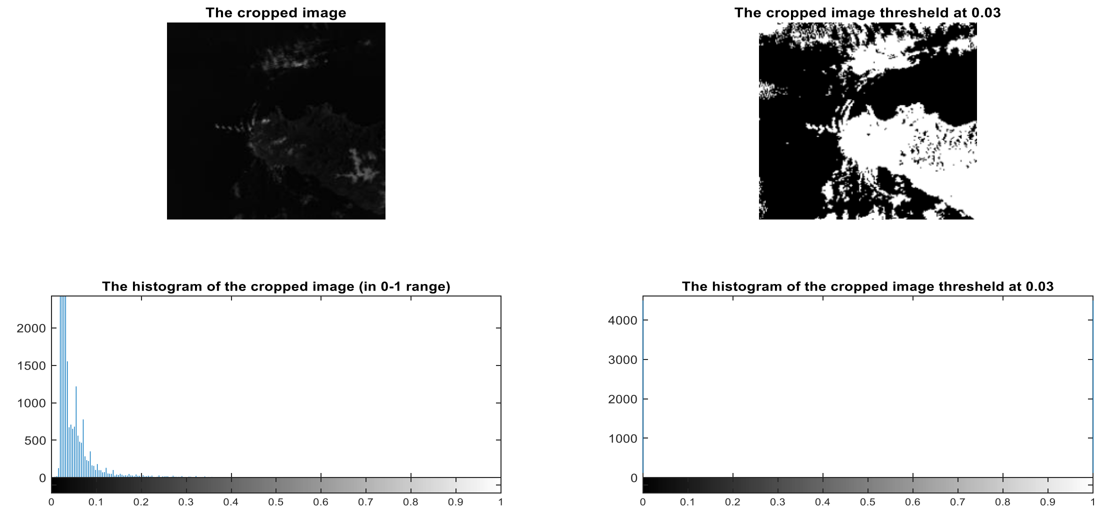


Figure 18. Thresholding at 0.03 to improve image's contrast

If the choose a too low threshold, the image will end up too white as all pixels go to maximum intensity. And on the other hand if it is too high the image will go black. After some trials I found the threshold value of 0.03 in the 0-1 range suitable. From the figure above we see that the histogram now only contains 0 and 1 with nothing in between – the result of thresholding. The important thing for this method is to choose the threshold value.

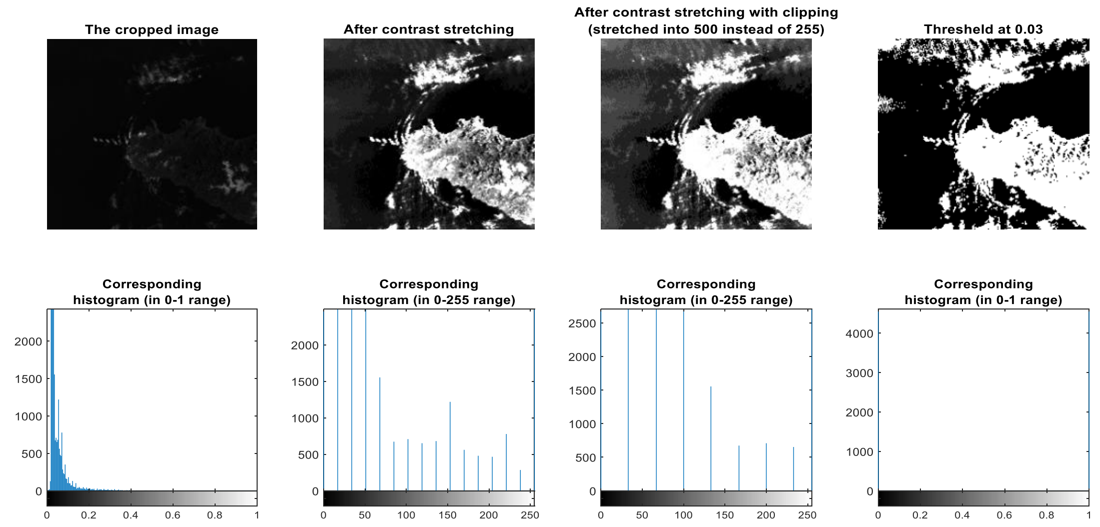


Figure 19. Comparing the results of 3 methods

Then I compared the 3 methods. All of them helped increase the contrast of an image by redistributing the histogram. The effectiveness varies by how well we choose the values. But note that only the 1st method is linear, which means we can recover to original image as there exists a 1-1 mapping between the histograms. With clipping and thresholding, this is impossible.

In this case where we have only a grayscale image and we only want to increase its contrast, a piecewise linear transformation is sufficient.

2. Histogram matching:

Given a color and a grayscale image, I want to study the histogram of each color, then to matching it with reference histogram which belongs to the grayscale image. To do this, I separated the colors RGB and find their corresponding histograms. In the figure below there are the color histograms and the reference one.

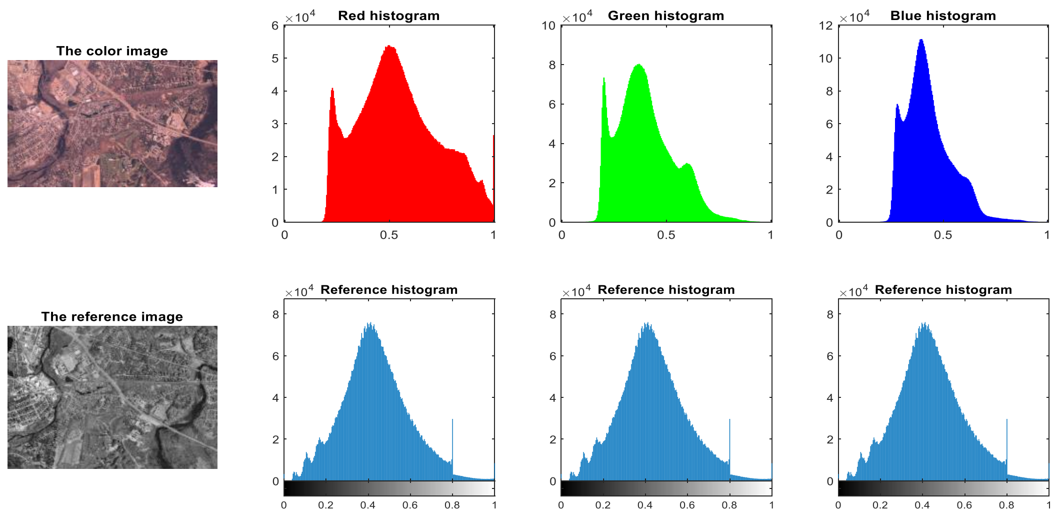


Figure 20. The color and reference histograms

Now, I need to suppose that each histogram follows a Gaussian distribution. By definition the means and standard deviation are defined by:

$$\begin{cases} Mean = \dfrac{\sum intensity \times number\ of\ pixels\ of\ that\ intensity}{total\ number\ of\ pixels} \\ SD = \sqrt{\dfrac{\sum number\ of\ pixels\ of\ that\ intensity \times (intensity\ -\ Mean)^2}{total\ number\ of\ pixels}} \end{cases}$$

I made some FOR loops to calculate these parameters for each color and the reference histogram. The values of mean and standard deviations are as follow:

	Red	**Green**	**Blue**	**Reference**
Mean	0.542415097387107	0.398178097965948	0.427399395149768	0.434601594719573
SD	0.195421429432720	0.136726459771172	0.111049702692827	0.158655041698607

Figure 21. Table of calculated means and standard deviations

After obtaining these values, I used this operation to match the histograms by shifting and scaling them:

$$M_i^{matched} = (M_i - \mu_i)\frac{\sigma_{ref}}{\sigma_i} + \mu_{ref}$$

Then I obtained the following results for the histograms:

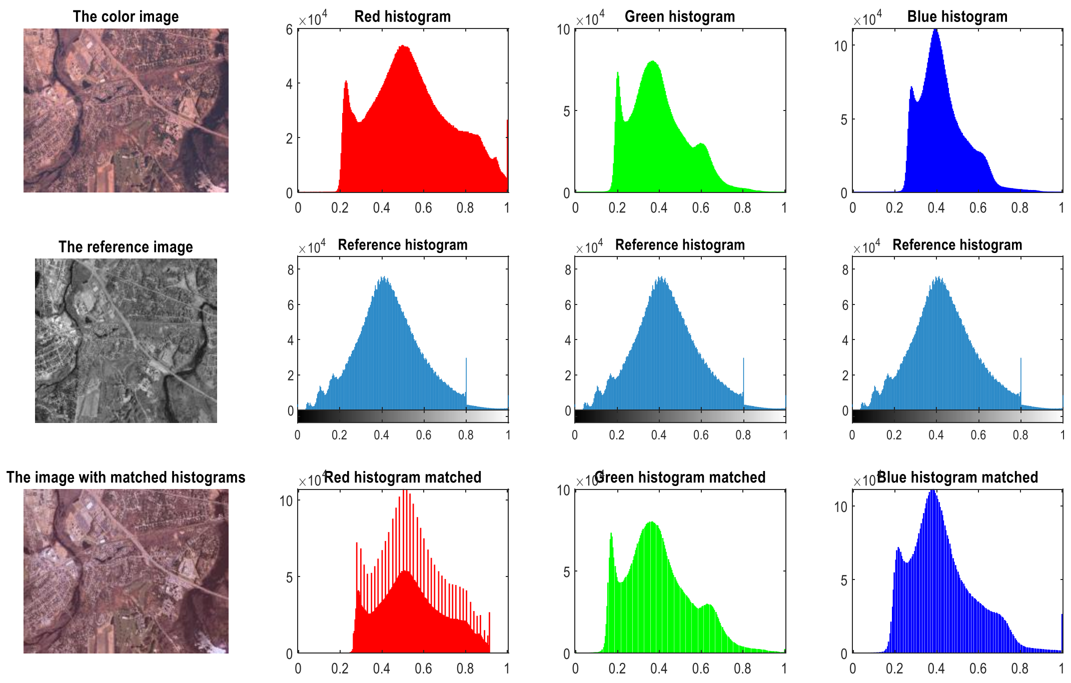


Figure 22. Results obtained from histogram matching

We see that the color histograms are now shifted and scaled to fit better with the reference one. We achieved better balance among the 3 colors. The result is a new image with better visual consistency, a combination of the 2 given images.

Conclusion: This Lab helped understand the 2 methods downsampling and upsampling with the problems behind them and the solutions that are filtering and interpolation, besides the fact that we can change an image's size by a non integer factor. It also allowed me to practice the use of basis changing in an image and how to work with saturation, hue and brightness. Lastly, I got to understand the use of histograms as well as the different operations that we can perform on histograms to enhance an image's contrast or to achieve better color balance.