22437 - Industrial Vision
Lab 3: Image Enhancement (Histogram Processing)

Miguel Ángel Calafat Torrens, Manuel Piñar Molina

Universitat de les Illes Balears

Histogram Stretching

Useful functions: *imhist*, *imadjust*, *stretchlim*.

Histogram stretching is a simple image enhancement technique that attempts to improve the contrast of an image modifying an input range of intensities to a desired output range of intensities. Unlike histogram equalization, contrast stretching is restricted to a linear mapping of input to output values. The result is less dramatic, but tends to avoid the sometimes artificial appearance of equalized images. As a result, the histogram of the resulting image tends to maintain its original shape. The simplest way of performing a histogram stretching is by means of the following transformation:

$$g(x,y) = (\mathrm{O_{max}} - \mathrm{O_{min}}) \frac{f(x,y) - \mathrm{I_{min}}}{\mathrm{I_{max}} - \mathrm{I_{min}}} + \mathrm{O_{min}}$$

where:

- $g(x,y)$: is the intensity value in the output image.
- $f(x,y)$: is the intensity value in the input image.
- $\mathrm{I_{min}}$: is the minimum intensity in the input image.
- $\mathrm{I_{max}}$: is the maximum intensity in the input image.
- $\mathrm{O_{min}}$: is the minimum desired intensity in the output image.
- $\mathrm{O_{max}}$: is the maximum desired intensity in the output image.

Normally this transformation is applied using the full intensity range for the output image and considering the minimum and the maximum intensity values present in the input image. This information can be obtained from the histogram.

Perform the following tasks:

1. Display the images *pollenlow.jpg*, *pollenblack.jpg*, *pollenwhite.jpg* along with their corresponding histograms using the *imhist* function. What can you say about the contrast of each of these images?
2. Write a function in Matlab to adjust the contrast of an image using the above-mentioned formula and avoiding specific Matlab functions designed for this purpose. The function signature should be:

   ```
   function imadj = adjust(image)
   ```

   where *image* is the original image containing values in the range $[0.0, 1.0]$ and *imadj* is the resulting adjusted image.
3. Using the function created in the previous point, adjust the contrast of the three images of exercise 1 and display the resulting images along with their corresponding histograms. Compare the histograms of the adjusted images with the original ones.

4. Apply the function provided by Matlab for adjusting the contrast of the three images, and display the resulting images with their histograms. Are these histograms the same as the ones obtained in the previous exercise? Why?

5. Would it be possible to use the Matlab's contrast adjusting function to obtain the complement (negative) of an image?. Think about it, try to perform this operation using the three images and show the results.

6. Given the three versions of the images: the original one, the adjusted one obtained with your function and the adjusted one using the Matlab's function, which one would you use for a further processing step? Why?

Histogram Equalization

Useful functions: *histeq*, *adapthisteq*.

Histogram equalization is a global technique to enhance the image contrast of an image. The idea is to find a transformation between the input and output gray levels which ideally generates the same number of pixels for each intensity value in the resulting image. To this end, we need to use the probability of occurrence of each intensity level in the original image, which can be approximated by a normalized histogram of the image:

$$h_n(i) = \frac{h(i)}{NM} \forall i \in L$$

where h_n is the normalized histogram, h is the original histogram, NM is the total number of pixels in the image and L is the total number of gray levels. Note that the sum of all values in the normalized histogram should be one. For discrete quantities, the histogram equalization transformation is defined as:

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} h_n(j)$$

where s_k is the intensity value in the output image corresponding to the r_k in the input image. This is the cumulative sum of normalized histogram values, expanded along all the gray level intensities.

Perform the tasks listed below:

1. Write a function in Matlab for computing the cumulative sum of a normalized histogram. The function signature should be:

   ```
   function cdf = csum(hnorm)
   ```

   where *hnorm* is the normalized histogram of an image and *cdf* is the resulting cumulative sum.

2. Write a function in Matlab to equalize an image using a uniform transformation with 256 gray levels. The function signature should be:

   ```
   function imageeq = equalize(image)
   ```

   where *image* is the input image and *imageeq* is the resulting equalized image. This function must use the *csum* function implemented in the previous exercise.

3. Equalize the images *pollenlow.jpg*, *pollenblack.jpg*, *pollenwhite.jpg* using the function written in the previous point. Display each resulting image and its corresponding histogram in figures.

4. Equalize the images *pollenlow.jpg*, *pollenblack.jpg*, *pollenwhite.jpg* using the function provided by Matlab. Display the resulting images and their histograms, and compare the results with the histograms computed in the previous exercise.

5. Compare the results obtained with histogram stretching and equalization and explain briefly the main differences between both approaches.

6. Examine the *adapthisteq* Matlab's function. How does it work?. Equalize the histograms of the images *pollenlow.jpg*, *pollenblack.jpg* and *pollenwhite.jpg* using two different grid sizes and display the results.