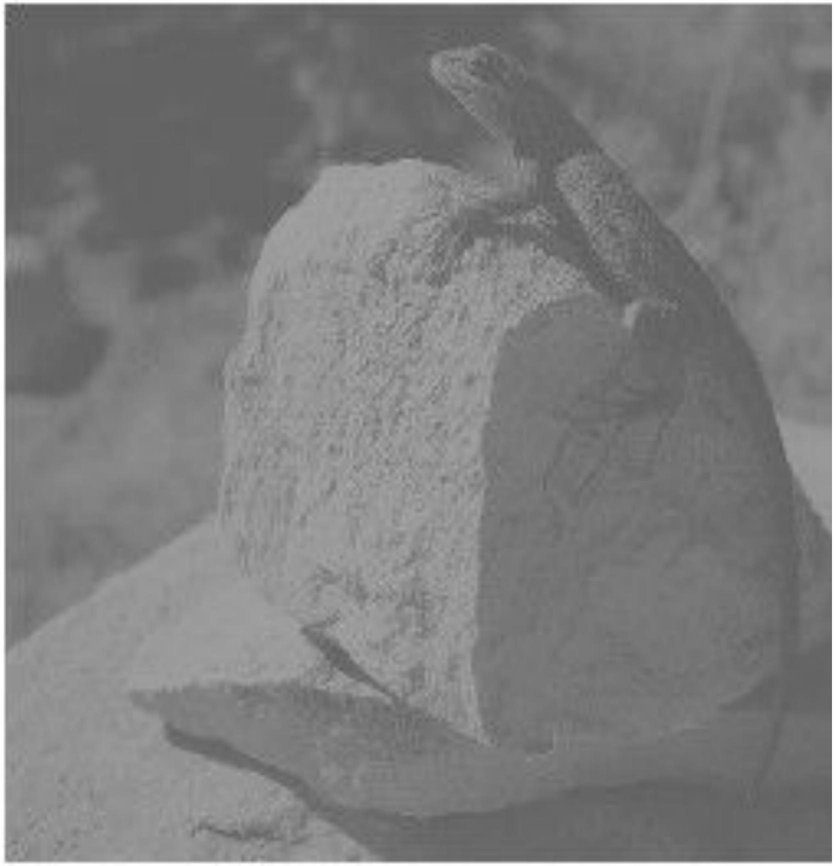
Lesson 4: Contrast Manipulation, Equalization and LUT

Jorge Beltrán de la Cita, Arturo de la Escalera

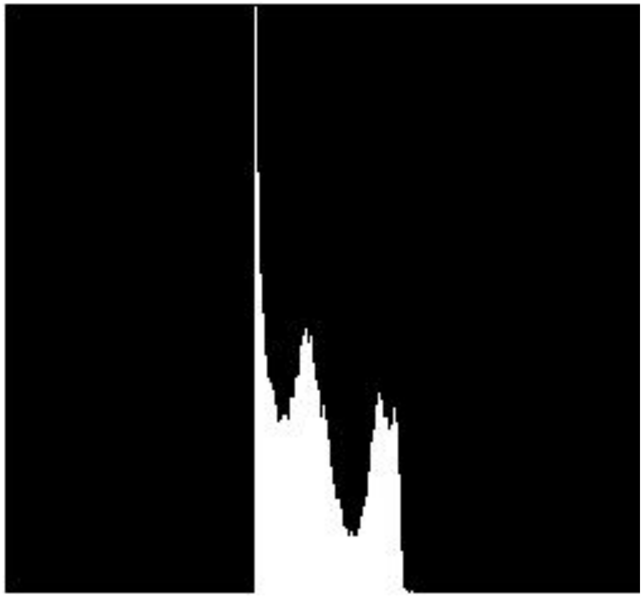
Perception Systems

Course 2019-2020

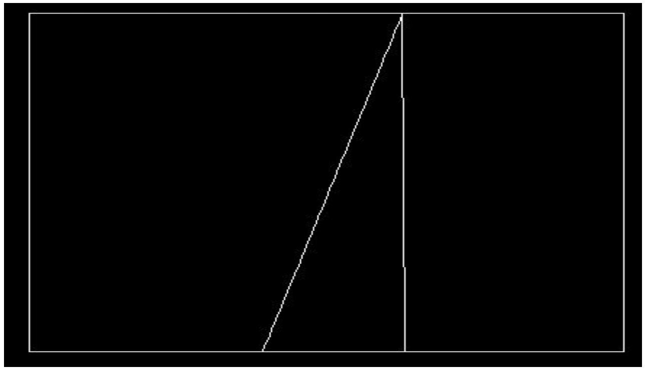
- Scale Amplitude



Original image



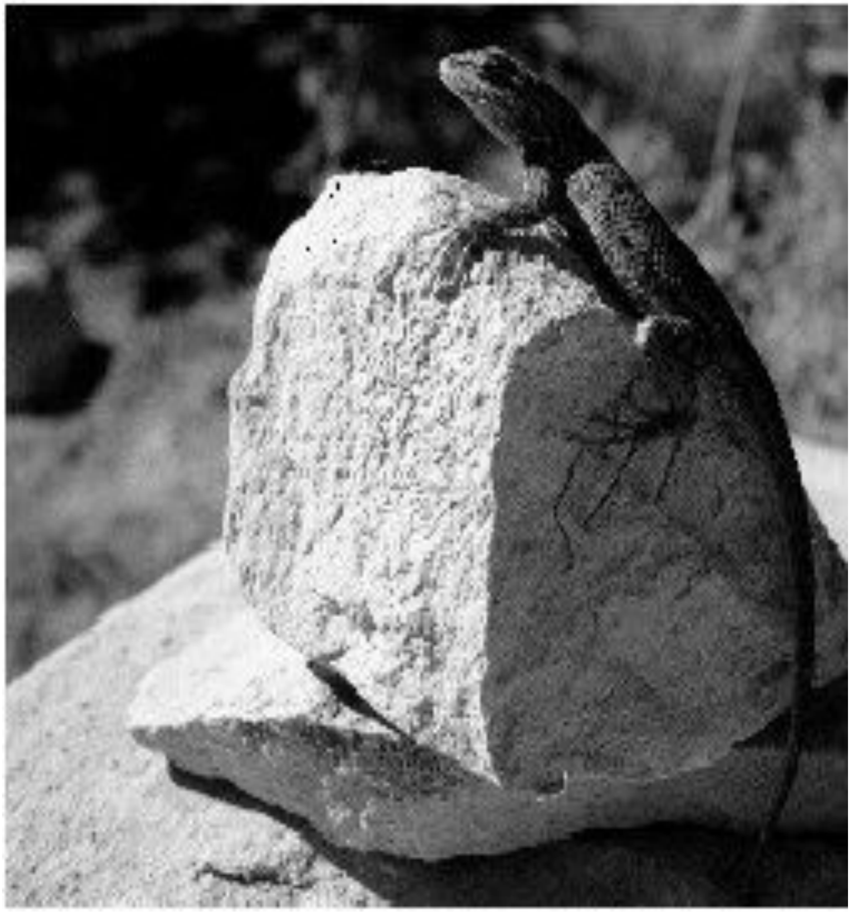
Original histogram



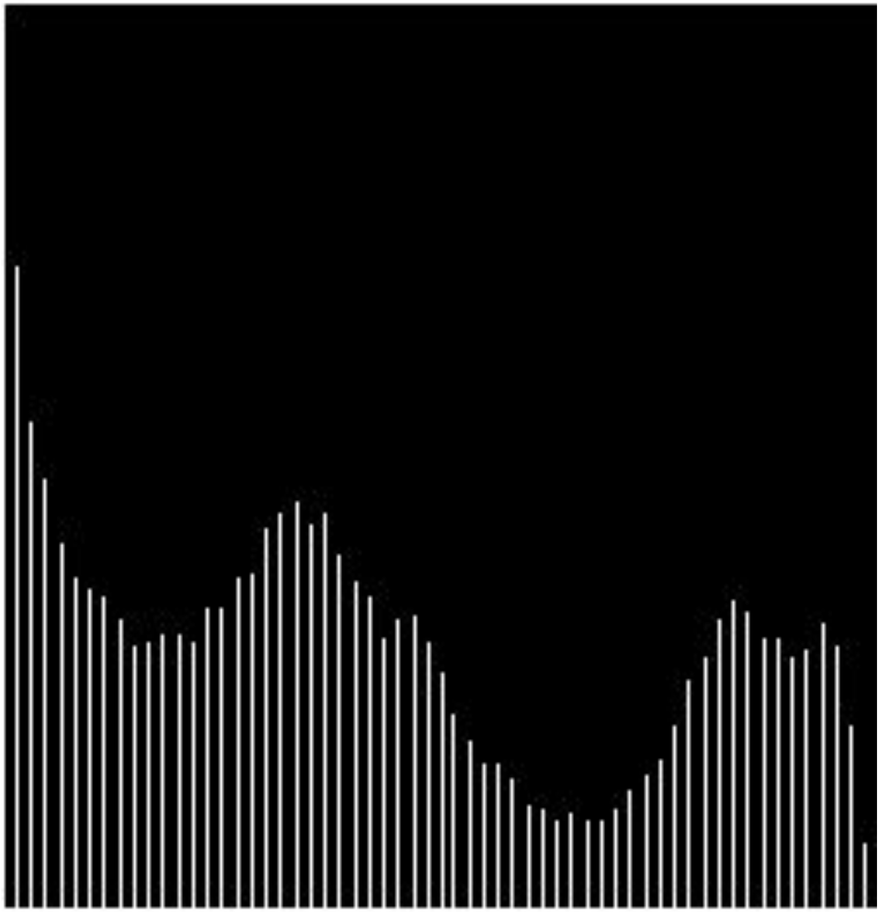
Scale amplitude

Lesson 4: Contrast manipulation, Equalization and LUTs

- Scale Amplitude



New image with scale amplitude



New histogram

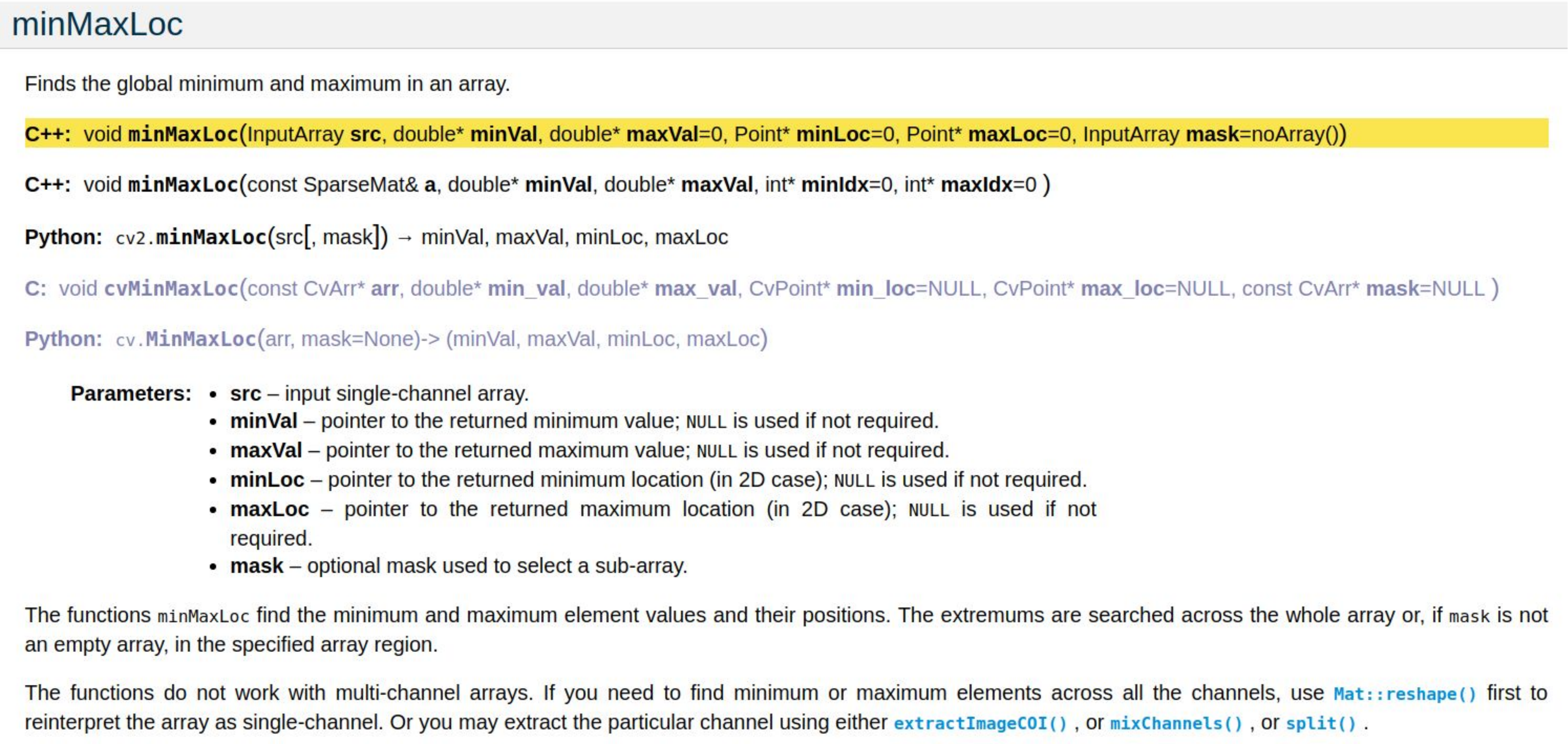
Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 1: amplitude of a histogram

- Load a grayscale image
- Compute the histogram of the image
- Print the values of the histogram
- Get the maximum and minimum values of the histogram
- Plot the original image and the histogram
- Free memory

Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 1: amplitude of a histogram



minMaxLoc

Finds the global minimum and maximum in an array.

C++: void **`minMaxLoc`**(InputArray **src**, double* **minVal**, double* **maxVal**=0, Point* **minLoc**=0, Point* **maxLoc**=0, InputArray **mask**=noArray())

C++: void **`minMaxLoc`**(const SparseMat& **a**, double* **minVal**, double* **maxVal**, int* **minIdx**=0, int* **maxIdx**=0)

Python: `cv2.`**`minMaxLoc`**(src[, mask]) → minVal, maxVal, minLoc, maxLoc

C: void **`cvMinMaxLoc`**(const CvArr* **arr**, double* **min_val**, double* **max_val**, CvPoint* **min_loc**=NULL, CvPoint* **max_loc**=NULL, const CvArr* **mask**=NULL)

Python: `cv.`**`MinMaxLoc`**(arr, mask=None)-> (minVal, maxVal, minLoc, maxLoc)

Parameters:

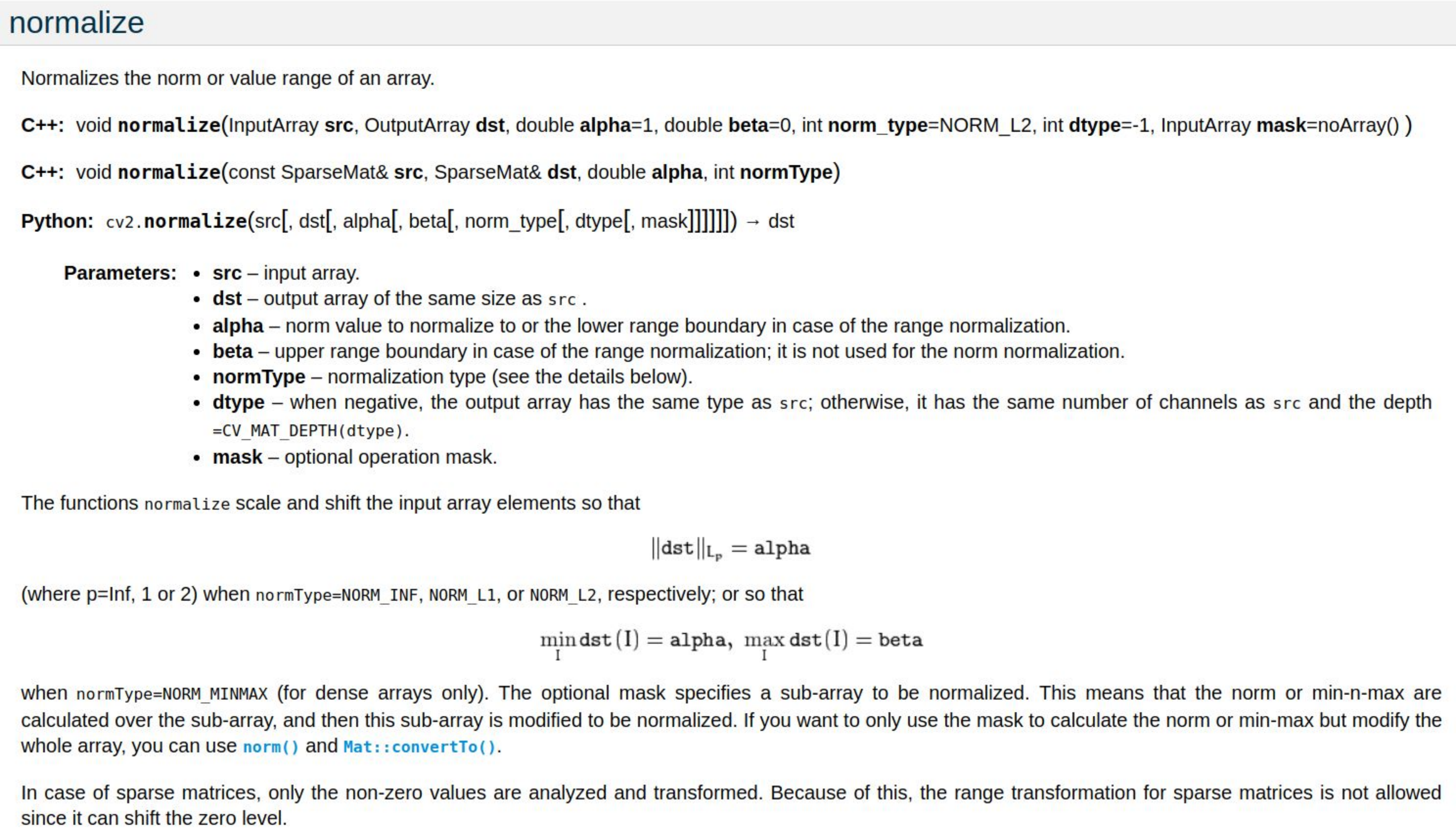
- **src** – input single-channel array.
- **minVal** – pointer to the returned minimum value; `NULL` is used if not required.
- **maxVal** – pointer to the returned maximum value; `NULL` is used if not required.
- **minLoc** – pointer to the returned minimum location (in 2D case); `NULL` is used if not required.
- **maxLoc** – pointer to the returned maximum location (in 2D case); `NULL` is used if not required.
- **mask** – optional mask used to select a sub-array.

The functions `minMaxLoc` find the minimum and maximum element values and their positions. The extremums are searched across the whole array or, if `mask` is not an empty array, in the specified array region.

The functions do not work with multi-channel arrays. If you need to find minimum or maximum elements across all the channels, use `Mat::reshape()` first to reinterpret the array as single-channel. Or you may extract the particular channel using either `extractImageCOI()` , or `mixChannels()` , or `split()` .

Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 1: amplitude of a histogram



normalize

Normalizes the norm or value range of an array.

C++: void **normalize**(InputArray **src**, OutputArray **dst**, double **alpha**=1, double **beta**=0, int **norm_type**=NORM_L2, int **dtype**=-1, InputArray **mask**=noArray())

C++: void **normalize**(const SparseMat& **src**, SparseMat& **dst**, double **alpha**, int **normType**)

Python: cv2.**normalize**(src[, dst[, alpha[, beta[, norm_type[, dtype[, mask]]]]]]) → dst

Parameters:
- **src** – input array.
- **dst** – output array of the same size as `src` .
- **alpha** – norm value to normalize to or the lower range boundary in case of the range normalization.
- **beta** – upper range boundary in case of the range normalization; it is not used for the norm normalization.
- **normType** – normalization type (see the details below).
- **dtype** – when negative, the output array has the same type as `src`; otherwise, it has the same number of channels as `src` and the depth `=CV_MAT_DEPTH(dtype)`.
- **mask** – optional operation mask.

The functions `normalize` scale and shift the input array elements so that

$$\|\mathtt{dst}\|_{L_p} = \mathtt{alpha}$$

(where p=Inf, 1 or 2) when `normType=NORM_INF`, `NORM_L1`, or `NORM_L2`, respectively; or so that

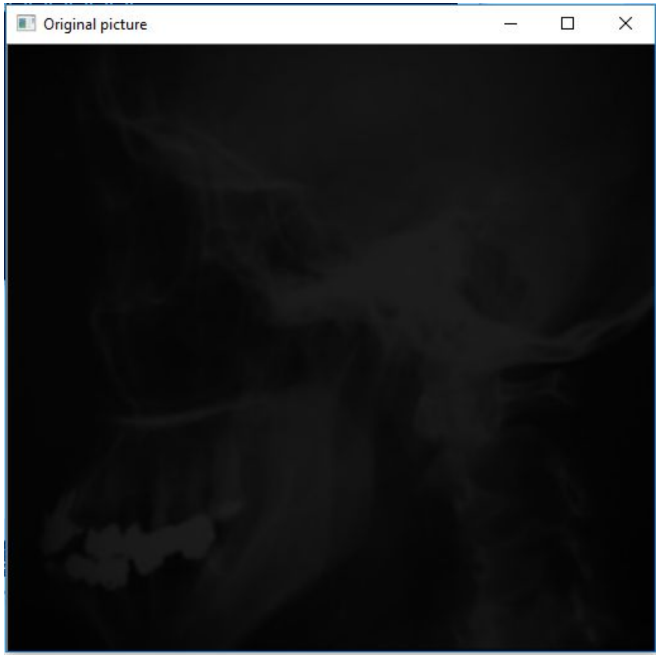
$$\min_I \mathtt{dst}(I) = \mathtt{alpha}, \ \max_I \mathtt{dst}(I) = \mathtt{beta}$$

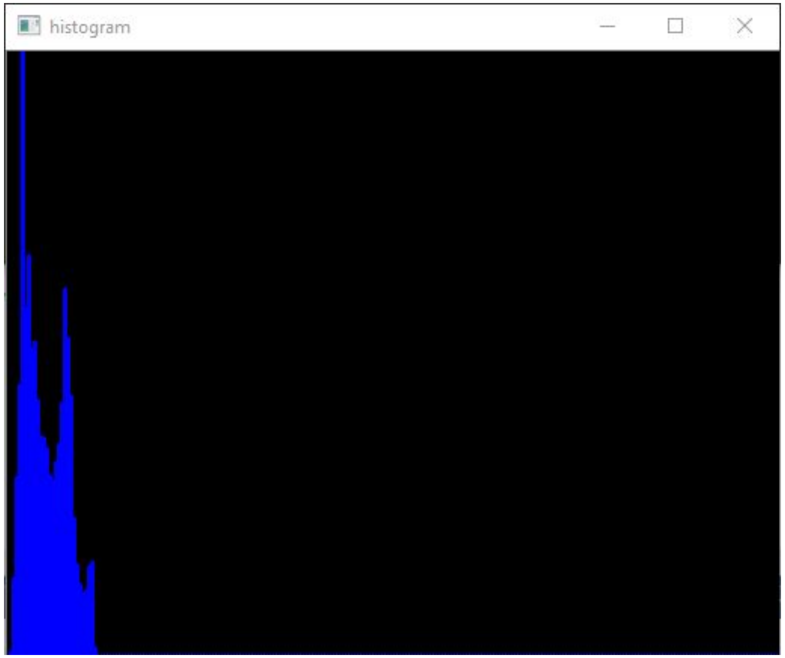
when `normType=NORM_MINMAX` (for dense arrays only). The optional mask specifies a sub-array to be normalized. This means that the norm or min-n-max are calculated over the sub-array, and then this sub-array is modified to be normalized. If you want to only use the mask to calculate the norm or min-max but modify the whole array, you can use `norm()` and `Mat::convertTo()`.

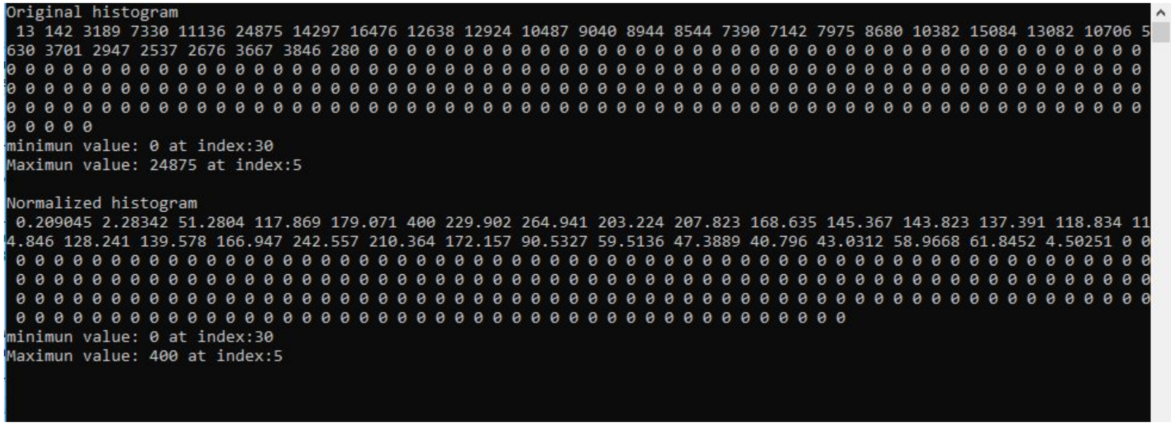
In case of sparse matrices, only the non-zero values are analyzed and transformed. Because of this, the range transformation for sparse matrices is not allowed since it can shift the zero level.

Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 1: amplitude of a histogram





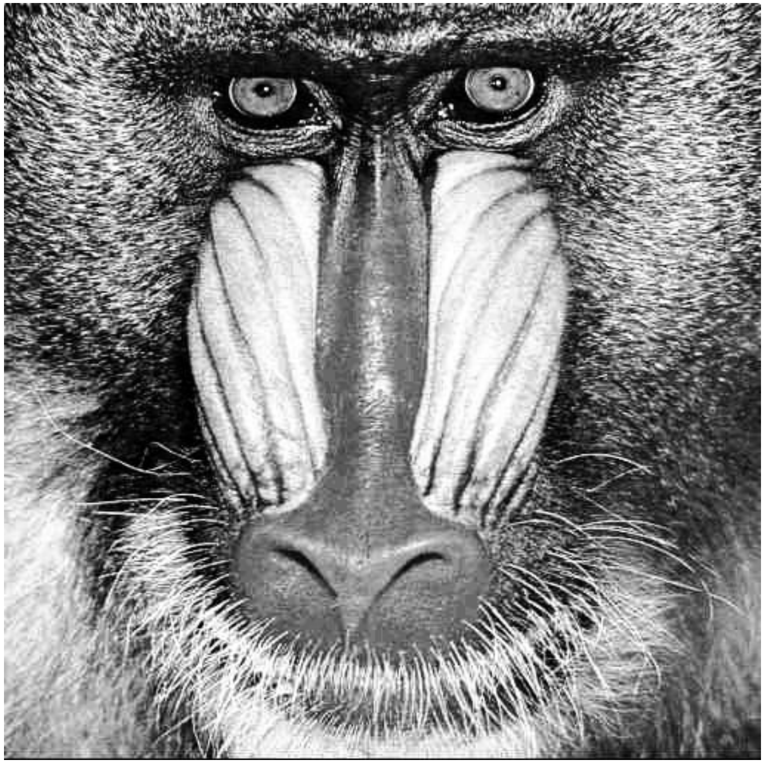


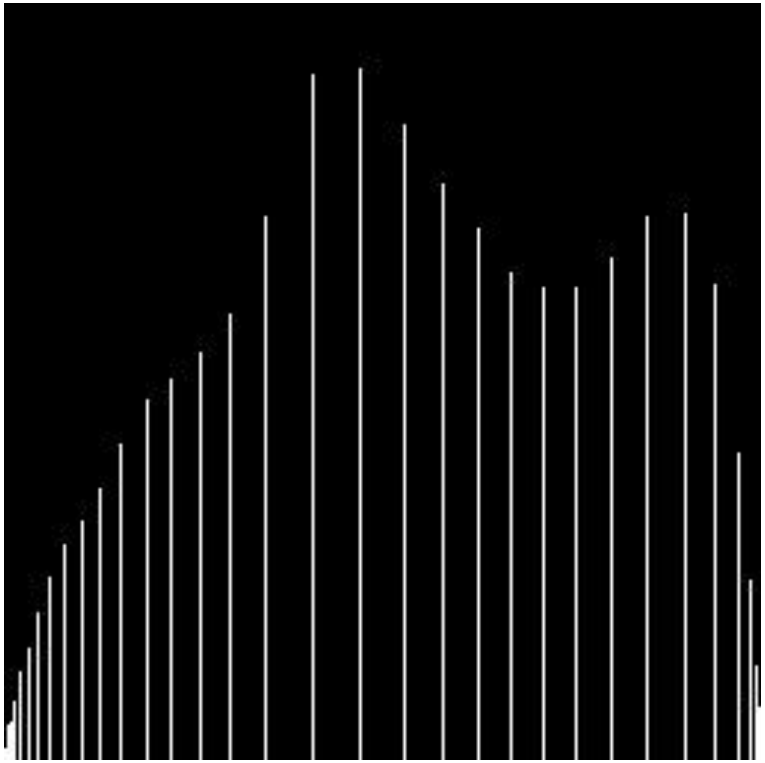
```
Original histogram
 13 142 3189 7330 11136 24875 14297 16476 12638 12924 10487 9040 8944 8544 7390 7142 7975 8680 10382 15084 13082 10706 5
630 3701 2947 2537 2676 3667 3846 280 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
minimun value: 0 at index:30
Maximun value: 24875 at index:5

Normalized histogram
 0.209045 2.28342 51.2804 117.869 179.071 400 229.902 264.941 203.224 207.823 168.635 145.367 143.823 137.391 118.834 11
4.846 128.241 139.578 166.947 242.557 210.364 172.157 90.5327 59.5136 47.3889 40.796 43.0312 58.9668 61.8452 4.50251 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
minimun value: 0 at index:30
Maximun value: 400 at index:5
```

Lesson 4: Contrast manipulation, Equalization and LUTs

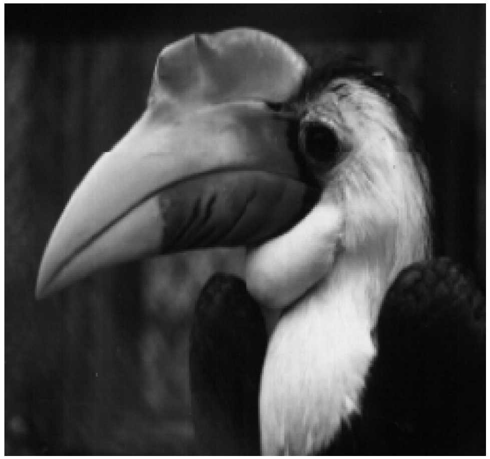
- Histogram equalization

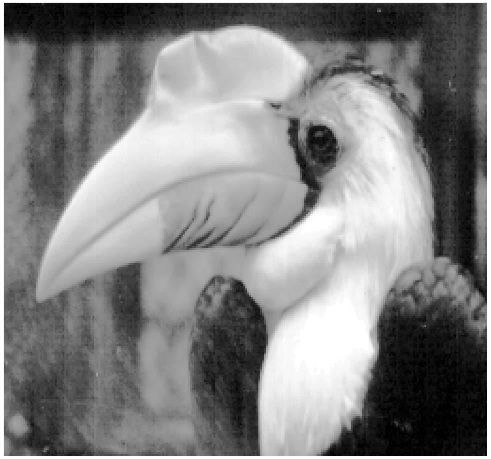




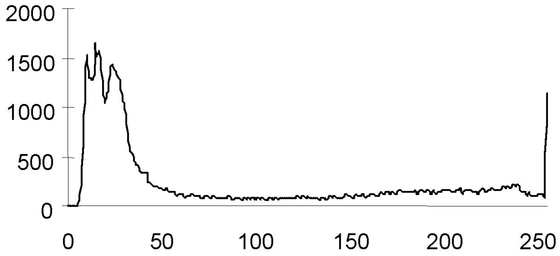
Lesson 4: Contrast manipulation, Equalization and LUTs

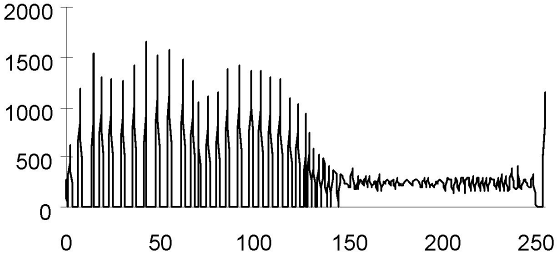
- Histogram equalization



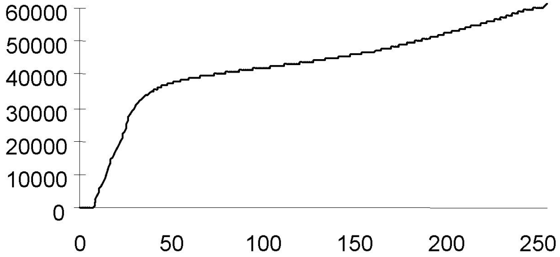


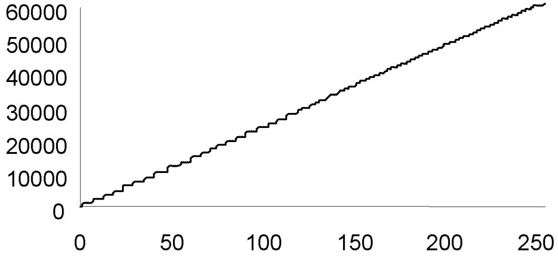
Histogram





Accumulated Histogram



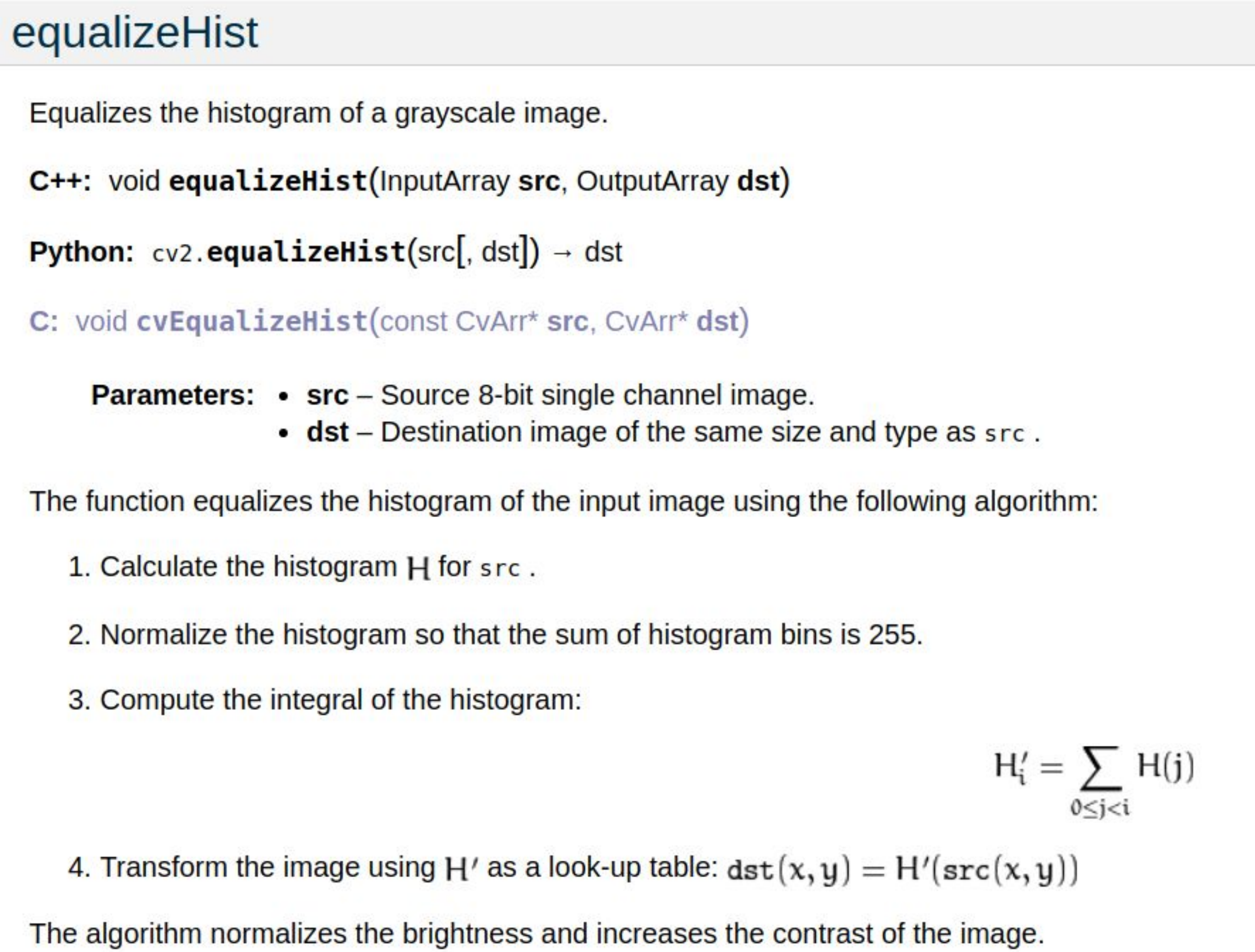


Exercise 2. Equalize the histogram of an image

- Load a grayscale image
- Compute the histogram of the image
- Print the values of the histogram
- Equalize the histogram
- Print the values of the equalized histogram
- Plot both histograms
- Plot both the original and the equalized image
- Free memory

Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 2. Equalize the histogram of an image



equalizeHist

Equalizes the histogram of a grayscale image.

C++: void **equalizeHist**(InputArray **src**, OutputArray **dst**)

Python: cv2.**equalizeHist**(src[, dst]) → dst

C: void **cvEqualizeHist**(const CvArr* **src**, CvArr* **dst**)

Parameters:
- **src** – Source 8-bit single channel image.
- **dst** – Destination image of the same size and type as `src` .

The function equalizes the histogram of the input image using the following algorithm:

1. Calculate the histogram H for `src` .
2. Normalize the histogram so that the sum of histogram bins is 255.
3. Compute the integral of the histogram:

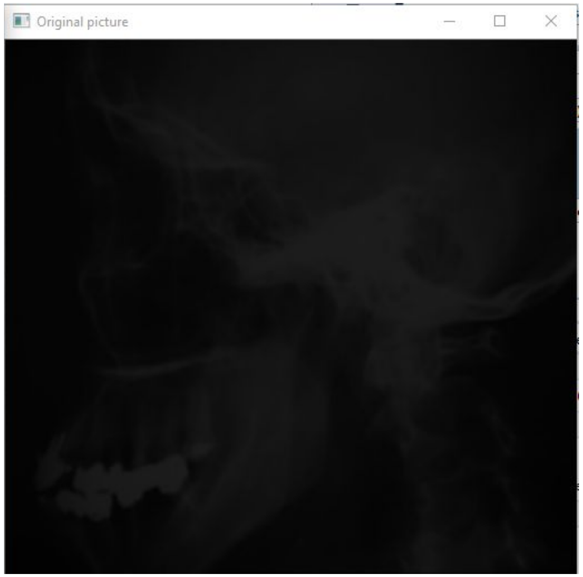
$$H'_i = \sum_{0 \le j < i} H(j)$$

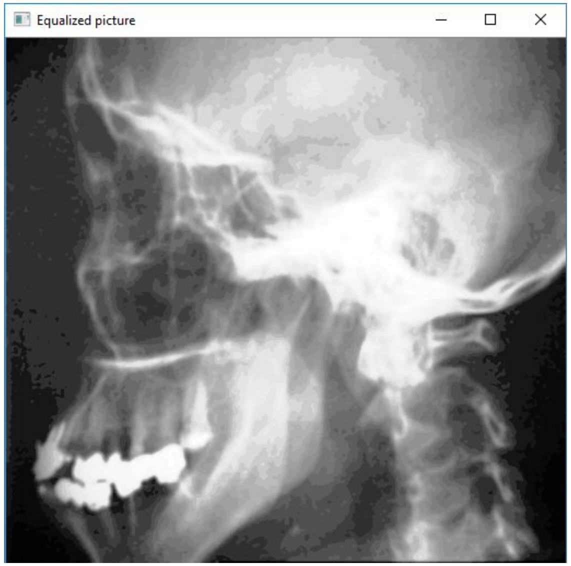
4. Transform the image using H' as a look-up table: $\texttt{dst}(x, y) = H'(\texttt{src}(x, y))$

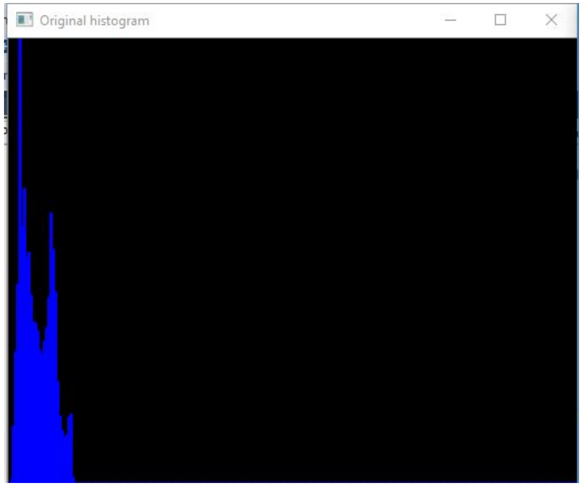
The algorithm normalizes the brightness and increases the contrast of the image.

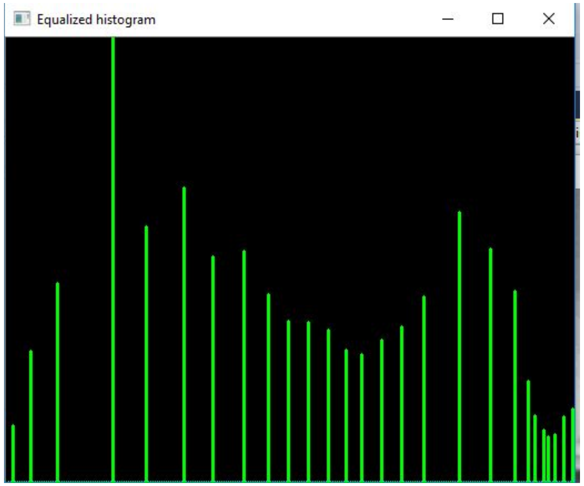
Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 2. Equalize the histogram of an image



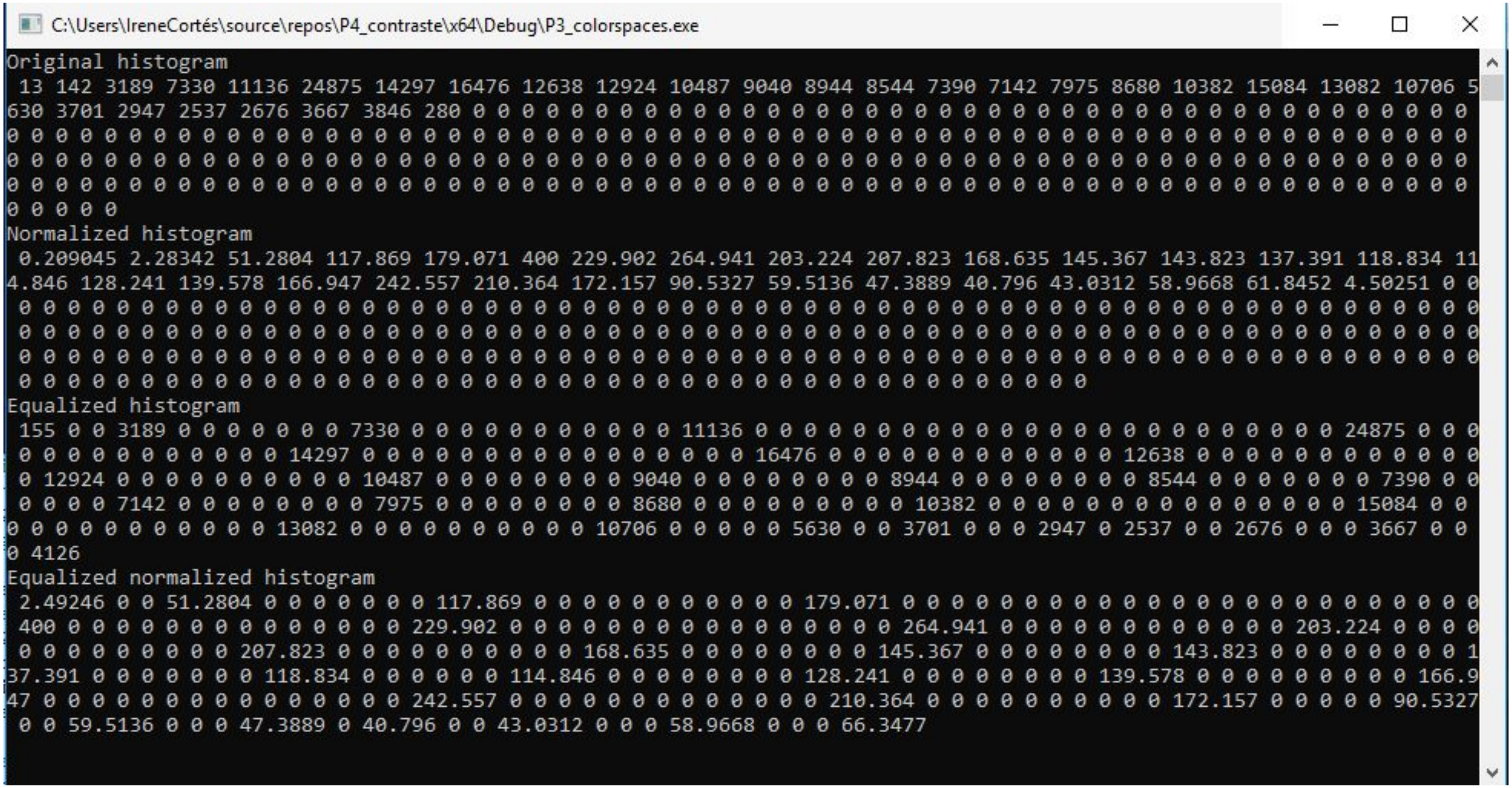






Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 2. Equalize the histogram of an image



C:\Users\IreneCortés\source\repos\P4_contraste\x64\Debug\P3_colorspaces.exe

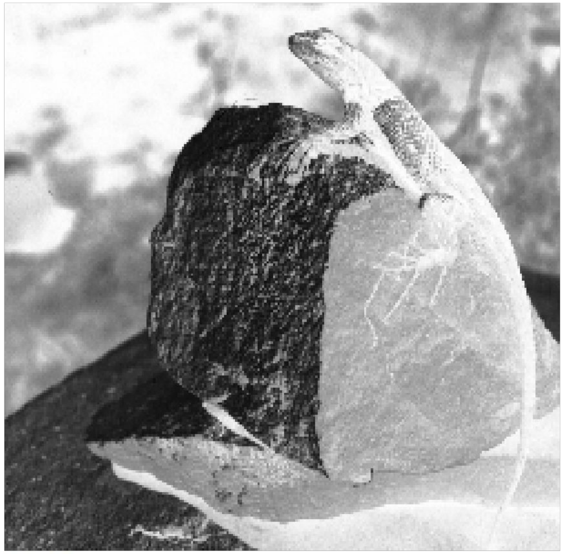
```
Original histogram
 13 142 3189 7330 11136 24875 14297 16476 12638 12924 10487 9040 8944 8544 7390 7142 7975 8680 10382 15084 13082 10706 5
630 3701 2947 2537 2676 3667 3846 280 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
Normalized histogram
 0.209045 2.28342 51.2804 117.869 179.071 400 229.902 264.941 203.224 207.823 168.635 145.367 143.823 137.391 118.834 11
4.846 128.241 139.578 166.947 242.557 210.364 172.157 90.5327 59.5136 47.3889 40.796 43.0312 58.9668 61.8452 4.50251 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Equalized histogram
 155 0 0 3189 0 0 0 0 0 0 0 7330 0 0 0 0 0 0 0 0 0 0 0 11136 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 24875 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 14297 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 16476 0 0 0 0 0 0 0 0 0 0 0 0 12638 0 0 0 0 0 0 0 0 0 0 0 0
 0 12924 0 0 0 0 0 0 0 0 0 0 10487 0 0 0 0 0 0 0 0 9040 0 0 0 0 0 0 0 0 8944 0 0 0 0 0 0 0 0 8544 0 0 0 0 0 0 0 7390 0 0
 0 0 0 0 7142 0 0 0 0 0 0 0 0 7975 0 0 0 0 0 0 0 0 8680 0 0 0 0 0 0 0 0 0 10382 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15084 0 0
0 0 0 0 0 0 0 0 0 0 0 13082 0 0 0 0 0 0 0 0 0 0 10706 0 0 0 0 0 5630 0 0 3701 0 0 0 2947 0 2537 0 0 2676 0 0 0 3667 0 0
0 4126
Equalized normalized histogram
 2.49246 0 0 51.2804 0 0 0 0 0 0 0 117.869 0 0 0 0 0 0 0 0 0 0 0 179.071 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 400 0 0 0 0 0 0 0 0 0 0 0 0 0 0 229.902 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 264.941 0 0 0 0 0 0 0 0 0 0 0 0 203.224 0 0 0 0
 0 0 0 0 0 0 0 0 0 207.823 0 0 0 0 0 0 0 0 0 0 168.635 0 0 0 0 0 0 0 0 145.367 0 0 0 0 0 0 0 0 143.823 0 0 0 0 0 0 0 0 1
37.391 0 0 0 0 0 0 0 118.834 0 0 0 0 0 0 114.846 0 0 0 0 0 0 0 0 128.241 0 0 0 0 0 0 0 0 139.578 0 0 0 0 0 0 0 0 0 166.9
47 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 242.557 0 0 0 0 0 0 0 0 0 0 0 0 0 210.364 0 0 0 0 0 0 0 0 0 0 172.157 0 0 0 0 0 90.5327
 0 0 59.5136 0 0 0 47.3889 0 40.796 0 0 43.0312 0 0 0 58.9668 0 0 0 66.3477
```

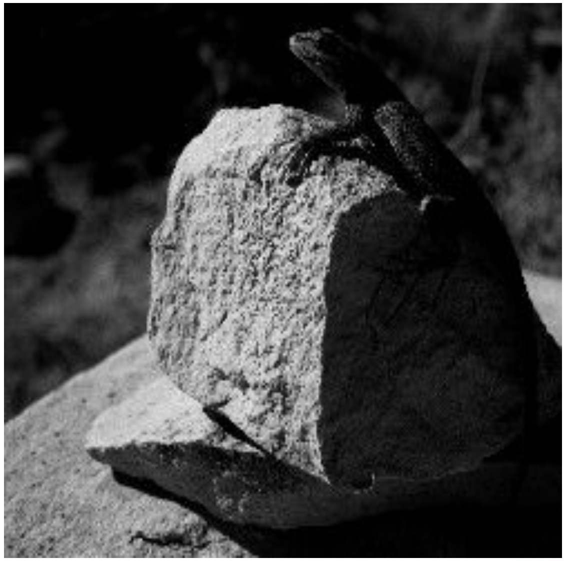
Lesson 4: Contrast manipulation, Equalization and LUTs

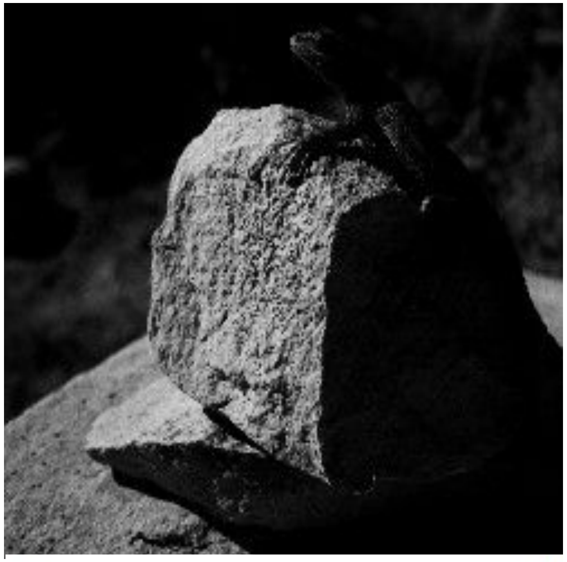
- LUTs. Contrast manipulation. Usual transformations:

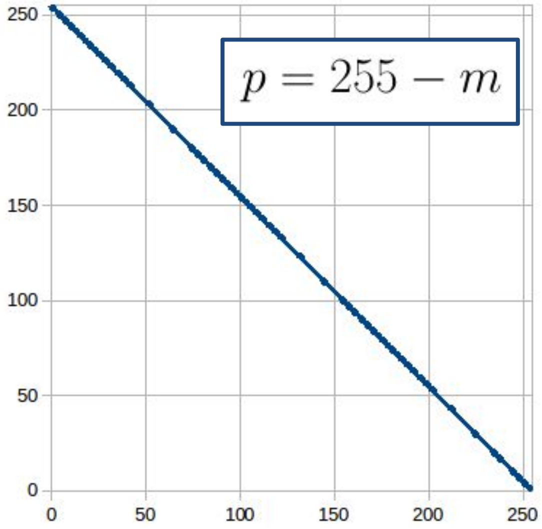
Inverse function	$p = 255 - m$
Quadratic function	$p = \frac{m^2}{255}$
Cubic function	$p = \frac{m^3}{255^2}$
Square root function	$p = \sqrt{255m}$
Cubic root function	$p = \sqrt[3]{255^2 m}$
Logaritmic function	$p = 255\frac{\ln(1+m)}{\ln(1+255)}$

Lesson 4: Contrast manipulation, Equalization and LUTs

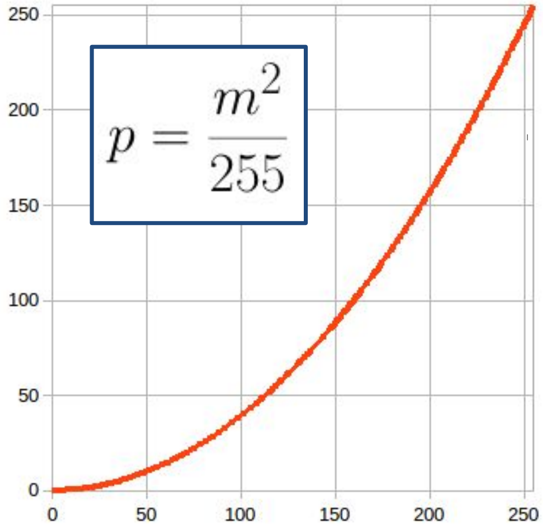




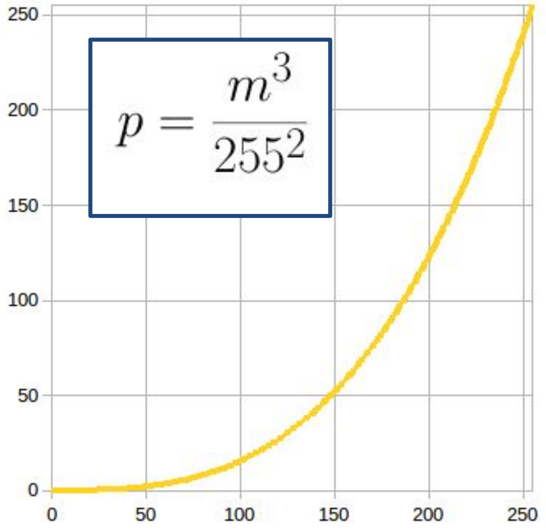




$$p = 255 - m$$

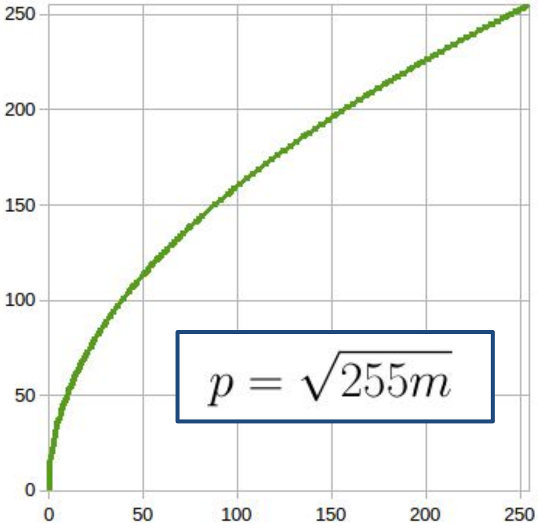


$$p = \frac{m^2}{255}$$

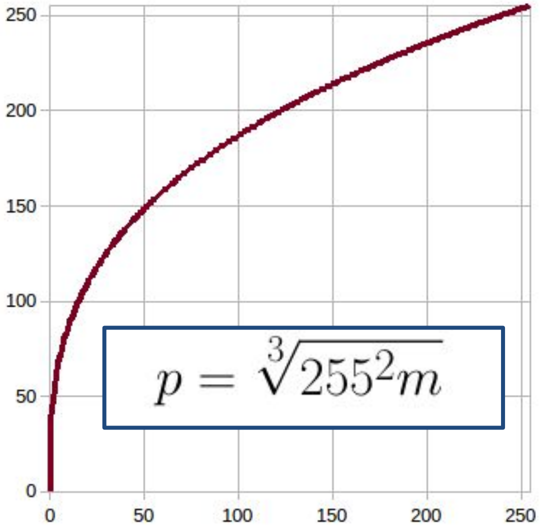


$$p = \frac{m^3}{255^2}$$

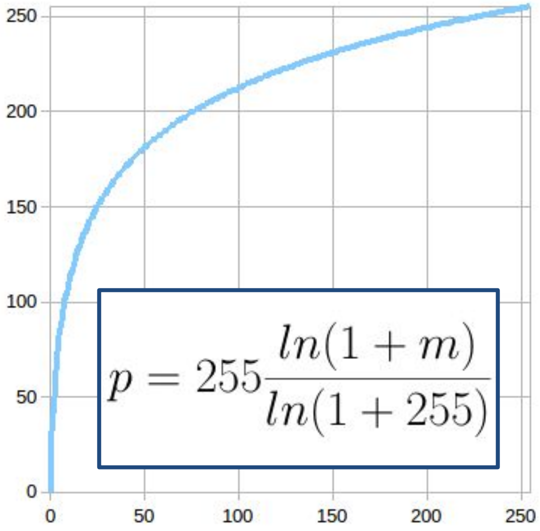
Lesson 4: Contrast manipulation, Equalization and LUTs



$$p = \sqrt{255m}$$



$$p = \sqrt[3]{255^2 m}$$



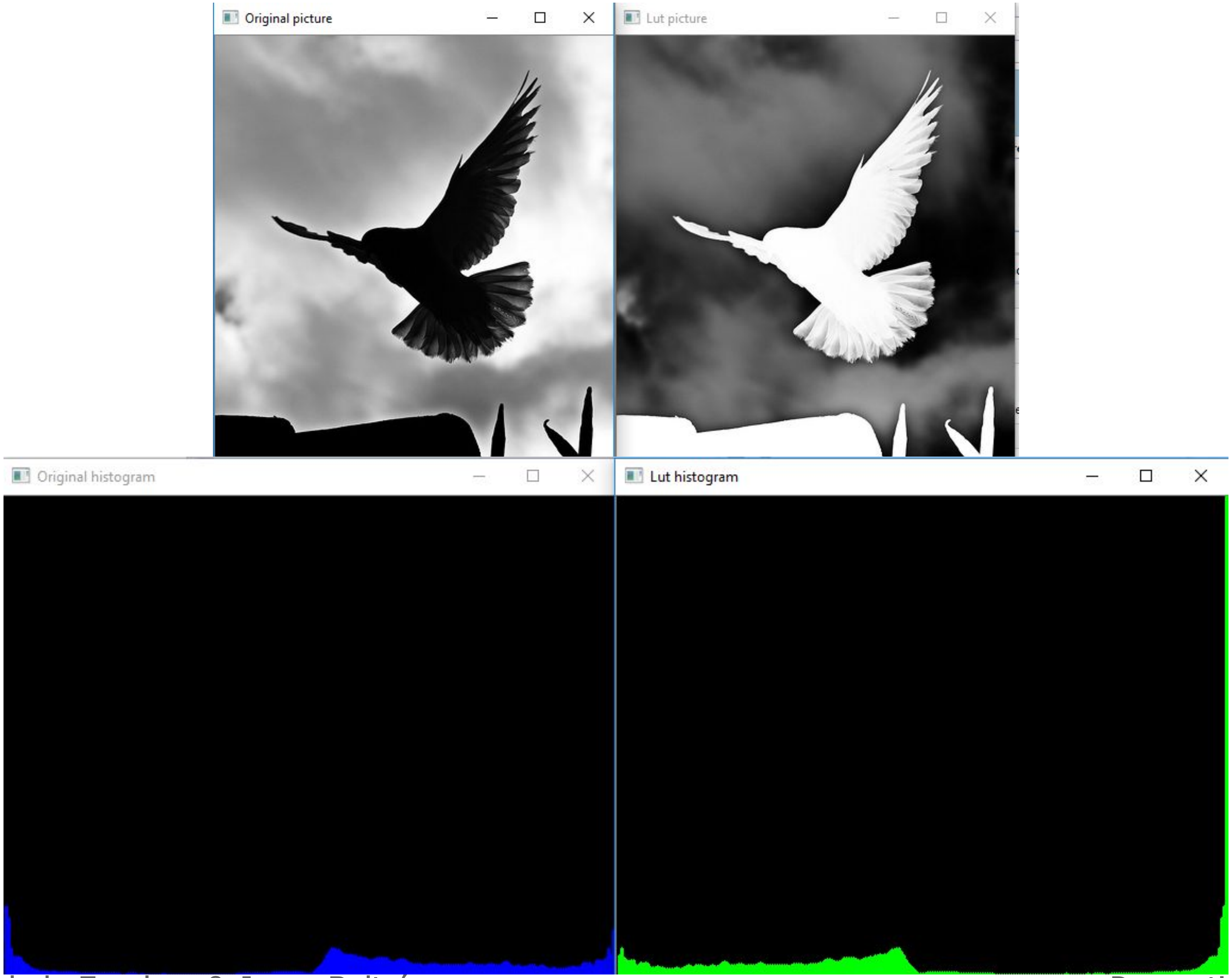
$$p = 255\frac{ln(1+m)}{ln(1+255)}$$

Exercise 3. Modifying the contrast, usual transformations

- Load a grayscale image
- Compute the histogram of the image
- Print the values of the histogram
- Define and apply all the common LUT functions
- Compute the histogram of the modified images
- Print the values of the new histograms
- Plot the original and the new histograms
- Plot both the original and the modified images
- Free memory

Lesson 4: Contrast manipulation, Equalization and LUTs

Exercise 3. Modifying the contrast, usual transformations



Lesson 4: Contrast Manipulation, Equalization and LUT

Jorge Beltrán de la Cita, Arturo de la Escalera

Perception Systems

Course 2019-2020