Computer Vision

Arithmetic and Logic Operations
Histograms

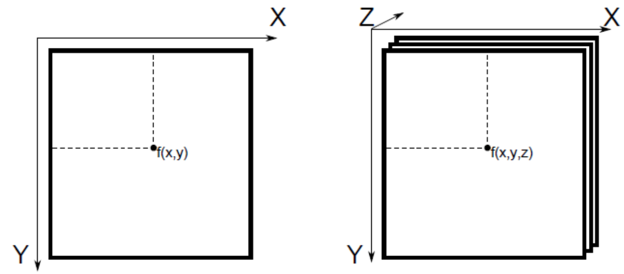
Fátima Leal

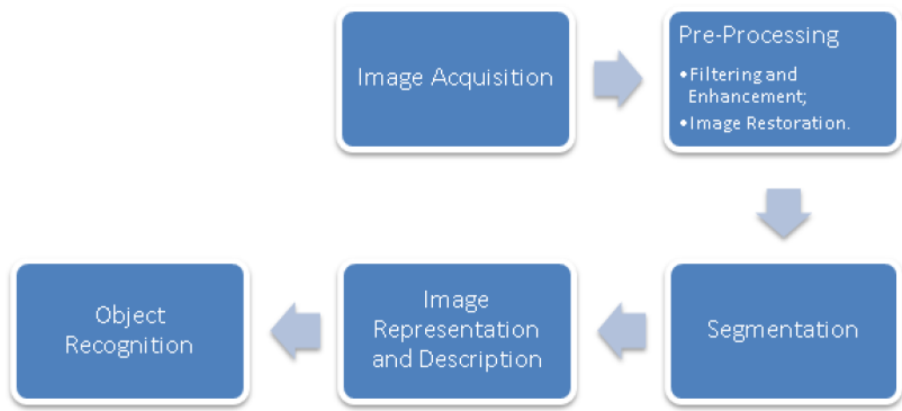
DEPARTAMENTO CIÊNCIA E TECNOLOGIA

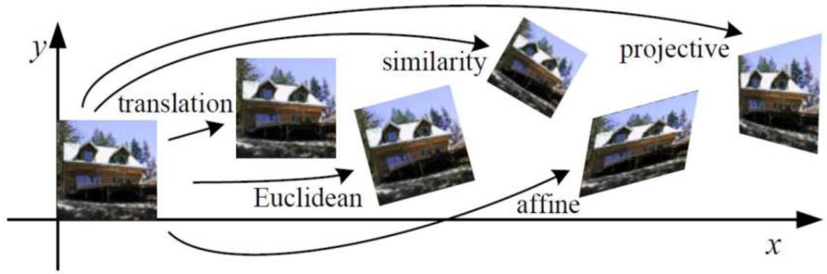
UNIVERSIDADE PORTUCALENSE

What we have learnt

- Fundamentals image concepts
- Machine vision systems
- Geometric operations







DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Content

- Arithmetic operations
- Logic operations
- Histograms

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Arithmetic Operations

- We all know basic **arithmetic operations** like **addition** and **subtraction**.

- The idea is to apply a simple function:

$$y = f(x)$$

To each image values:

$$y = x \pm C$$

$$y = xC$$

Where C is a constant.

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Arithmetic Operations

- Note that, with **arithmetic operations** we need to keep in mind the limits of our colour space and data type

- **RGB images** have pixels that fall within the **range [0, 255]**

- What happens if we are examining a pixel with **intensity 250** and we try to **add 10** to it?

- Under **normal arithmetic** rules, we would end up with a **value of 260**.

- However, since **RGB images** are represented **as 8-bit unsigned integers**, 260 is **not a valid** value.

Arithmetic Operations

- So, what should happen? There isn't no correct way to handle image additions and subtractions that fall outside the range of [0, 255].

- It simply depends on how you are manipulating your pixels and your goals.

- The most common procedures are:
  - **Normalization**: store the result temporarily in a bigger type variable and recalculate the result using equation:

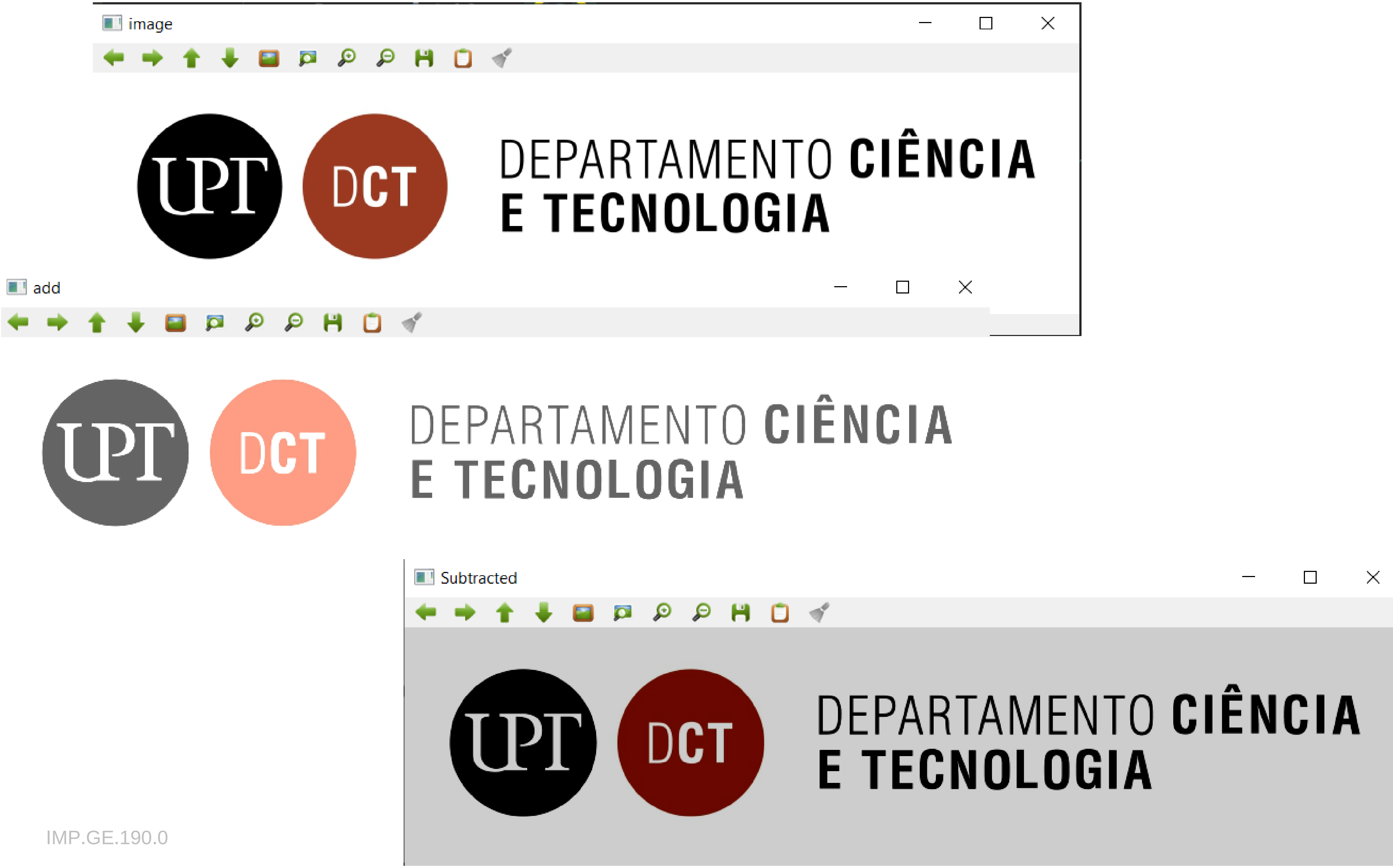
$$g = \frac{Lmax}{fmax - fmin}(\mathrm{f} - \mathrm{fmin})$$

  - **Truncation**: cut off everything that goes outside the range of used type.

Arithmetic Operations

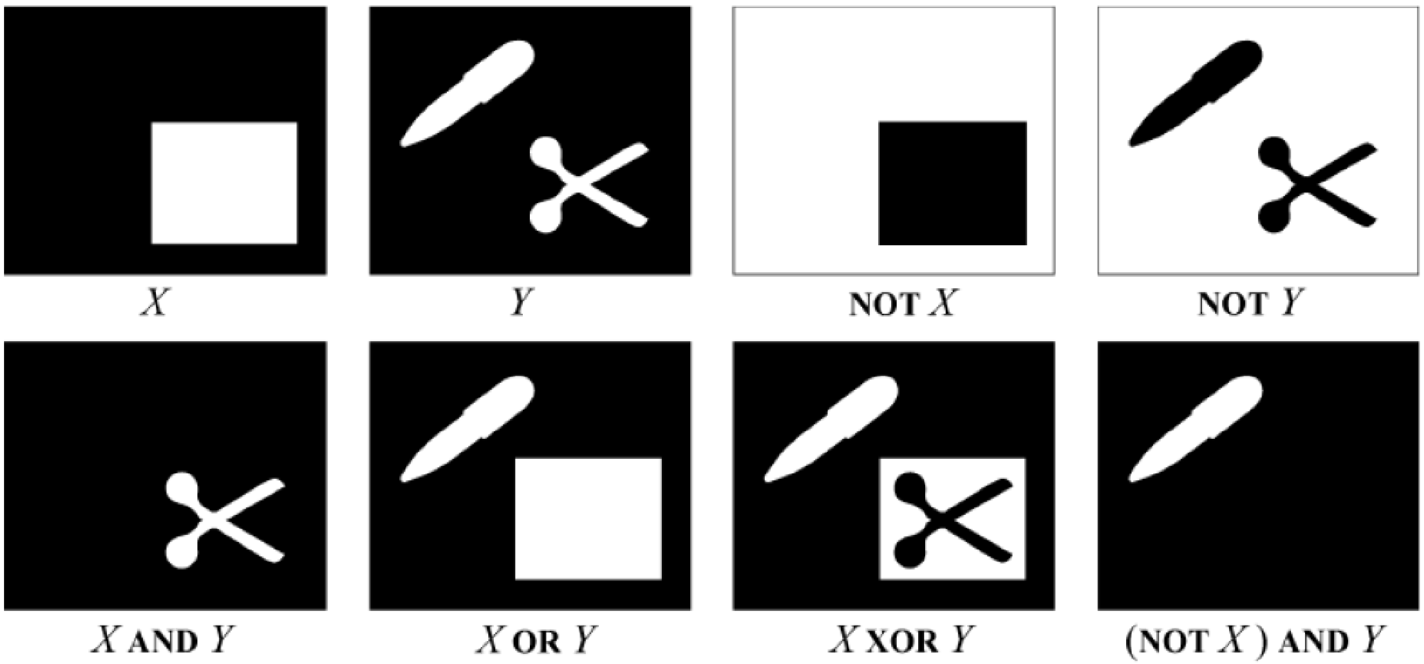
- Take note that there is a difference between OpenCV and NumPy addition
  - OpenCV ensures pixel values never fall outside the range [0, 255]
  - NumPy will perform a modular arithmetic and wrap around
  - *e.g.*: 200 + 100 =
    - **OpenCV** – max: 255 and min 0
    - **Numpy** - 44

Arithmetic Operations



Logic Operations

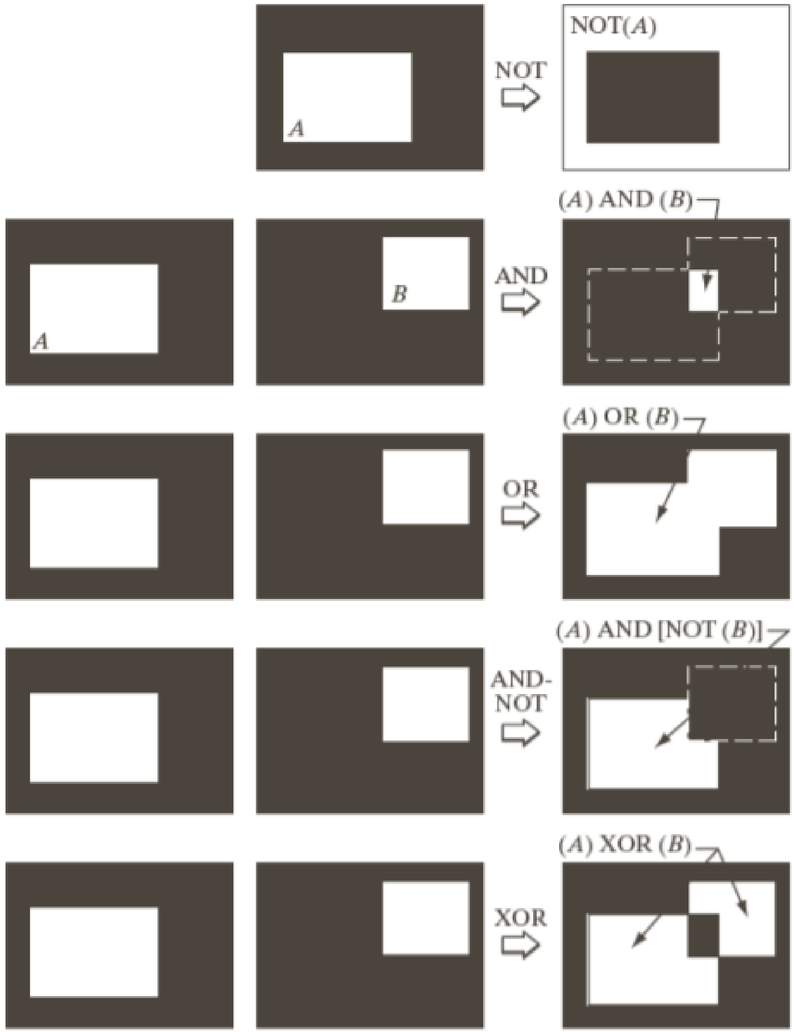
- Logic operations operate in a binary manner and are represented as grayscale images
- A given pixel is turned “off” if it has a value of zero, and it is turned “on” if the pixel has a value greater than zero



Logic Operations

- **AND**: A bitwise AND is true if and only if both pixels are greater than zero
- **OR**: A bitwise OR is true if either of the two pixels are greater than zero
- **XOR**: A bitwise XOR is true if and only if either of the two pixels are greater than zero, but not both
- **NOT**: A bitwise NOT inverts the “on” and “off” pixels in an image.

Logic Operations

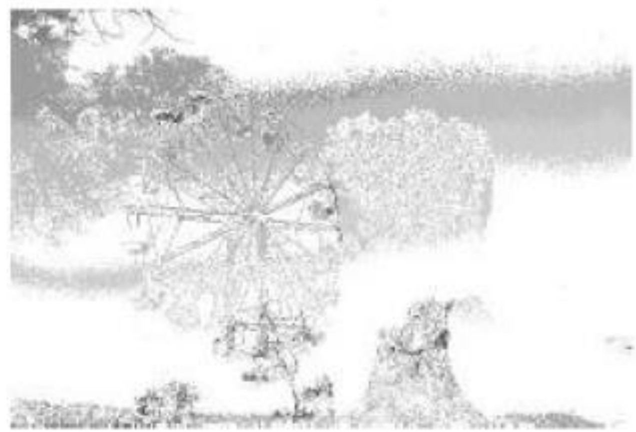


Logic Operations: AND

A bitwise AND is true if and only if both pixels are greater than zero

Logic Operations: OR

A bitwise OR is true if either of the two pixels are greater than zero

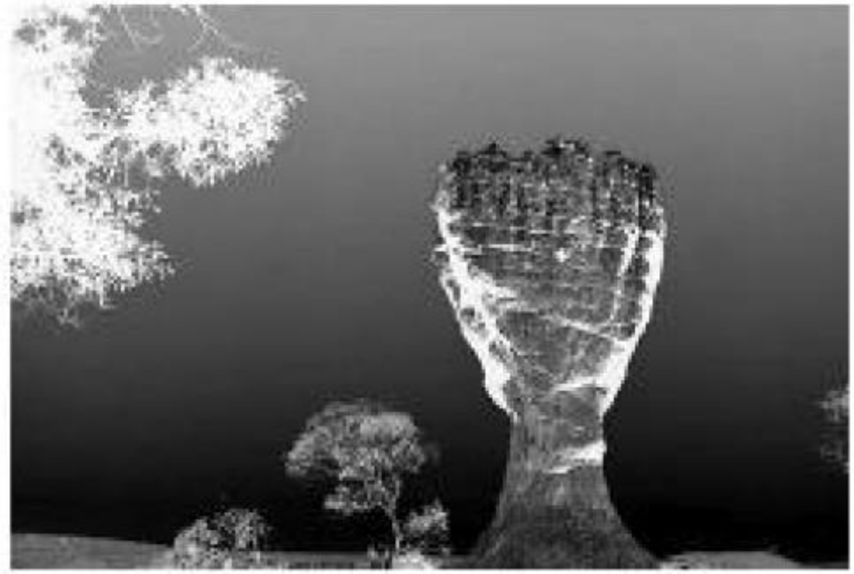


DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Logic Operations: XOR

A bitwise XOR is true if and only if either of the two pixels are greater than zero, but not both

Logic Operations: NOT



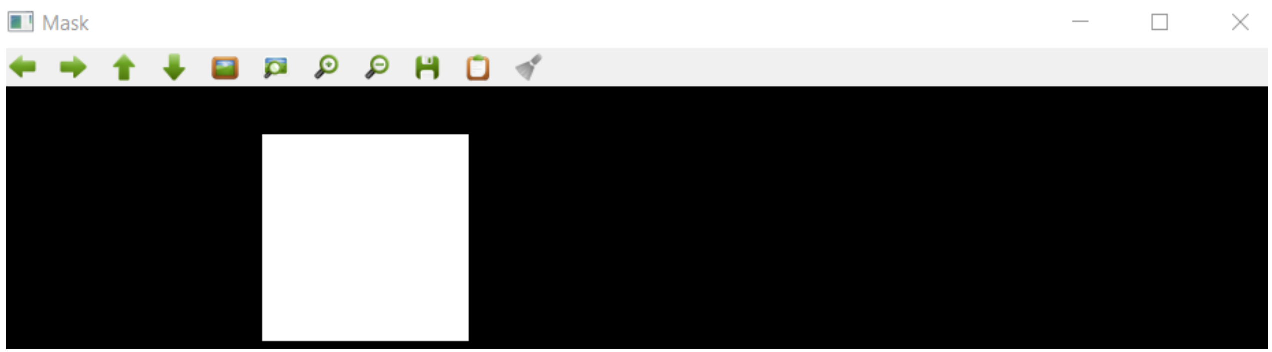
A bitwise NOT inverts the “on” and “off” pixels in an image.

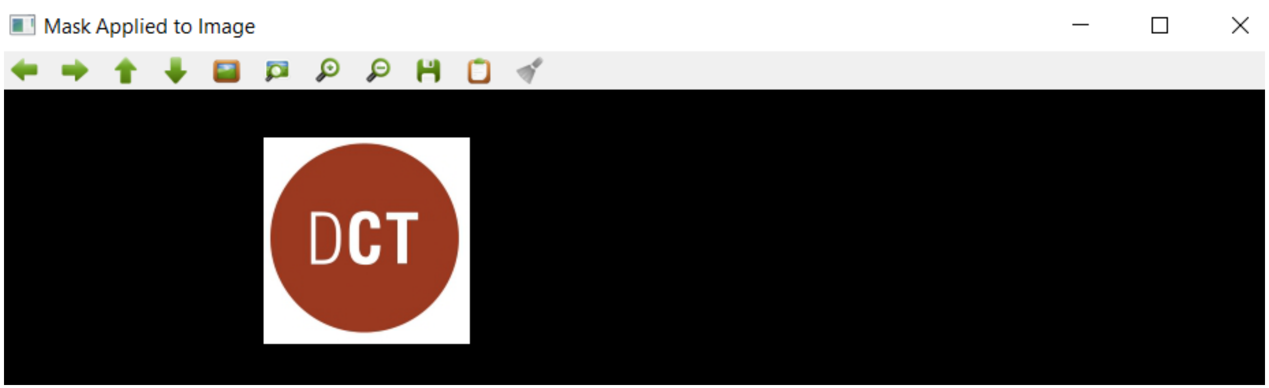
Masking

- **Masking** is an extremely powerful and useful technique in computer vision and image processing
- It allows to **focus only on the portions** of the image that **interests** to us
- Imagine we were building a computer vision system to recognize faces. The only parts of the image we want are those which contain faces
- The remaining content is discarded.

Masking

using the `cv2.bitwise_and` function





Splitting and Merging Channels

- A colour image consists of **multiple channels**: RGB
- We can **access** those components **via NumPy** arrays
- Is it possible to **split an image** into its respective components?

YES!

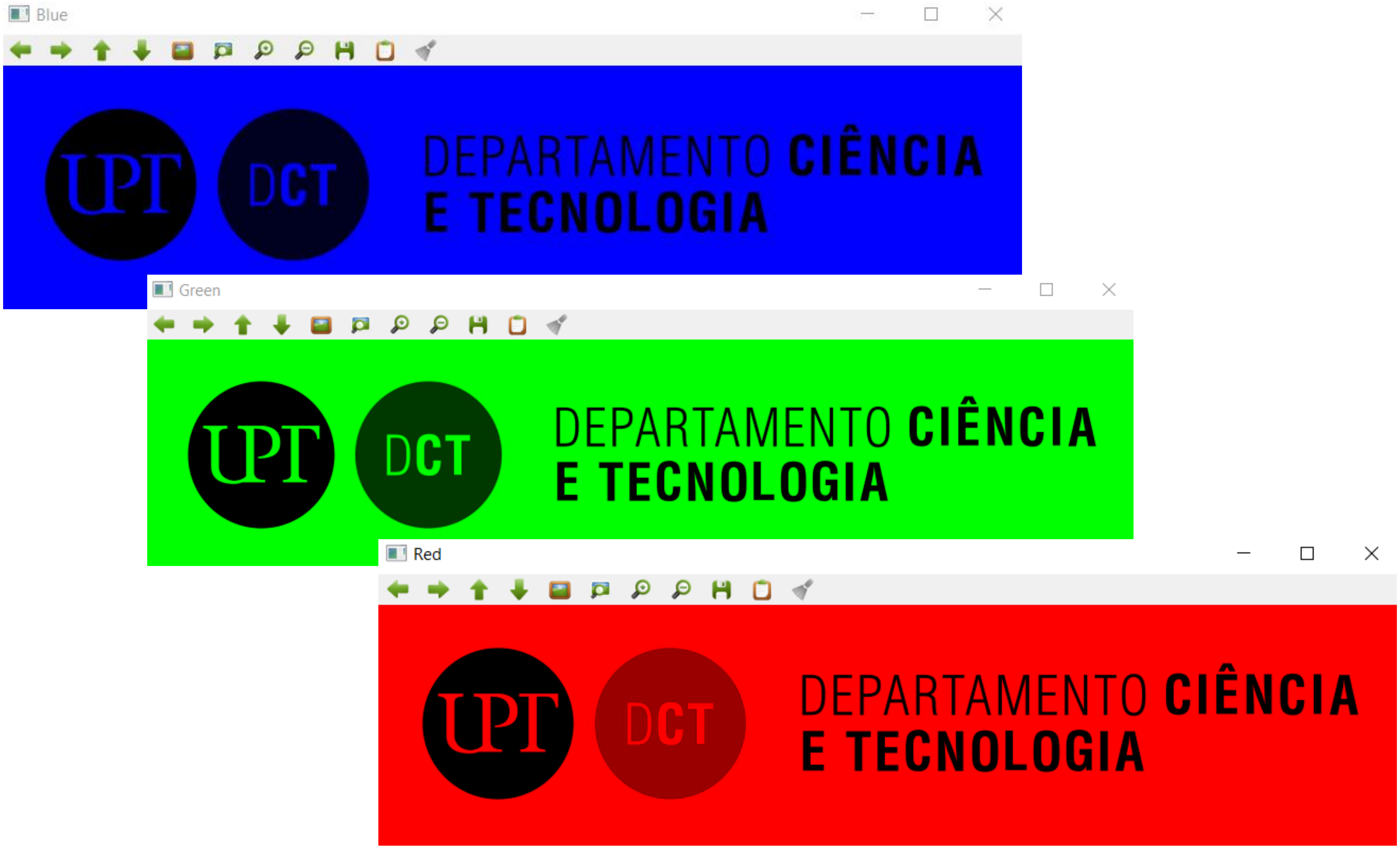
DEPARTAMENTO CIÊNCIA E TECNOLOGIA

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Splitting


Blue
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA
(x=438, y=4) ~ L:255
Green
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA
Red
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA

Merging

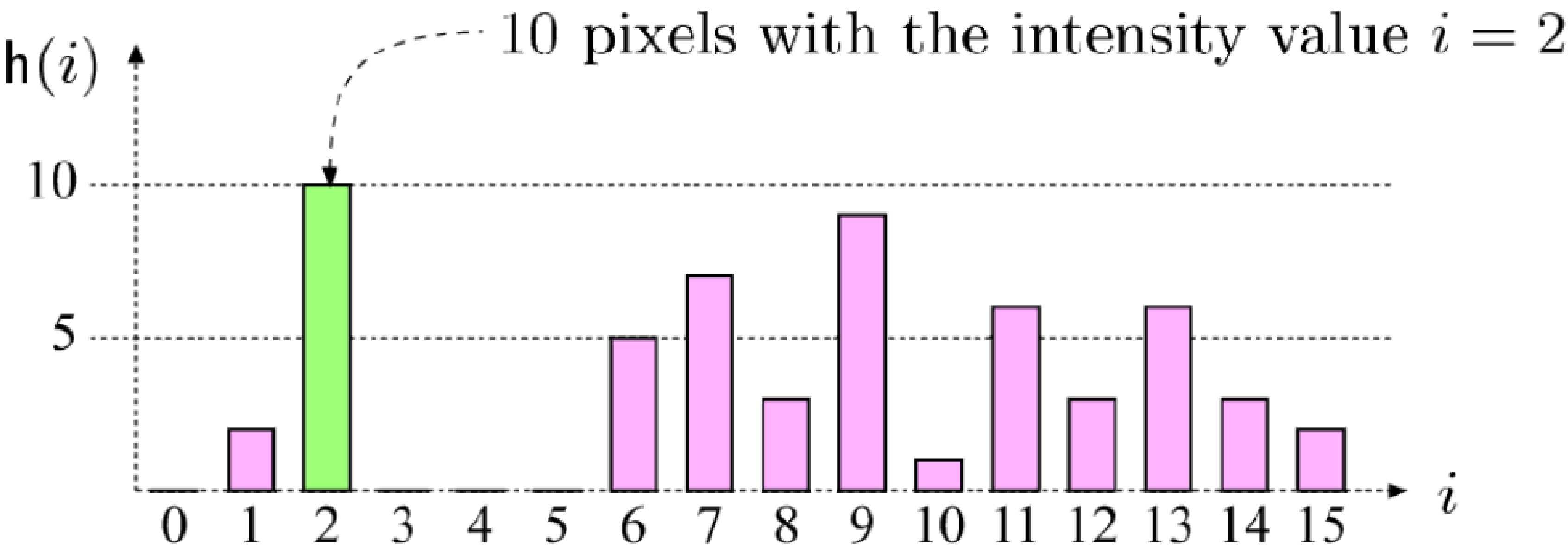

Blue
UPT
DCT
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA
Green
UPT
DCT
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA
Red
UPT
DCT
DEPARTAMENTO CIÊNCIA
E TECNOLOGIA

Colour spaces

Histograms

- **Histograms** have become a popular tool for **image statistics**
- It helps to determine certain problems in an image
- A histogram represents the **distribution of pixel intensities**
- In histograms, the X-axis serves as our “bins”
- If we construct a histogram with 256 bins, then we are effectively counting the number of times each pixel value occurs

Histograms



h(i)	0	2	10	0	0	0	5	7	3	9	1	6	3	6	3	2
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Histograms

- A **histogram is not enough** to draw **qualitative conclusions** about the overall quality of the image (presence or absence of noise, etc.)

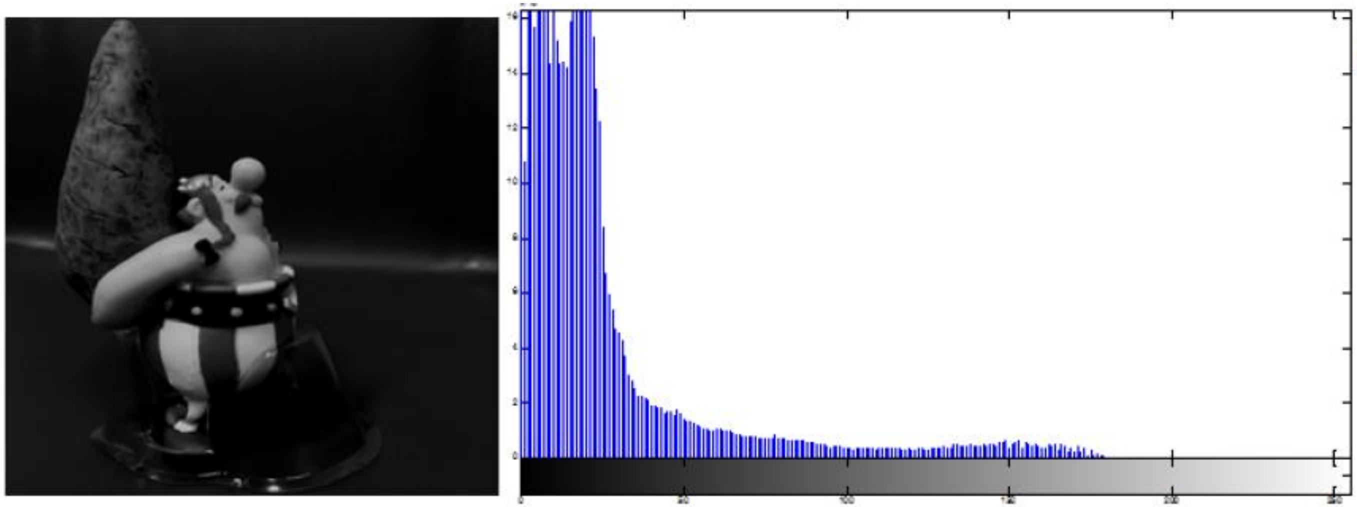
- Nevertheless, it **carries significant qualitative and quantitative** information about the corresponding image (*e.g.*, minimum, average, and maximum grey level values, dominance of bright or dark pixels, etc.).

- By simply **examining the image histogram**, we have a general understanding regarding the **contrast**, **brightness**, and **intensity distribution**

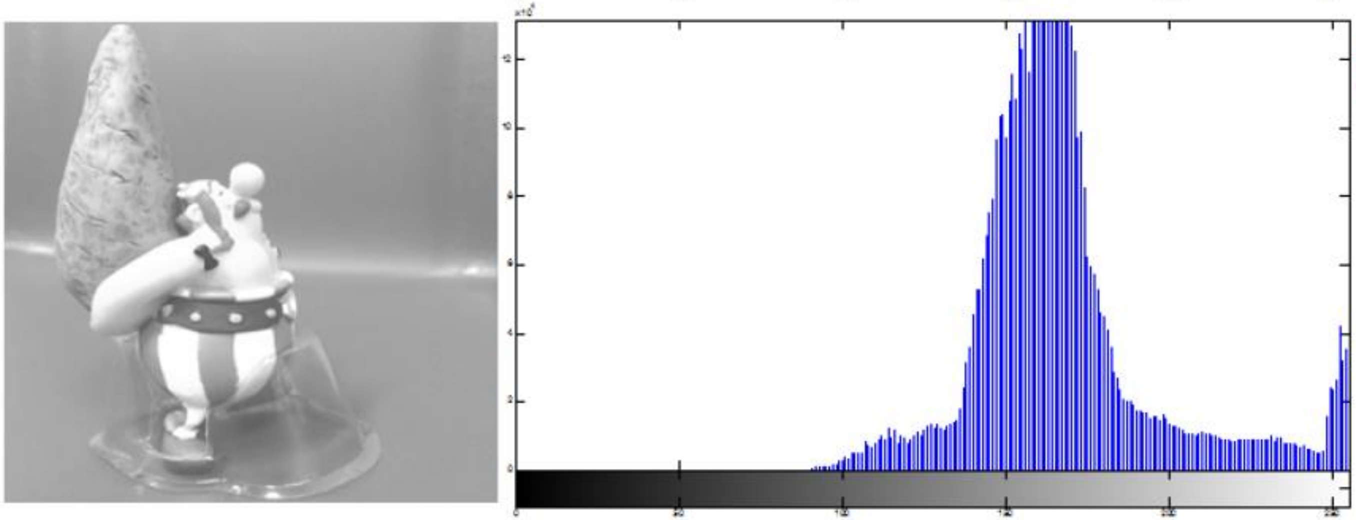
- The **left side** of the histogram corresponds to **lower pixel values.** If the frequency at lower pixel values is very high, it indicates darkness

- The **right side** of the histogram corresponds to **higher pixel values**. If the frequency at higher pixel values is very high, it indicates saturation.

Histograms

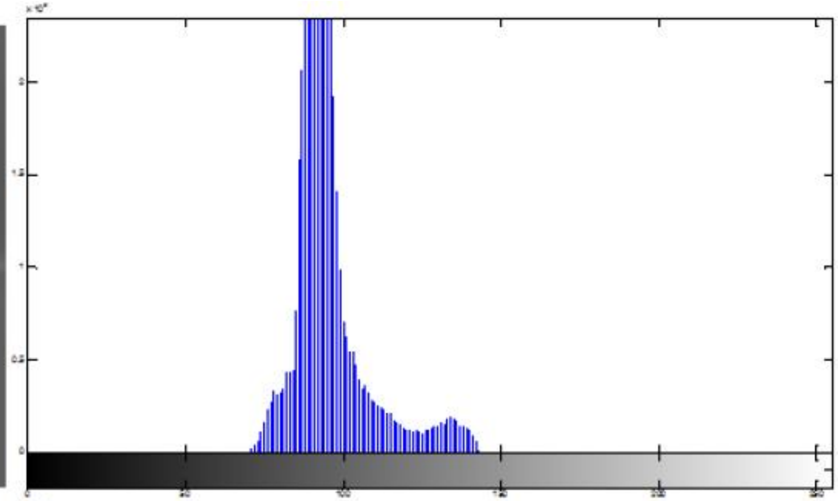


Dark image

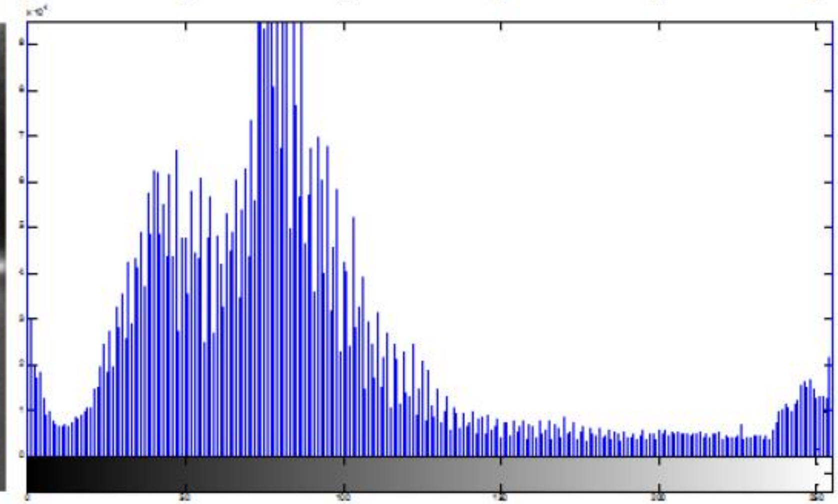


Bright image

Histograms



Low contrast image

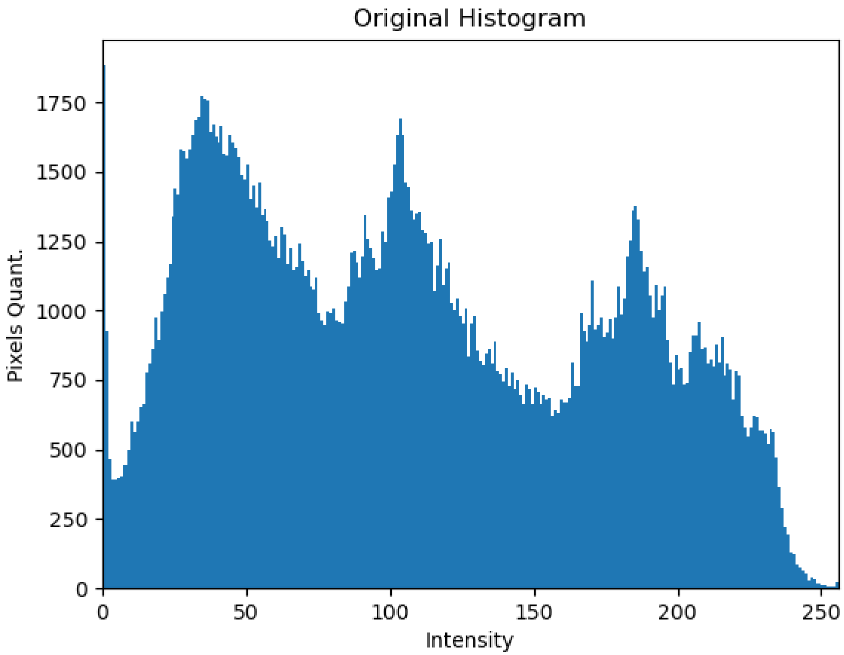


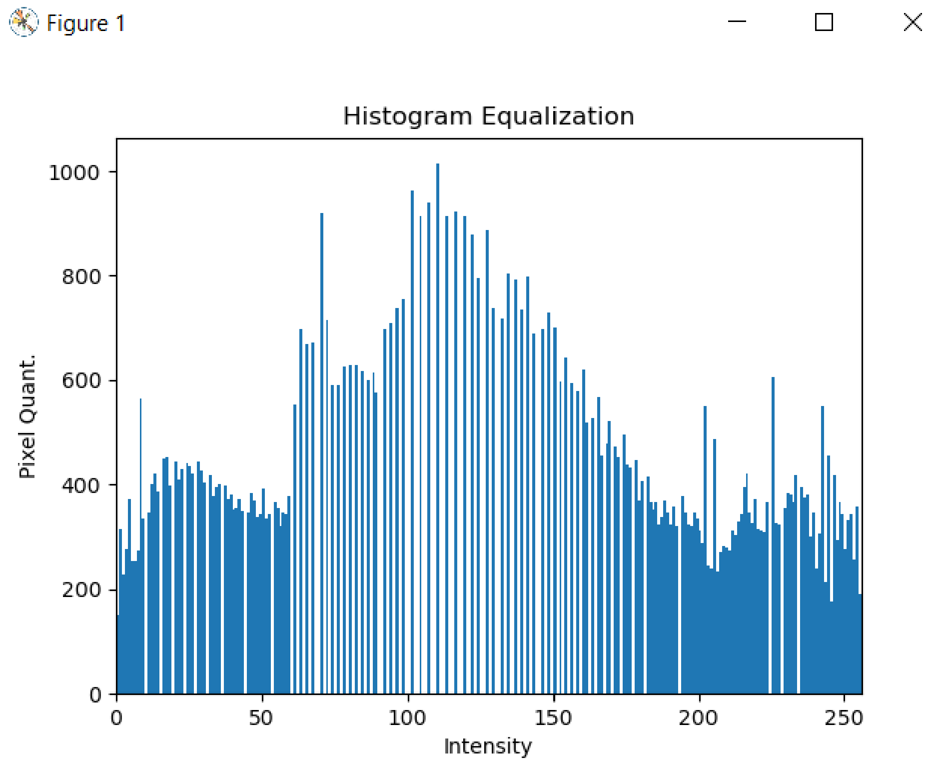
High contrast image

Histograms Equalization

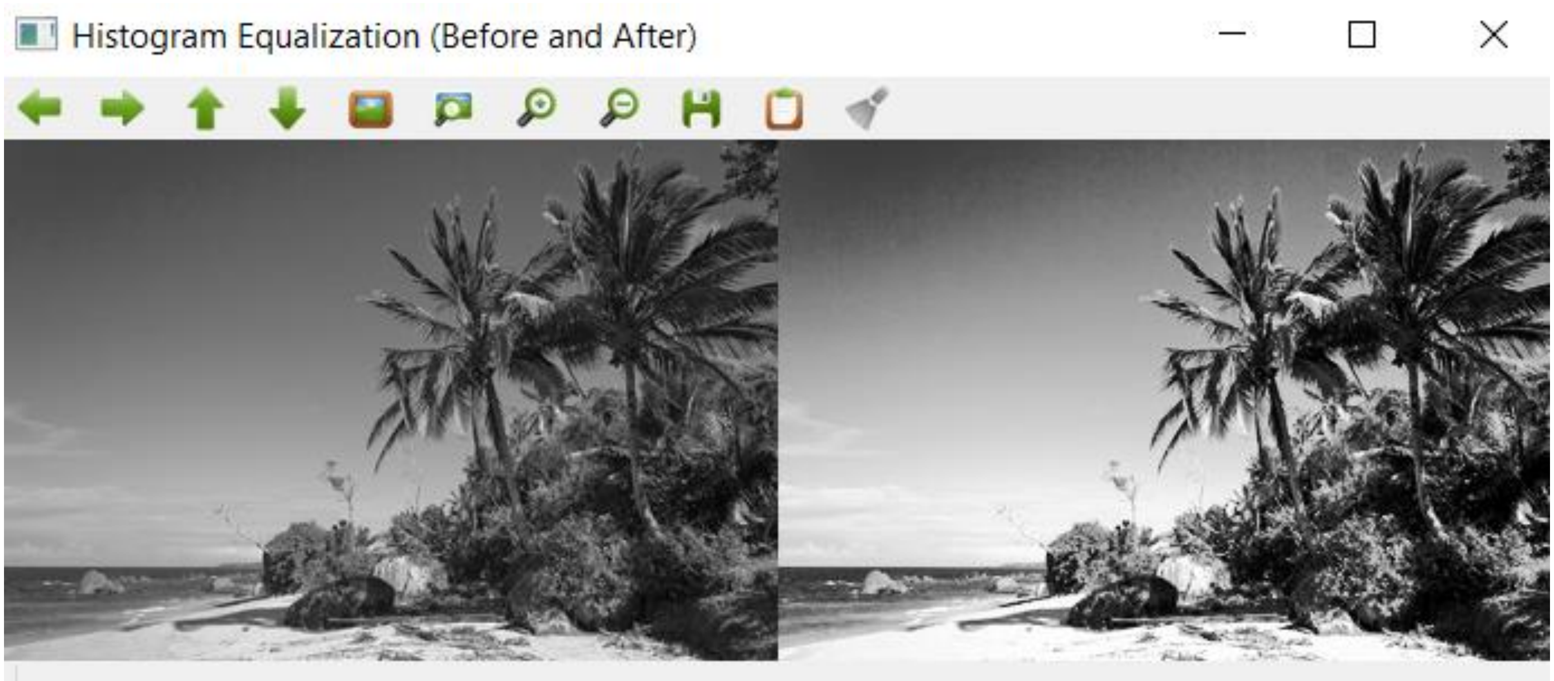
- The goal is to **improve the contrast** of an image by rescaling the histogram
- Consider a histogram with a large peak in the centre. Applying histogram equalization, the **pixel intensity will be distributed through the image**
- Histogram equalization is applied to **grayscale images**
- This method is useful when an image contains **foregrounds and backgrounds** that are both **dark or light**
- It tends to produce **unrealistic effects** in photographs; however, it is normally useful when **enhancing the contrast of medical or satellite** images.

Histograms Equalization

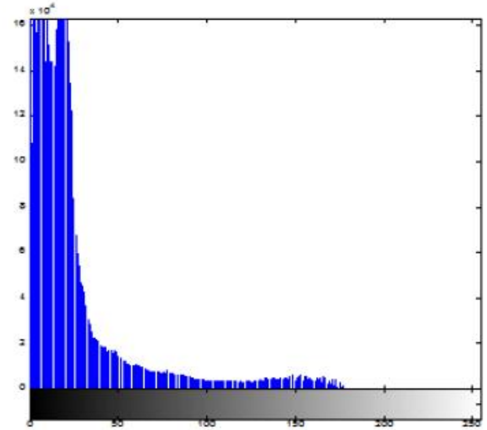


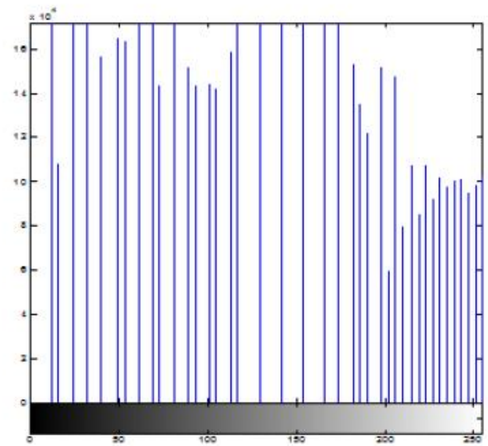


Histograms Equalization



Histograms Equalization





Histograms Equalization

- The histogram equalization is very useful to recognize objects.

- It uses the following algorithm:

  - Compute the histogram
  - Normalize the histogram to ensure that all values are between 0 and 255
  - Cumulative distribution function
  - Transform the image

  - The contrast enhancement is ensured

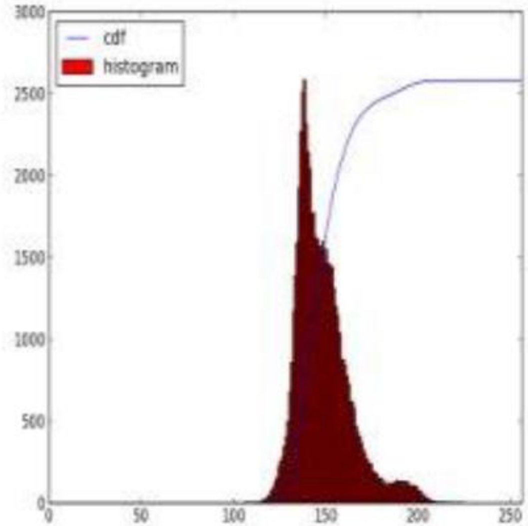
Histograms Equalization

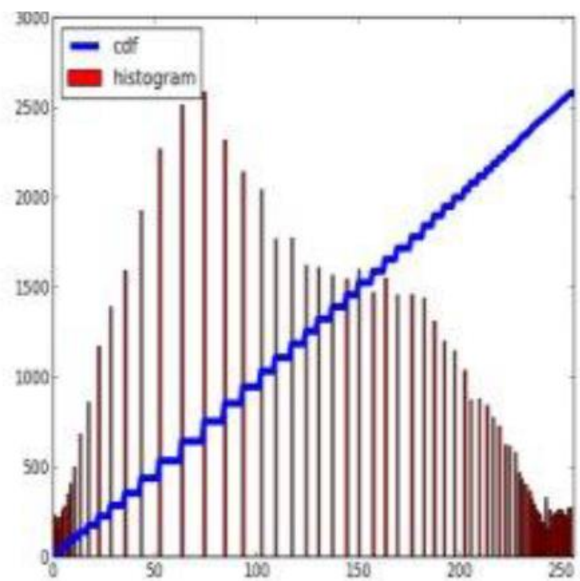
Grey level i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n_i	15	0	0	0	0	0	0	0	0	70	110	45	80	40	0	0

Grey level i	n_i	Σn_i	$(1/24)\Sigma n_i$	Rounded value
0	15	15	0.63	1
1	0	15	0.63	1
2	0	15	0.63	1
3	0	15	0.63	1
4	0	15	0.63	1
5	0	15	0.63	1
6	0	15	0.63	1
7	0	15	0.63	1
8	0	15	0.63	1
9	70	85	3.65	4
10	110	195	8.13	8
11	45	240	10	10
12	80	320	13.33	13
13	40	360	15	15

Original grey level i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Final grey level j	1	1	1	1	1	1	1	1	1	4	8	10	13	15	15	15

DEPARTAMENTO CIÊNCIA E TECNOLOGIA


cdf
histogram
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250


cdf
histogram
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250

Let’s play with images!

Examples

- Consider the matrix [250] and [10]. Add both matrixes using numpy and cv2. See the differences.
- Download images arithmetic1.jpg and arithmetic2.jpg from Moodle. Then, using the arithmetic operation add both images and see the result.
- Execute logic operations with both images: and, or, not. See the results
- Plot the histogram of arithmetic1.jpg and analyse the result

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

UNIVERSIDADE PORTUCALENSE

Do conhecimento à prática.