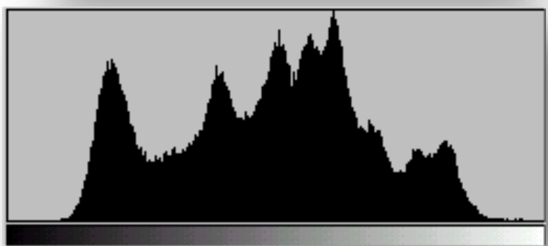
# Histogram

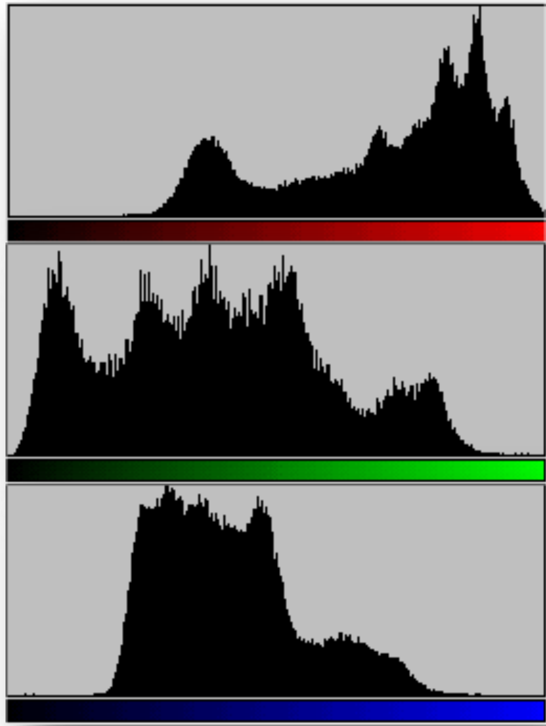
- How can I use the color information to characterize an image?
- The **histogram** is a simple way to describe the color distribution of a picture
- It can be seen as a probability density function
- It represents the occurrence of all colors on a graph
- It's a statistical representation of the pixel values
- For an MxN image

$$hist(p) = \frac{\#\, pixels: I(x,y) = p}{M \times N} \cong f(p)$$

# Histogram

- The equation holds for one component
- In case of more components, a histogram can be obtained from each of them



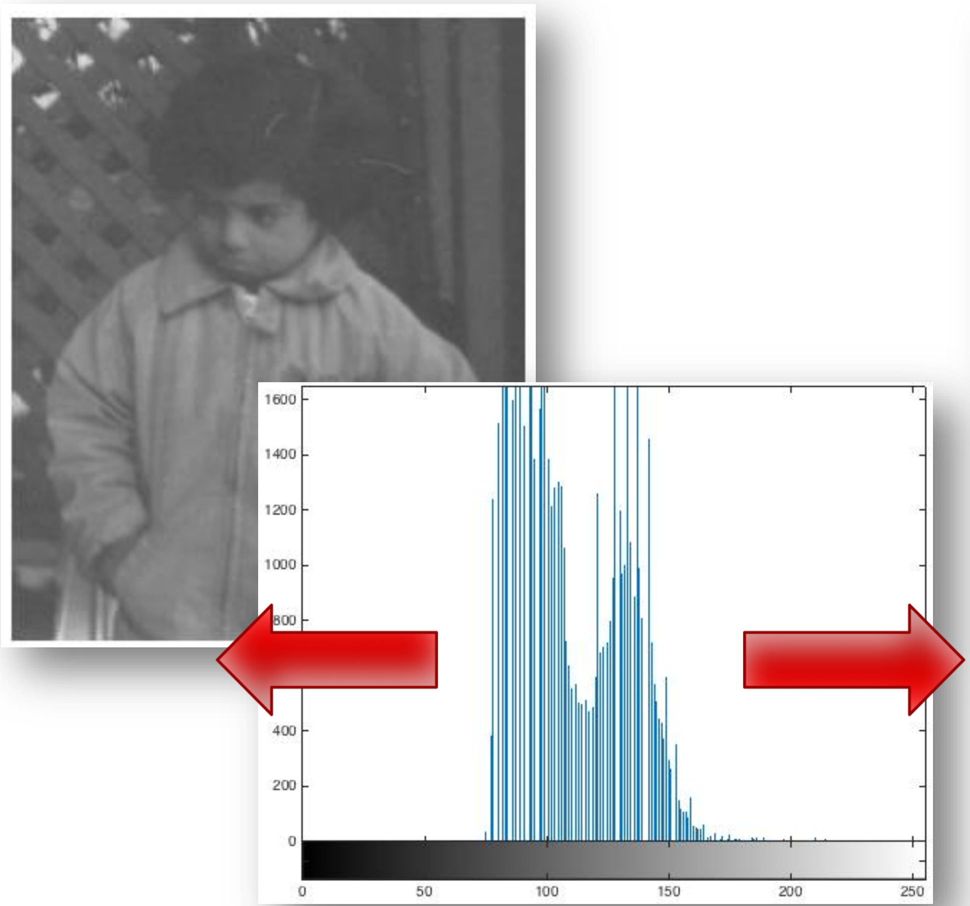


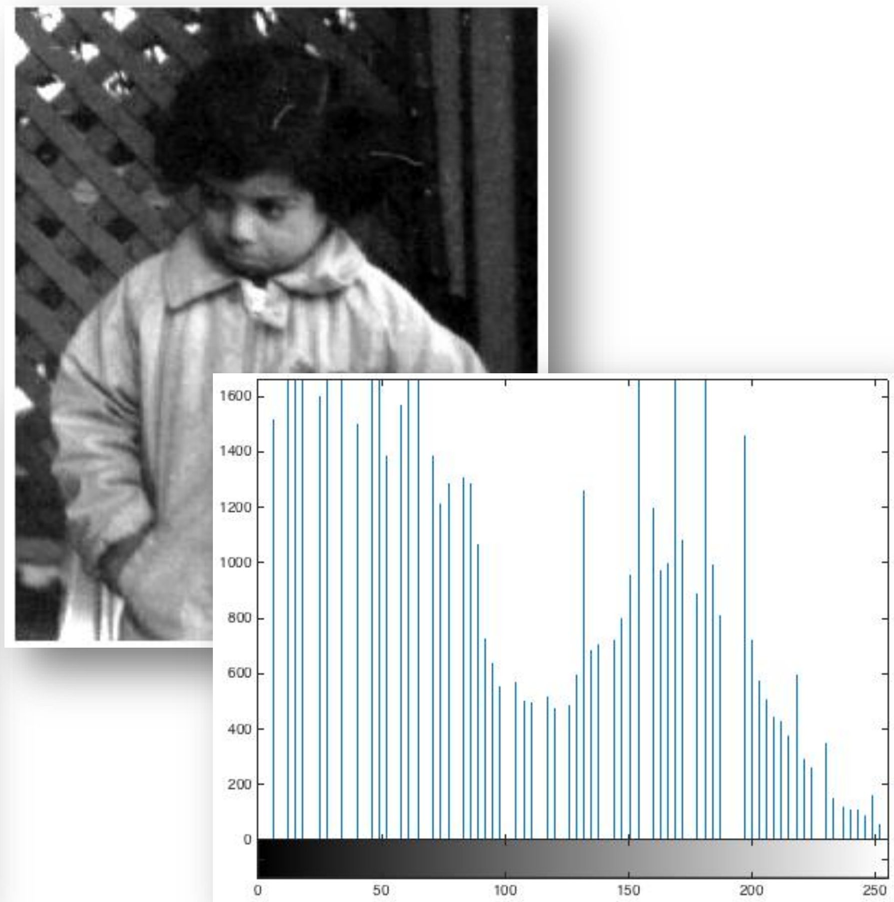
# Histogram

- What kind of information can we obtain from the histogram?
  - Is the image dark or bright?
  - Are colors distributed equally?
  - Are there dominant colors?
  - Are there colors missing?
- Applications
  - Is the illumination of a scene correct for environmental monitoring?
  - How has the background changed in the past 2 hours?
  - Can I use the histogram to determine whether the moving object I'm observing is A or B?
- In general it can be seen as a "signature" to be applied in different domains

# Simple operations: stretching

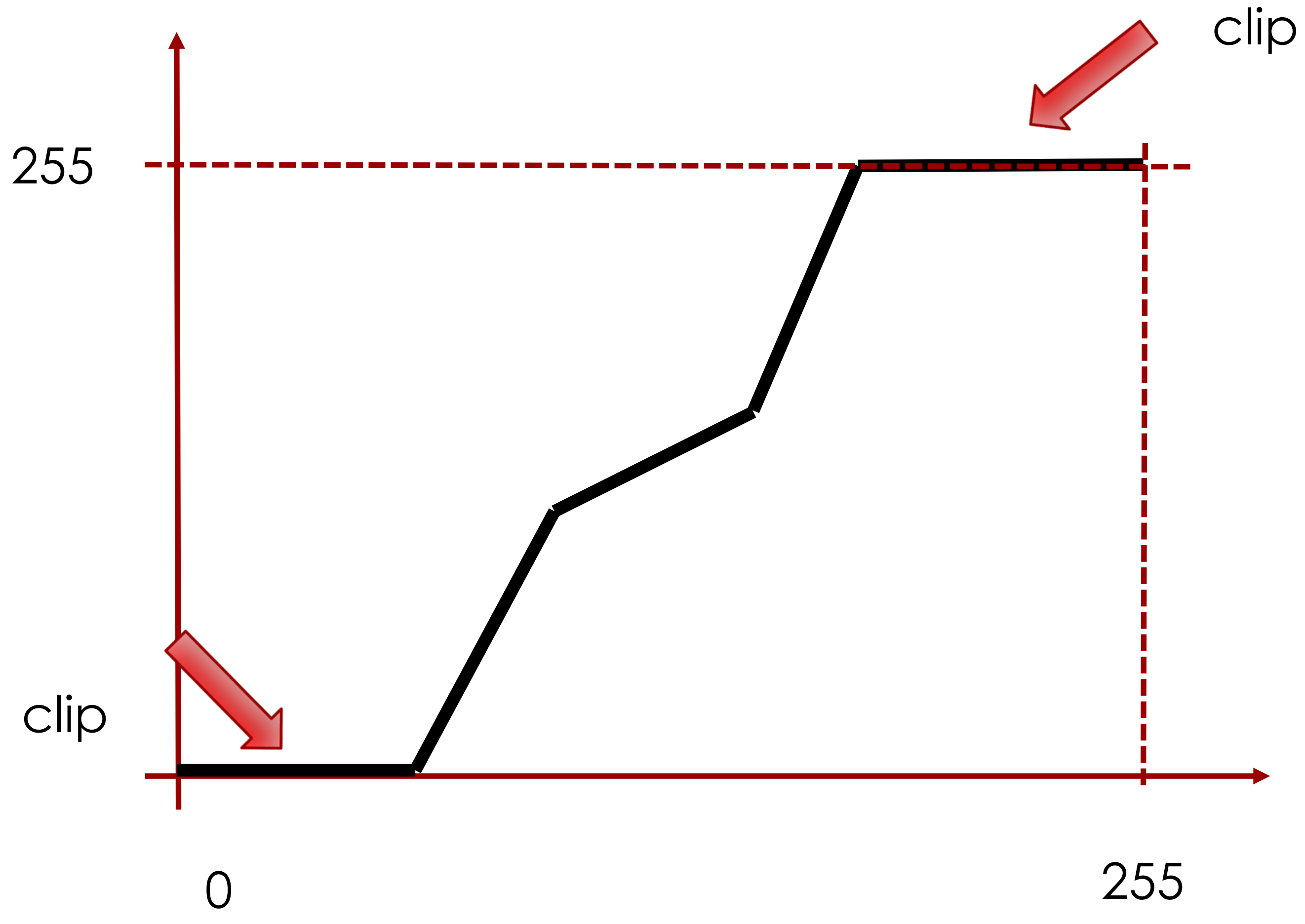
- Change the dynamic range of the image by stretching the histogram and increasing the contrast



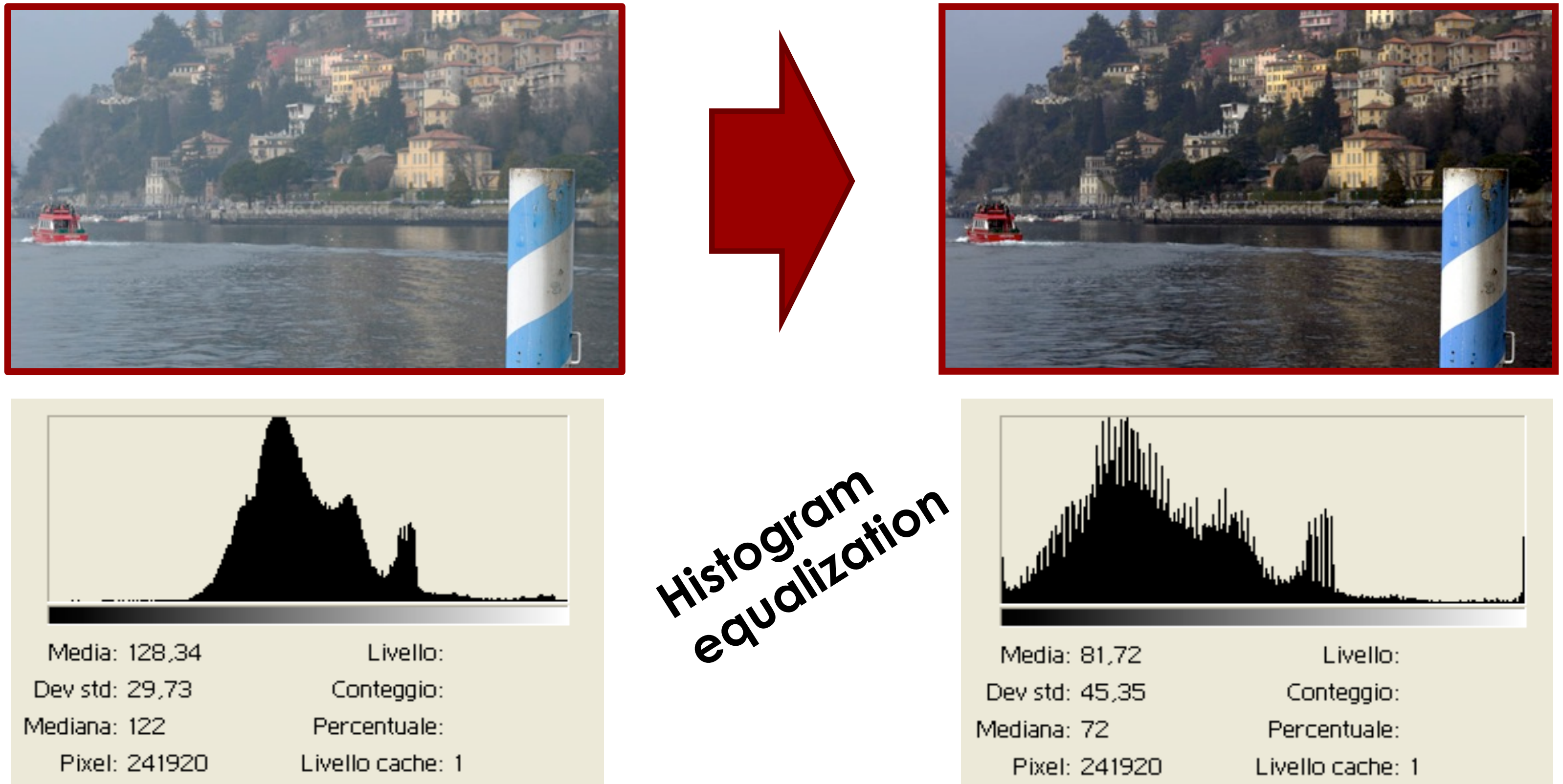
# Simple operations: stretching

- Usually apply a piecewise linear function



# Simple operations: equalization

- Working on the statistics of the pixels it is possible to improve the quality of an image

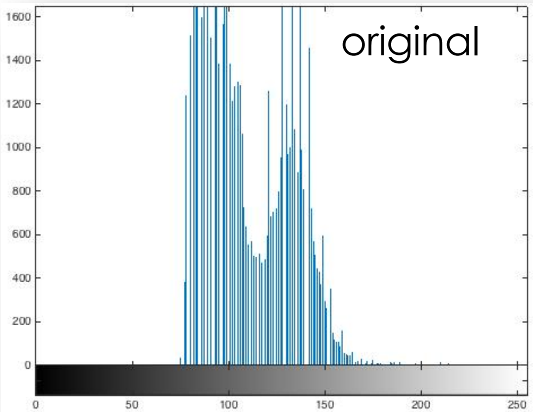


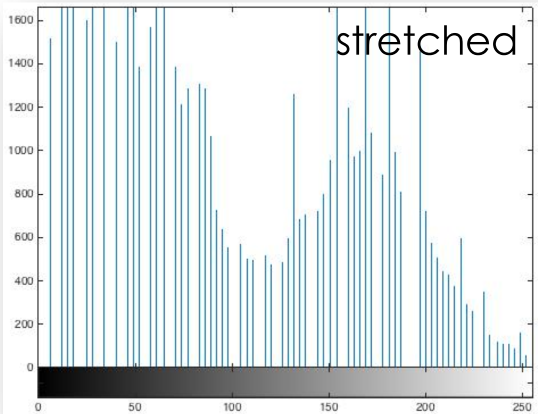
# Simple operations: equalization

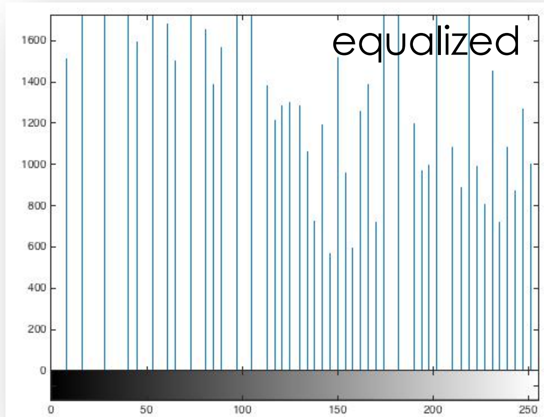
- **Ideally** we'd like to obtain a FLAT histogram

- To do so:
  - Compute the cumulative histogram (equivalent to CDF)

$$CHist_I(p) = \sum_{k=0}^{p} hist(k) \qquad hist_{eq}(p) = \frac{CHist(p) - CHist_{min}}{MxN - 1} \times 255$$



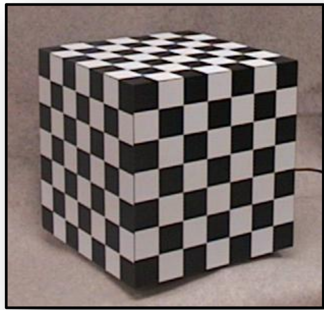
# Edge extraction

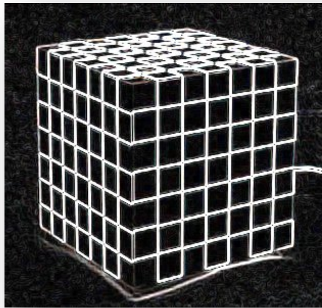
# Why and what

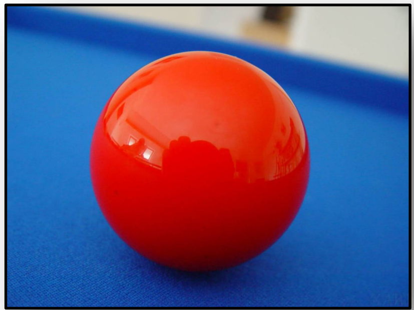
- We perceive objects through
  - Color (appearance model)
  - Shape
- Shape is the boundary between the object and the rest of the picture
- Shape helps us recognizing things
- What is an edge?
  - It can be seen as a sharp change in image brightness

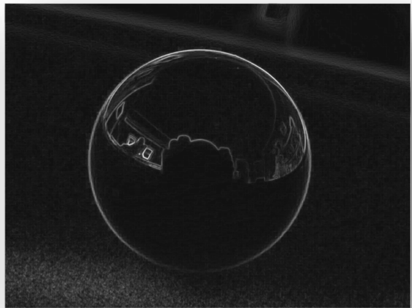
# Examples

- The checkerboard has strong contours, both outer and inner
- If the dice rotates, we can see that!
- The ball has a strong contour with respect to the background
- The ball is "flat" inside
  - It's difficult to say what happens inside
  - It may be spinning but we cannot say for sure…



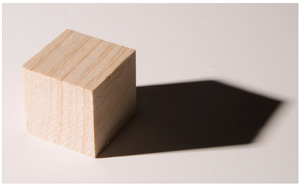


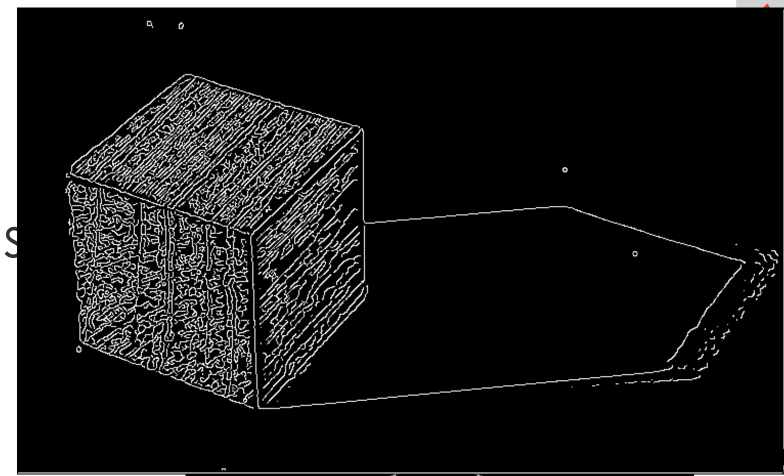




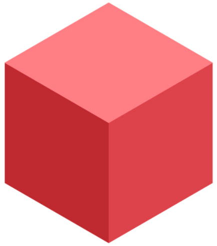
# Not only

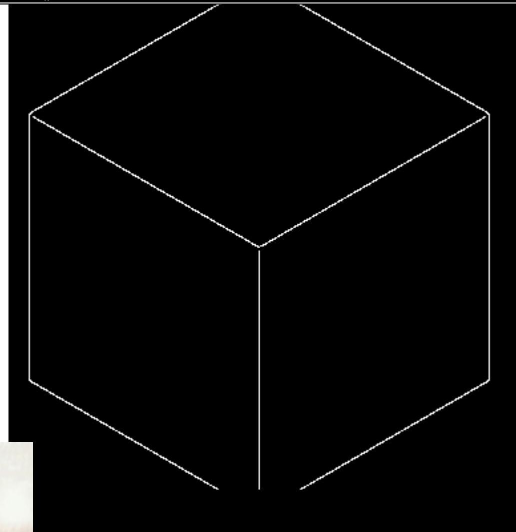
- Other environmental conditions
  - Cast shadows



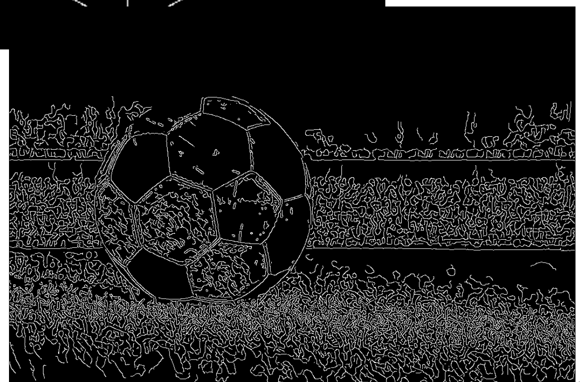


  - Surface orientations





- Edges are not always meaningful:
  - Noise
  - Textured areas



# Steps

- Determine intensity, and possibly the direction of an edge for each pixel location
  - Gradient
  - Laplacian
- Find a threshold and binarize
- The first step is the most difficult one
- Different tools are available, we'll focus here on the gradient-based algorithms

# Edge extraction by gradient analysis

- As for mono-dimensional signals, the goal is to find maxima and minima
- Compared to the 1D case, we also have a **direction**
- This means finding a gradient along a line *r* oriented in the direction $\theta$

$$\frac{\partial f}{\partial r} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial r} = f_x \cos\theta + f_y \sin\theta$$

# Edge extraction by gradient analysis

- The maximum with respect to θ is when

$$\left(\frac{\partial}{\partial\theta}\right)\left(\frac{\partial f}{\partial r}\right) = 0$$

- From which we obtain

$$-f_x \sin\theta_g + f_y \cos\theta_g = 0 \Rightarrow \theta_g = arctg\left(\frac{f_y}{f_x}\right)$$

$$\left(\frac{\partial f}{\partial r}\right)_{\max} = \sqrt{f_x^2 + f_y^2}$$

- The direction $\theta_g$ maximizes the gradient

# Operators

- The edge extraction process consists of computing the 1st order derivative in two orthogonal directions
  - $f_1$
  - $f_2$
- To each of them we associate an amplitude
- And an orientation

$$\vartheta_g(m,n) = arctg \frac{f_2(m,n)}{f_1(m,n)}$$

# Edge extraction in practice

- Typically FIR filters are adopted
- Choose a **convolution mask** and apply it to the picture

Roberts: $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Prewitt: $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

- The convolution is computed for both masks
- A threshold is chosen to highlight only the strongest edges

# The Sobel operator

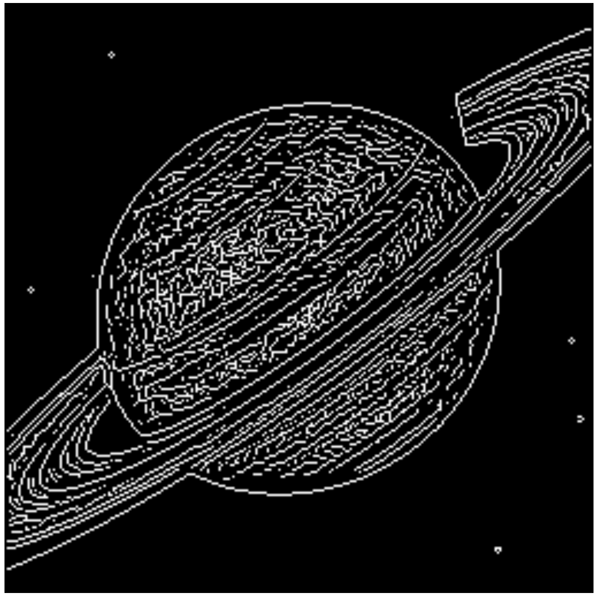
- Apply two masks, one for each orthogonal direction
- Compute the gradient for each point
- Threshold

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad D_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

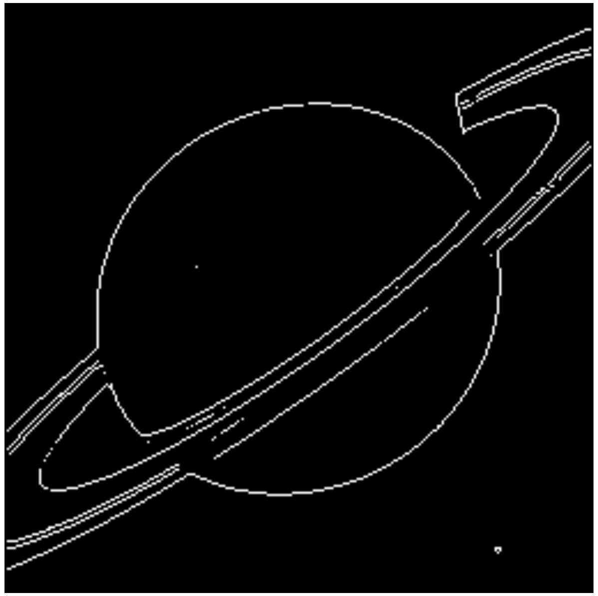
$$D = \sqrt{D_x^2 + D_y^2}$$

# Example

Original



After Sobel



After threshold

# How To

- The convolution with the FIR masks is performed similarly to the 1D convolution
  - Take the mask
  - Rotate
  - Slide from left to right
  - Associate to the central point the value of the convolution

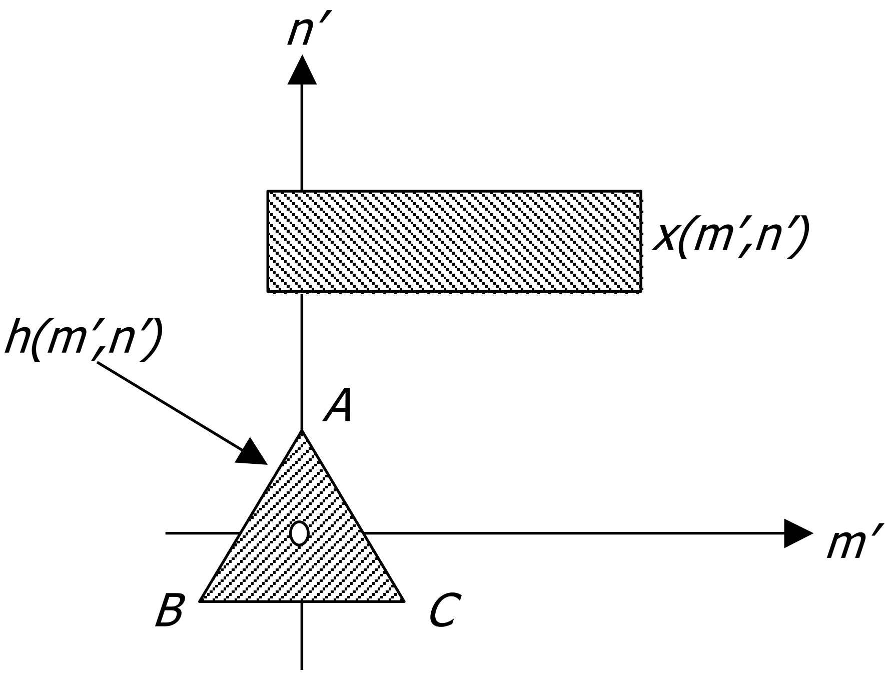
In space: $g(x,y) = f(x,y) * h(x,y)$

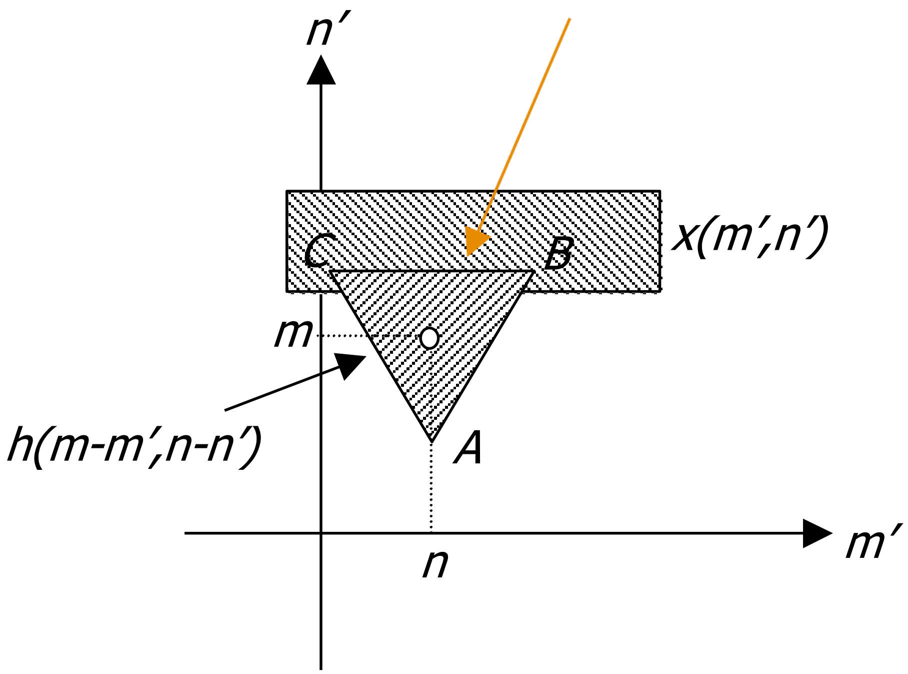
In frequency: $G(u,v) = F(u,v) \cdot H(u,v)$

$$y(m,n) = x(m,n) * h(m,n) = \sum_{m'=-\infty}^{\infty} \sum_{n'=-\infty}^{\infty} h(m-m',n-n') \cdot x(m',n')$$

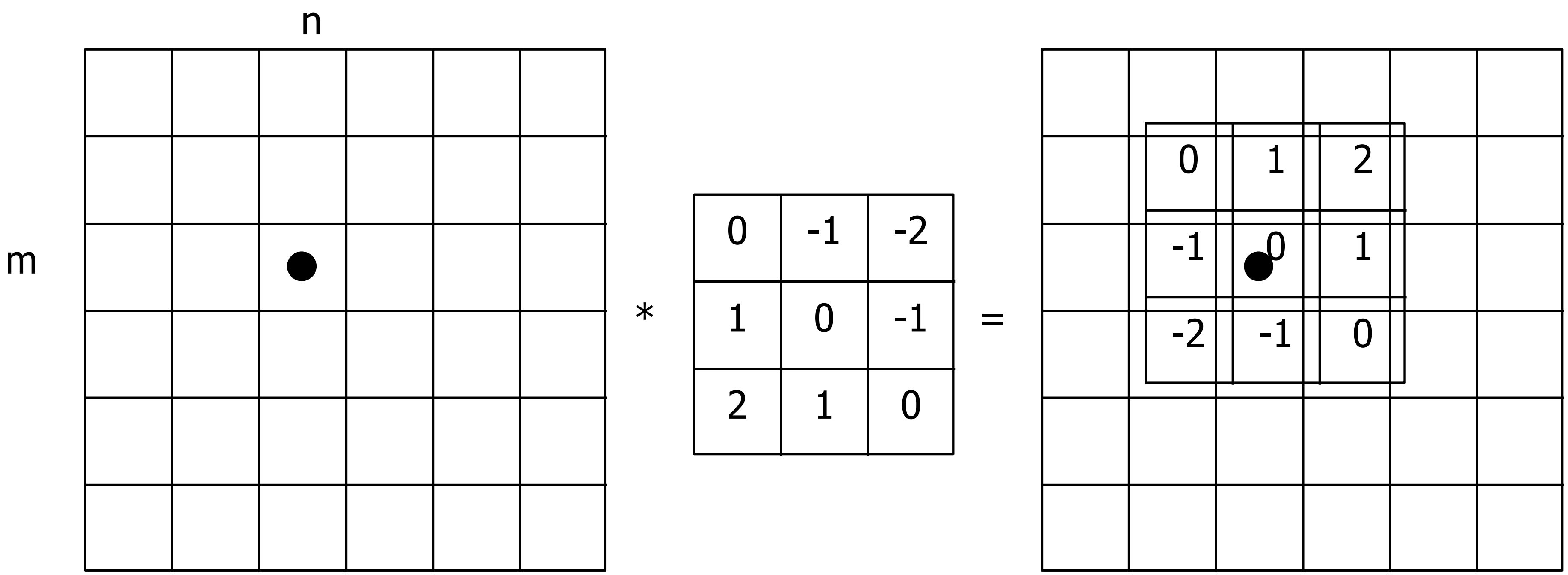
# How To

Rotate by 180°
and shift (m,n)



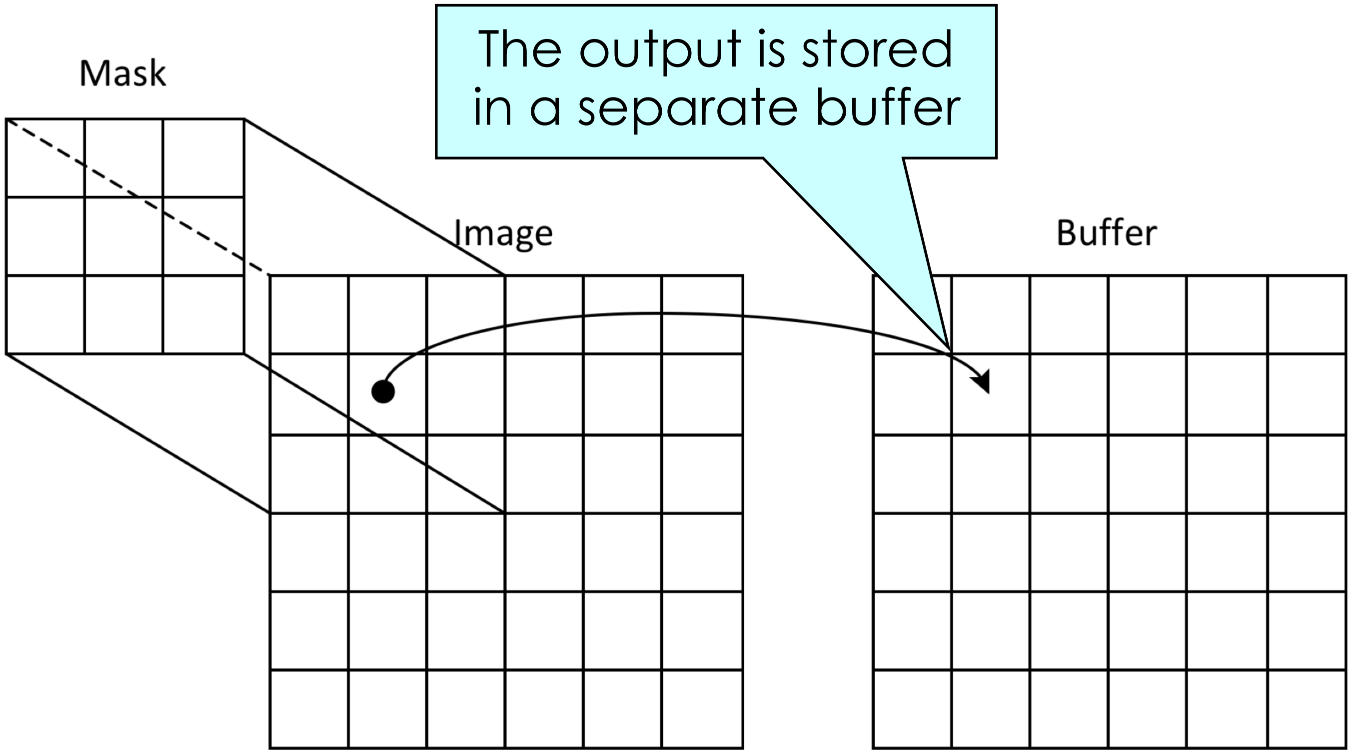



# Example: 2D convolution




Discrete convolution of $I(x,y)$ with a 3x3 mask

# Example: 2D convolution



# Examples

- (based on shape) Can we say it's the same cat?

# Smoothing/ Enhancement

# Low-pass filtering

- Easiest way is to average the values in the sliding window
- Sum up the pixel value and divide
- For a 5x5 window:

$$I_{LP}(x,y) = \frac{1}{25} \sum_{x=-2}^{+2} \sum_{y=-2}^{+2} I(x,y)$$

# Gaussian filtering

- Gaussian usually centered in the center of the filter mask

$$g(x,y) = ce^{-\frac{x^2+y^2}{2\sigma^2}}$$

- c set big enough to make sure values in the mask are integer

- Using a Gaussian as a mask is convenient as:
  - In the Fourier domain it is still a Gaussian!
  - It is isotropic
  - No need to flip the mask

$\mathbf{G_{3\times 3}} =$

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

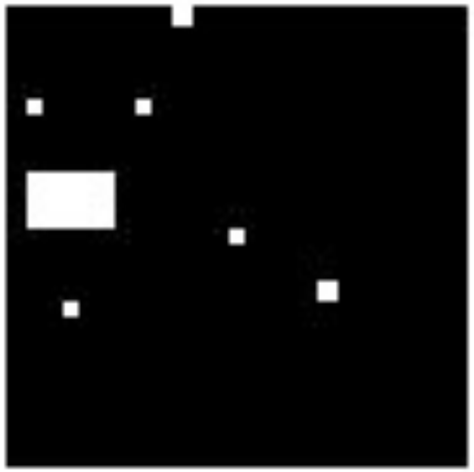
;

$\mathbf{G_{7\times 7}} =$

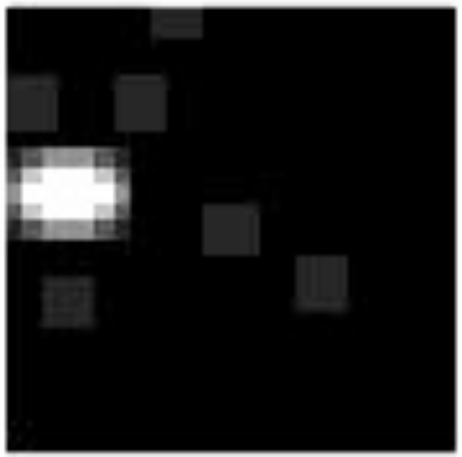
| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 3 | 7 | 9 | 7 | 3 | 1 |
| 3 | 12 | 26 | 33 | 26 | 12 | 3 |
| 7 | 26 | 55 | 70 | 55 | 26 | 7 |
| 9 | 33 | 70 | 90 | 70 | 33 | 9 |
| 7 | 26 | 55 | 70 | 55 | 26 | 7 |
| 3 | 12 | 26 | 33 | 26 | 12 | 3 |
| 1 | 3 | 7 | 9 | 7 | 3 | 1 |

# Median filtering

- In case the noise is zero-mean, smoothing is fine.
- When you have spikes, LP filtering blurs also the noise.



noise



Trento

low-pass

median

# Morphology

# Morphology in Computer vision

- Non-linear filtering
- Morphology refers to the shape of a region
- Goals:
  - Check whether a certain shape fits into another
  - Check whether a picture has holes of a certain size
  - Remove areas smaller than a threshold and with certain shape
  - …

# Binary morphology operations

- Binary image
- Binary *Structuring Element*, a known arbitrary shape
- Common shapes are rectangles and disks
- Structuring elements have an origin (typically central point, but not necessarily)

# Common operations

- Four main operations:
  - Erosion
  - Dilation
  - Opening
  - Closing
- Erosion and Dilation are self-explanatory
  - Reduce the area of a shape
  - Enlarge the area of a shape
- Opening and closing are combination of erosion/dilation
  - Opening gets rid of small portions of the image close to the boundaries of relevant areas
  - Closing fills holes and makes region boundaries smoother

# Structuring Elements

| | | | |
|---|---|---|---|
| | 1 | 1 | |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| | 1 | 1 | |

DISK

RECTANGLES

| | | | |
|---|---|---|---|
| | | | |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| | | | |

| | | | |
|---|---|---|---|
| | 1 | 1 | |
| 1 | | | 1 |
| 1 | | | 1 |
| | 1 | 1 | |

RING

Depending on the type of shape we want to edit, the right element must be chosen

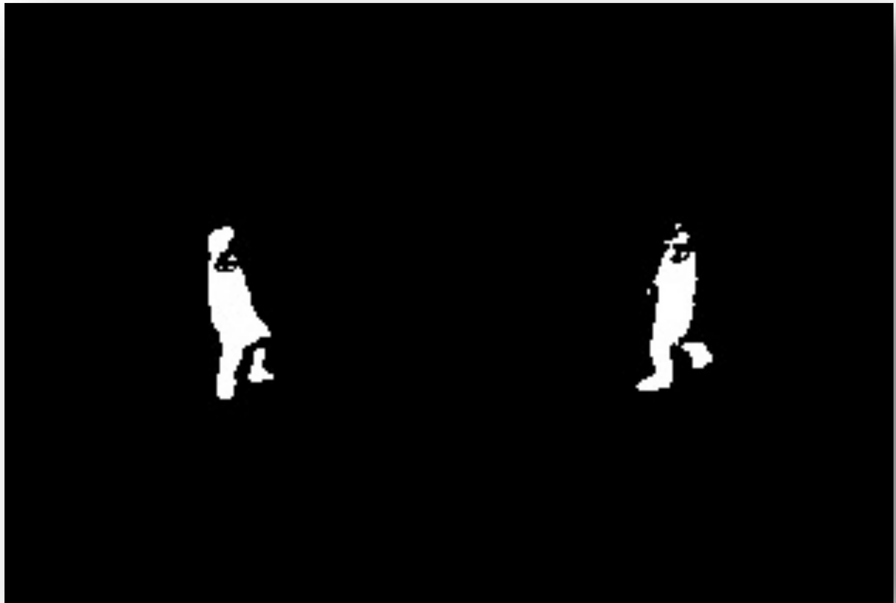
| | | | |
|---|---|---|---|
| | 1 | 1 | |
| | 1 | 1 | |
| | 1 | 1 | |
| | 1 | 1 | |

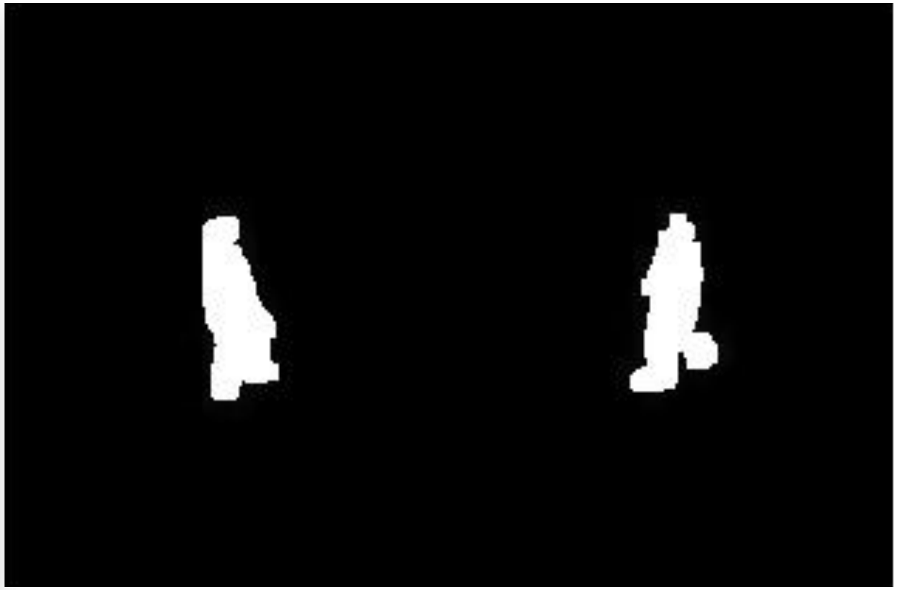
# Dilation

$$B \oplus S = \bigcup_{b \in B} S_b$$

- Sweep the structuring element on the whole image
- As the origin of the structuring element touches a "1" of the image all pixels of the structuring element are *OR'ed* to the output image

# Dilation - Example



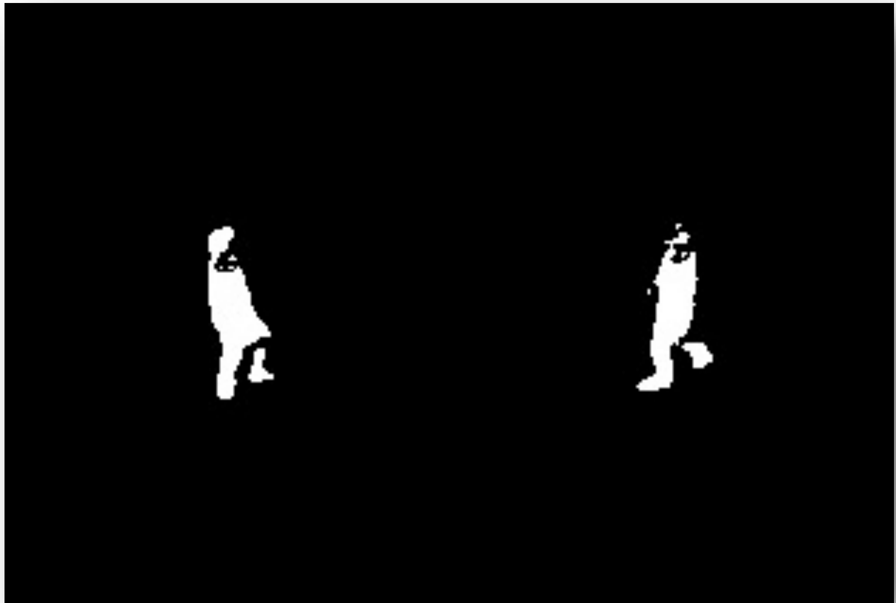


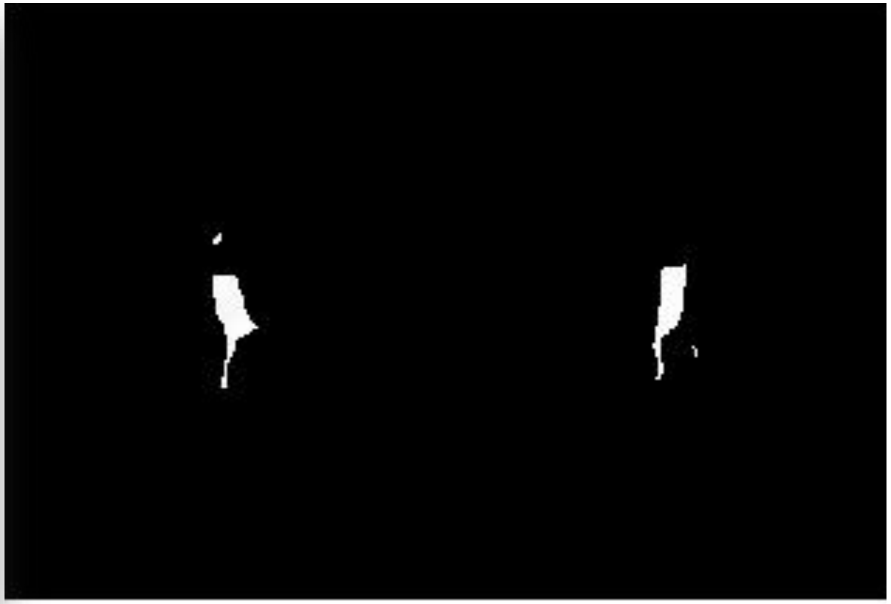
# Erosion

$$B \ominus S = \left\{ b \mid b + S \in B \qquad \forall s \in S \right\}$$

- Sweep the structuring element on the whole image
- At each position where **every 1-pixel of the structuring element covers a 1-pixel of the binary image** the binary image corresponding to the origin is OR'ed with the output image
- Erosion of *A* by *B* can be understood as the locus of points reached by the center of B when **B moves inside A**

# Erosion - Example





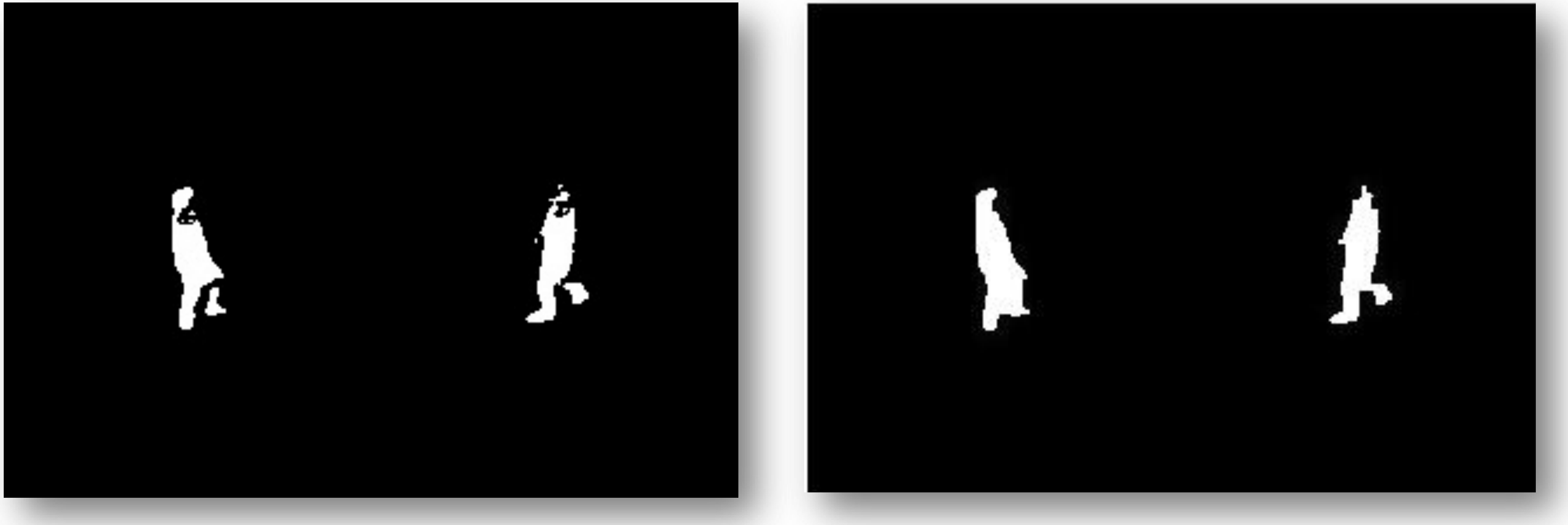
# Closing and Opening

- Closing → first dilate and then erode
- Opening → first erode and then dilate

- These operations are in general non-reversible

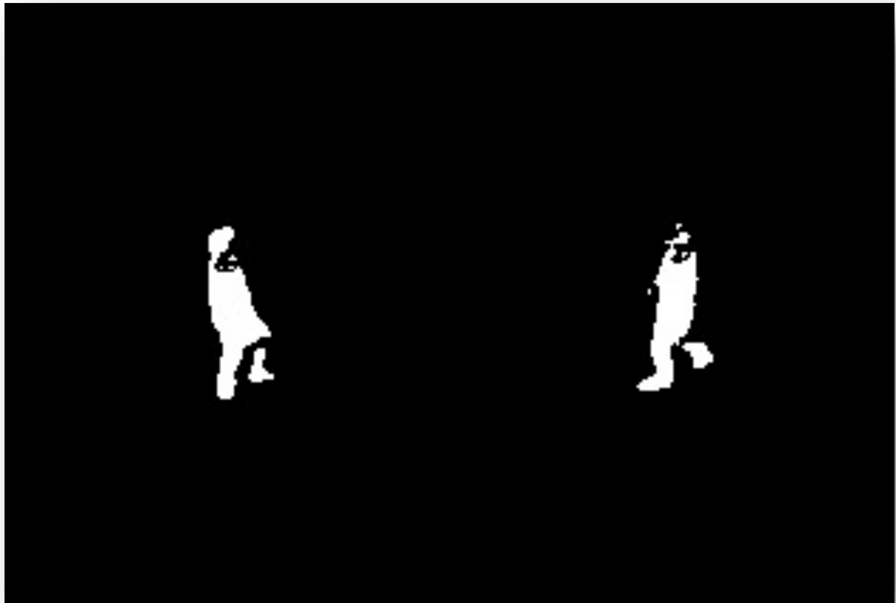
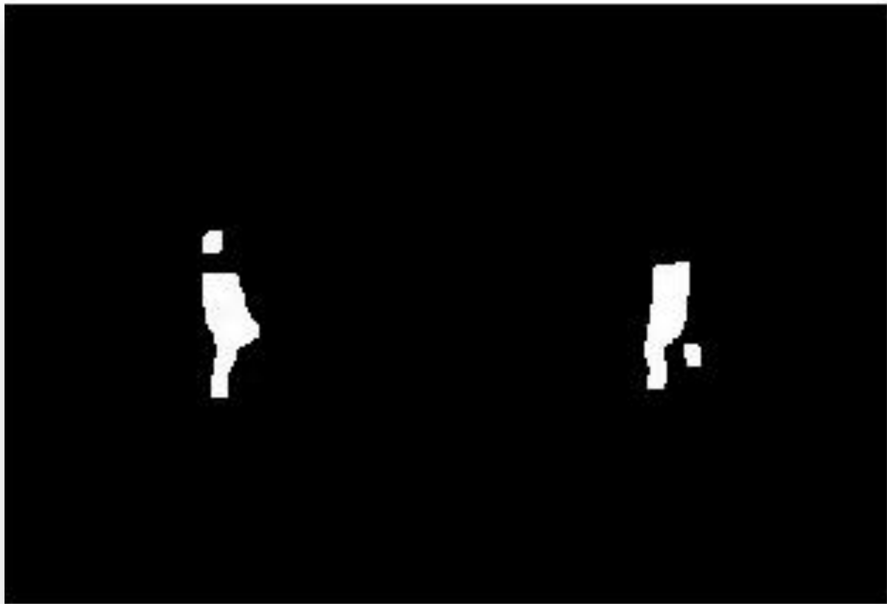
$$A \circ B = \left(A \ominus B\right) \oplus B$$
$$A \bullet B = \left(A \oplus B\right) \ominus B$$

# Closing - Example



- Non-contiguous regions are first merged
- Borders are then refined

# Opening - Example

- First small areas are removed
- Residual information is then refined