

# Basic image processing



Yoni Chechik

[www.AliMath.com](http://www.AliMath.com)



# References

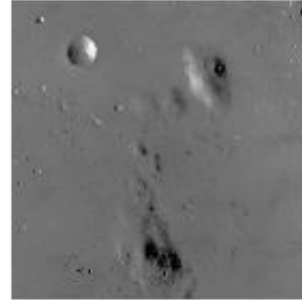
- <http://szeliski.org/Book/>
- <http://www.cs.cornell.edu/courses/cs5670/2019sp/lectures/lectures.html>
- <http://www.cs.cmu.edu/~16385/>

# Some motivation



Art  
(Photoshop color grading)

Low contrast image



Contrast stretching



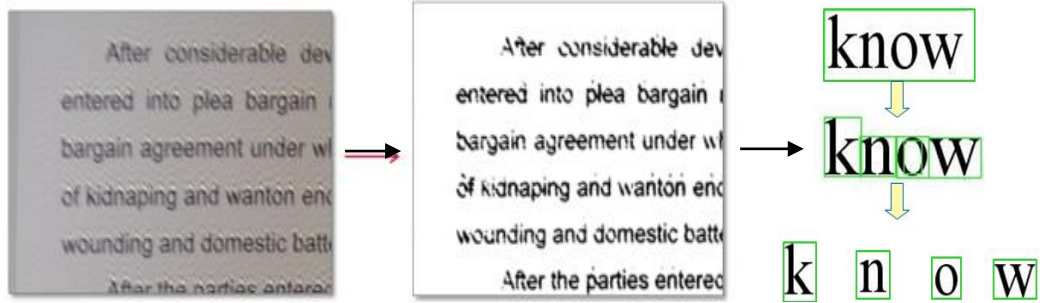
Histogram equalization



Adaptive equalization



Science and space  
(image enhancement)



Robotics  
(OCR – optical character recognition)



Agriculture  
(color ripeness detection)

# contents

- **Image representation**
- Pixel-wise operations
- Histogram equalization
- Template matching
- Morphology operators
- Connected components
- Color space



# Image representation

- We can think of an image as a 3d matrix of discrete RGB values.
- The values mark the intensity of each color channel and are usually of type `uint8 = {0, ..., 255}`.



# Image representation

- We can also think of an image as a function  $f(x, y)$  .



# contents

- Image representation
- **Pixel-wise operations**
- Histogram equalization
- Template matching
- Morphology operators
- Connected components
- Color space

# Pixel-wise operators

- Pixel-wise operators, or point operators, are defined as such that each output pixel's value depends on only the corresponding input pixel value.



# Pixel-wise operators

original



$x$

darken



lower contrast



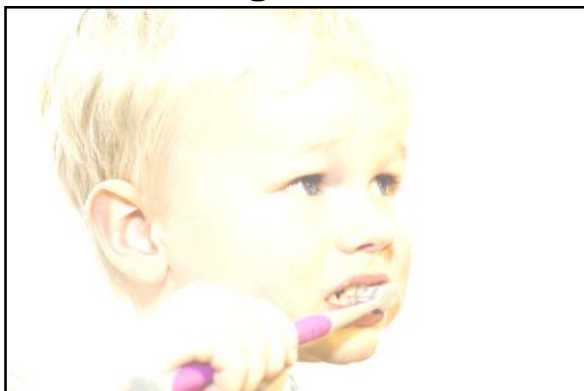
Gamma compression



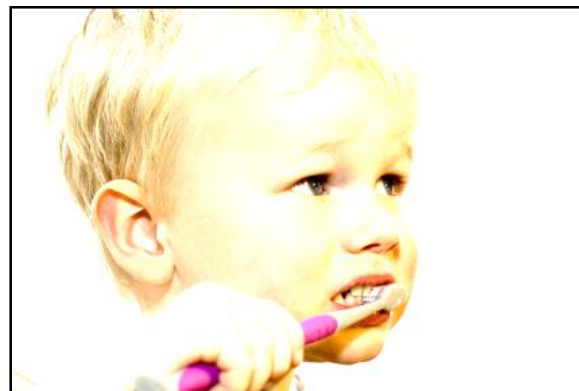
invert



lighten



raise contrast



Gamma expansion



# Pixel-wise operators

original



$x$

darken



lower contrast



Gamma compression

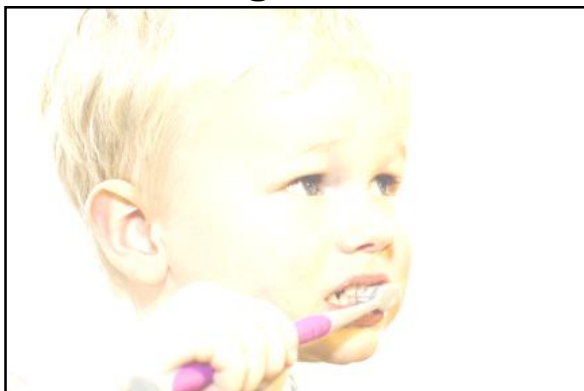


invert

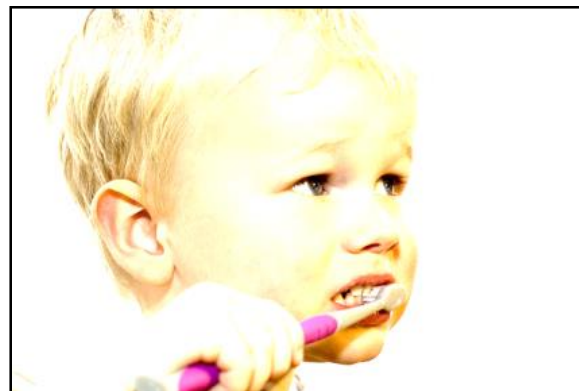


$255 - x$

lighten



raise contrast



Gamma expansion



# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



Gamma compression

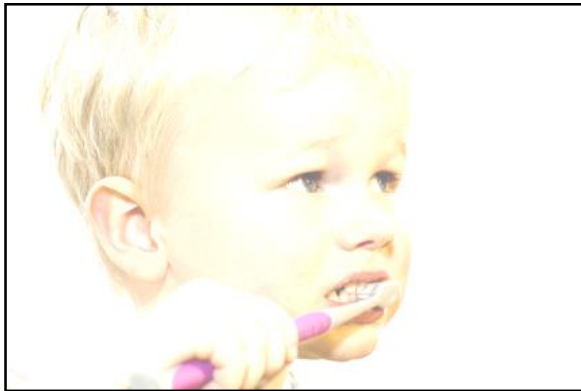


invert



$$255 - x$$

lighten



raise contrast



Gamma expansion





# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



Gamma compression

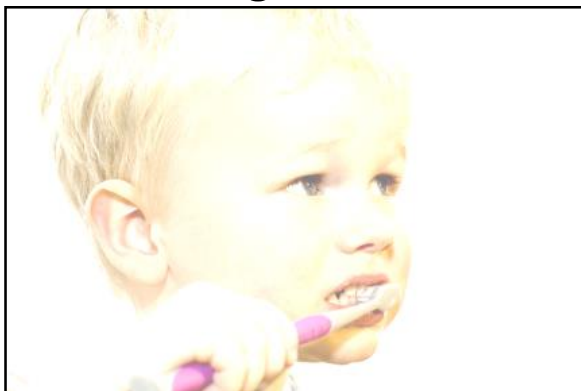


invert



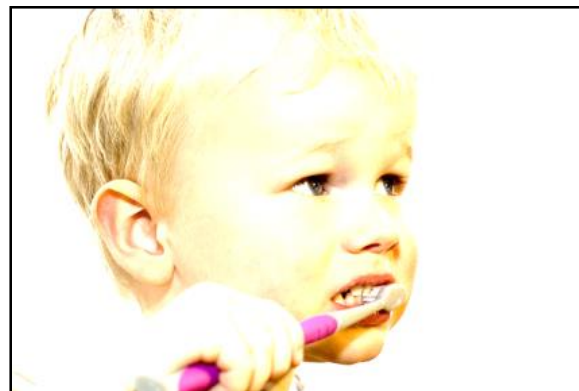
$$255 - x$$

lighten



$$x + 128$$

raise contrast



Gamma expansion



# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



$$\frac{x}{2}$$

Gamma compression



invert



$$255 - x$$

lighten



$$x + 128$$

raise contrast



Gamma expansion





# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



$$\frac{x}{2}$$

Gamma compression

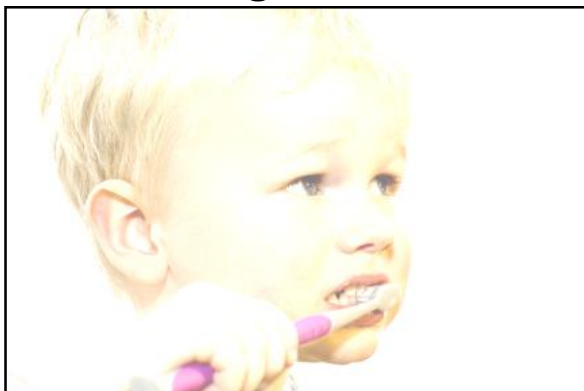


invert



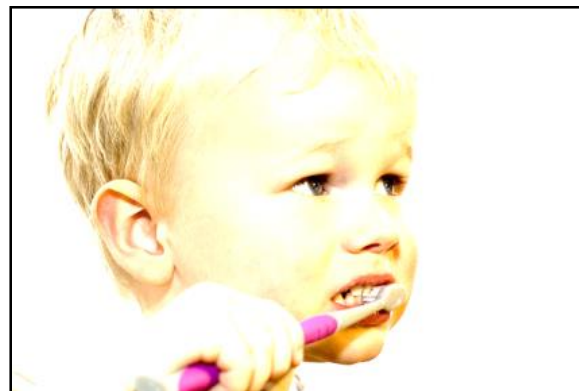
$$255 - x$$

lighten



$$x + 128$$

raise contrast



$$x \times 2$$

Gamma expansion



# Contrast

- **Contrast** in visual perception is the difference in appearance of two or more parts of a seen field.
- The human visual system is more sensitive to contrast than absolute luminance;
- **Contrast ratio**, or **dynamic range**, is the ratio between the largest and smallest values of the image or:

$$CR = \frac{V_{max}}{V_{min}}$$



# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



$$\frac{x}{2}$$

Gamma compression



invert



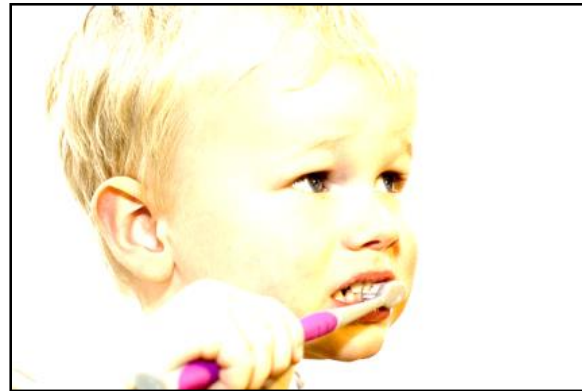
$$255 - x$$

lighten



$$x + 128$$

raise contrast



$$x \times 2$$

Gamma expansion





# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



$$\frac{x}{2}$$

Gamma compression



$$\left(\frac{x}{255}\right)^{1/3} \times 255$$

invert



$$255 - x$$

lighten



$$x + 128$$

raise contrast



$$x \times 2$$

Gamma expansion



# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



$$\frac{x}{2}$$

Gamma compression



$$\left(\frac{x}{255}\right)^{1/3} \times 255$$

invert



$$255 - x$$

lighten



$$x + 128$$

raise contrast



$$x \times 2$$

Gamma expansion

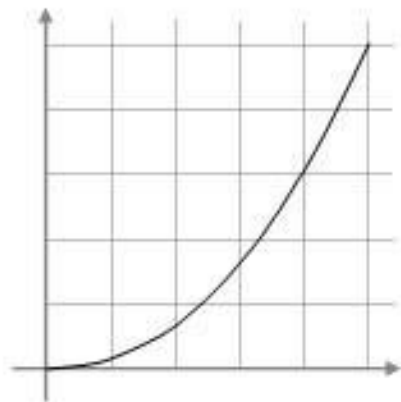


$$\left(\frac{x}{255}\right)^2 \times 255$$



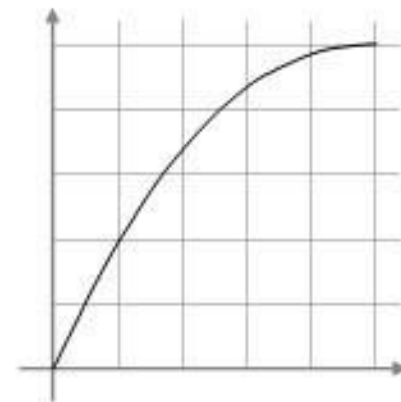
# Gamma correction

- Originally, Due to non-linearities in the old CRT televisions, intensities was seen different then they are.



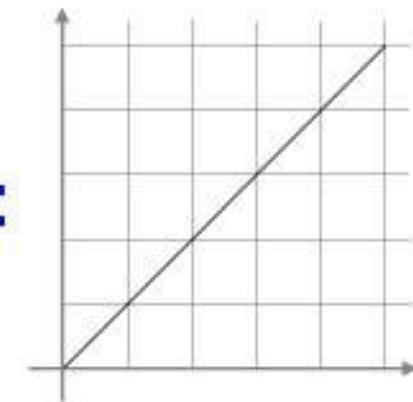
Gamma characteristics of monitors

×

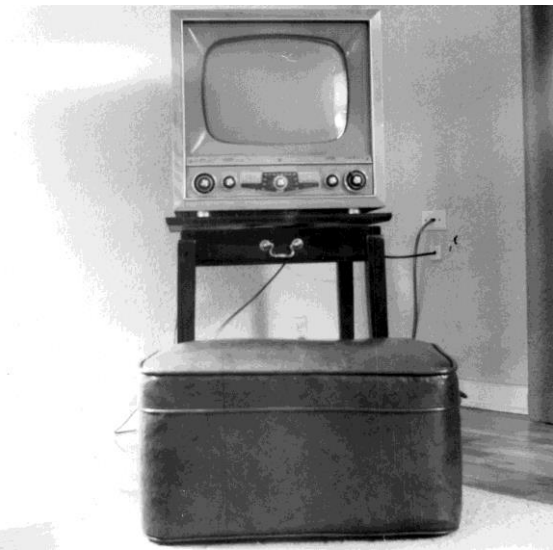


Color information adjusted to match gamma characteristics

=



Color handling approaching the "y = x" idealcs

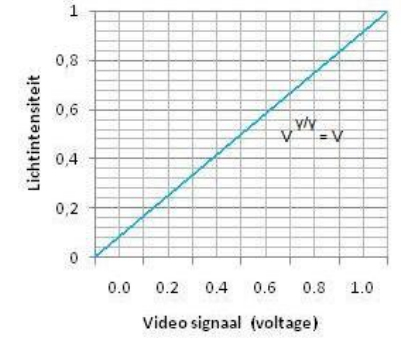
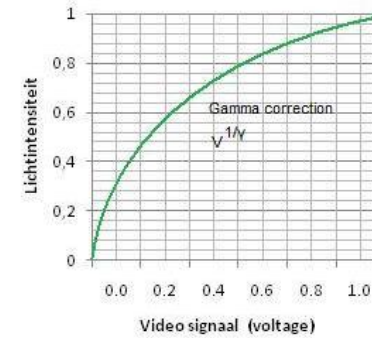
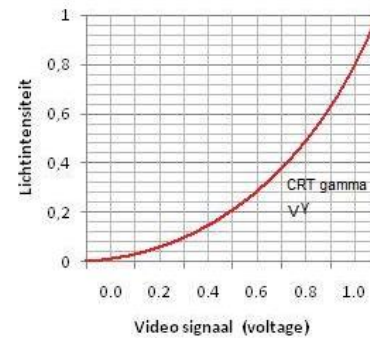
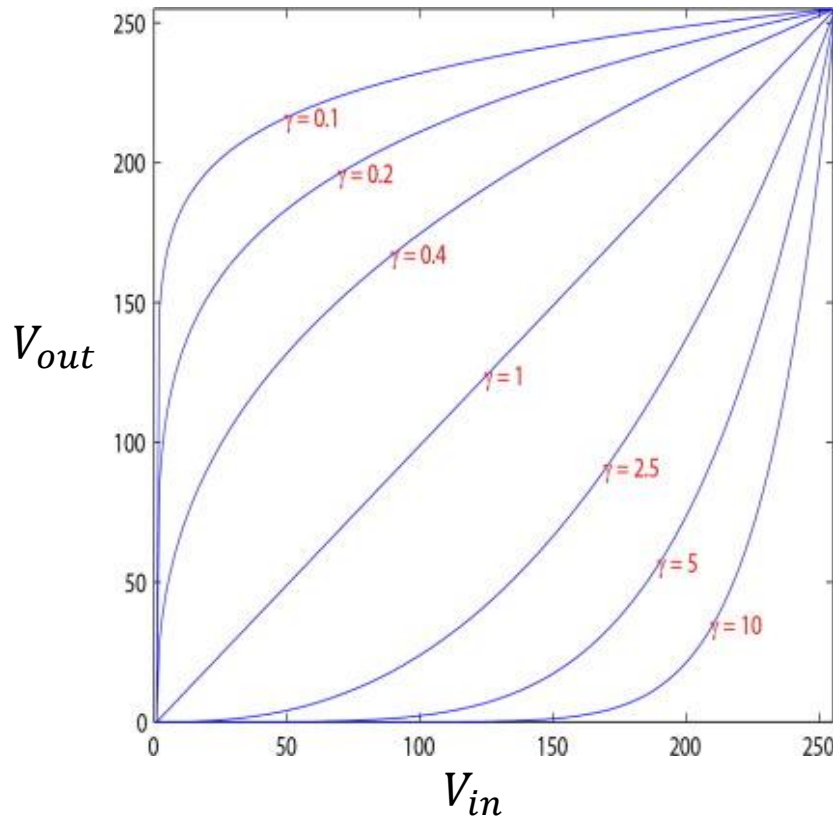


# Gamma correction

- To correct this non-linear transformation, gamma correction was done:

$$V_{out} = \left( \frac{V_{in}}{255} \right)^\gamma \cdot 255 \quad (V_{in}, V_{out} \in \{0, 1, \dots, 255\})$$

- This is, of course, also applicable for image enhancements.





# Some more point- wise operators



# contents

- Image representation
- Pixel-wise operations
- **Histogram equalization**
- Template matching
- Morphology operators
- Connected components
- Color space

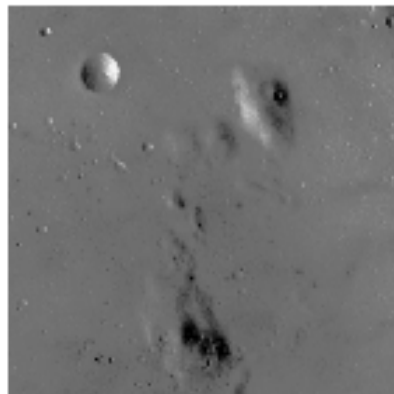
# Histogram equalization

- **Histogram equalization** is a method in image processing of contrast adjustment using the image's histogram.
- This method is used to increase the global contrast of an image and is useful in images with backgrounds and foregrounds that are both bright or both dark.
- Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values.





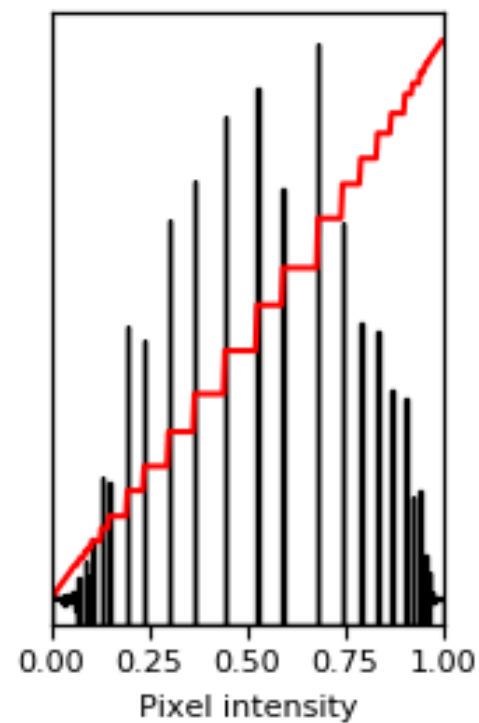
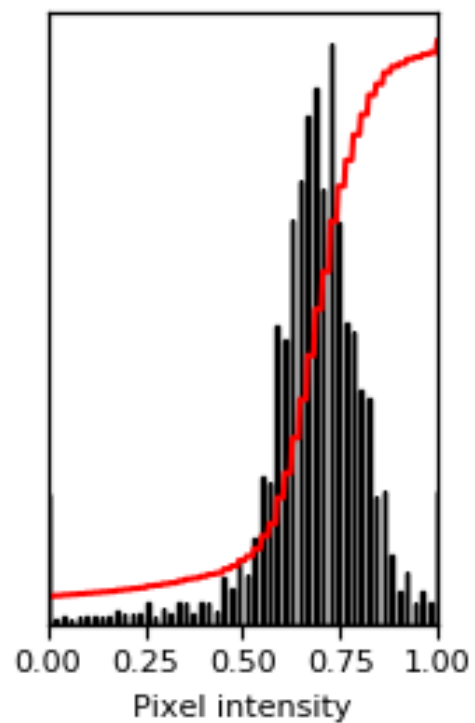
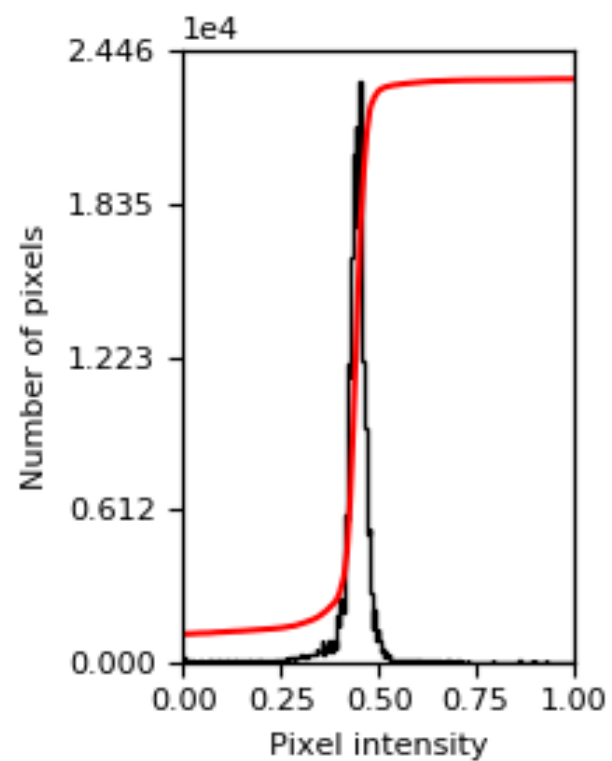
Low contrast image



Contrast stretching

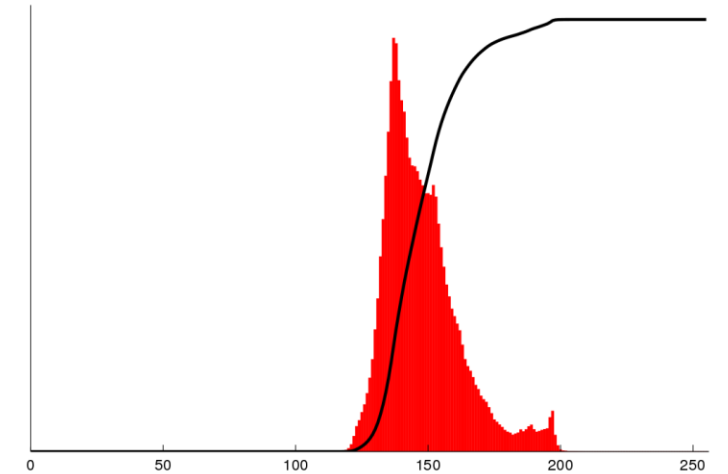


Histogram equalization



# Histogram equalization

- A histogram is a discrete form representation of the distribution of numerical data.
- We will assume at first that our image is continuous in the range  $[0,255]$  for better understanding.
- Instead of a histogram we will talk about the **probability density function (PDF)**  $f_X(x)$  of the data.



# PDF and CDF

- **cumulative distribution function (CDF)** of a real-valued random variable  $X$  is the probability that  $X$  will take a value less than or equal to  $x$ :

$$F_X(x) = P(X \leq x)$$

- Properties of CDF:

- $\lim_{x \rightarrow -\infty} F_X(x) = 0, \quad \lim_{x \rightarrow +\infty} F_X(x) = 1$

- Monotonically non decreasing.

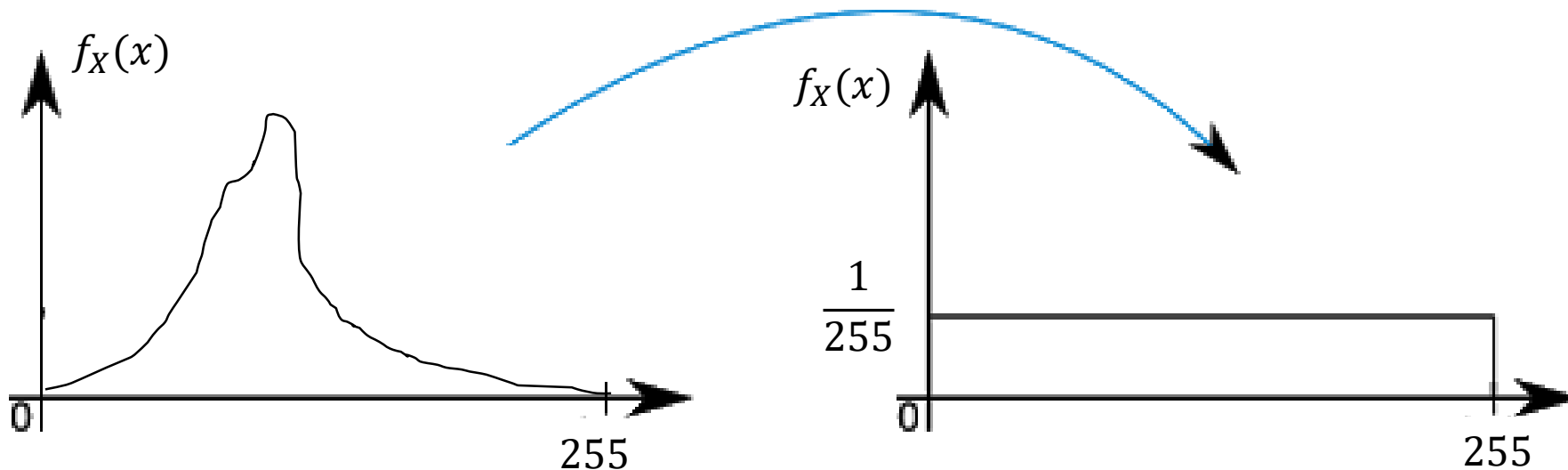
- The **probability density function (PDF)** of a continuous random variable can be determined from the cumulative distribution function by differentiating.

$$f_X(x) = \frac{dF_X(x)}{dx} \quad \text{OR} \quad F_X(x) = \int_{-\infty}^x f_X(t) dt$$

# PDF equalization

- We want that our resulting PDF will be constant for any value in the range  $[0,255]$ .
- If the PDF is constant, that means that the CDF is linear in  $[0,255]$ , and so we get the final CDF as:

$$F_Y(y) = P(Y \leq y) = \begin{cases} 0 & : y < 0 \\ y & : 0 \leq y \leq 255 \\ 1 & : y > 255 \end{cases}$$



# PDF equalization

- So we are looking for a transformation function such that:

$$y = T(x)$$

- In the interesting area  $[0,1]$ :

$$P(Y \leq y) = y$$

$$P(T(X) \leq y) = y$$

$$\text{assuming } T \text{ is invertible: } P(X \leq T^{-1}(y)) = y$$

$$\text{change of variables } z = T^{-1}(y): P(X \leq z) = T(z)$$

- The answer is  $F_X(x) = T(x)$ , and in fact  $T$  is invertible since  $F_X$  is Monotonically non decreasing



# Back to histogram equalization

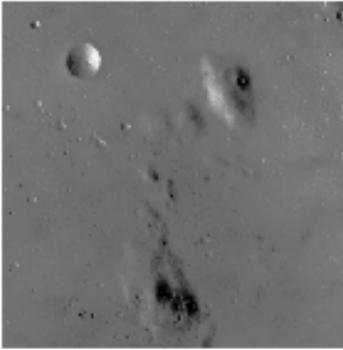
- The same result is also applicable for discrete space like actual images and their histograms.

- Build histogram of a given image.
- To make the histogram act like a discrete pdf, we will normalize the PDF: divide each bin by the sum of all bins.
- Build the discrete CDF.
- Un-normalize the CDF and round the results back to uint8:

$$f_{eq}(x) = \text{round}(CDF(x) * 255)$$

# Other variants of histogram equalization

Low contrast image



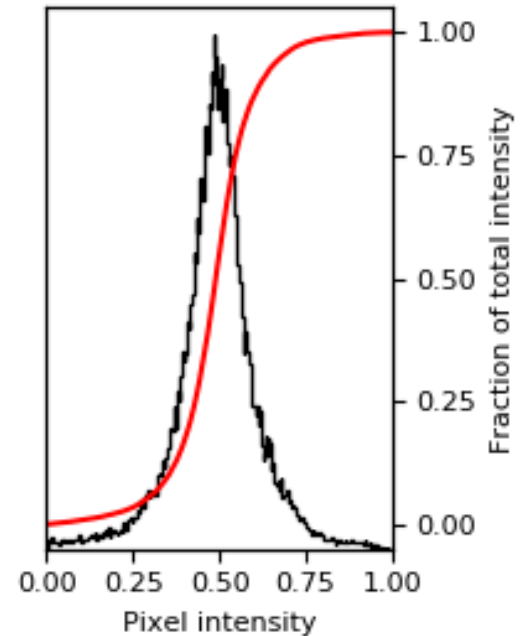
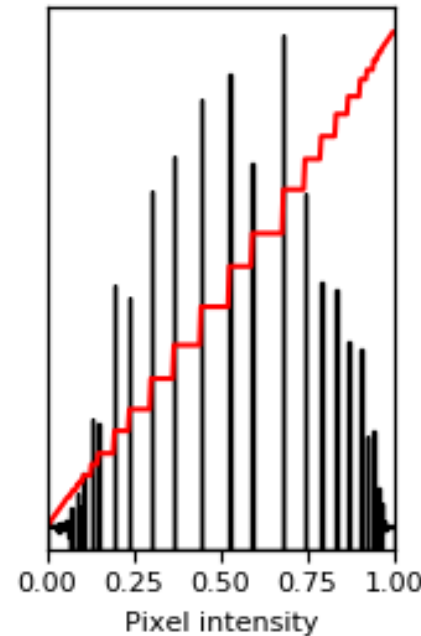
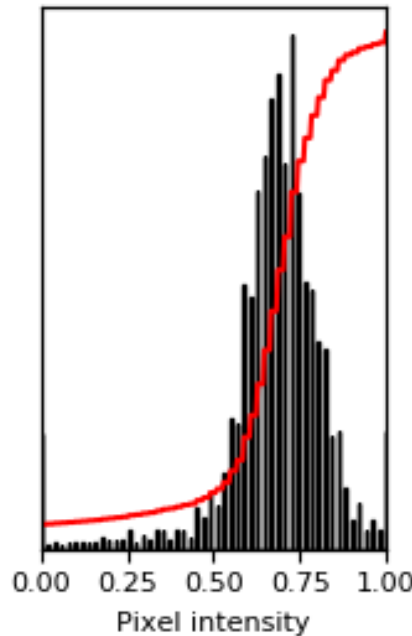
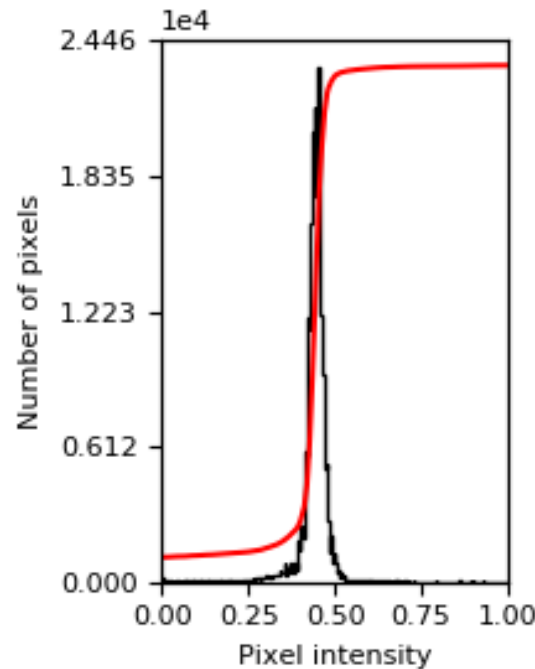
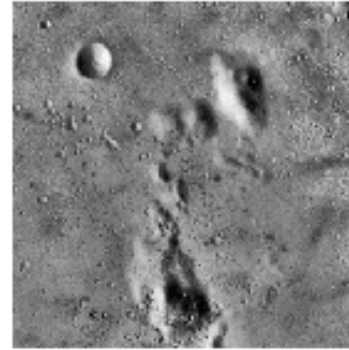
Contrast stretching



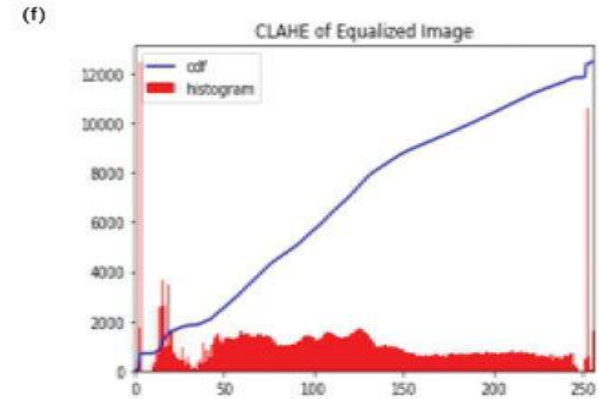
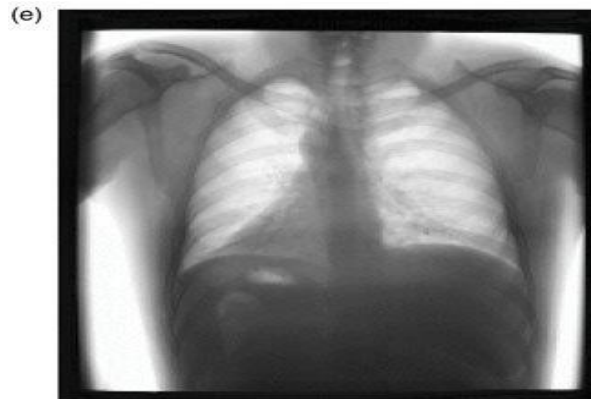
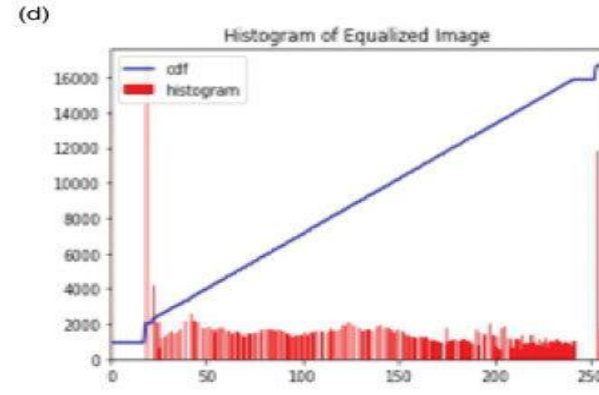
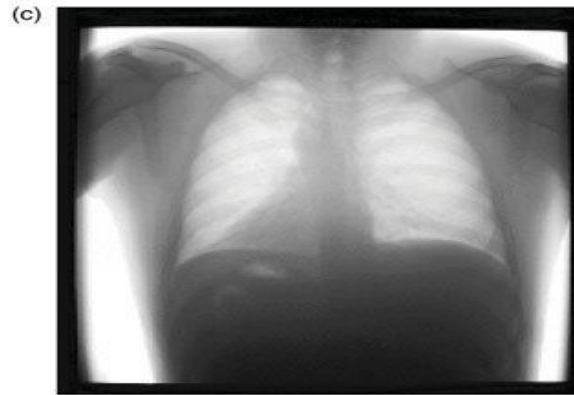
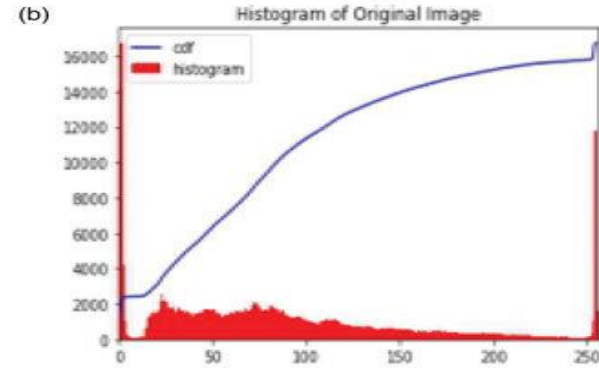
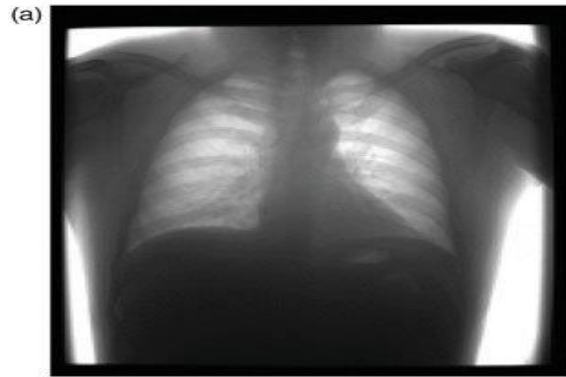
Histogram equalization



Adaptive equalization



# Other variants of histogram equalization

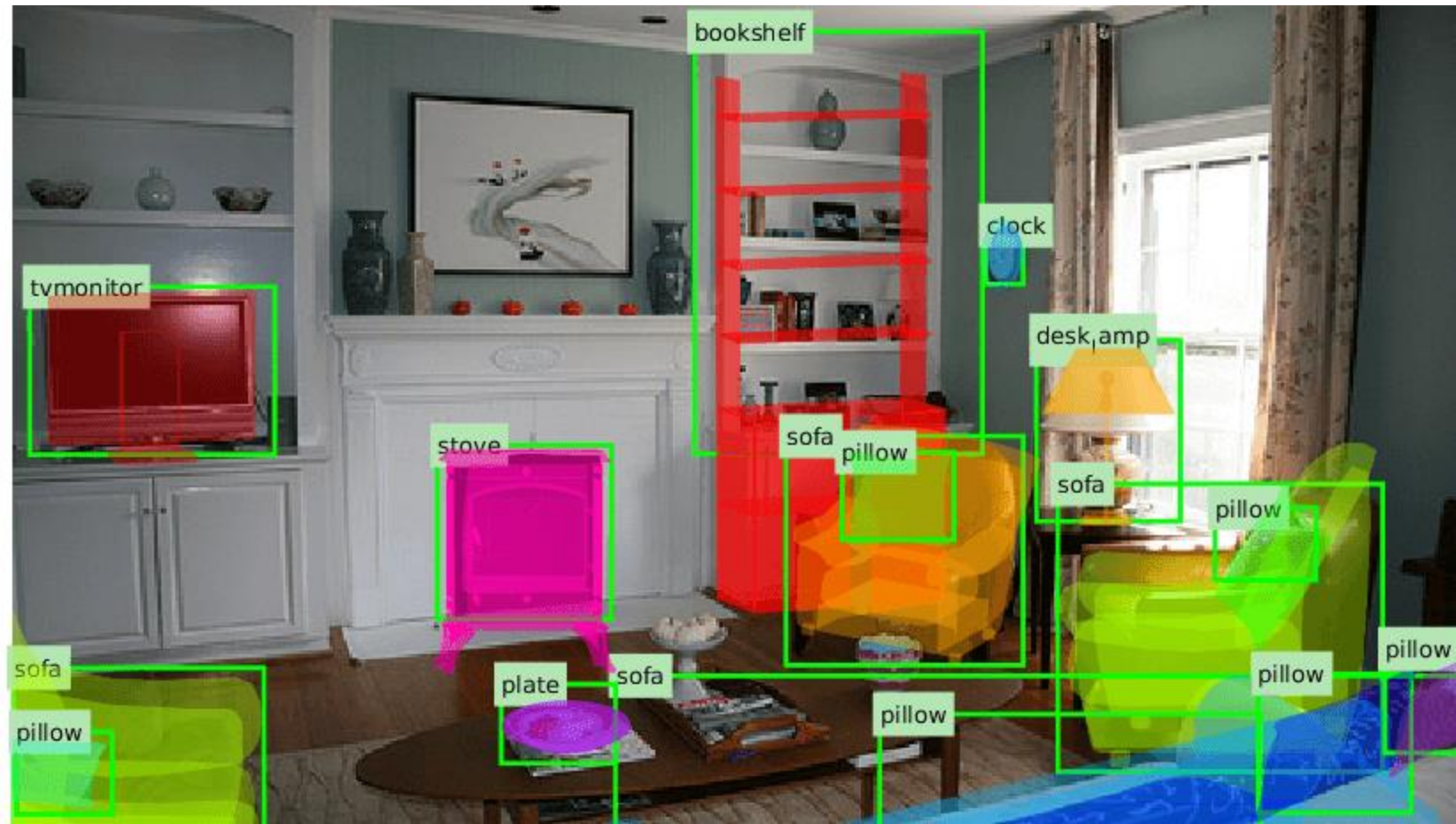


# contents

- Image representation
- Pixel-wise operations
- Histogram equalization
- **Template matching**
- Morphology operators
- Connected components
- Color space

# Template matching

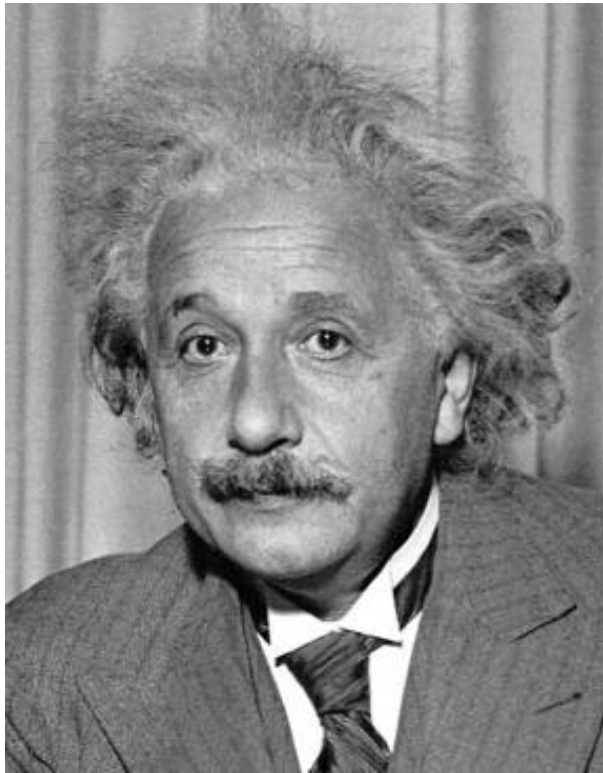
- Given an image template- find it in another image.
- Template matching is a sub-field in **object recognition**.
  - We will see it **a lot** of this topic in this course:
    - SSD
    - Cross correlation
    - Feature based – SIFT
    - Neural networks






# SSD – sum squared distances

How do we detect the template  in the following image?

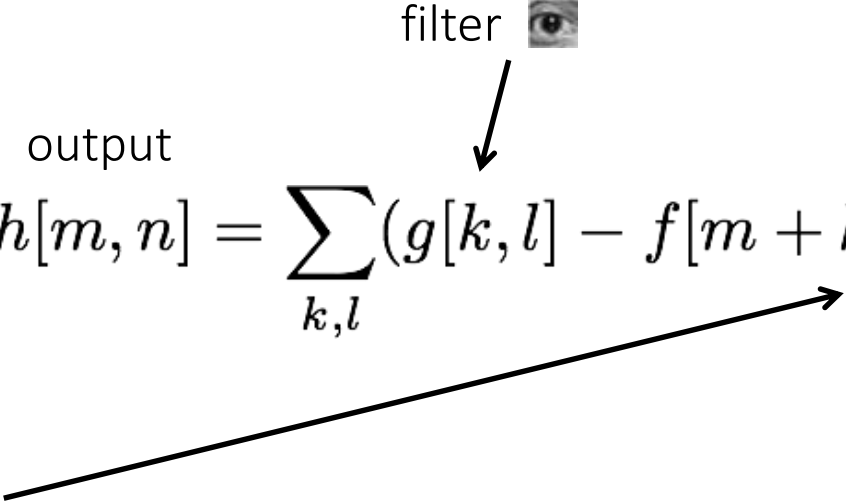


output

filter 

$$h[m, n] = \sum_{k, l} (g[k, l] - f[m + k, n + l])^2$$

image

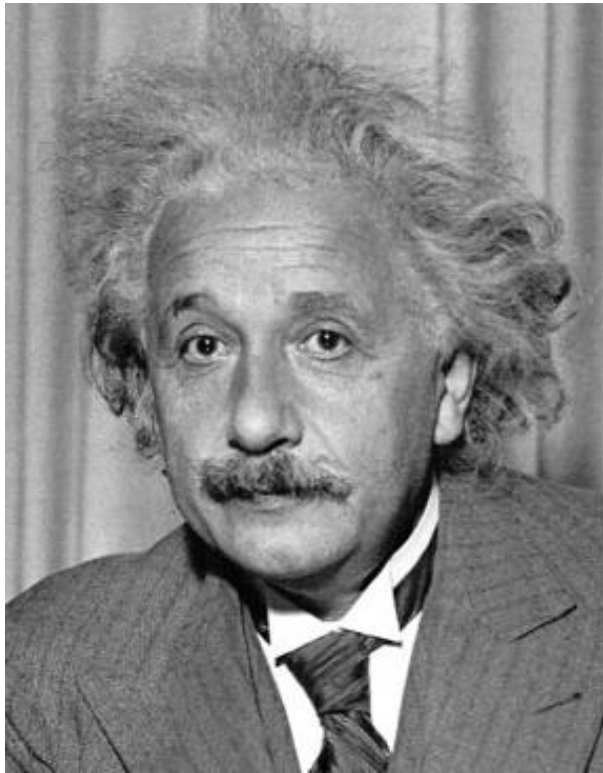


The diagram illustrates the SSD formula. An arrow labeled 'image' points from the bottom left towards the formula. Another arrow labeled 'filter' points from the 'eye' icon towards the formula. A long arrow points from the bottom left towards the formula, indicating the input image.

What will the output look like?


# SSD – sum squared distances

How do we detect the template  in the following image?



output

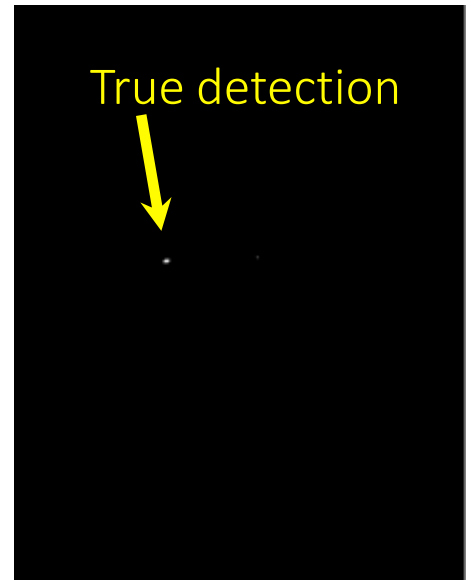
$$h[m, n] = \sum_{k, l} (g[k, l] - f[m + k, n + l])^2$$

filter 

image

thresholding

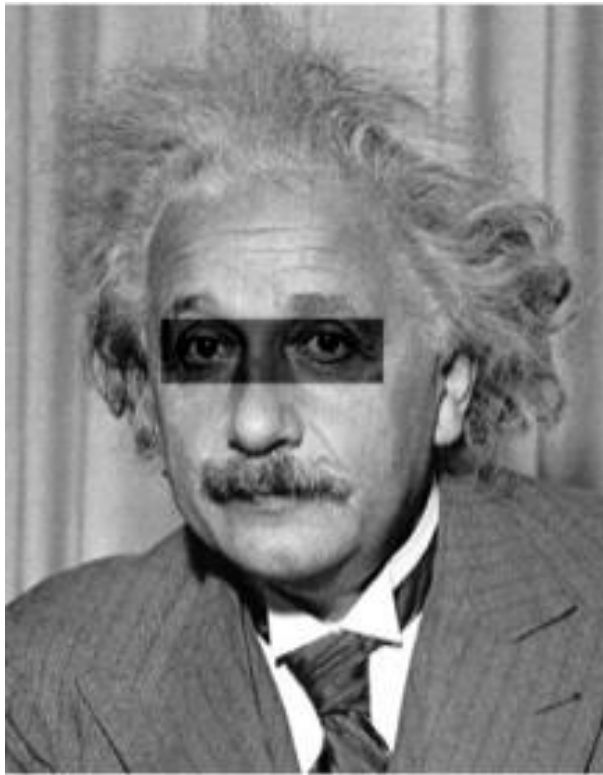
1-output




What could go wrong?

# SSD – sum squared distances

How do we detect the template  in the following image?



output

filter 

$$h[m, n] = \sum_{k, l} (g[k, l] - f[m + k, n + l])^2$$

image

The diagram shows the SSD process. An arrow labeled 'filter' points from the eye icon to the equation. An arrow labeled 'image' points from the Einstein image to the equation. The word 'output' is placed above the equation.

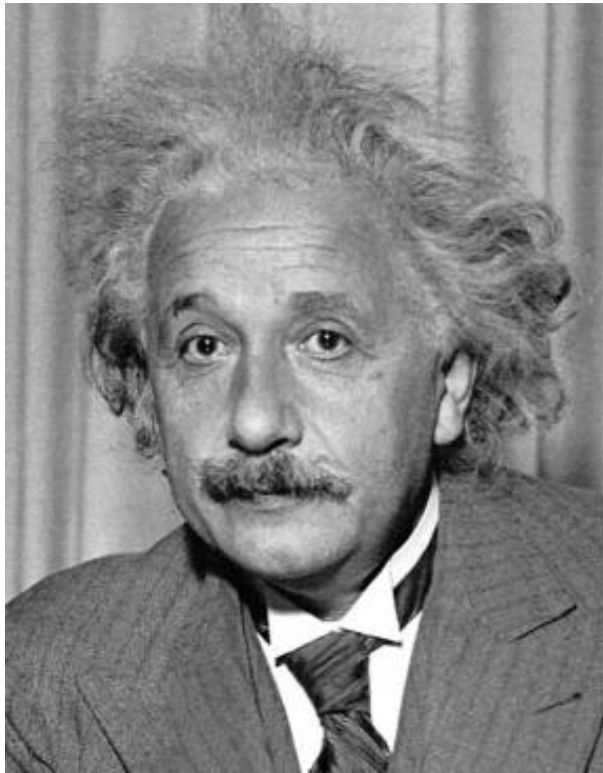
1-output



Not robust to local intensity changes


# CC – cross correlation

How do we detect the template  in the following image?

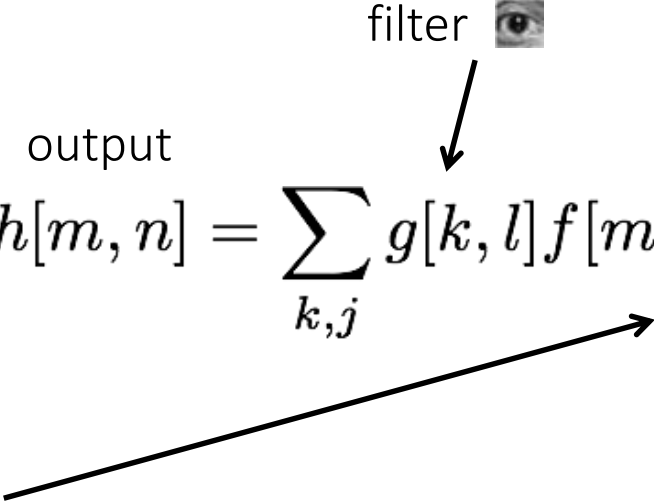


output

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

filter 

image

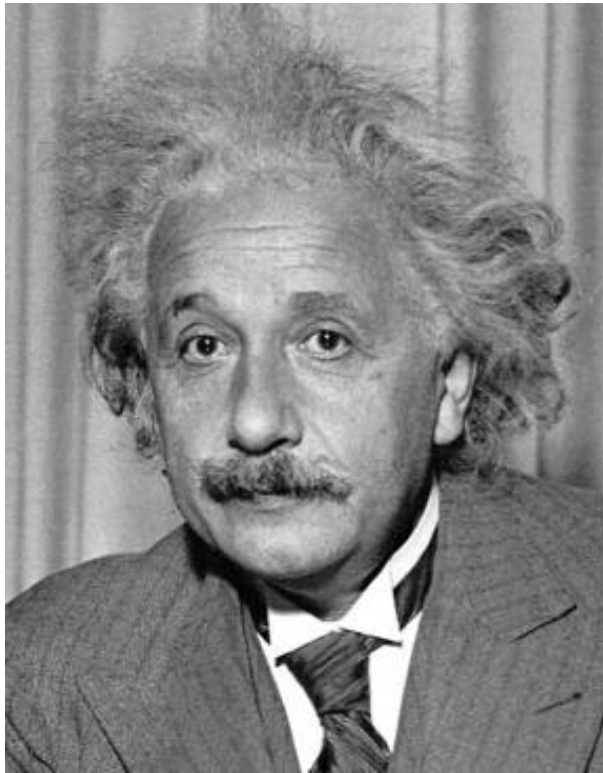


What will the output look like?




# CC – cross correlation

How do we detect the template  in the following image?

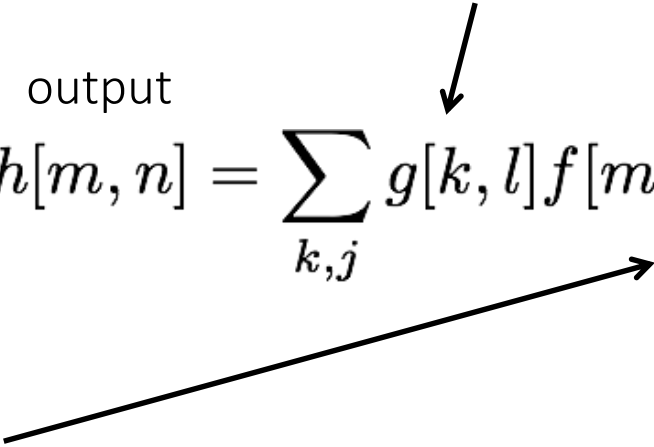


output

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

filter 

image

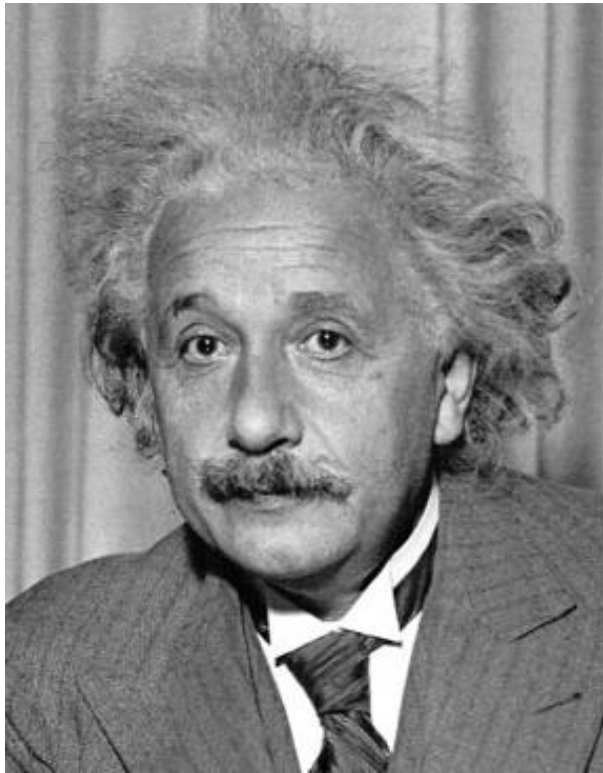




What went wrong?


# CC – cross correlation

How do we detect the template  in the following image?

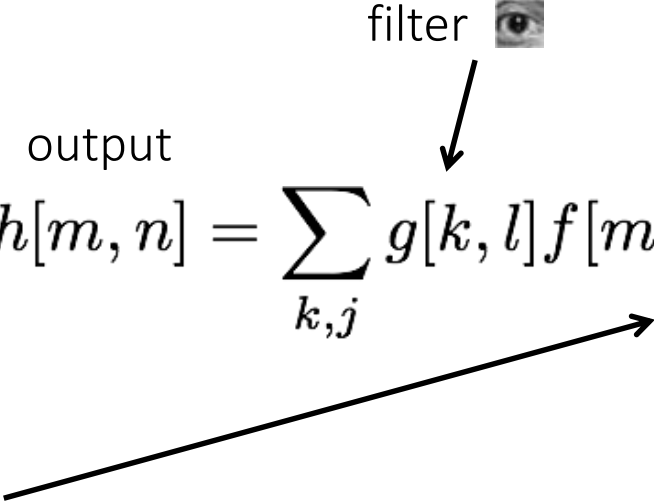


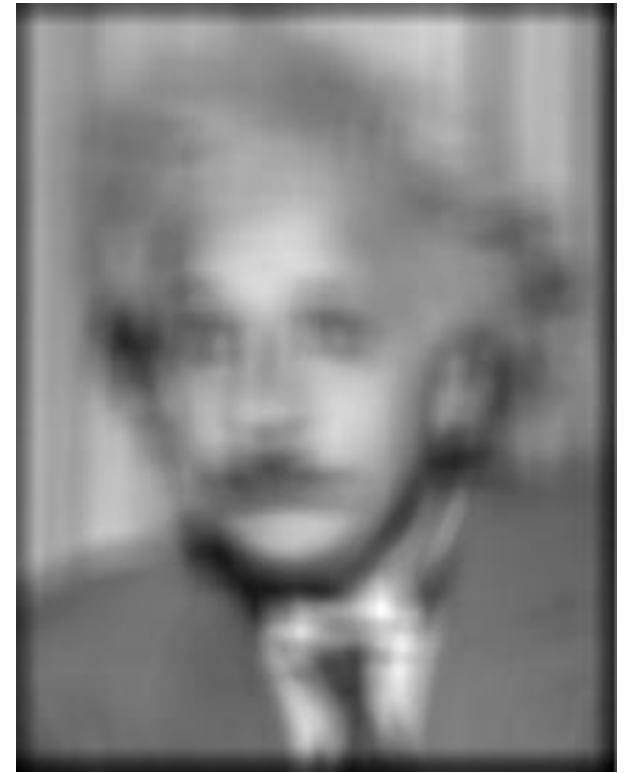
output

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

filter 

image

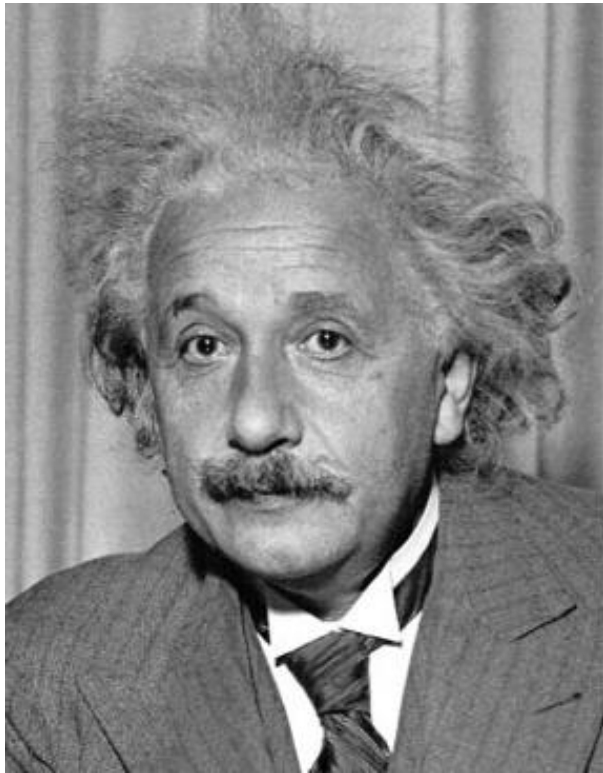




Increases for higher  
local intensities.


# Zero mean cross correlation

How do we detect the template  in the following image?



output

$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g}) f[m + k, n + l]$$

filter  template mean

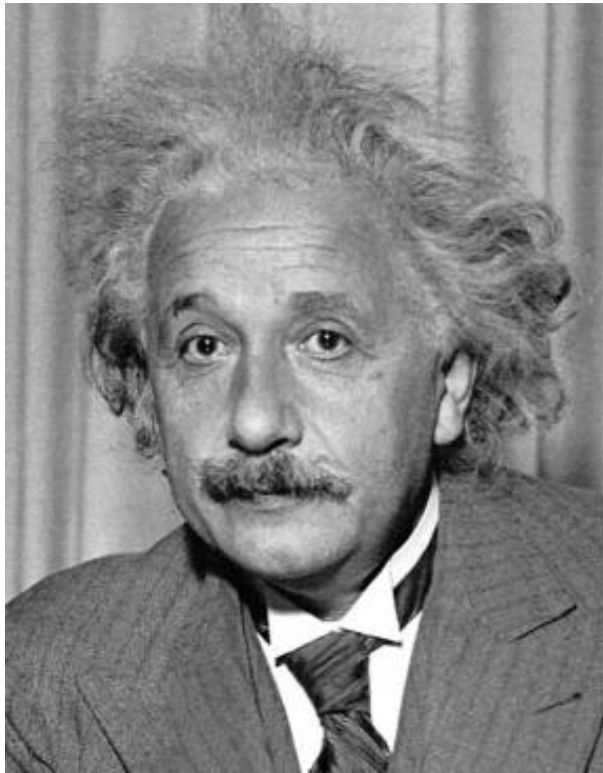
image

The diagram shows the equation for zero mean cross correlation. An arrow points from the word 'filter' to the term  $g[k, l]$  in the equation. Another arrow points from the words 'template mean' to the term  $\bar{g}$ . A third arrow points from the word 'image' to the term  $f[m + k, n + l]$ . The word 'output' is placed above the equation.


What will the output look like?

# Zero mean cross correlation

How do we detect the template  in the following image?



output

filter 

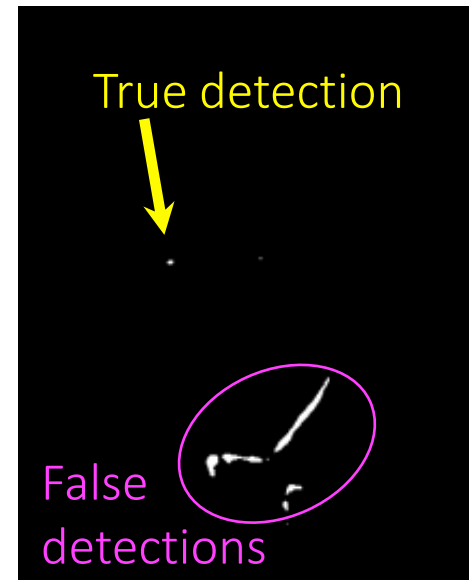
template mean

$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g}) f[m + k, n + l]$$

image

thresholding

output

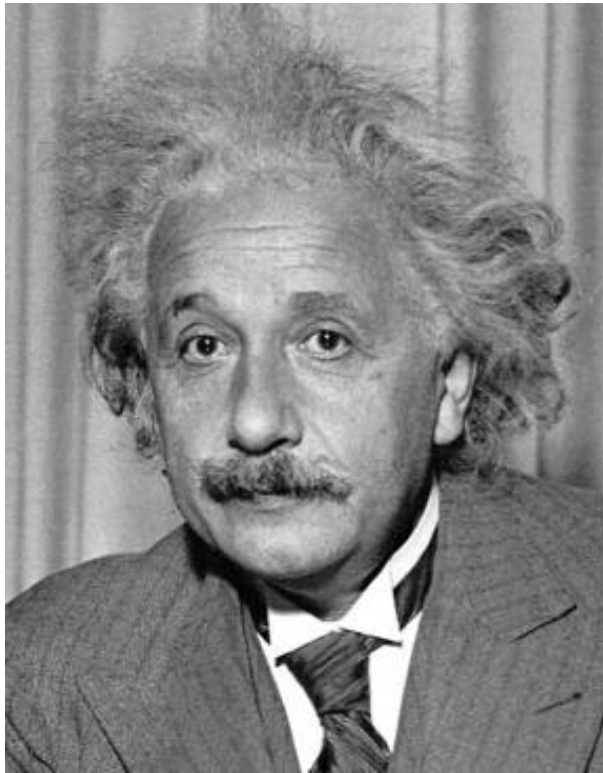


What went wrong?




# Zero mean cross correlation

How do we detect the template  in the following image?



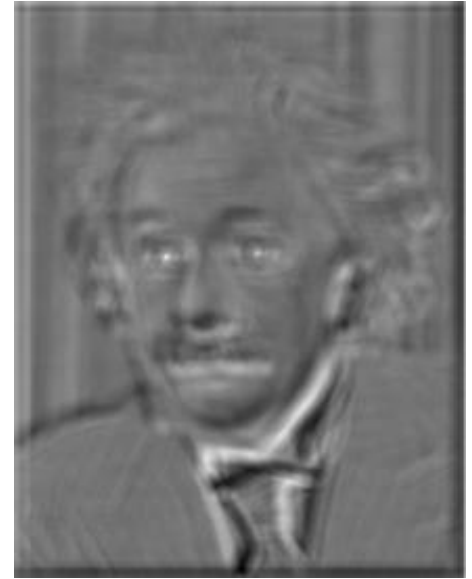
output

$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g}) f[m + k, n + l]$$

filter 

template mean

output

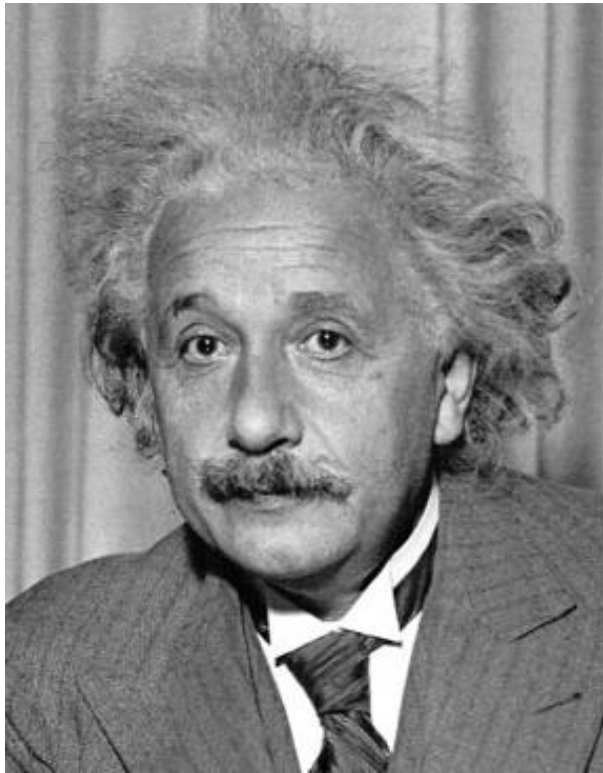


Not robust to high-contrast areas


image

# ZNCC – zero mean normalized cross correlation

How do we detect the template  in the following image?



What will the output look like?

filter  template mean

output

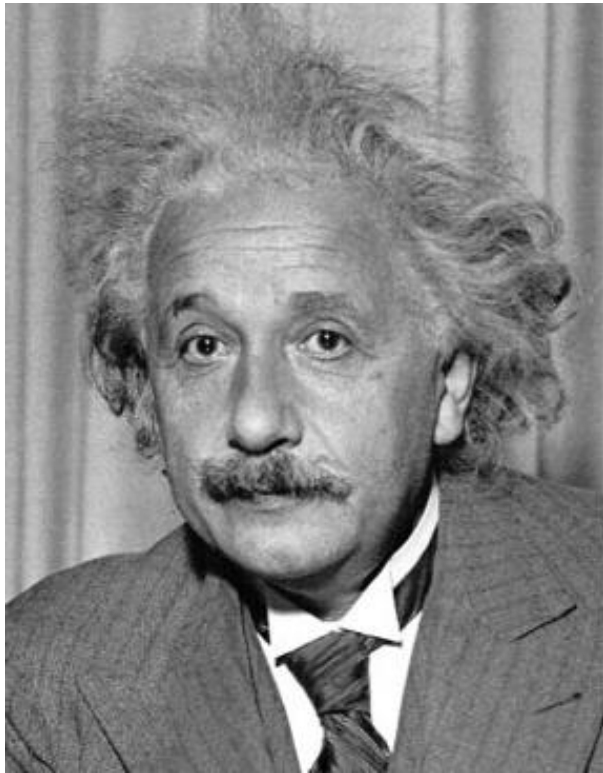
$$h[m, n] = \frac{\sum_{k,l} (g[k, l] - \bar{g})(f[m + k, n + l] - \bar{f}_{m,n})}{\sqrt{(\sum_{k,l} (g[k, l] - \bar{g})^2 \sum_{k,l} (f[m + k, n + l] - \bar{f}_{m,n})^2)}}$$

image

local patch mean

# ZNCC – zero mean normalized cross correlation

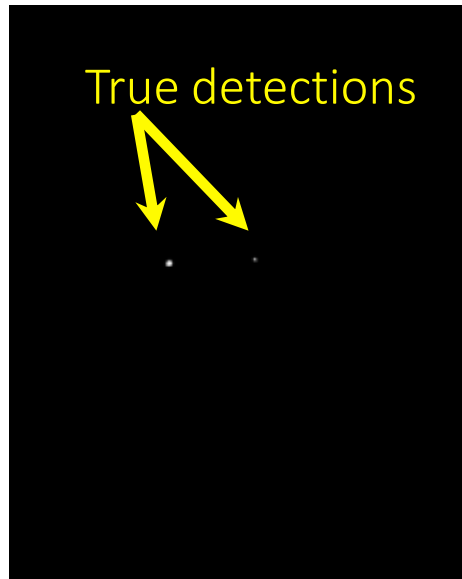
How do we detect the template  in the following image?



1-output

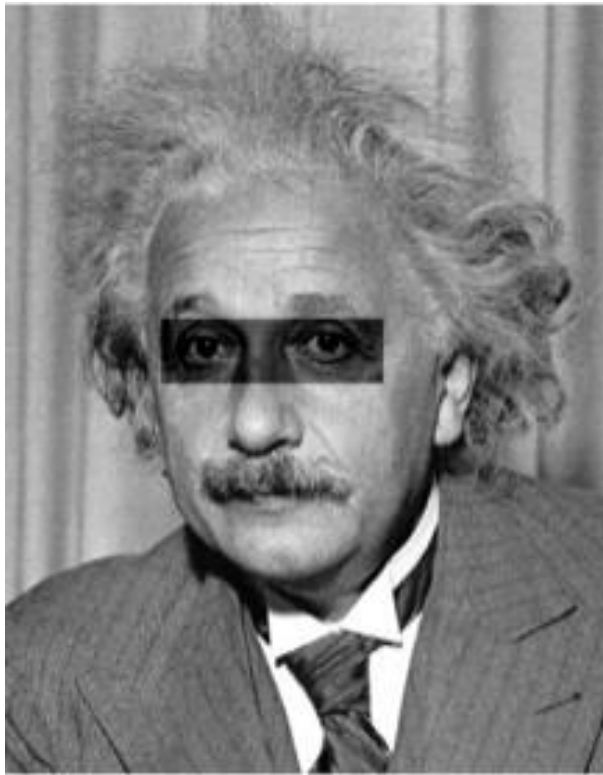


thresholding



# ZNCC – zero mean normalized cross correlation

How do we detect the template  in the following image?



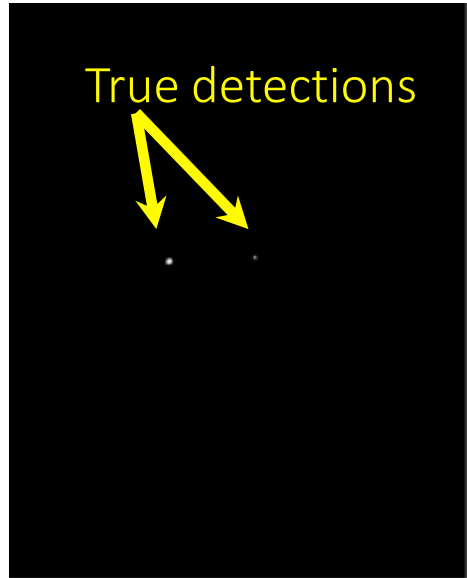
1-output



thresholding

robust to change in  
intensities

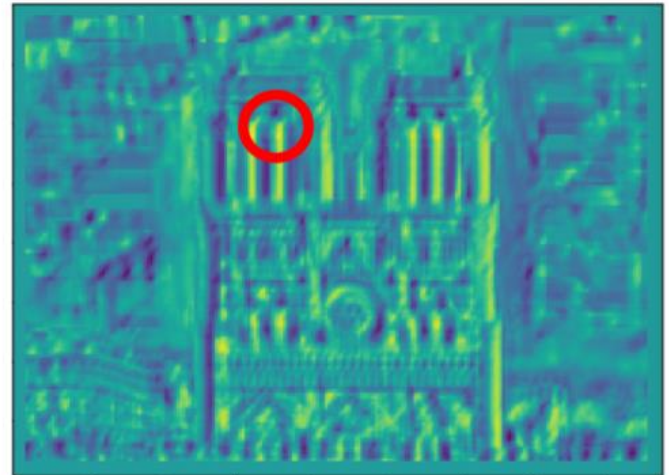
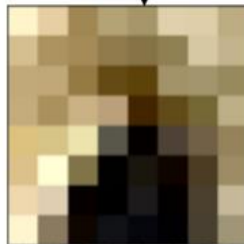
True detections





# Template matching- SSD, ZNCC

- Good for very carefully constructed scenarios.
- Can't handle change in rotation and scale.



# contents

- Image representation
- Pixel-wise operations
- Histogram equalization
- Template matching
- **Morphology operators**
- Connected components
- Color space

# Morphology

- Handy tool whenever needed to clean up binary images.
- Each morphology operator is constructed as such:
  1. Select a structure element (binary kernel)

$$s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

# Morphology

- Handy tool whenever needed to clean up binary images.
- Each morphology operator is constructed as such:
  1. Select a structure element (binary kernel)
  2. Cross-correlate with input binary image  $g = f \star s$

$$s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

# Morphology

- Handy tool whenever needed to clean up binary images.
- Each morphology operator is constructed as such:
  1. Select a structure element (binary kernel)
  2. Cross-correlate with input binary image  $g = f \star s$
  3. Threshold the output

$$s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\theta_{TH}(x, t) = \begin{cases} 1 & \text{if } x \geq t, \\ 0 & \text{else} \end{cases}$$



# Morphology

- Handy tool whenever needed to clean up binary images.

- Each morphology operator is constructed as such:

1. Select a structure element (binary kernel)
2. Cross-correlate with input binary image  $g = f \star s$
3. Threshold the output

$$s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

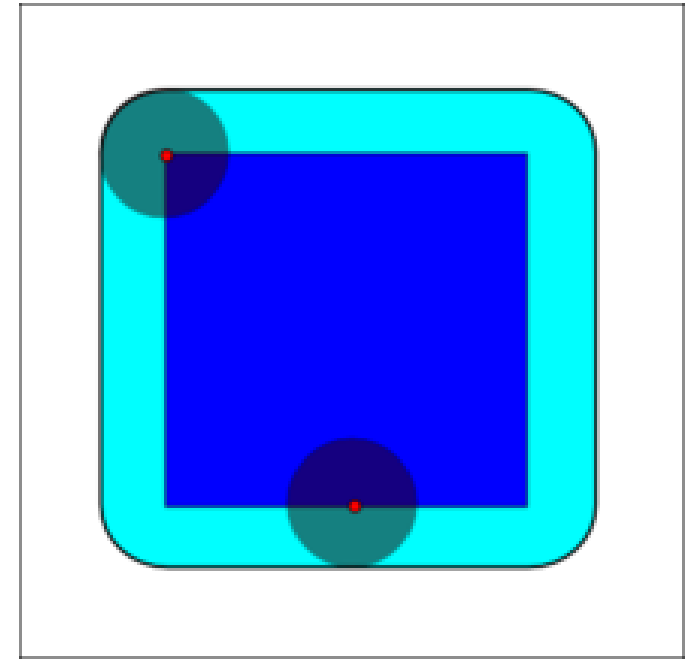
$$\theta_{TH}(x, t) = \begin{cases} 1 & \text{if } x \geq t, \\ 0 & \text{else} \end{cases}$$

- Overall morphologic operation should look like so:

$$k = \theta_{TH}(f \star s, t)$$

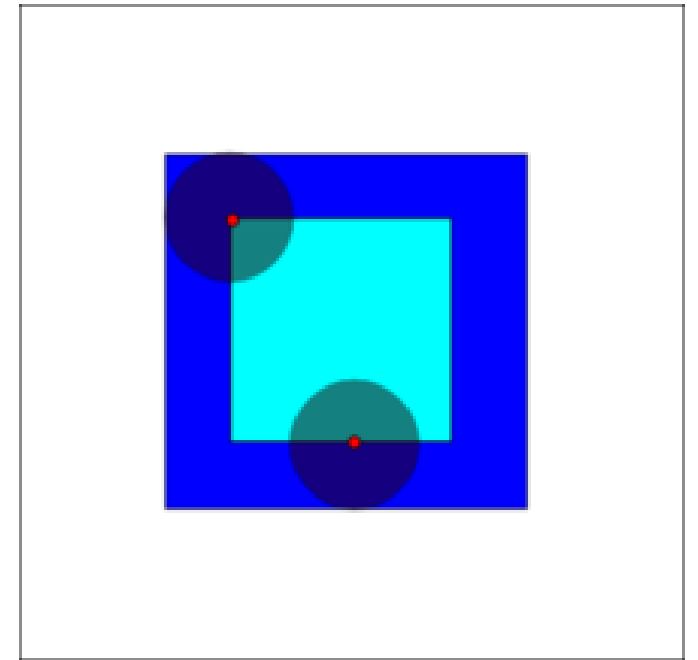
# Dilation

- $k = \theta_{TH}(f \star s, t)$
- Dilation:  $t = 1$ 
  - Skinny lines will get thicker



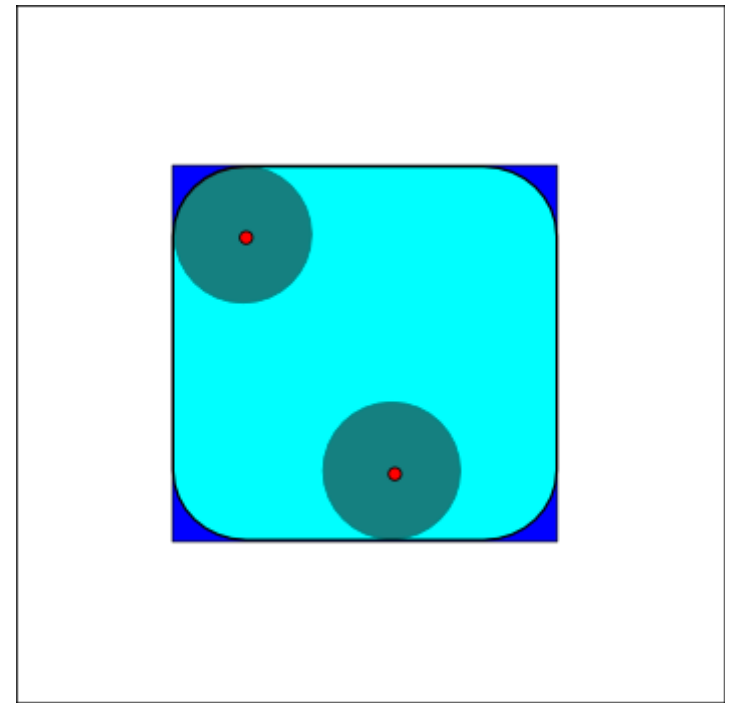
# Erosion

- $k = \theta_{TH}(f \star s, t)$
- Erosion:  $t = \text{sum}(s)$ 
  - Thicker lines will get skinny



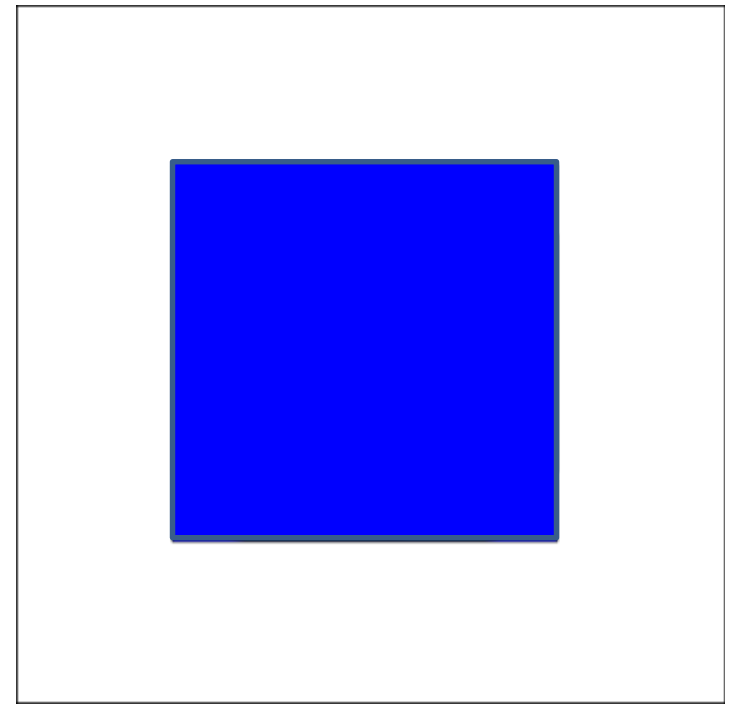
# Opening

- Erosion followed by dilation.
  - The effect is of rounding off sharp edges.



# Closing

- Dilation followed by erosion.
  - The effect is of closing of narrow gaps and holes.





# contents

- Image representation
- Pixel-wise operations
- Histogram equalization
- Template matching
- Morphology operators
- **Connected components**
- Color space

# Connected components

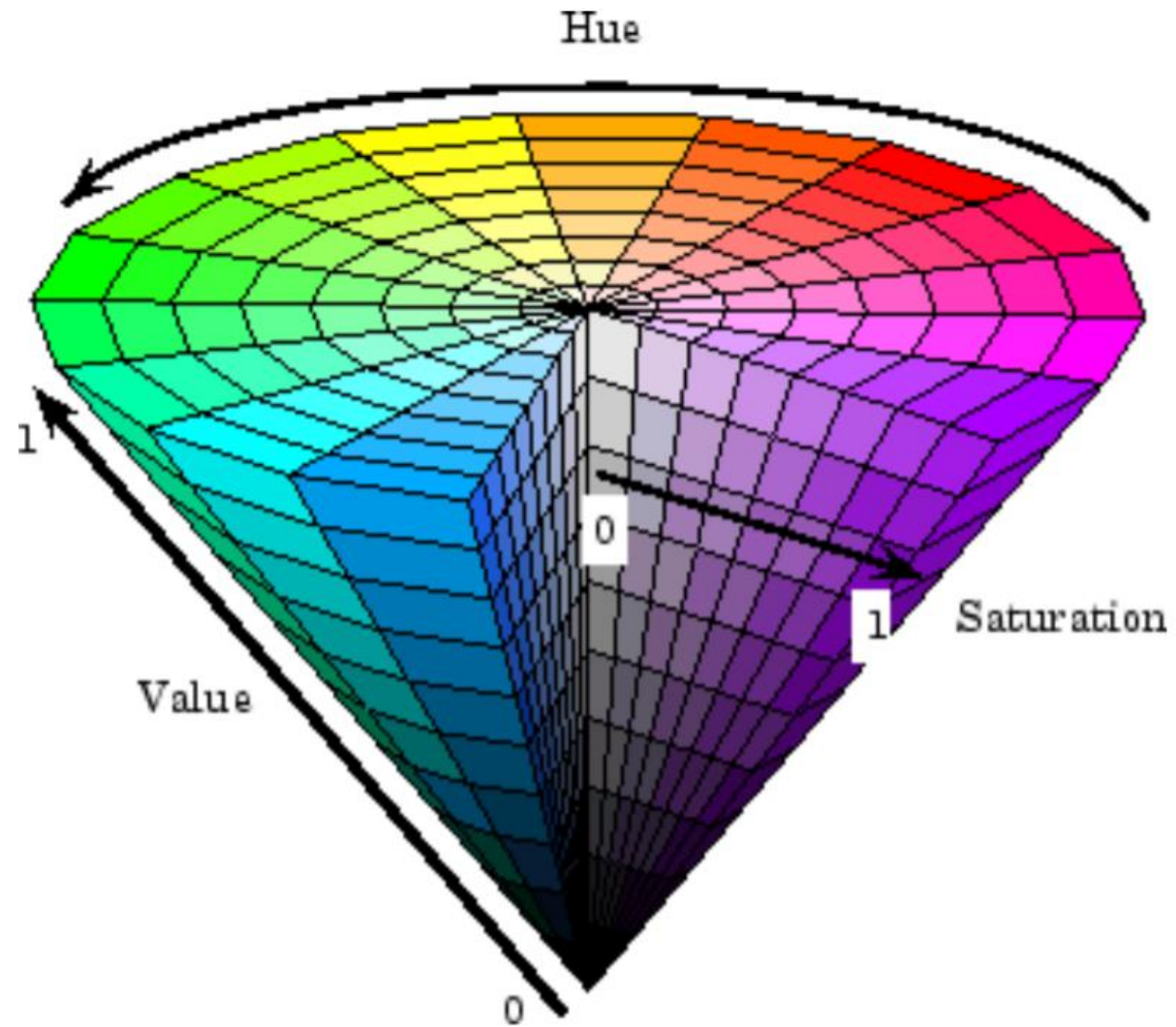
- Defined as regions of adjacent pixels that have the same value.
- Commonly used with binary images to find stand alone objects.
  - e.g.: letters in a document.



# contents

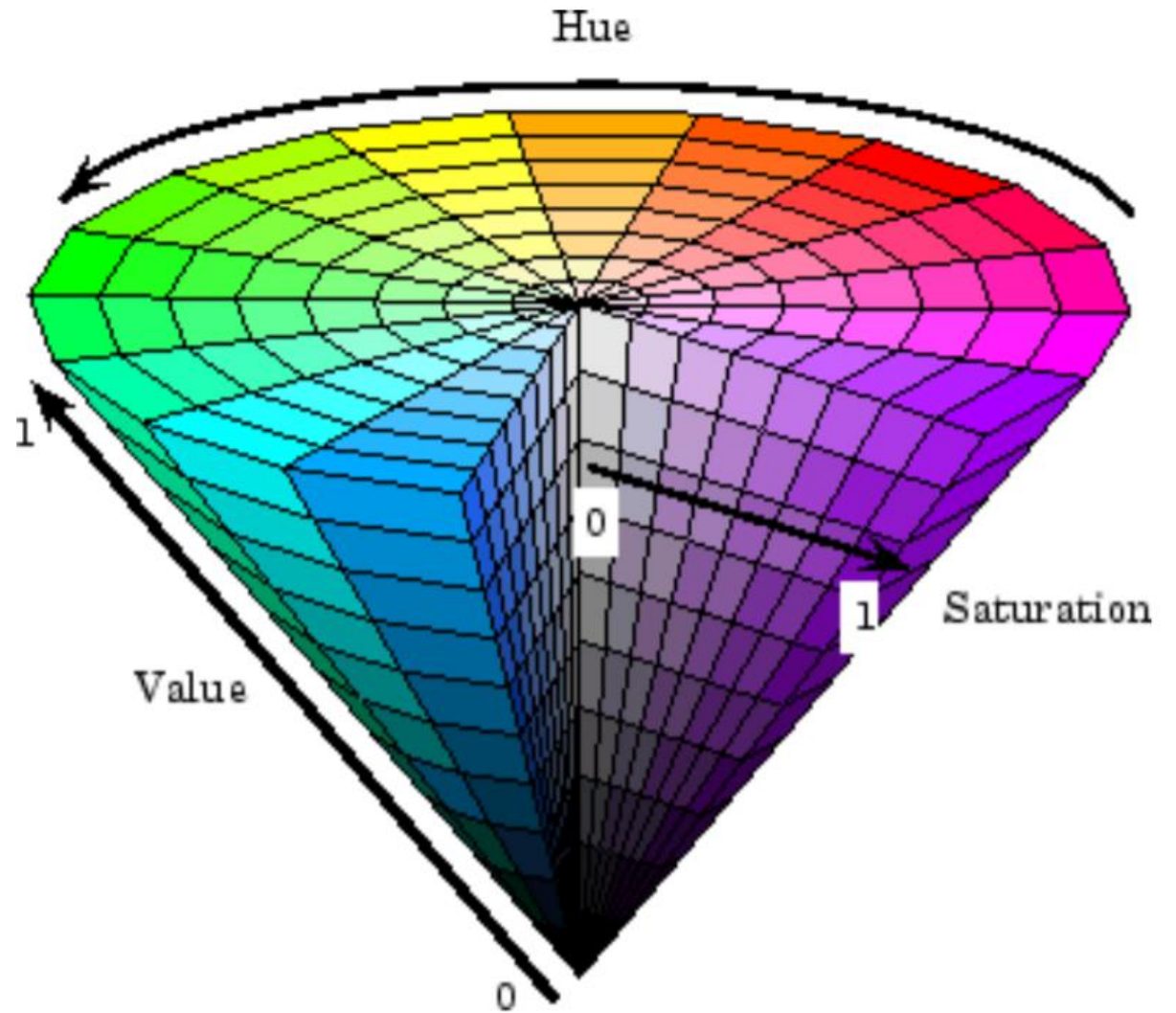
- Image representation
- Pixel-wise operations
- Histogram equalization
- Template matching
- Morphology operators
- Connected components
- **Color space**

# HSV

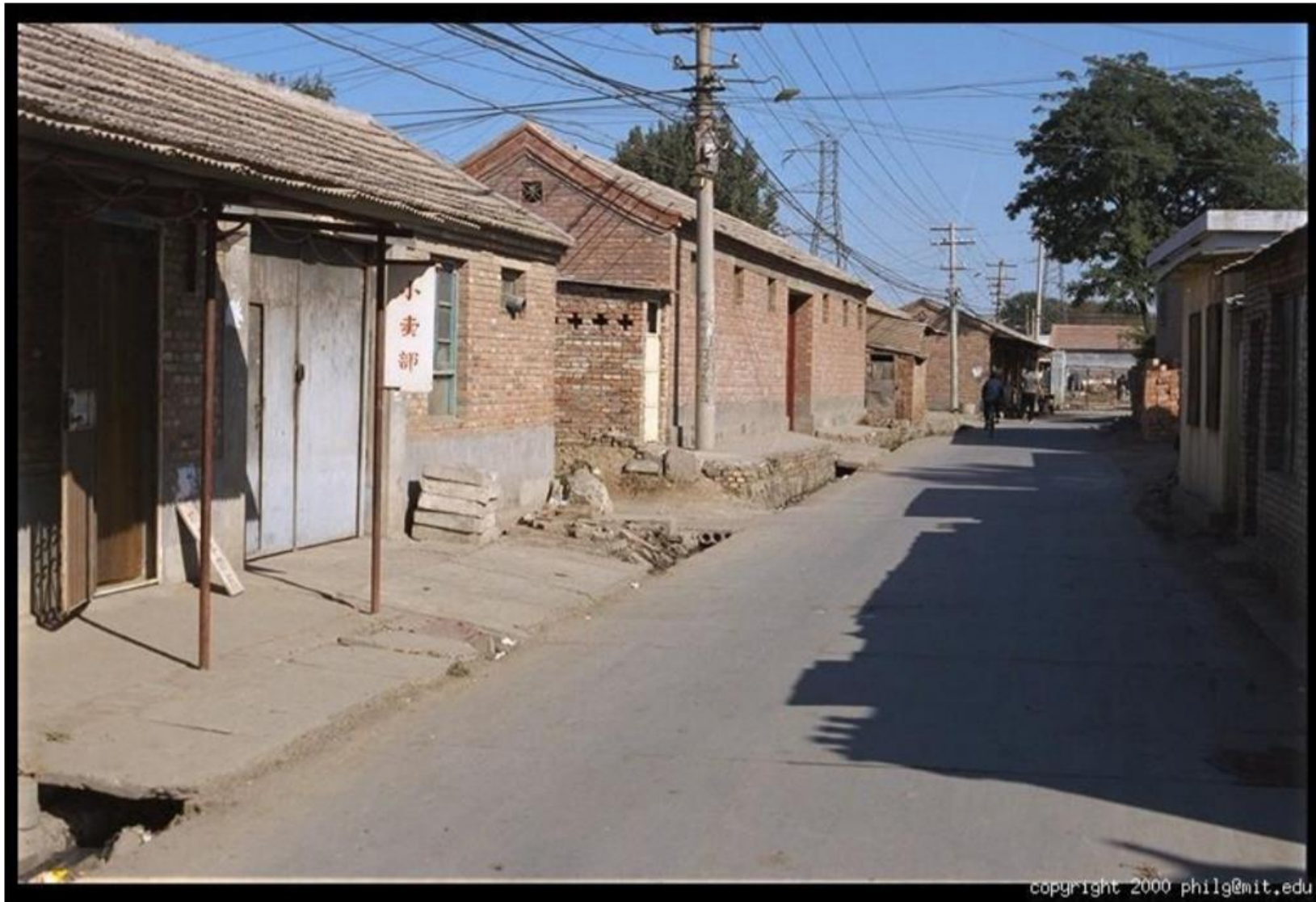


# HSV

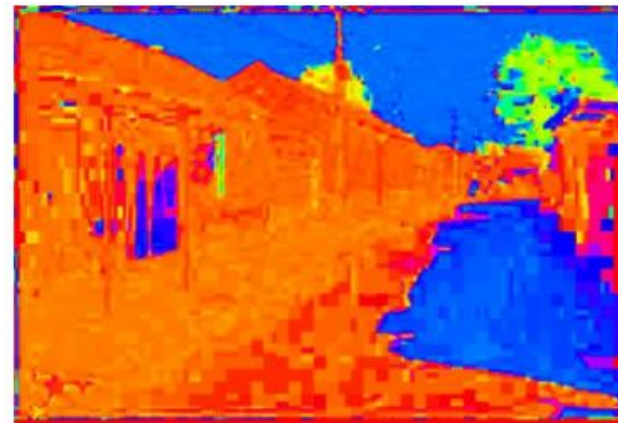
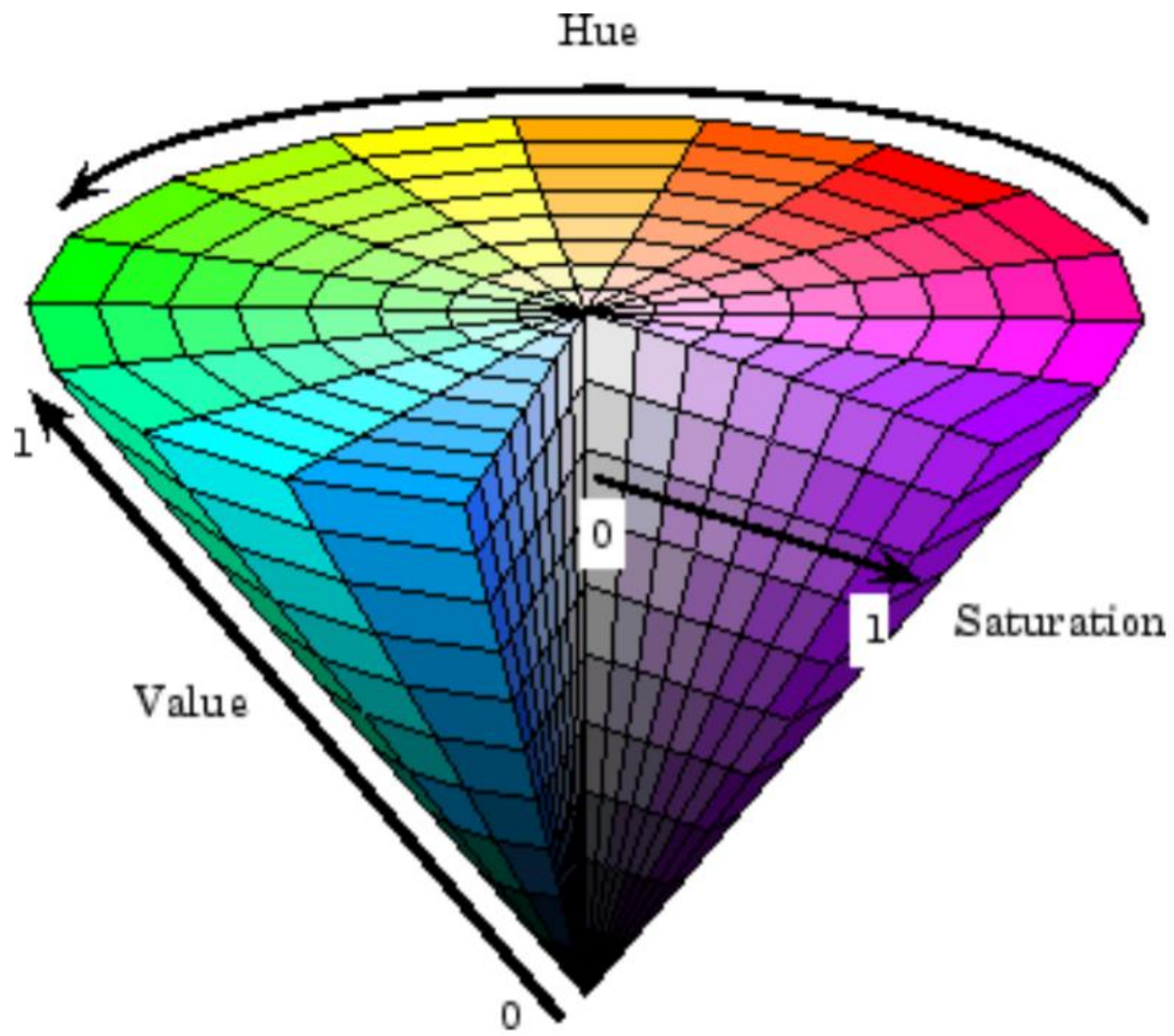
- **Hue:** The "attribute of a visual sensation according to which an area appears to be similar to one of the perceived colors: red, yellow, green, and blue, or to a combination of two of them"
- **Saturation:** The "colorfulness of a stimulus relative to its own brightness"
- **Value:** The "brightness relative to the brightness of a similarly illuminated white". Can also be called **brightness or intensity**.
  - [Wikipedia]







Original image



**H**  
(S=1,V=1)



**S**  
(H=1,V=1)

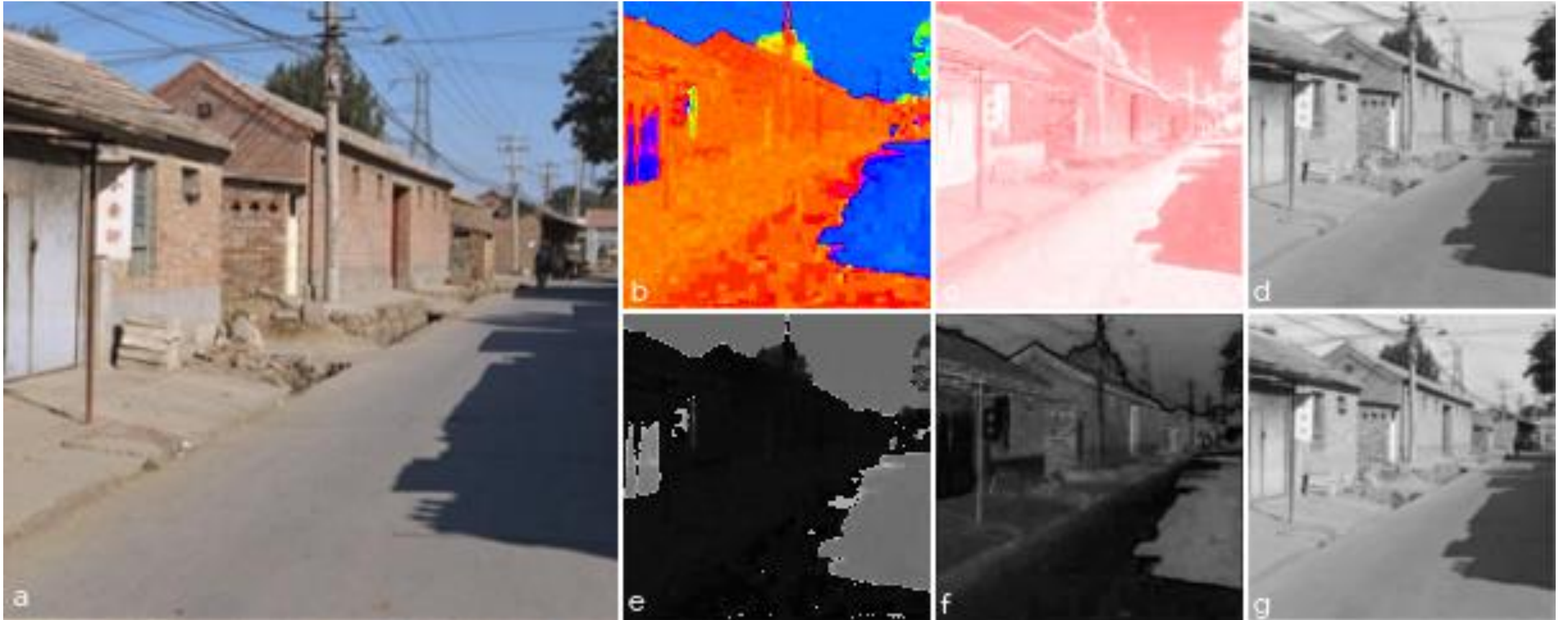


**V**  
(H=1,S=0)



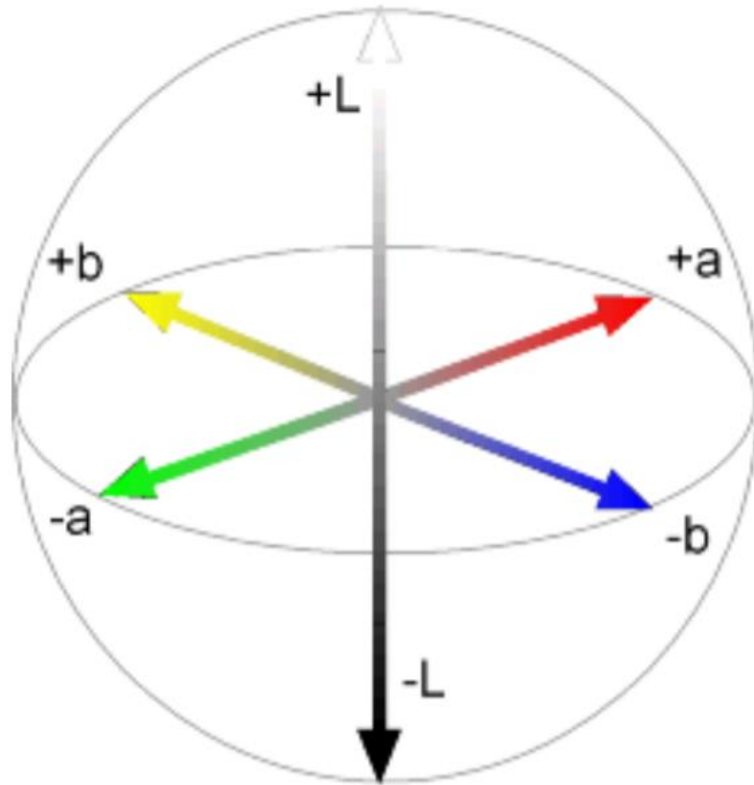
# HSV

- In e, f, g: single channel image representation.
- Conclusion: people are much more responsive to intensity than chroma.



# More color spaces: LAB

- L: lightness from black (0) to white (100).
- A: from green (−) to red (+).
- B: from blue (−) to yellow (+).



**L**  
(a=0,b=0)



**a**  
(L=65,b=0)



**b**  
(L=65,a=0)

# More color spaces: YUV

- Y: brightness/ intensity.
- U: blue projection.
- V: red projection.
- [Similar to YCbCr]

