

# Basic image processing



Yoni Chechik

[www.AlisMath.com](http://www.AlisMath.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



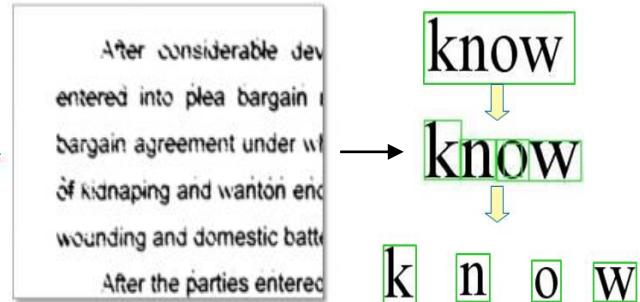
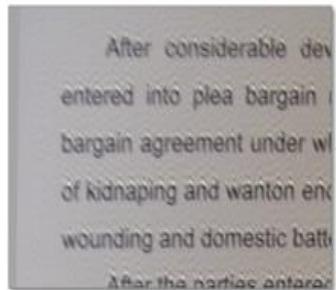
Contrast stretching



Histogram equalization



Adaptive equalization



Robotics  
(OCR – optical character recognition)



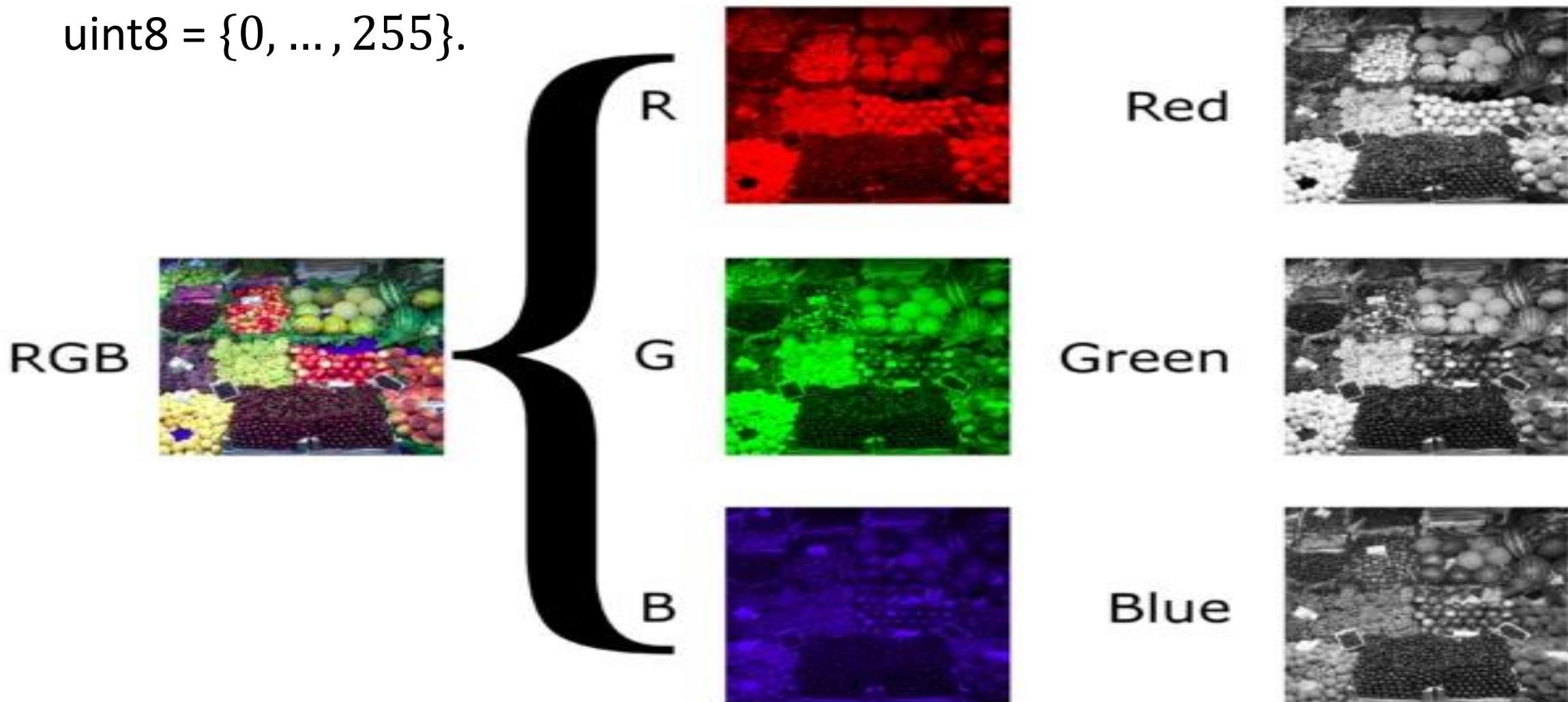
Agriculture  
(color ripeness detection)

# contents

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

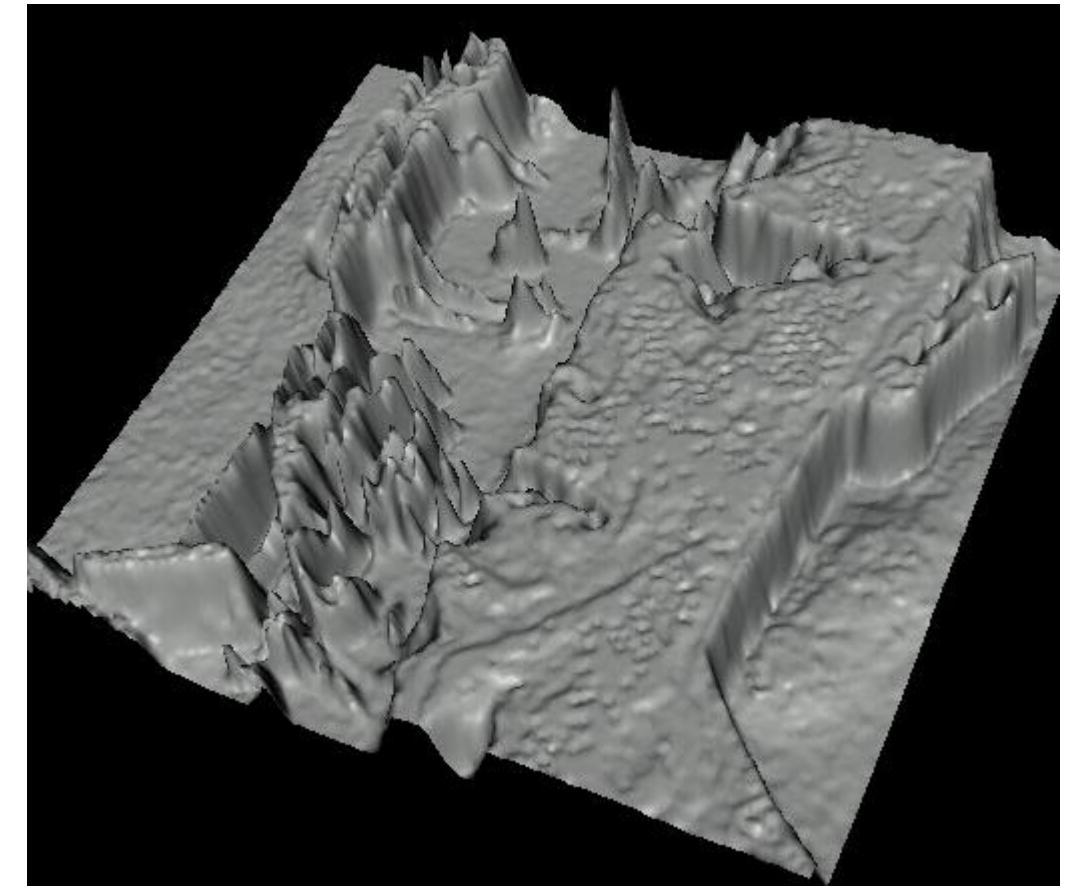
# Image representation

- 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  $\text{uint8} = \{0, \dots, 255\}$ .



# 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



darken



lower contrast



Gamma compression

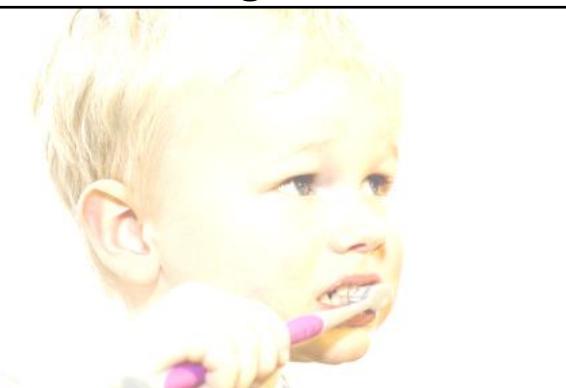


$x$

invert



lighten



raise contrast



Gamma expansion



# Pixel-wise operators

original



darken



lower contrast



Gamma compression

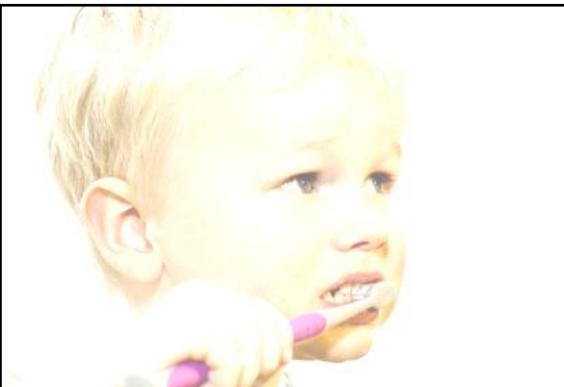


$x$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



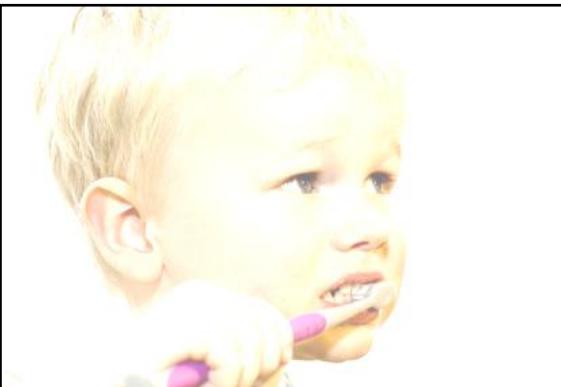
$x$

$x - 128$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



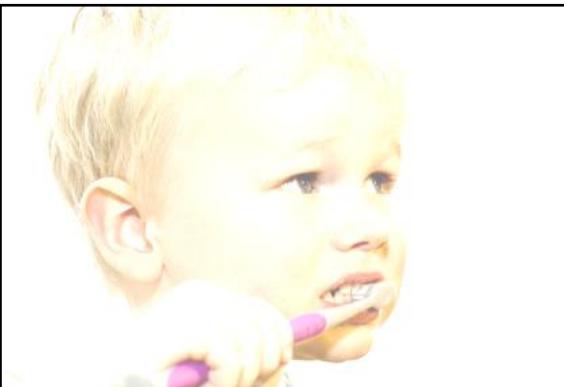
$x$

$x - 128$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

$x + 128$

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



$x$

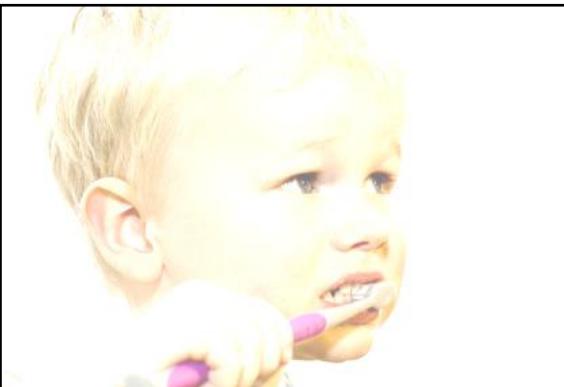
$x - 128$

$\frac{x}{2}$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

$x + 128$

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



$x$

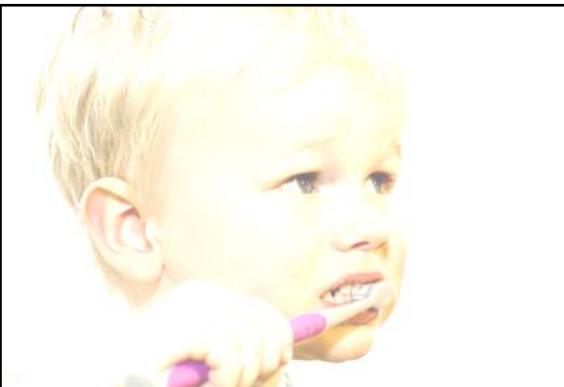
$x - 128$

$\frac{x}{2}$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

$x + 128$

$x \times 2$

# 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:

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



# Contrast

- Example of calculating contrast ratio in determining website accessibility:
  - <https://contrast-ratio.com/#%23000000-on-white>
  - <https://www.accessibility-developer-guide.com/knowledge/colours-and-contrast/how-to-calculate/>

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



$x$

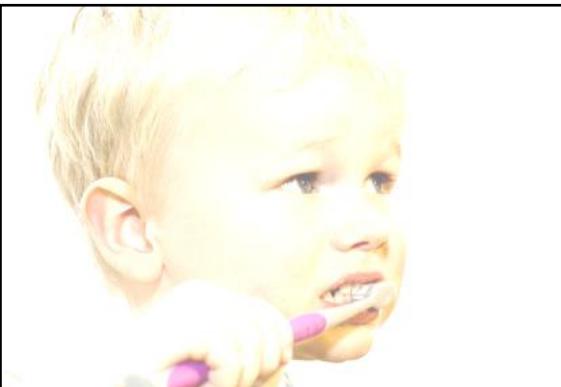
$x - 128$

$\frac{x}{2}$

invert



lighten



raise contrast



Gamma expansion



$255 - x$

$x + 128$

$x \times 2$

# Pixel-wise operators

original



darken



lower contrast



Gamma compression



$$x$$

$$x - 128$$

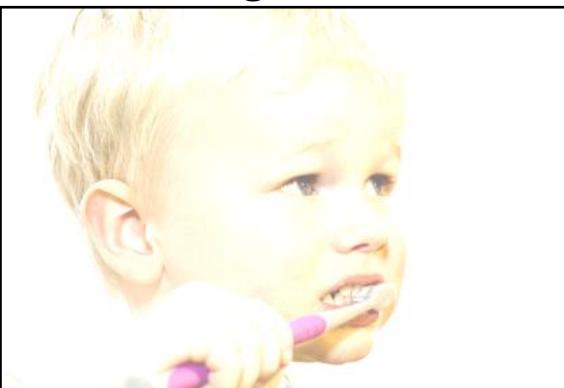
$$\frac{x}{2}$$

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

invert



lighten



raise contrast



Gamma expansion



$$255 - x$$

$$x + 128$$

$$x \times 2$$

# Pixel-wise operators

original



$$x$$

darken



$$x - 128$$

lower contrast



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

Gamma compression



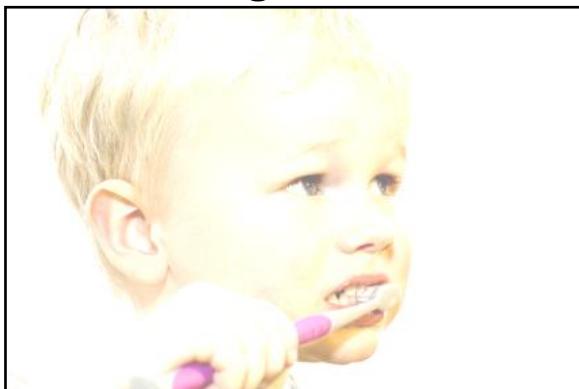
$$\left(\frac{x}{255}\right)^{1/3} \times 255$$

invert



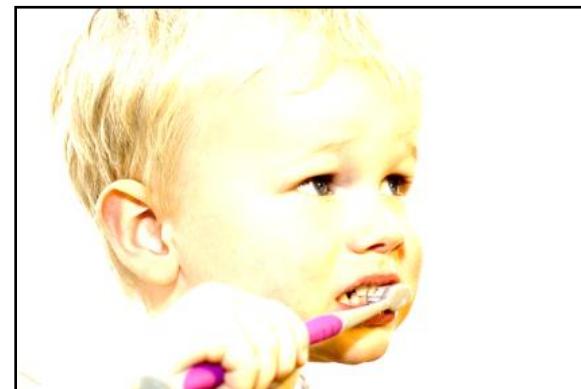
$$255 - x$$

lighten



$$x + 128$$

raise contrast



$$x \times 2$$

Gamma expansion



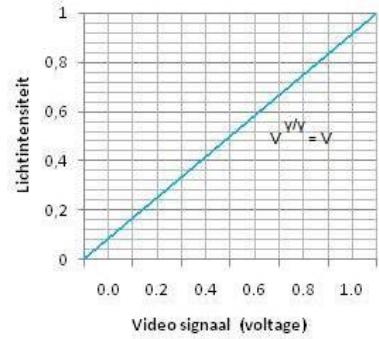
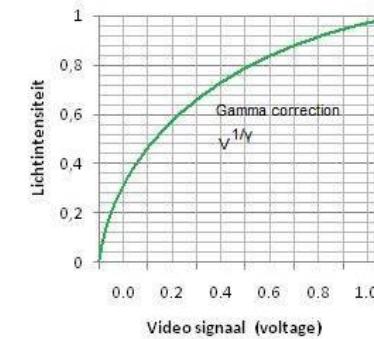
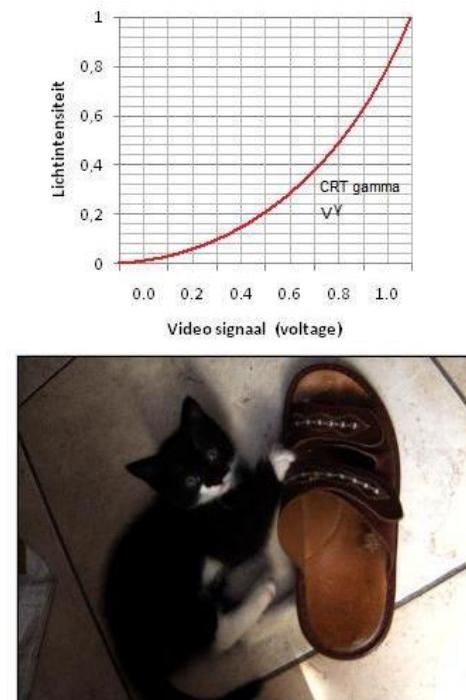
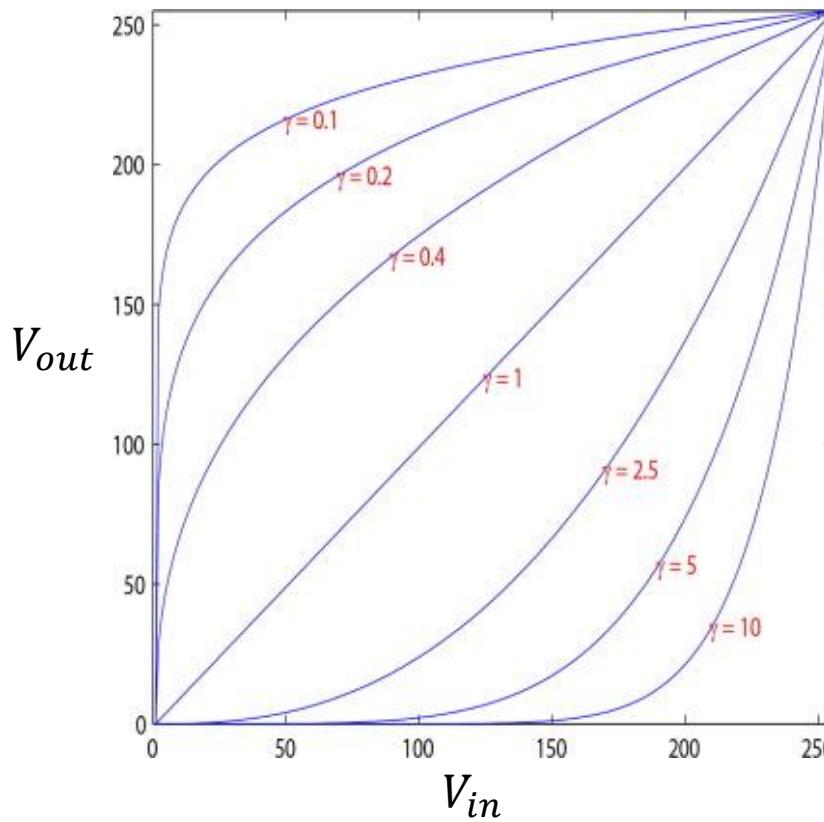
$$\left(\frac{x}{255}\right)^2 \times 255$$

# Gamma correction

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

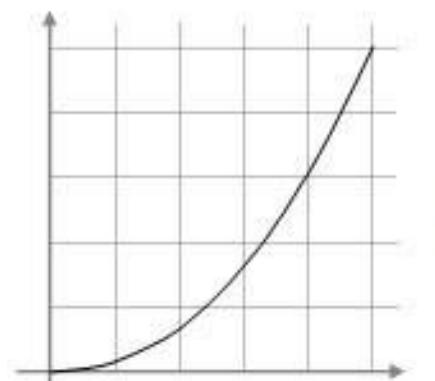
$$V_{out} = \left( \frac{V_{in}}{255} \right)^\gamma \cdot 255 \quad (V_{in}, V_{out} \in \{0, 1, \dots, 255\})$$

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

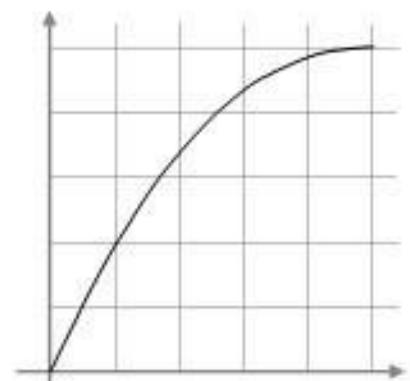


# Gamma correction

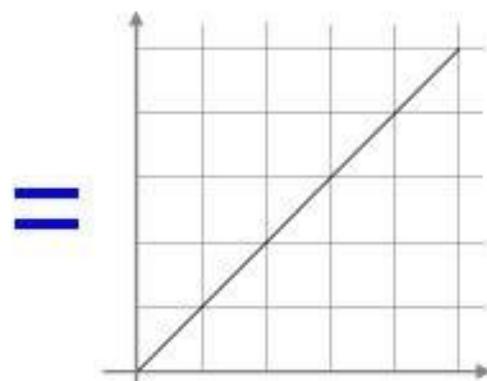
- Originally, Due to non-linearities in the old CRT televisions, intensities was seen different then they are.



Gamma characteristics  
of monitors



Color information adjusted  
to match gamma characteristics



Color handling approaching  
the " $y = x$ " ideals



$\gamma=1/4$

# Some more point- wise operators



# contents

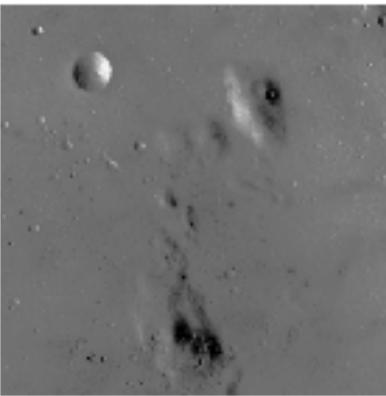
- Image representation
- Pixel-wise operations
- **Histogram equalization**
- Template matching
- Morphology operators
- Connected components
- Color space

# 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.**



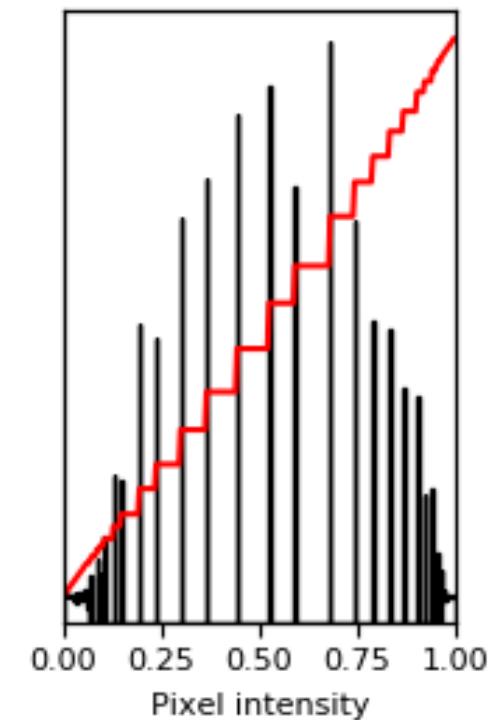
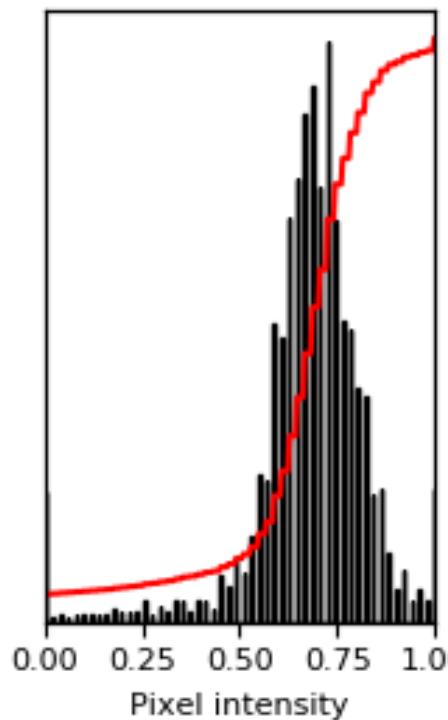
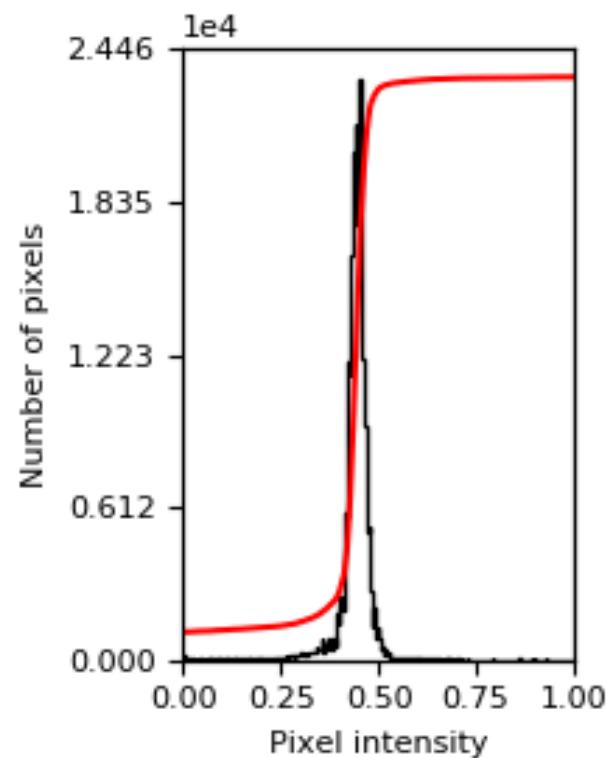
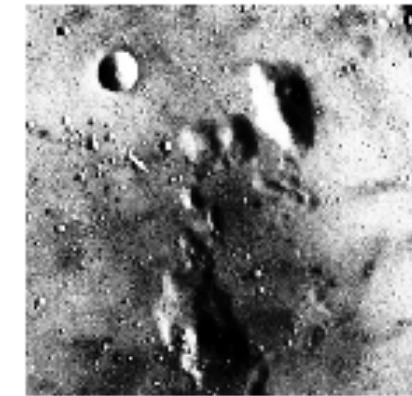
Low contrast image



Contrast stretching

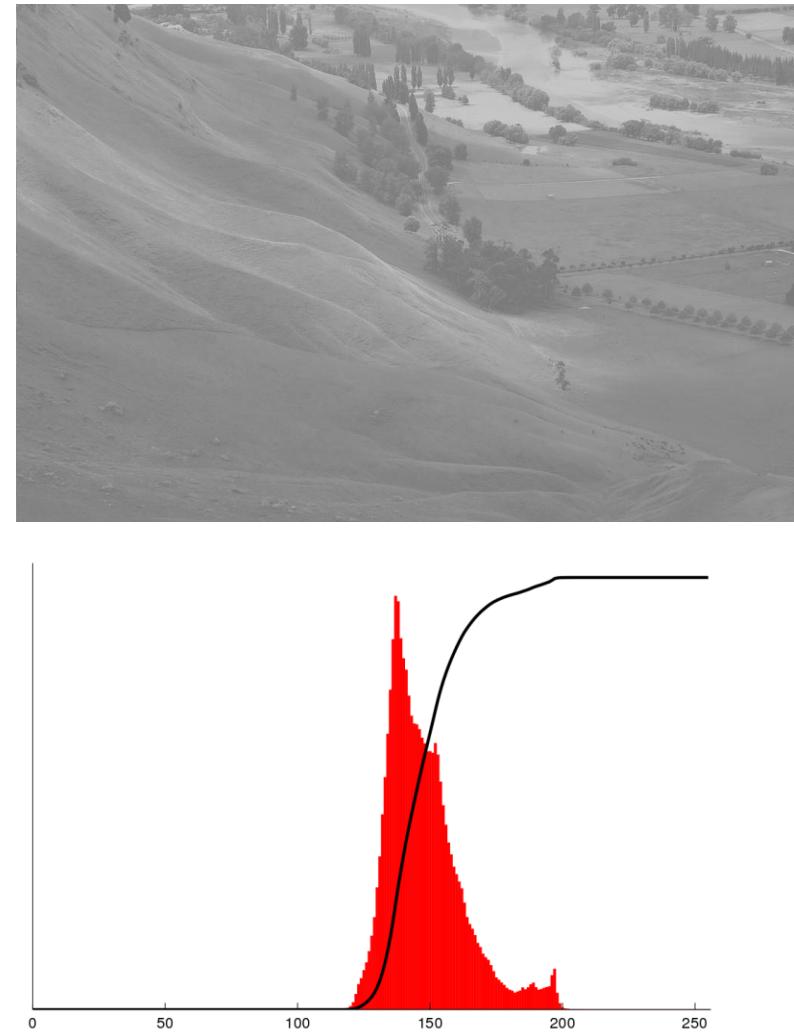


Histogram equalization



# 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.



# Reminder: 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:
  - $\lim_{x \rightarrow -\infty} F_X(x) = 0$ ,  $\lim_{x \rightarrow +\infty} F_X(x) = 1$
  - Monotonically non decreasing.
- The **probability density function (PDF)** of a continuous random variable can be determined from the cumulative distribution function by differentiating.

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

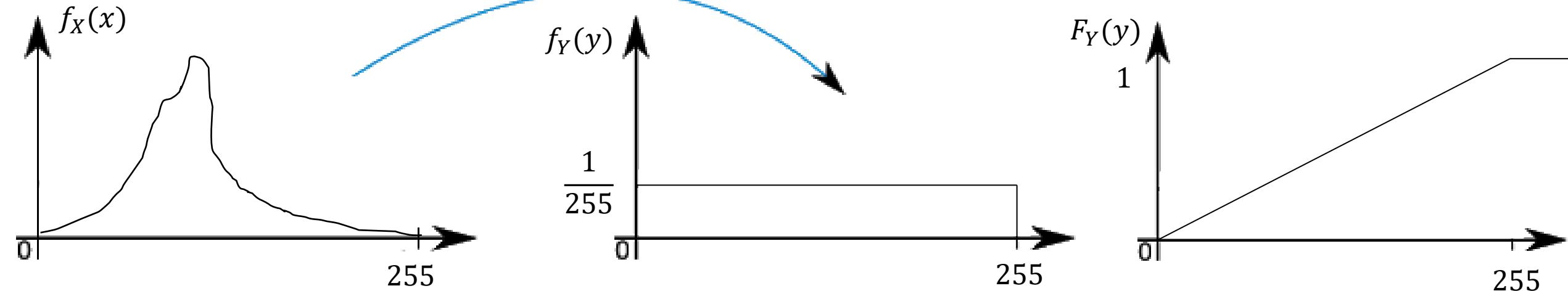
# PDF definition- Wikipedia

- **Probability density function (PDF)** can be interpreted as providing a "relative likelihood" that the value of the random variable would equal that sample.
  - The *absolute likelihood* for a continuous random variable to take on any particular value is 0 (since there are an infinite set of possible values to begin with).
- In a more precise sense, the PDF is used to specify the probability of the random variable falling within a particular range of values. This probability is given by the integral of this variable's PDF over that range and is actually the **CDF**.

# PDF equalization

- We want that our resulting PDF [ $f_Y(y)$ ] 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] (integration of constant is a linear function), and so we get the final CDF as:

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



# PDF equalization

- So we are looking for a transformation function of the random variable X to Y such that:

$$Y = T(X)$$

- In the interesting area [0,255]:

$$P(Y \leq y) = \frac{y}{255}$$

# PDF equalization

- So we are looking for a transformation function of the random variable X to Y such that:

$$Y = T(X)$$

- In the interesting area [0,255]:

$$P(Y \leq y) = \frac{y}{255}$$

$$255 \cdot P(T(X) \leq y) = y$$

# PDF equalization

- So we are looking for a transformation function of the random variable X to Y such that:

$$Y = T(X)$$

- In the interesting area [0,255]:

$$P(Y \leq y) = \frac{y}{255}$$

$$255 \cdot P(T(X) \leq y) = y$$

$$255 \cdot P(X \leq T^{-1}(y)) = y \quad (\text{assuming } T \text{ is invertible})$$

# PDF equalization

- So we are looking for a transformation function of the random variable X to Y such that:

$$Y = T(X)$$

- In the interesting area [0,255]:

$$P(Y \leq y) = \frac{y}{255}$$

$$255 \cdot P(T(X) \leq y) = y$$

$$255 \cdot P(X \leq T^{-1}(y)) = y \quad (\text{assuming } T \text{ is invertible})$$

$$255 \cdot P(X \leq z) = T(z) \quad (\text{change of variables } z = T^{-1}(y))$$

# PDF equalization

- So we are looking for a transformation function of the random variable  $X$  to  $Y$  such that:

$$Y = T(X)$$

- In the interesting area  $[0,255]$ :

$$P(Y \leq y) = \frac{y}{255}$$

$$255 \cdot P(T(X) \leq y) = y$$

$$255 \cdot P(X \leq T^{-1}(y)) = y \quad (\text{assuming } T \text{ is invertible})$$

$$255 \cdot P(X \leq z) = T(z) \quad (\text{change of variables } z = T^{-1}(y))$$

$$T(x) = F_X(x) \cdot 255$$

- In fact  $T$  is invertible since  $F_X$  is Monotonically non decreasing.

# Back to histogram equalization

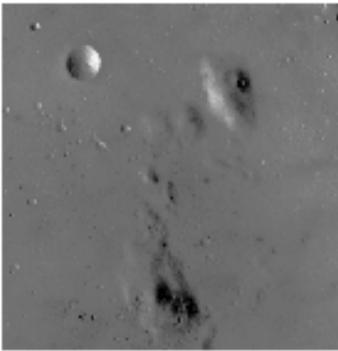
- The same result is also applicable for discrete space like actual images and their histograms.

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

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

# Other variants of histogram equalization

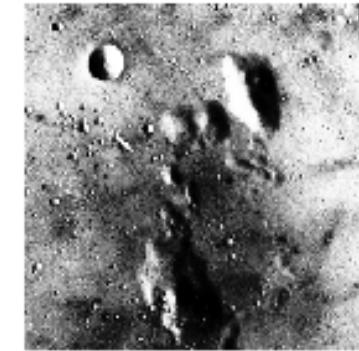
Low contrast image



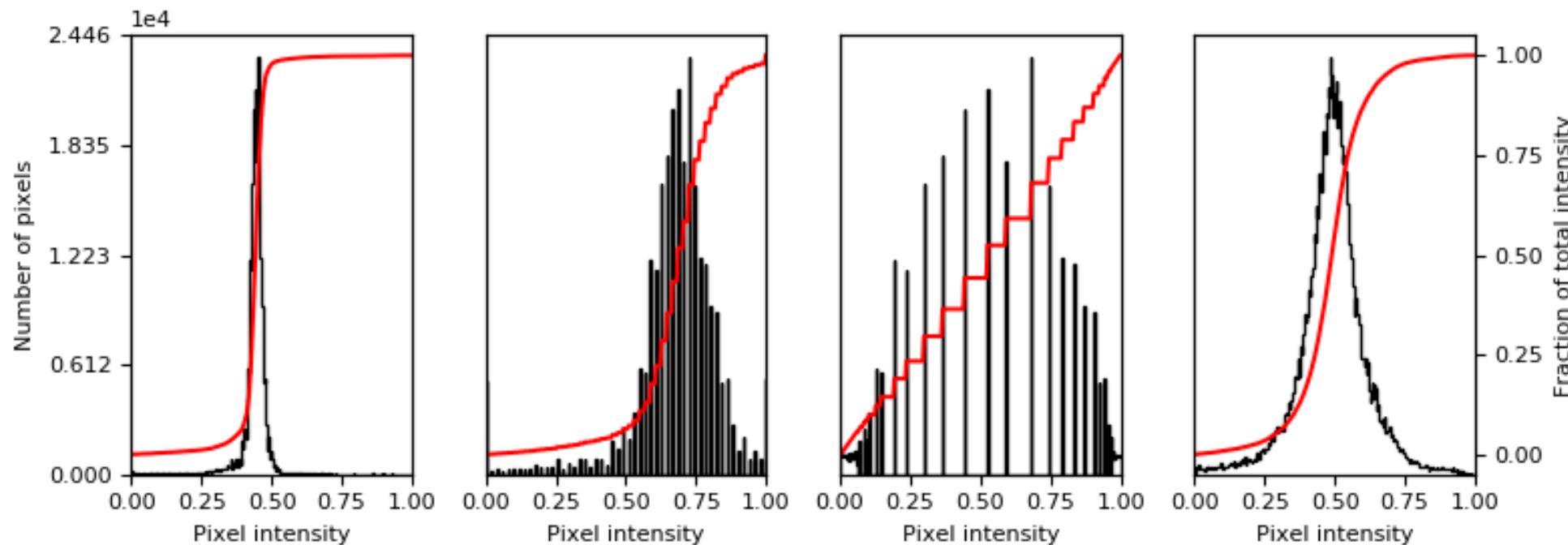
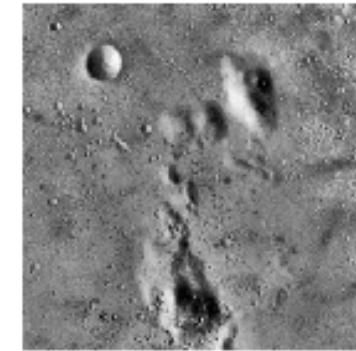
Contrast stretching



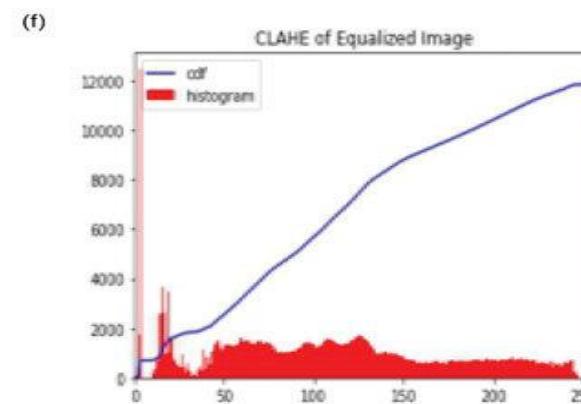
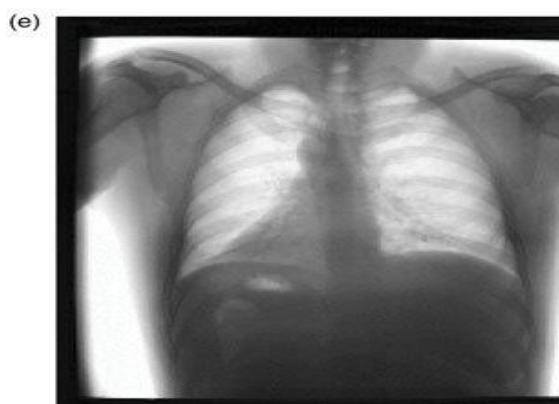
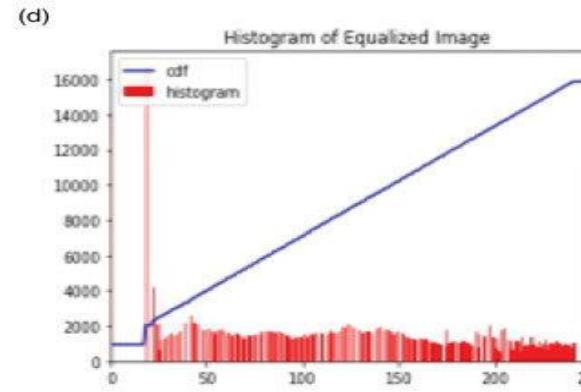
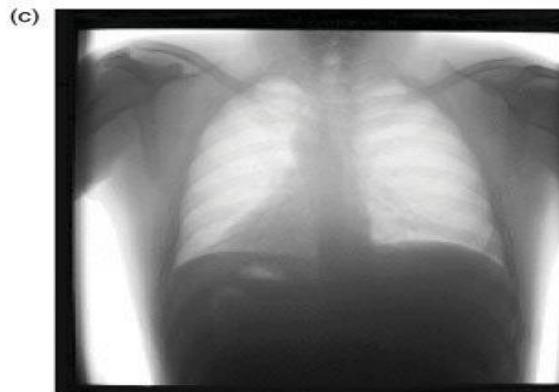
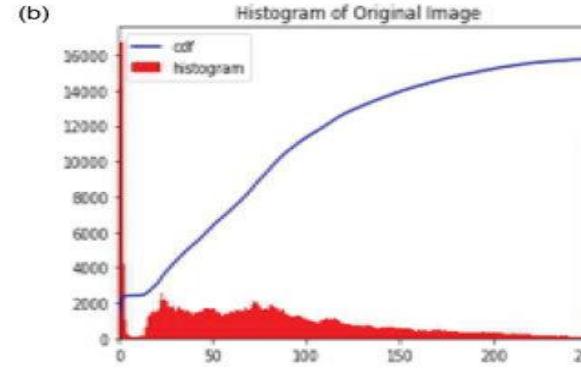
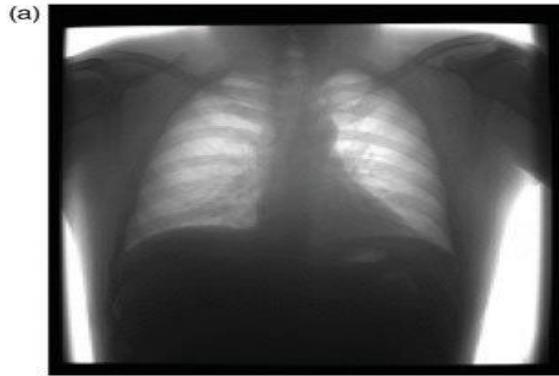
Histogram equalization



Adaptive equalization



# Other variants of histogram equalization

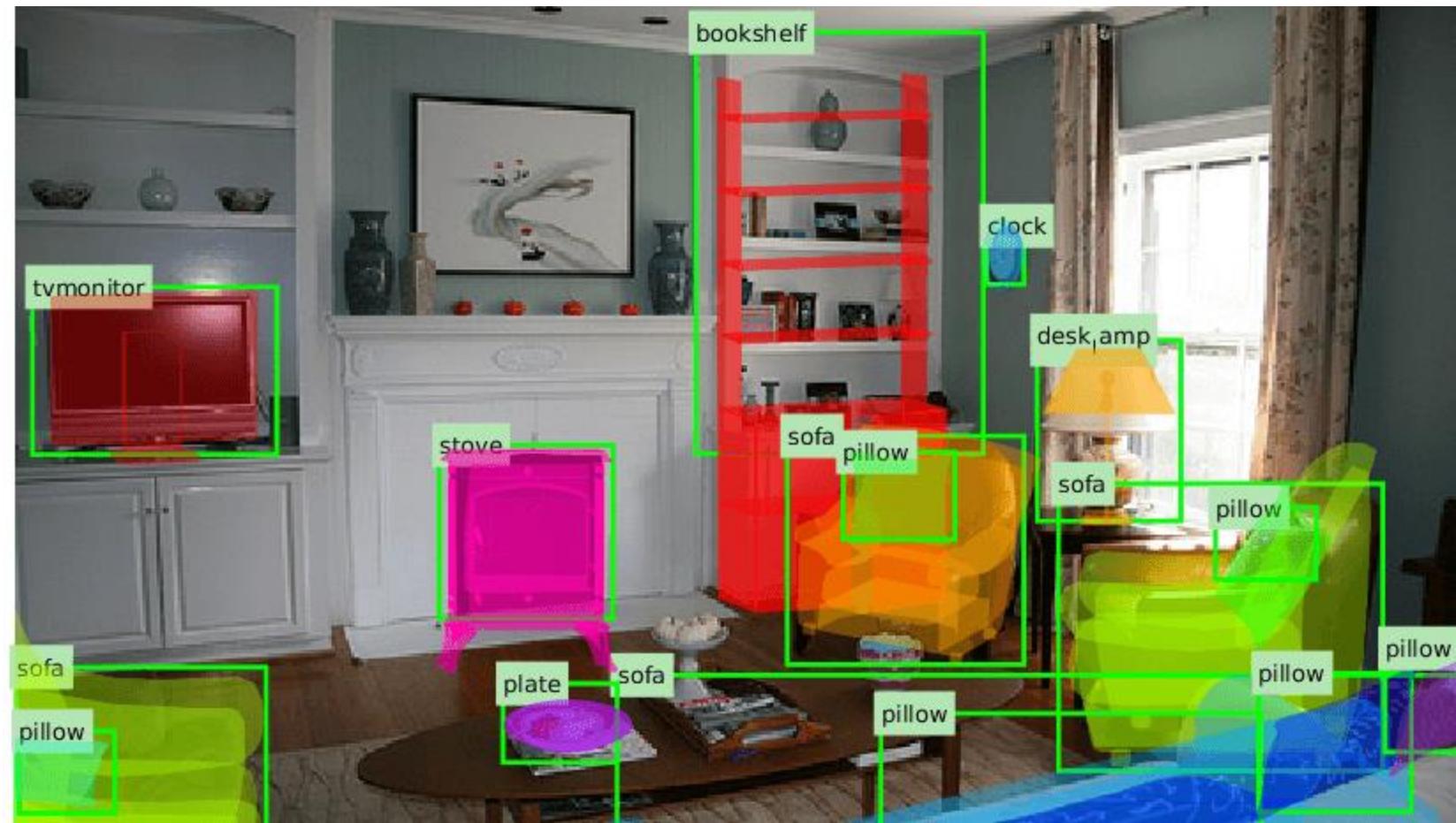


# contents

- 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:
    - Cross correlation
    - Feature based – SIFT
    - Neural networks



# Example output

- Run pixel-by-pixel for the entire image to look for a match to the template



# First- let's understand what is cross-correlation

$$\begin{matrix} \text{kernel} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{matrix}$$

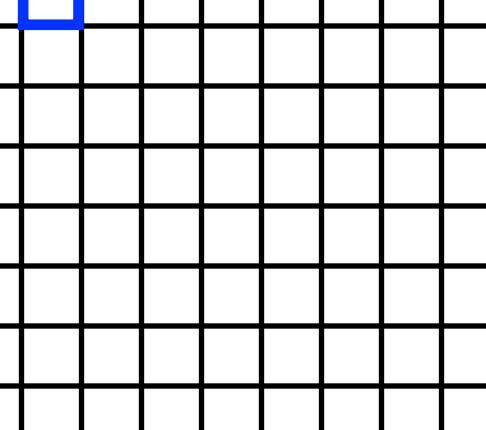
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

g

## image

f

## output

A 10x10 grid of black lines on a white background. The top-left square, located at the intersection of the first vertical and horizontal lines, is filled with blue.

h

# Run the filter

$$\begin{matrix} \text{kernel} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{matrix}$$

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

g

f

output

A 10x10 grid of empty cells. The cell at the top-left corner (row 1, column 1) is highlighted with a blue border.

h

# Run the filter

kernel

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$f$

output

0										

$h$

shift

# Run the filter

$$\begin{matrix} \text{kernel} \\ \hline \begin{matrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{matrix} \\ \hline g \end{matrix}$$

# Run the filter

kernel

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$f$

output

0	10	20								

$h$

# Run the filter

kernel

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$f$

output

0	10	20	30							

$h$

# Run the filter

kernel

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$f$

output

0	10	20	30	30					

$h$

# Run the filter

kernel

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 0 & 90 & 0 & 90 & 90 & 90 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$f$

output

$$\begin{bmatrix} & & & & & & \\ 0 & 10 & 20 & 30 & 30 & 30 & \\ & & & & & & \end{bmatrix}$$

$h$

# ... and the result is

kernel

$$g = \begin{bmatrix} 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$g$

image

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$f$

output

0	10	20	30	30	30	30	20	10	
0	20	40	60	60	60	60	40	20	
0	30	50	80	80	90	60	30		
0	30	50	80	80	90	60	30		
0	20	30	50	50	60	40	20		
0	10	20	30	30	30	20	10		
10	10	10	10	0	0	0	0		
10	10	10	10	0	0	0	0		

$h$

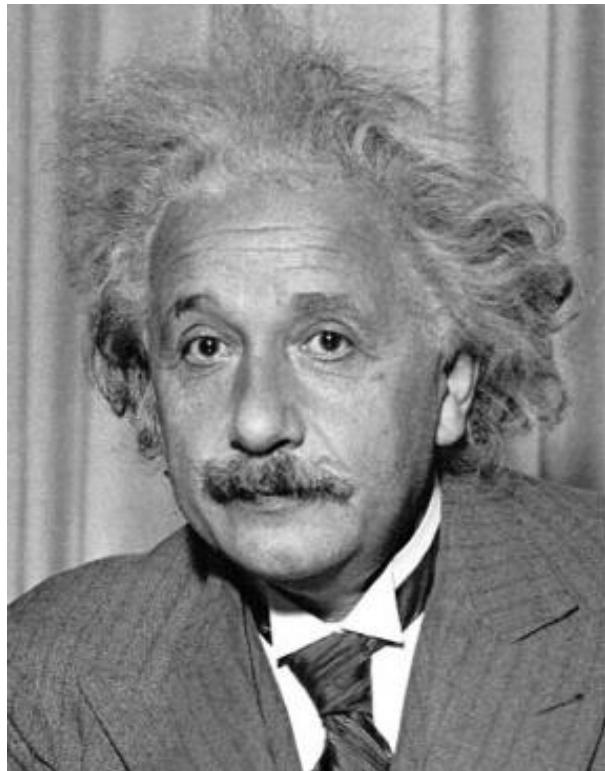
Cross correlation can also be more simply denoted as  $h = g \star f$

The full mathematical notation is this:

$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l] f[m + k, n + l]$$

# CC – cross correlation

Can cross-correlation be good for template matching? Let's take our template  that we want to find as our kernel



output

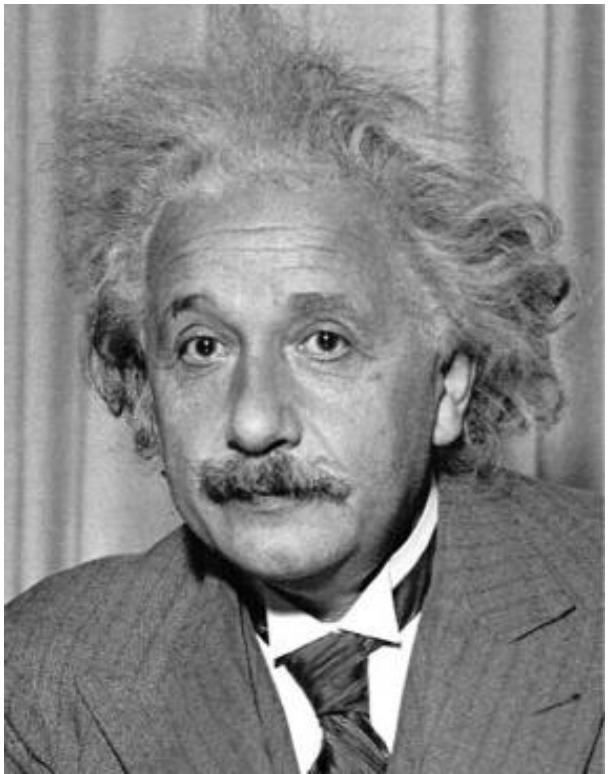
$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l] f[m + k, n + l]$$

image

A diagram illustrating the cross-correlation process. A small square containing an eye icon is positioned above a downward-pointing arrow. This arrow points to the summation formula. A long, thin arrow originates from the bottom-left and points towards the word "image" at the bottom left.

What will  
the output  
look like?

# CC – cross correlation



output

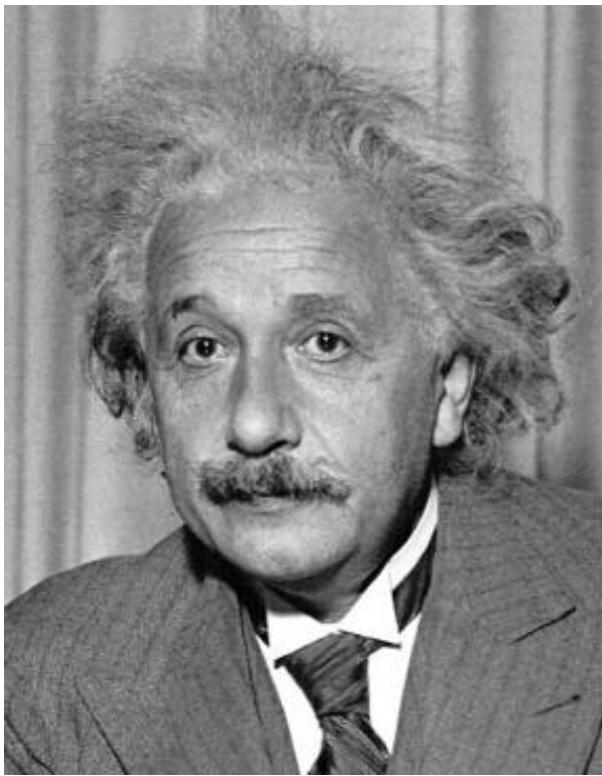
$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l]f[m + k, n + l]$$

image

A diagram illustrating the cross-correlation process. A small square containing an eye icon is positioned above a downward-pointing arrow. A long, thin arrow originates from the bottom left and points towards the right, indicating the direction of the output calculation.

What will  
the  
output  
look like?

# CC – cross correlation



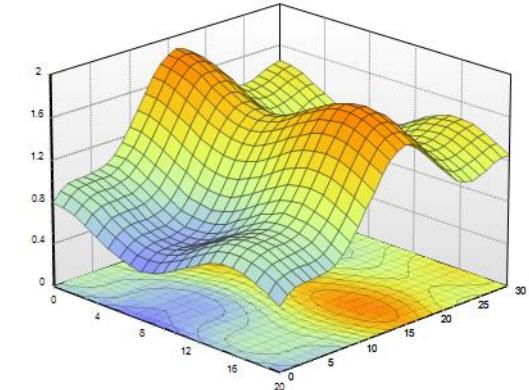
output

$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l] f[m + k, n + l]$$

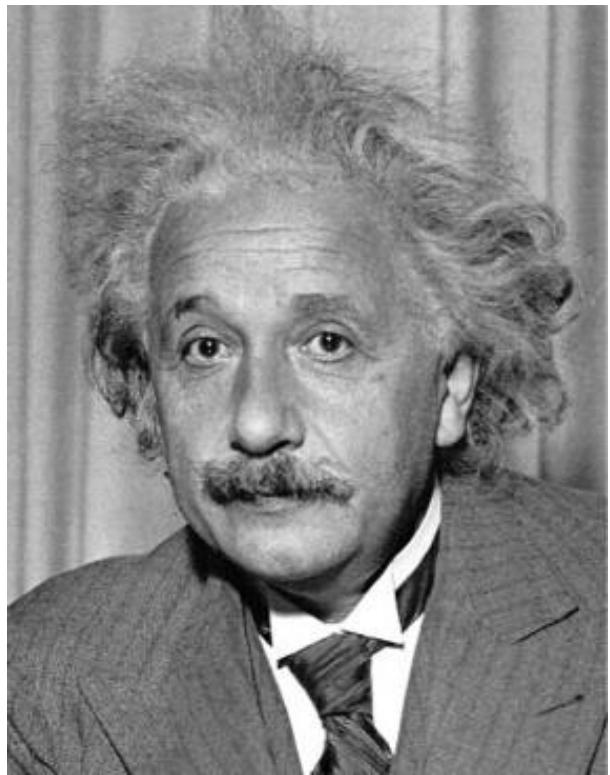
image

A diagram illustrating the cross-correlation process. A small square containing an eye icon is positioned above a downward-pointing arrow. A long diagonal arrow originates from the bottom left and points towards the mathematical equation, indicating that the input image is being processed to produce the output cross-correlation result.

The result will be a 2D heatmap from 0 to a very large number  
 $(\sim N * N * 255^2)$



# CC – cross correlation

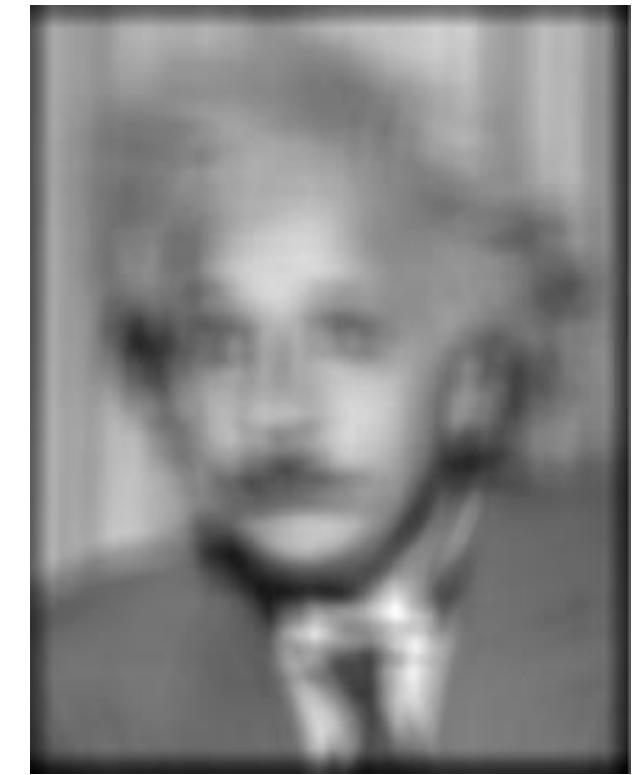


output

$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l] f[m + k, n + l]$$

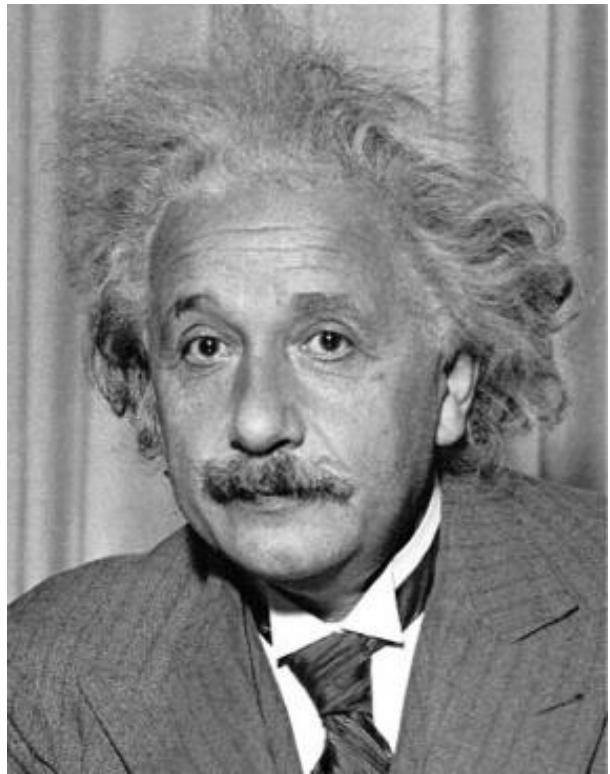
image

A mathematical equation for cross-correlation. It shows the output value  $h[m, n]$  as the sum of the product of two values:  $g[k, l]$  (from the template) and  $f[m + k, n + l]$  (from the image). The summation indices  $k$  and  $l$  both range from -1 to 1. A small diagram above the equation shows a template image with a single eye highlighted by a red box, with an arrow pointing to the term  $g[k, l]$ . Another arrow points from the term  $f[m + k, n + l]$  to the right side of the equation.



Is this good for  
template matching?

# CC – cross correlation

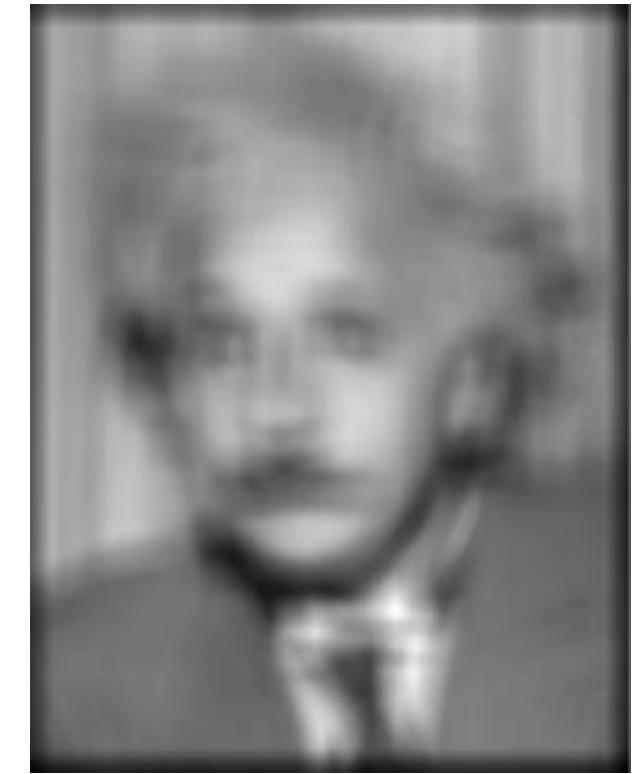


output

$$h[m, n] = \sum_{k=-1}^1 \sum_{l=-1}^1 g[k, l]f[m + k, n + l]$$

image

A mathematical equation for cross-correlation. It shows the output value  $h[m, n]$  as the sum of the product of two images. The first image,  $f$ , is the input image, represented by a right-pointing arrow labeled "image". The second image,  $g$ , is the kernel or template, represented by a downward-pointing arrow labeled "output". The summation indices  $k$  and  $l$  both range from -1 to 1, indicating a 3x3 neighborhood centered at the pixel  $(m, n)$ .



Increases for higher local intensities.

**CC**

255	255	255
255	0	255
255	255	255

*g*



255	255	255
255	255	255
255	255	255

*f*



$255^2$	$255^2$	$255^2$
$255^2$	0	$255^2$
$255^2$	$255^2$	$255^2$

1. Scalar multiplication



$8 * 255^2$
-------------

*h*



2. Summation

**CC**

128	128	128
128	0	128
128	128	128

*g*



255	255	255
255	255	255
255	255	255

*f*



128*	128*	128*
255	255	255
128*	0	128*
255		255
128*	128*	128*
255	255	255



261120
--------

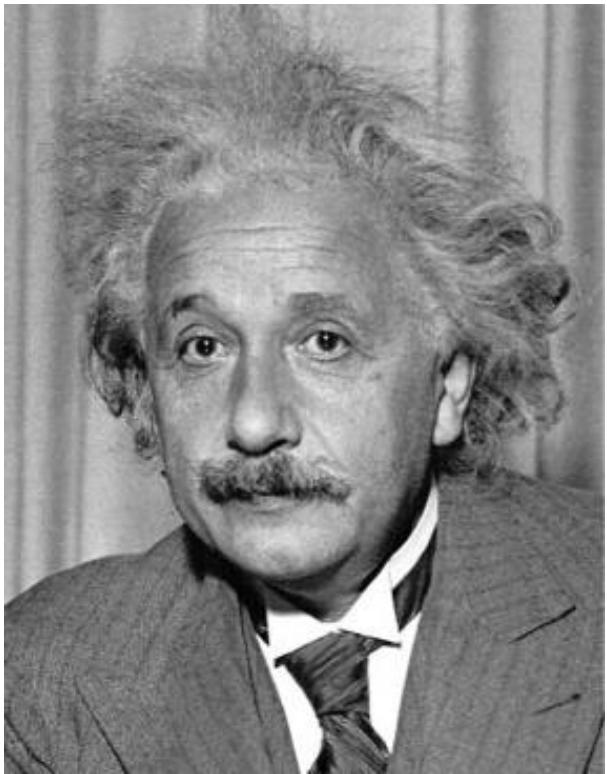


*h*

1. Scalar multiplication

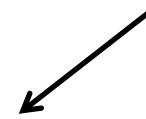
2. Summation

# Zero mean cross correlation



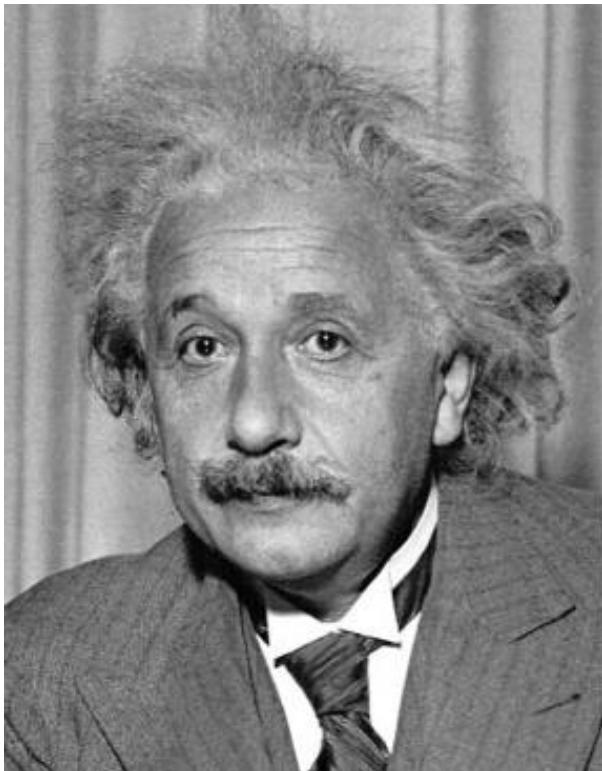
$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g})f[m + k, n + l]$$

template mean



What will  
the output  
look like?

# Zero mean cross correlation

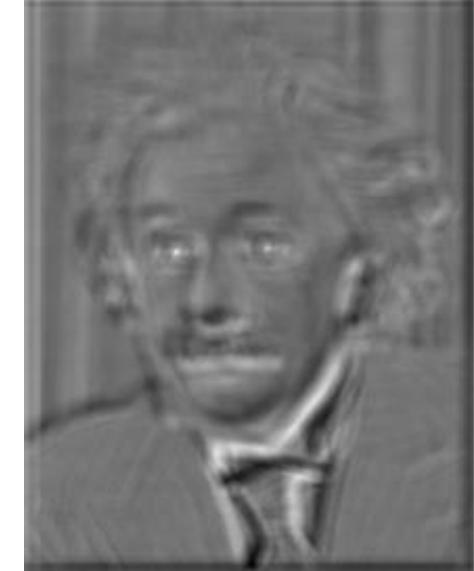


$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g})f[m + k, n + l]$$

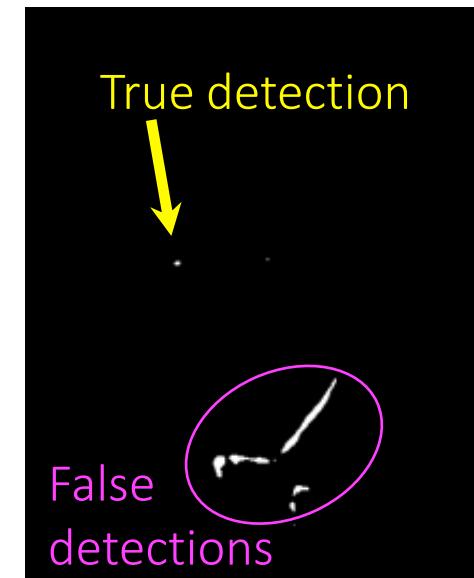
template mean

thresholding

output



True detection



Zero mean CC is good enough for most problems but can also cause false detections in high contrast areas.

# Zero mean CC

14	14	14
14	-113	14
14	14	14

$g - \bar{g}$



255	255	255
255	255	255
255	255	255

$f$



3570	3570	3570
3570	-28815	3570
3570	3570	3570



-255
------



$h$

1. Scalar multiplication

2. Summation

# Zero mean CC

14	14	14
14	-113	14
14	14	14

$g - \bar{g}$



128	128	128
128	0	128
128	128	128

$f$



1792	1792	1792
1792	0	1792
1792	1792	1792



14336
-------



$h$

1. Scalar multiplication

2. Summation

# Zero mean CC

14	14	14
14	-113	14
14	14	14

$g - \bar{g}$



0	0	255
0	0	255
255	255	255

$f$



0	0	3570
0	0	3570
3570	3570	3570



17850
-------

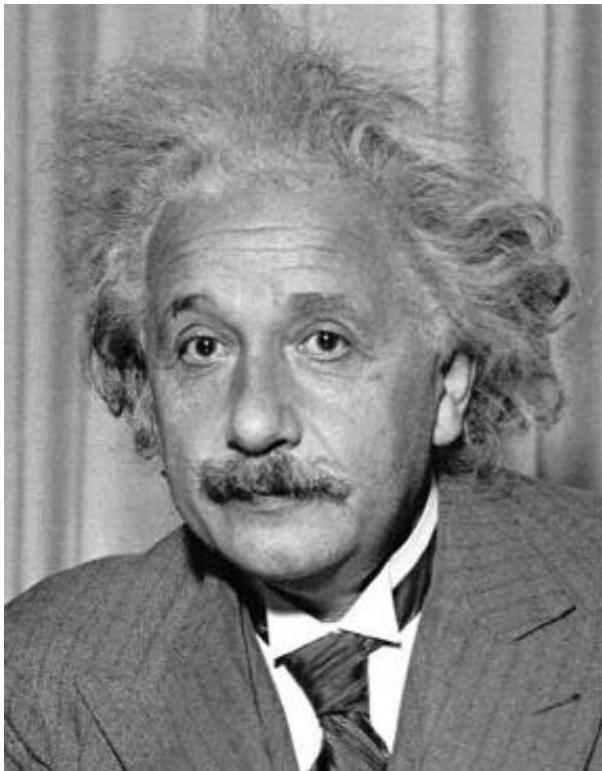


$h$

1. Scalar multiplication

2. Summation

# ZNCC – zero mean normalized cross correlation

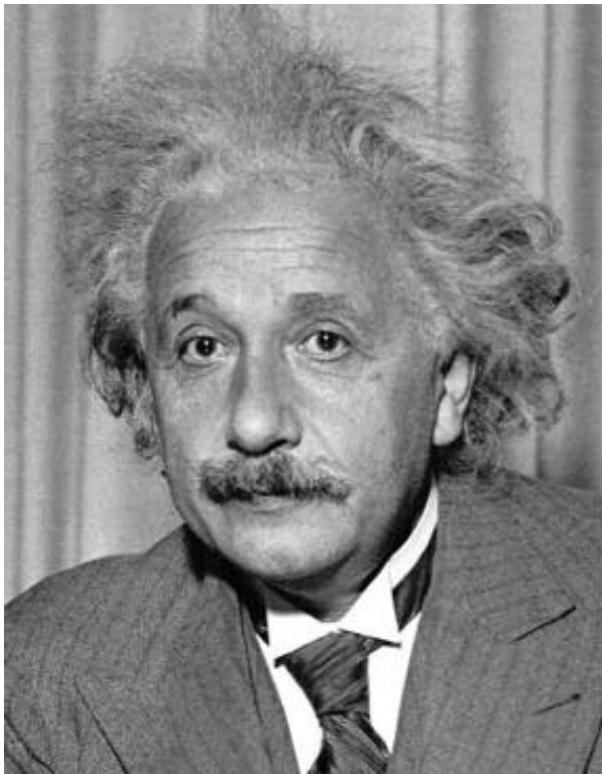


What will  
the output  
look like?

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

The below square root is the product of both template and patch STD. For clarity let's treat it as normalization of the image-patch and template [-1,1]

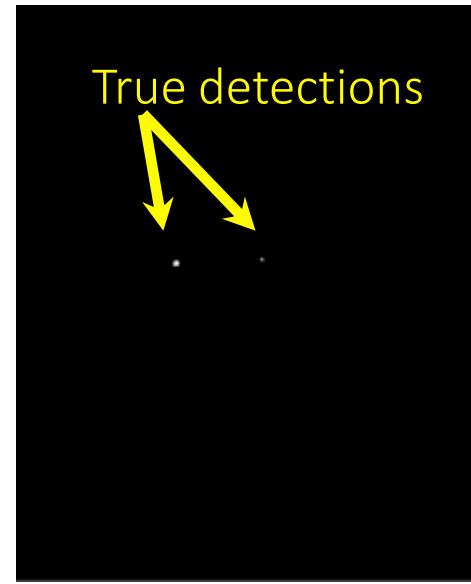
# ZNCC – zero mean normalized cross correlation



output

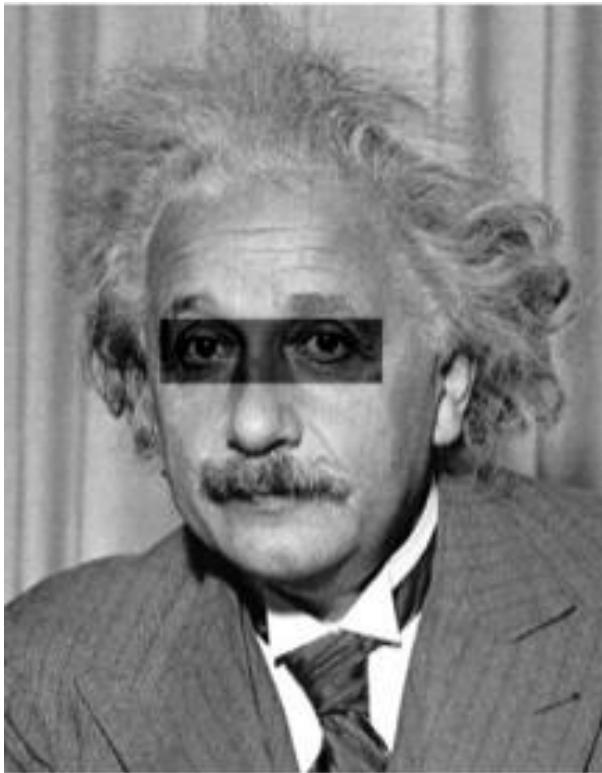


thresholding



True detections

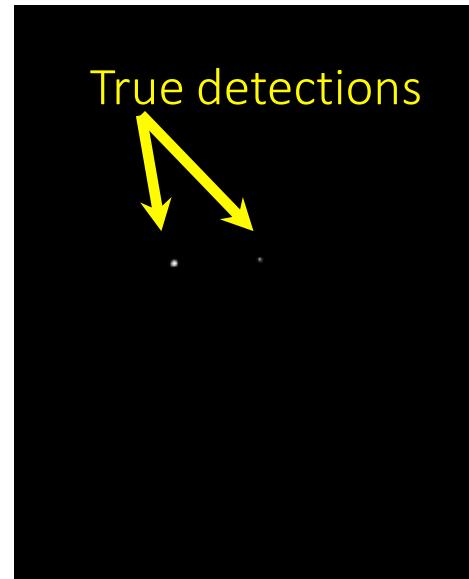
# ZNCC – zero mean normalized cross correlation



output



thresholding  
robust to change in  
intensities



# Zero mean normalized CC

1	1	1
1	-1	1
1	1	1

$$norm(g - \bar{g})$$



0	0	0
0	0	0
0	0	0

$$norm(f - \bar{f})$$



0
---



*h*

Summation

# Zero mean normalized CC

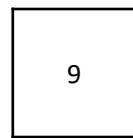
1	1	1
1	-1	1
1	1	1

$$norm(g - \bar{g})$$



1	1	1
1	-1	1
1	1	1

$$norm(f - \bar{f})$$



$$h$$



Summation

# Zero mean normalized CC

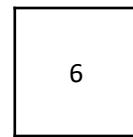
1	1	1
1	-1	1
1	1	1

$$norm(g - \bar{g})$$



-1	-1	1
-1	-1	1
1	1	1

$$norm(f - \bar{f})$$



$$h$$



Summation

# contents

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

# Morphology

Examples:

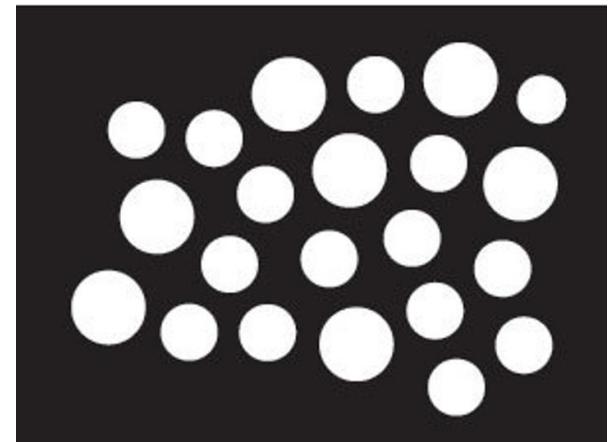
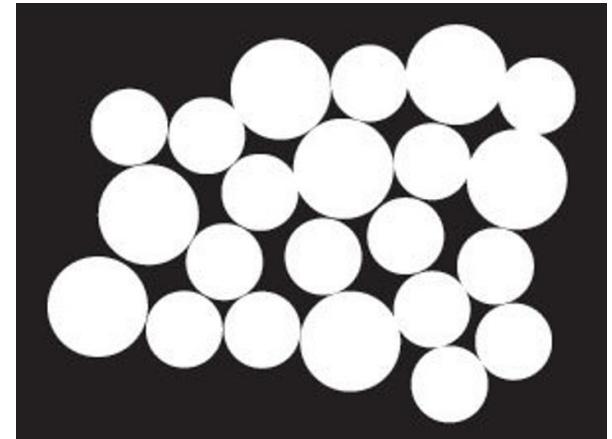
Image cleaning



Style



Coin counting  
(using connected components)



# The 4 basic operators

Dilate



Open (Erode ⇒ Dilate)



Erode

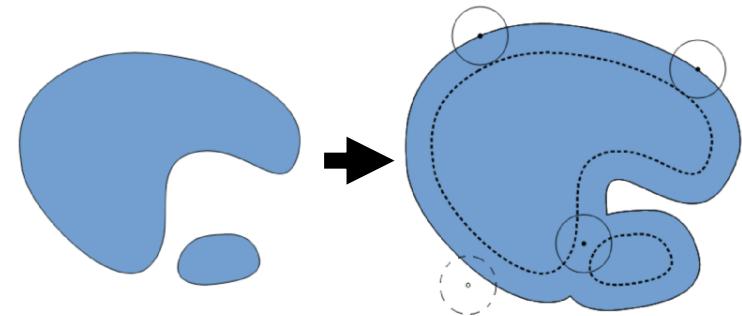


Close (Dilate ⇒ Erode)

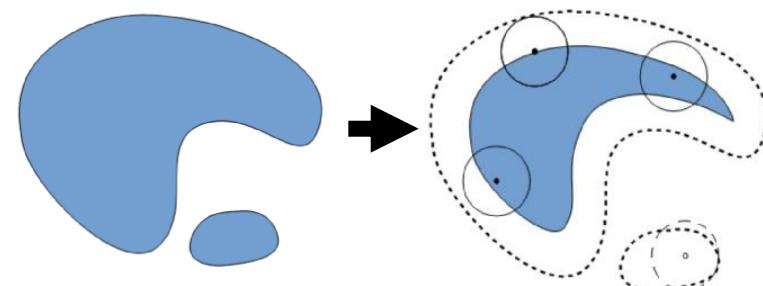


# Morphology: geometric interpretation

- Each kernel ( $g$ ) has an anchor point (usually in the kernel center).
- Dilation: the final shape is all points where the anchor point can be placed in which **the kernel touches a part of the original shape**.



- Erosion: the final shape is all points where the anchor point can be placed in which **all kernel points touch the original shape**.



# 1. Cross-correlation with the kernel

$$\begin{array}{ccccccccc}
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
 \hline
 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \hline
 \end{array}
 = \star
 \begin{array}{ccc}
 1 & 1 & 1 \\
 1 & 1 & 1 \\
 1 & 1 & 1
 \end{array}
 \quad g$$

## 2. Threshold the result

For **dilation**- threshold with 1

	2	3	3	2	2	1
	4	6	6	5	5	3
	6	9	9	7	5	2
	6	9	9	7	5	2
	6	8	7	4	2	0
	6	7	5	2	1	0

$\geq 1 =$

1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1	1	1	1	0	
1	1	1	1	1	1	0	

## 2. Threshold the result

For **erosion**- threshold with the sum of the kernel

	2	3	3	2	2	1		
	4	6	6	5	5	3		
	6	9	9	7	5	2		
	6	9	9	7	5	2		
	6	8	7	4	2	0		
	6	7	5	2	1	0		

$$\geq \text{sum}(g) =$$

	0	0	0	0	0	0		
	0	0	0	0	0	0		
	0	1	1	0	0	0		
	0	1	1	0	0	0		
	0	0	0	0	0	0		
	0	0	0	0	0	0		

$\text{sum}(g) = 9$  in this example

# Morphology: algorithm

- Each morphology operator is constructed as such:
  1. Select a structure element (binary kernel)
  2. Cross-correlate with input binary image  $h = f \star g$
  3. Threshold the output

$$g = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

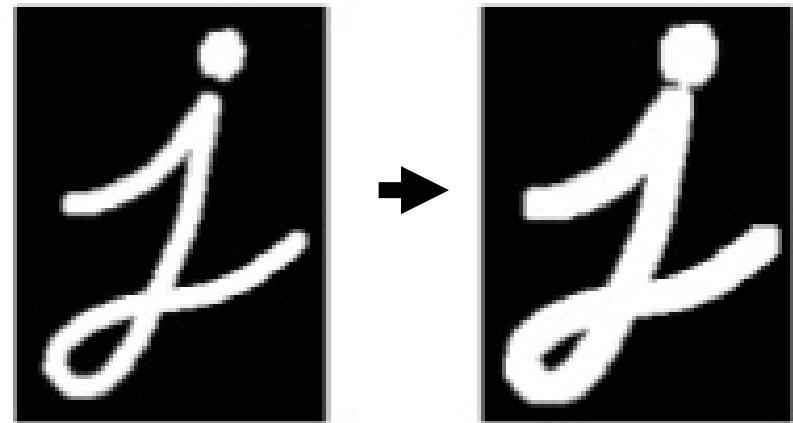
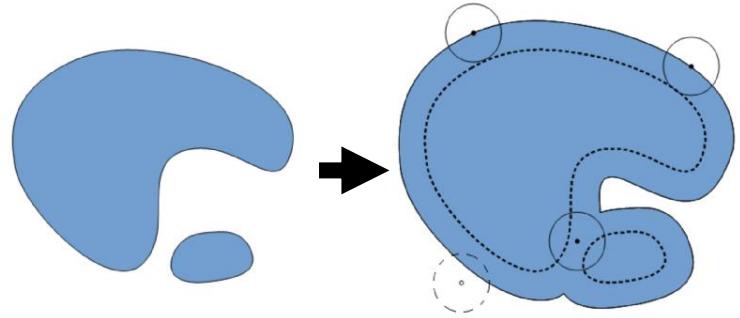
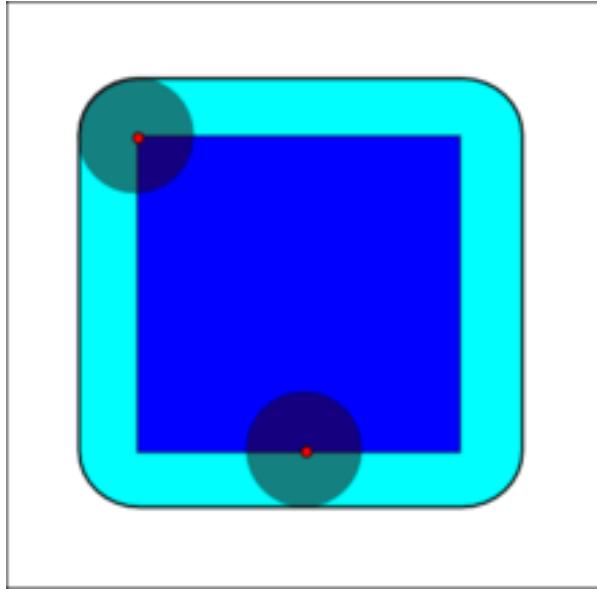
$$\theta_{TH}(x, t) = \begin{cases} 1 & \text{if } x \geq t, \\ 0 & \text{else} \end{cases}$$

- Overall morphologic operation should look like so:

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

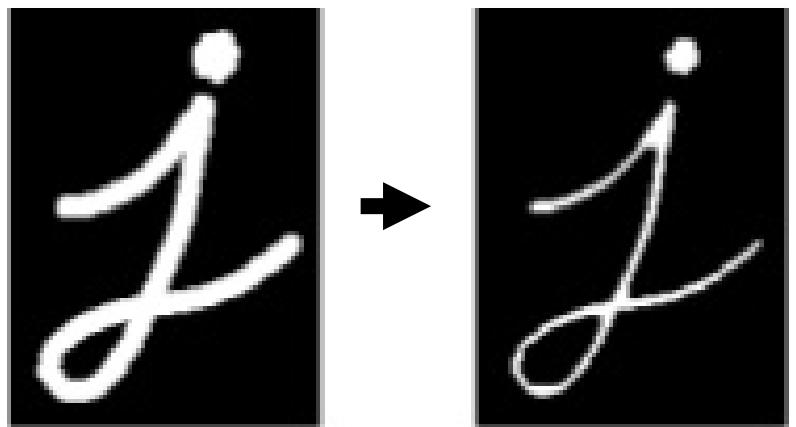
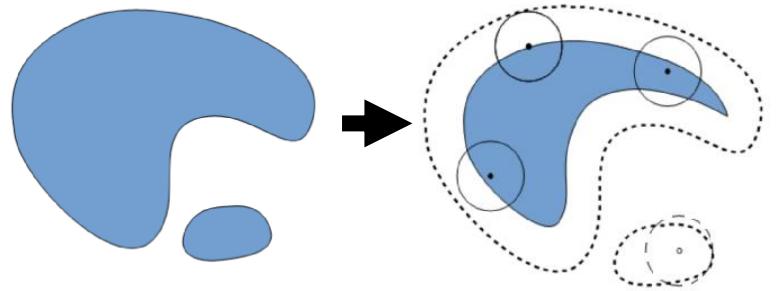
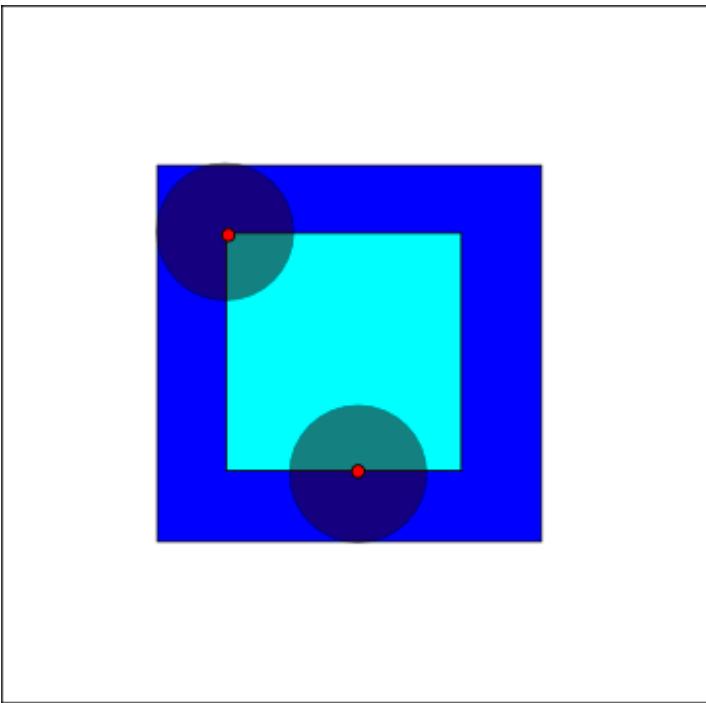
# Dilation- examples

- $k = \theta_{TH}(f \star g, t = 1)$



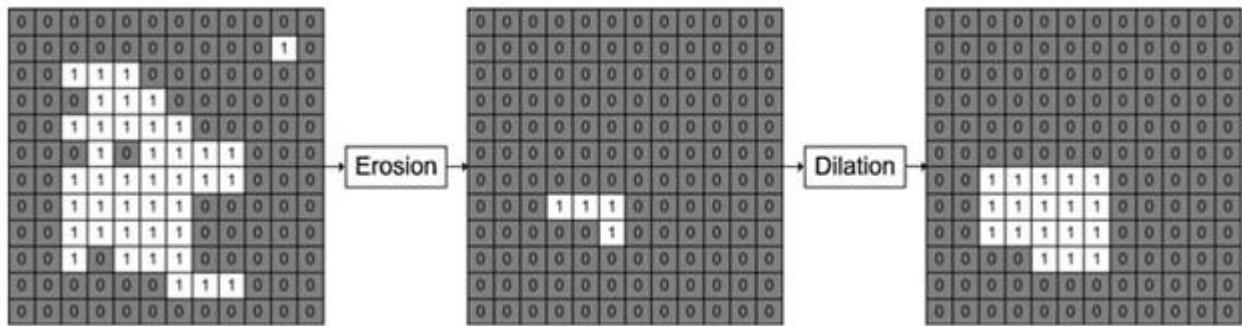
# Erosion- examples

- $k = \theta_{TH}(f * g, t = \text{sum}(g))$

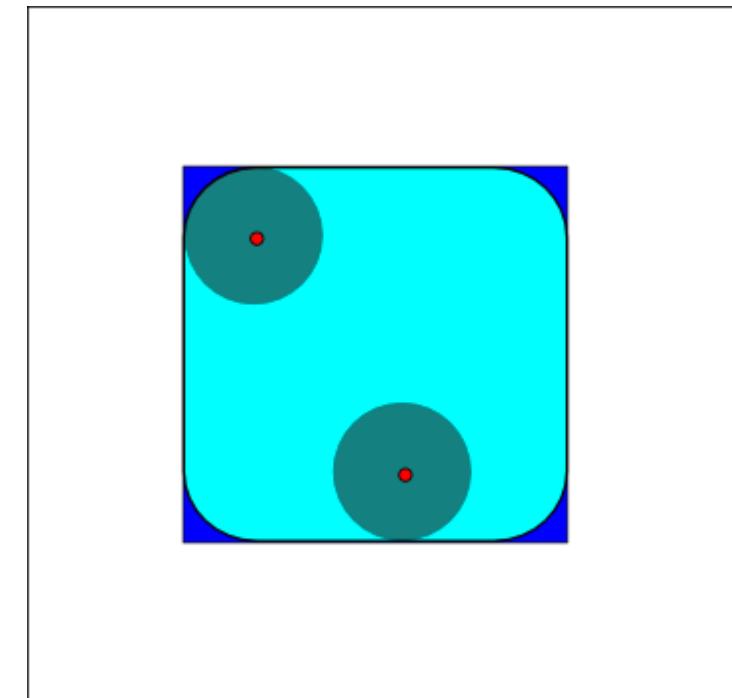


# Opening

- Erosion followed by dilation.
  - The effect is of removing noise or sharp edges.

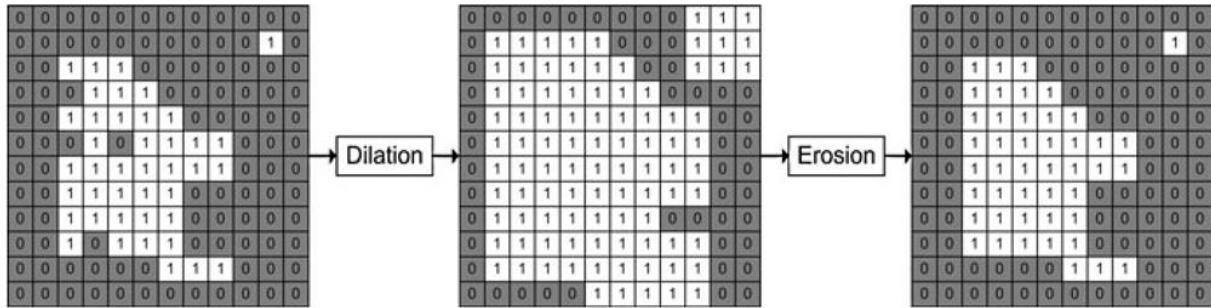


Open (Erode  $\Rightarrow$  Dilate)

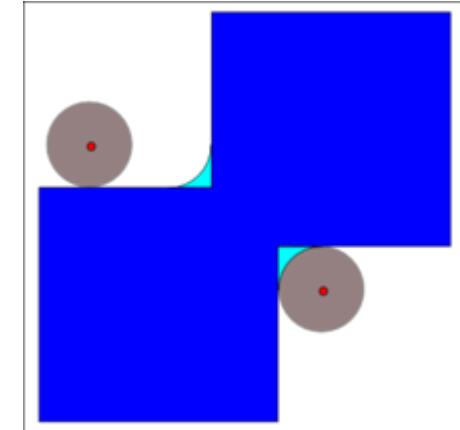


# Closing

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

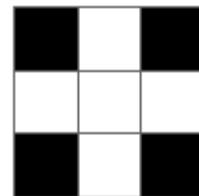


Close (Dilate  $\Rightarrow$  Erode)

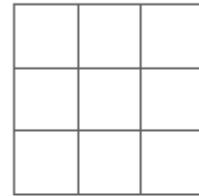


# Affect of different kernels

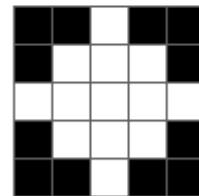
Dilation



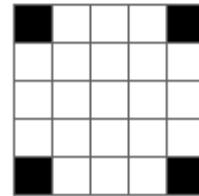
Dilation



Dilation



Dilation



Dilation

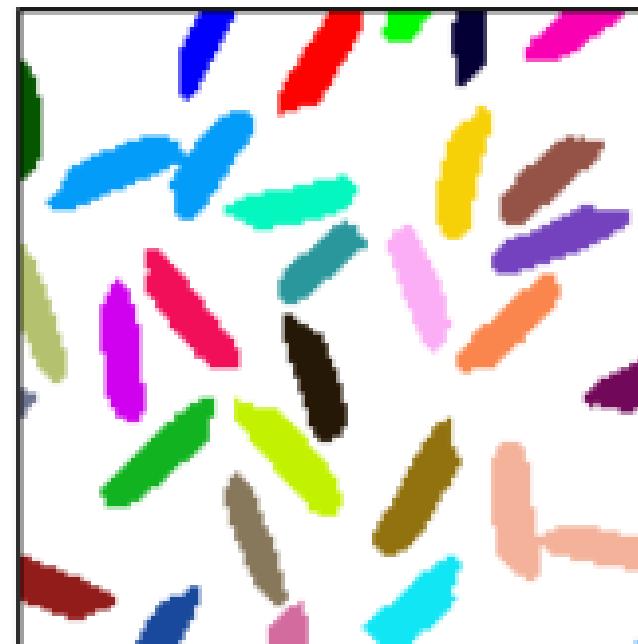
We will see non-symmetrical kernels in the HW

# contents

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

# Connected components

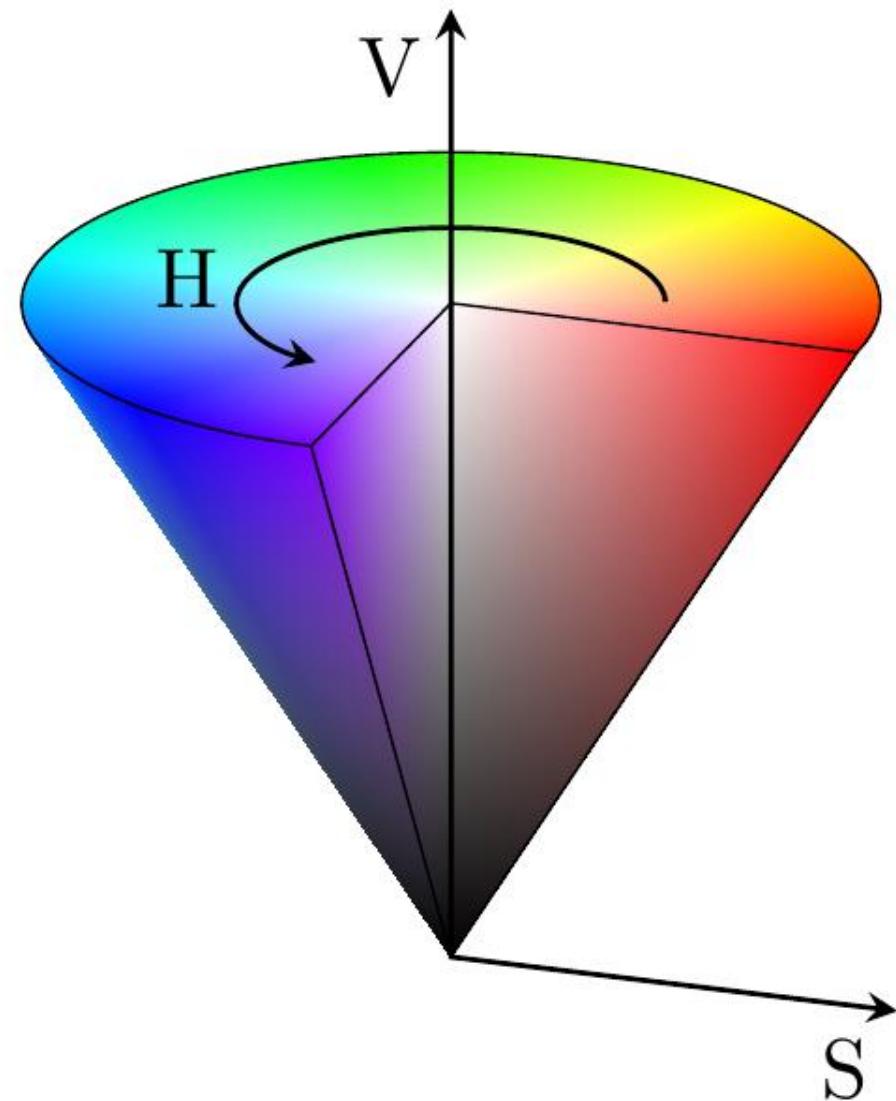
- Defined as regions of adjacent pixels that have the same value.
- Commonly used with binary images to find stand alone objects.
  - e.g.: letters in a document.



# contents

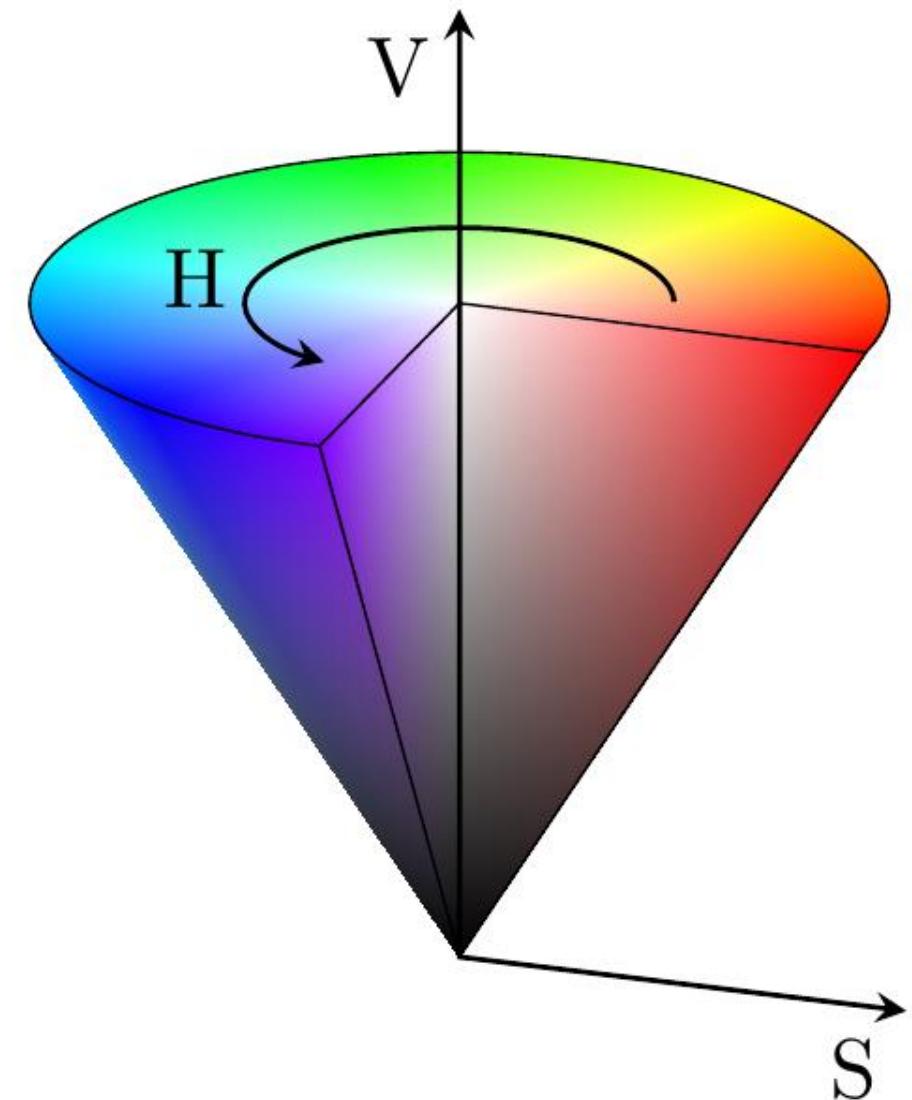
- Image representation
- Pixel-wise operations
- Histogram equalization
- Template matching
- Morphology operators
- Connected components
- **Color space**

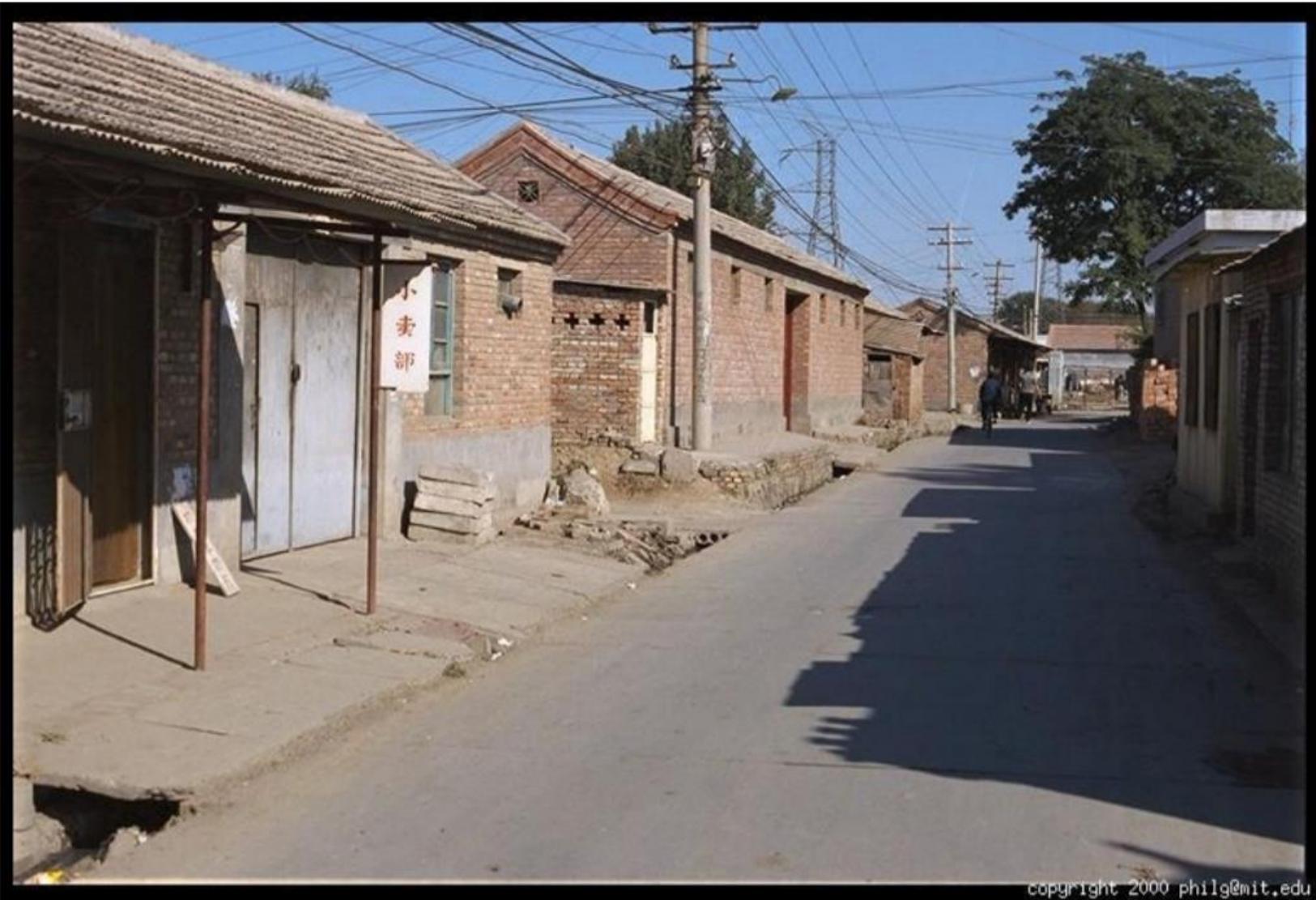
# HSV



# HSV

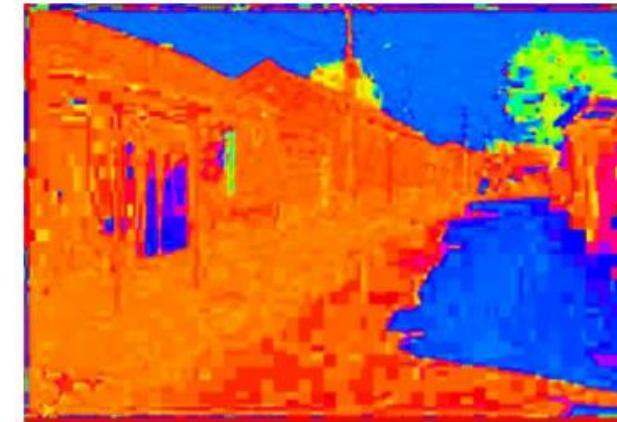
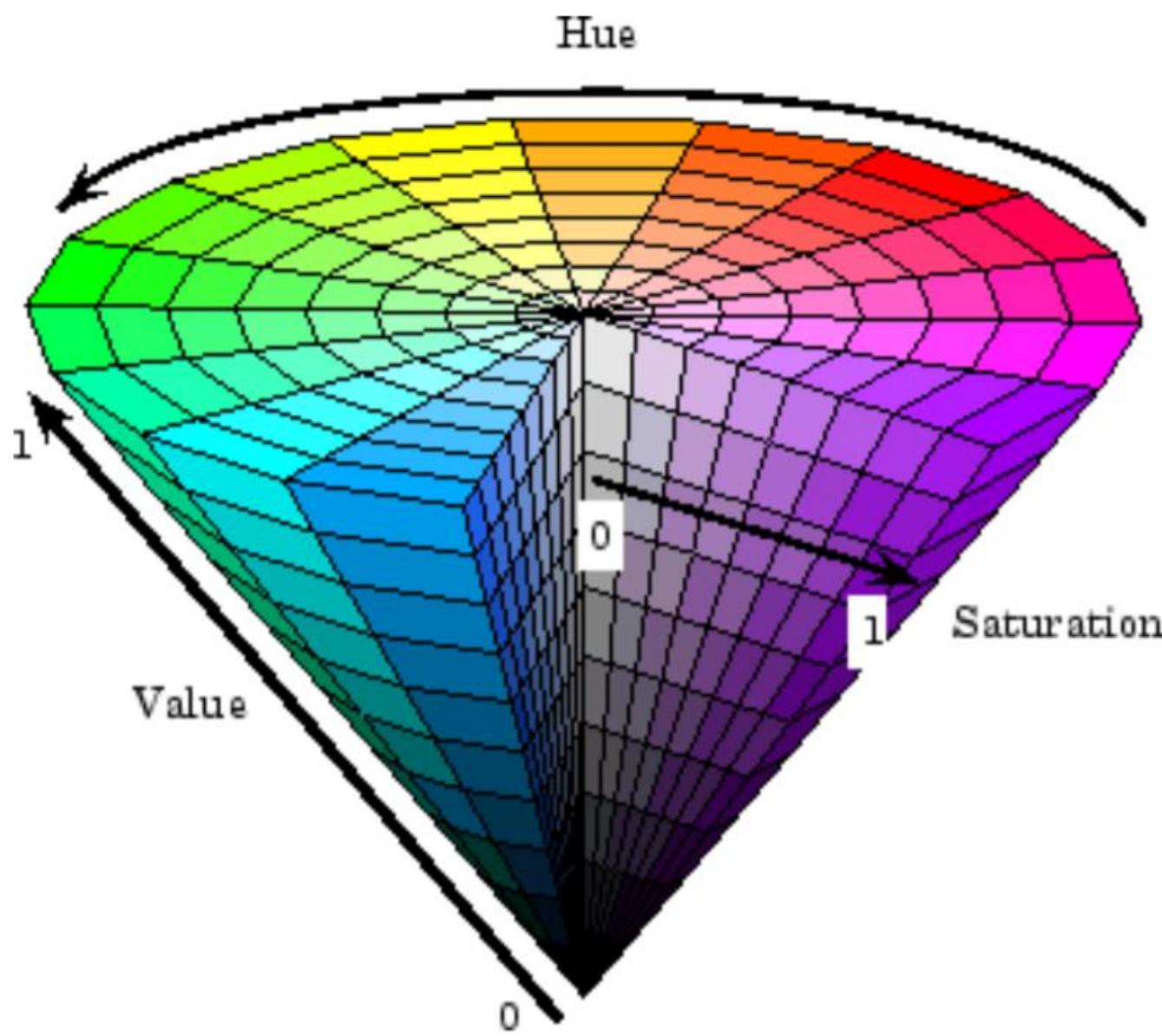
- **Hue:** The "attribute of a visual sensation according to which an area appears to be similar to one of the perceived colors: red, yellow, green, and blue, or to a combination of two of them"
- **Saturation:** The "colorfulness of a stimulus relative to its own brightness"
- **Value:** The "brightness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





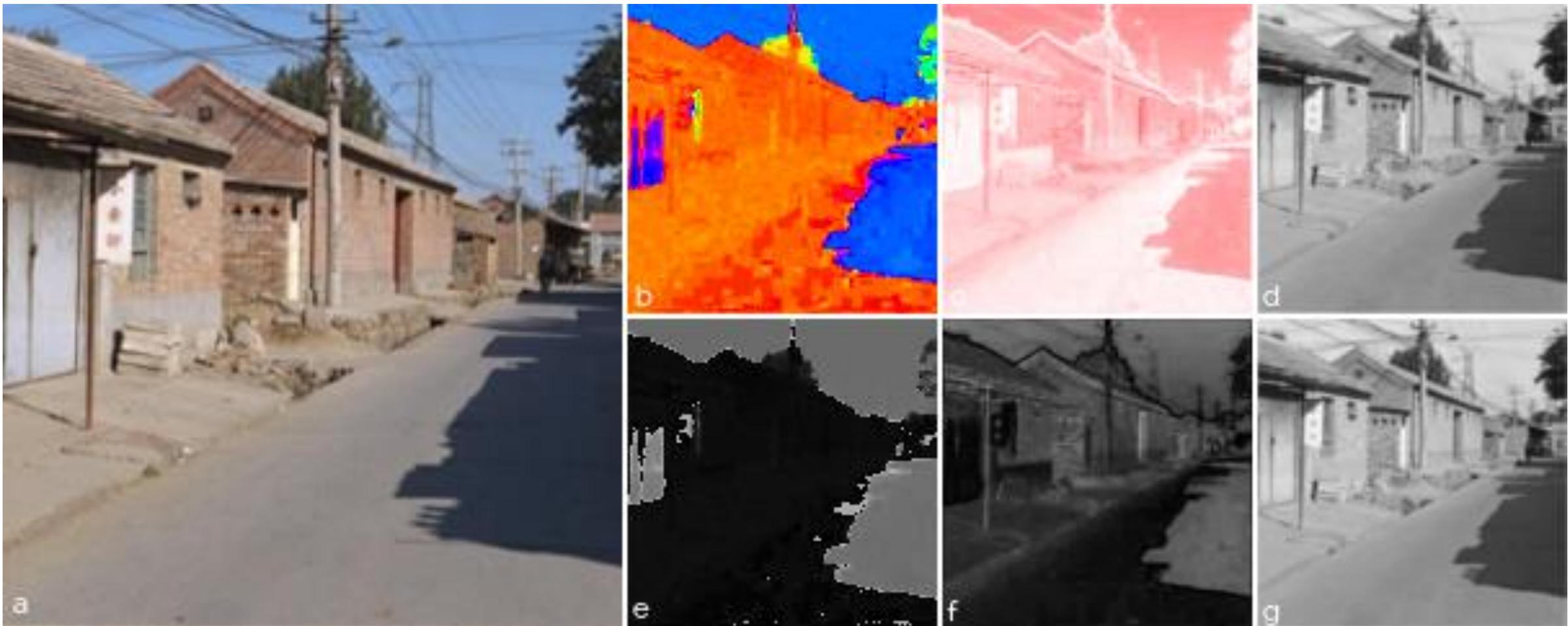
copyright 2000 philg@mit.edu

Original image



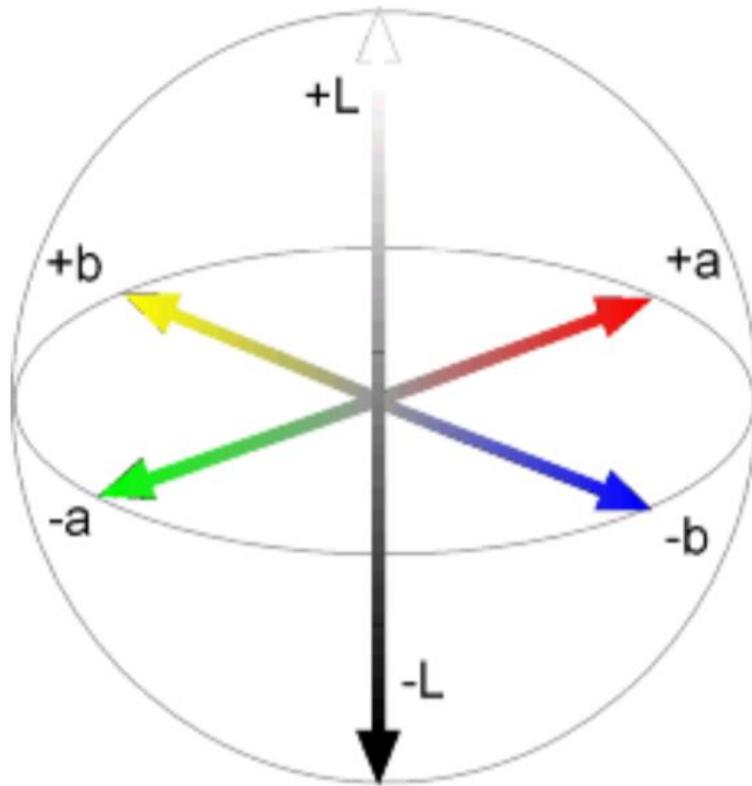
# HSV

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



# More color spaces: LAB

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



# More color spaces: YUV

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

