Basic image processing

Yoni Chechik

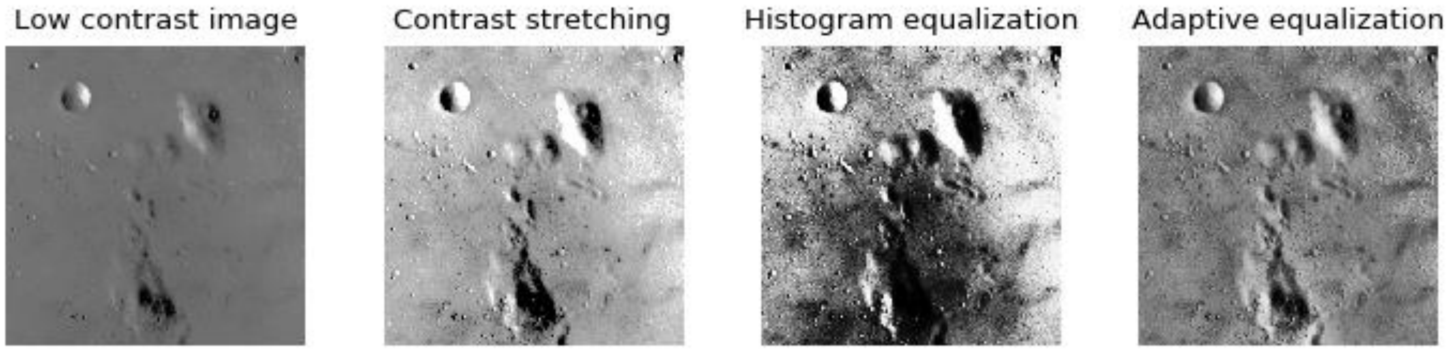
www.AlisMath.com

References

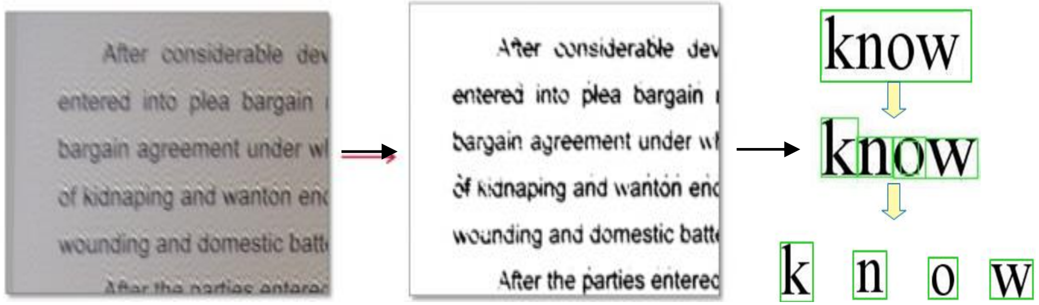
- http://szeliski.org/Book/
- http://www.cs.cornell.edu/courses/cs5670/2019sp/lectures/lectures.html
- http://www.cs.cmu.edu/~16385/

Some motivation

Art
(Photoshop color grading)



Science and space
(image enhancement)


After considerable dev
entered into plea bargain
bargain agreement under w
of kidnaping and wanton en
wounding and domestic batt
After the parties entered
know
know
k
n
o
w

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 = $\{0, \dots, 255\}$.

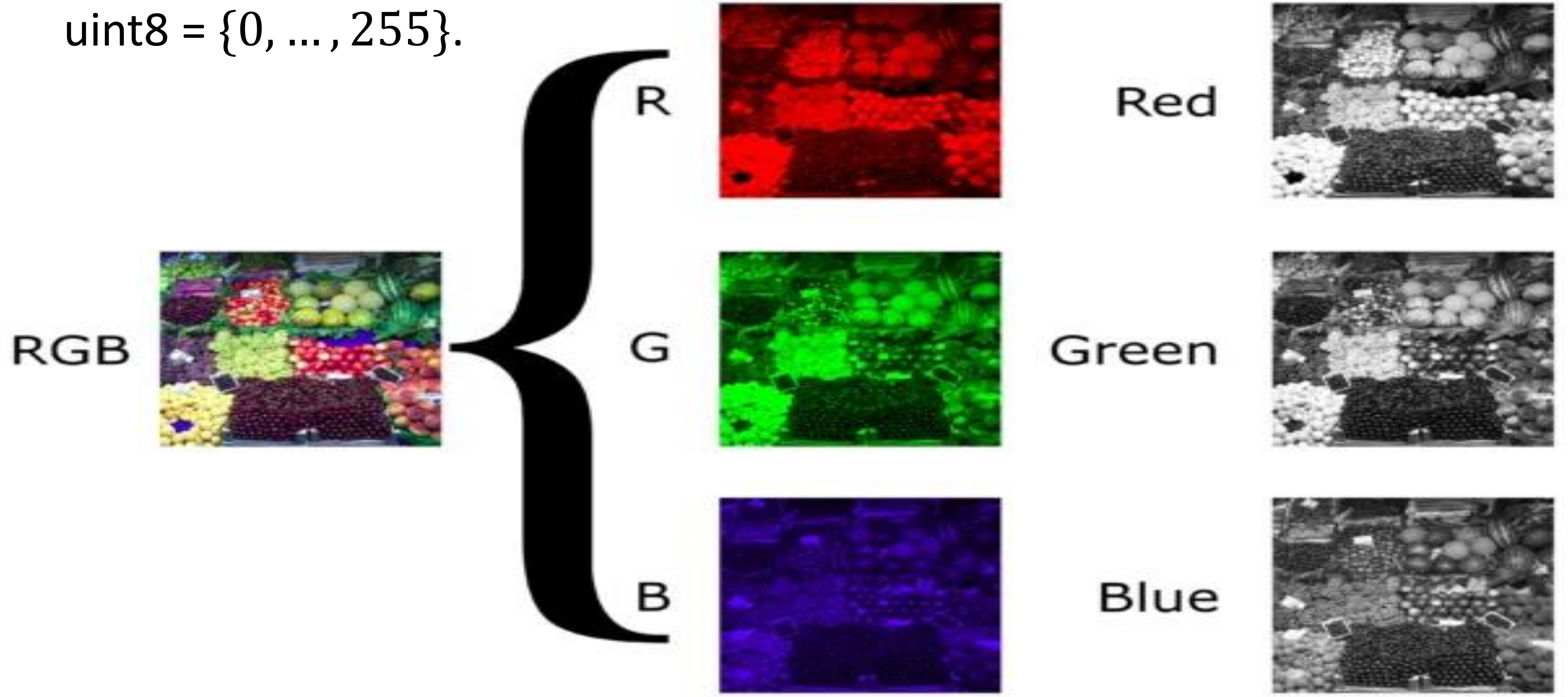
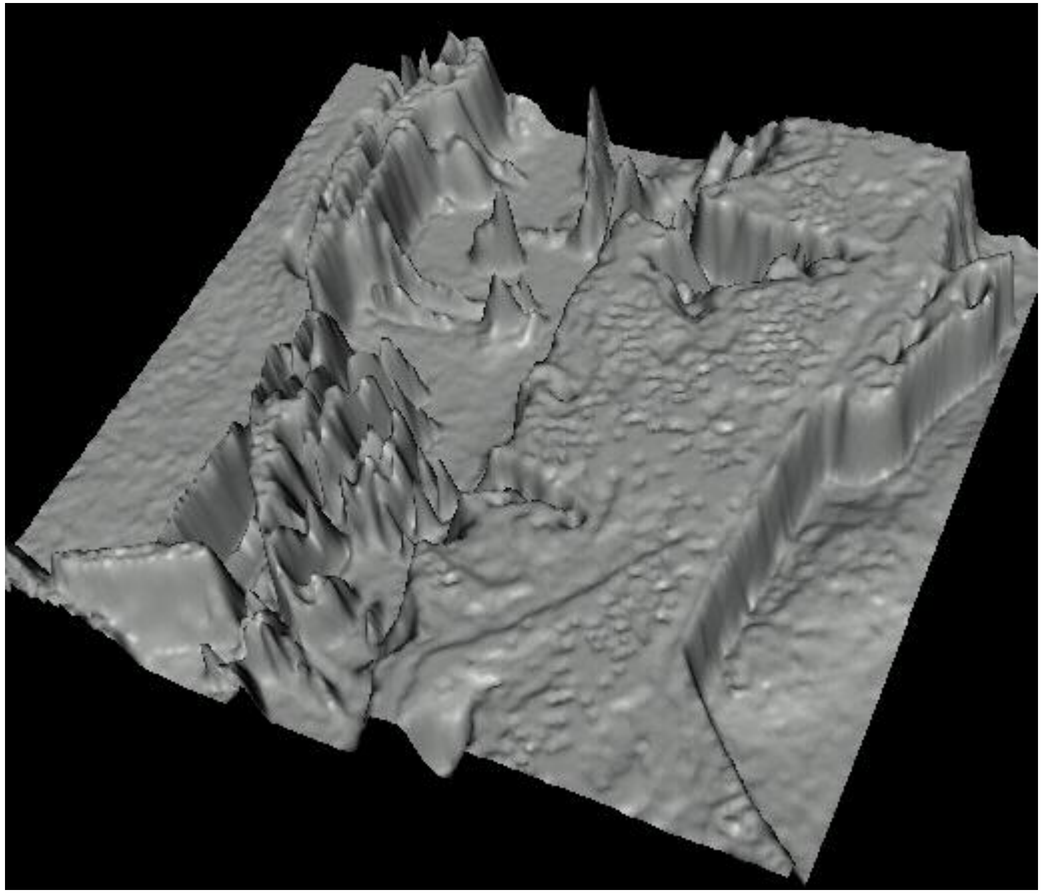


Image representation

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



contents

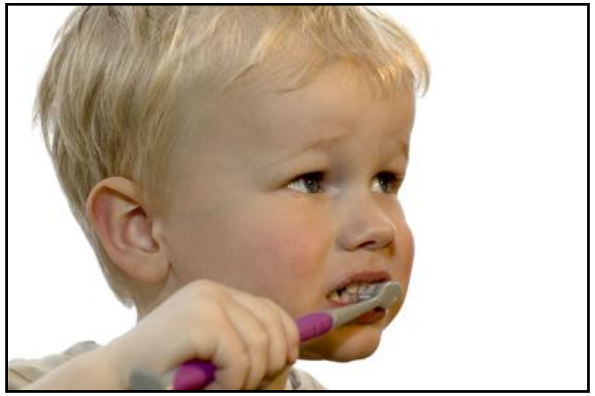
- Image representation
- **Pixel-wise operations**
- Histogram equalization
- Template matching
- Morphology operators
- Connected components
- Color space

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.

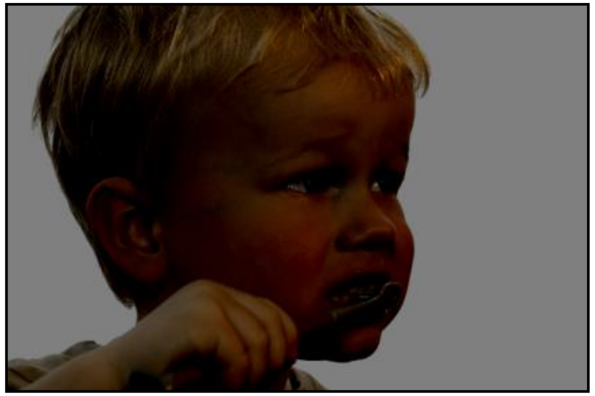
Pixel-wise operators

original

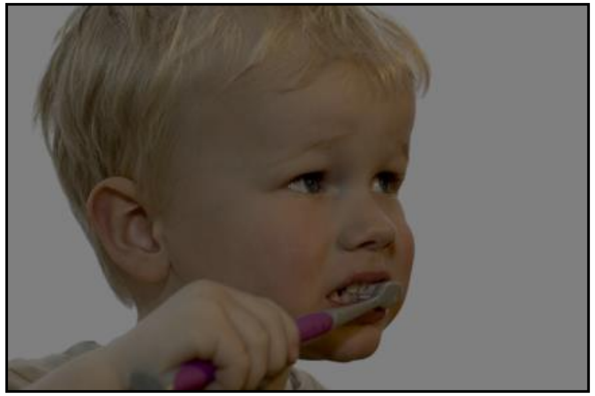


x

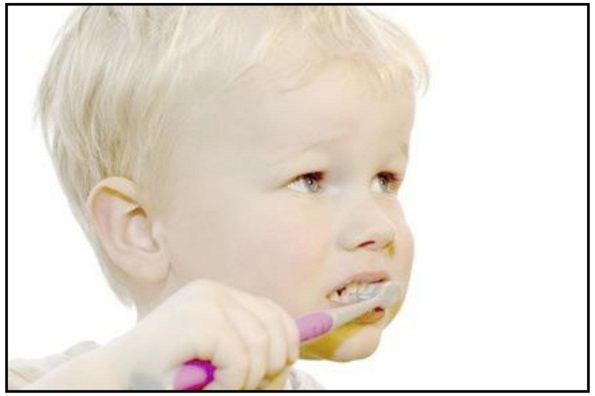
darken



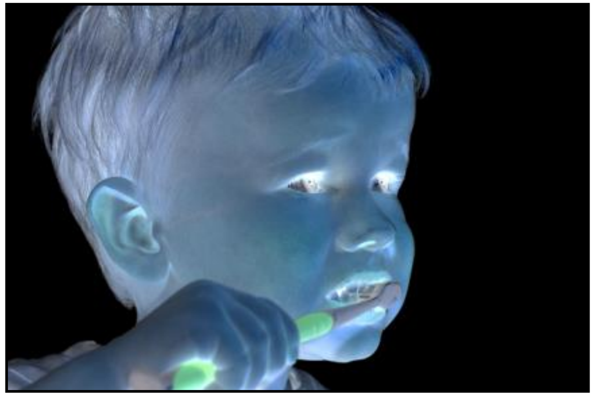
lower contrast



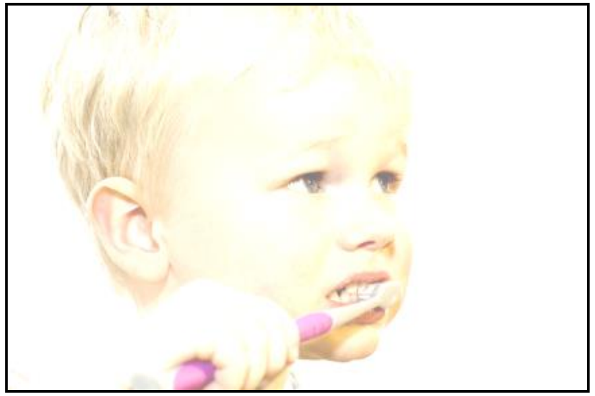
Gamma compression



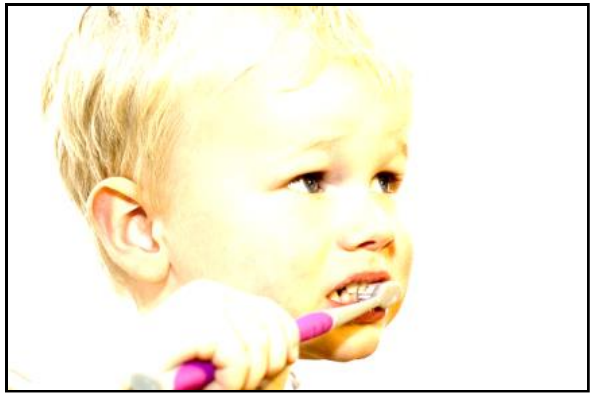
invert



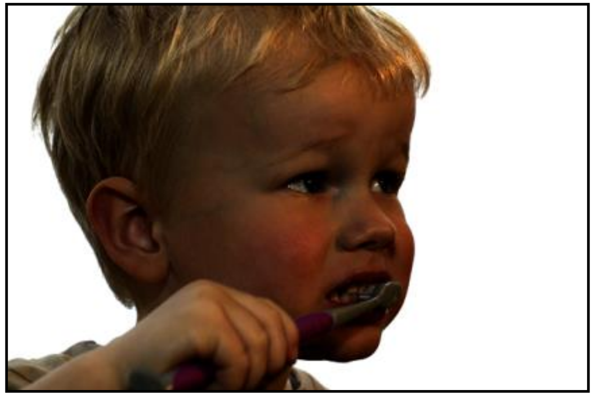
lighten



raise contrast

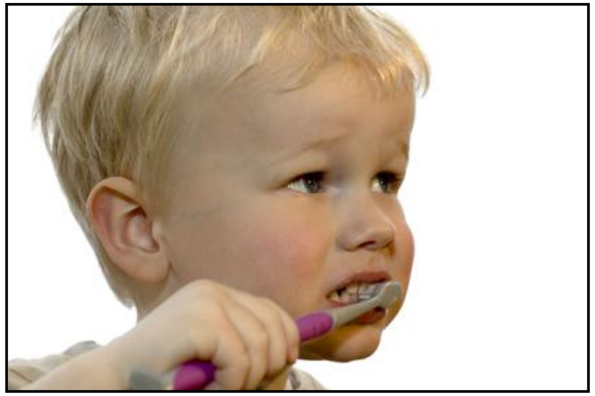


Gamma expansion



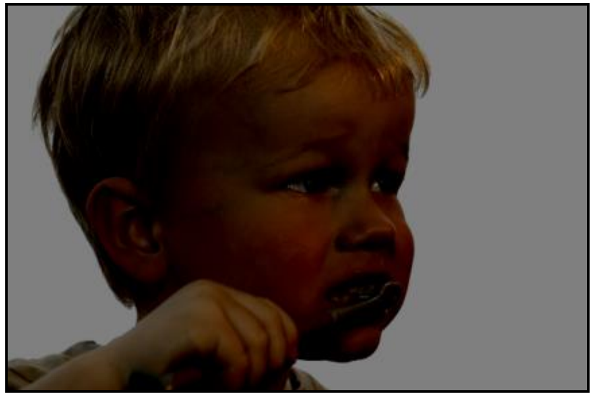
Pixel-wise operators

original

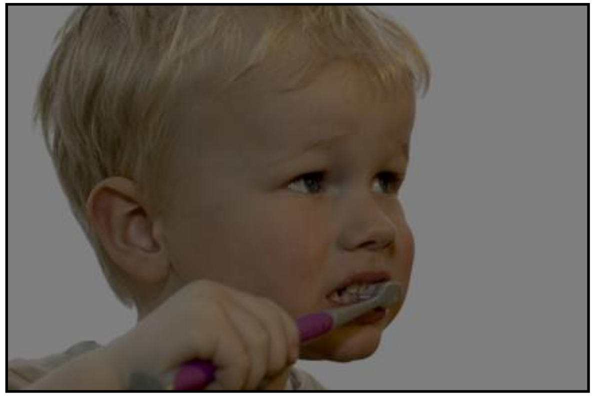


x

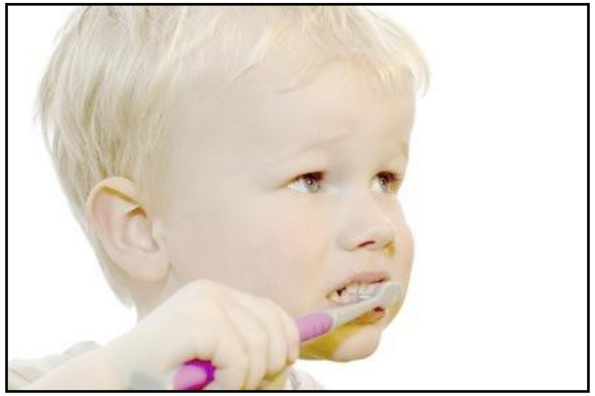
darken



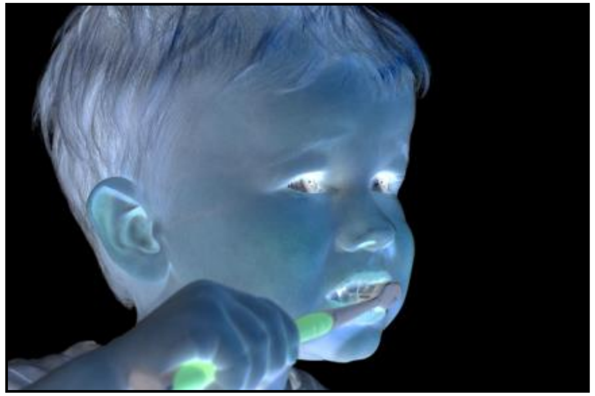
lower contrast



Gamma compression

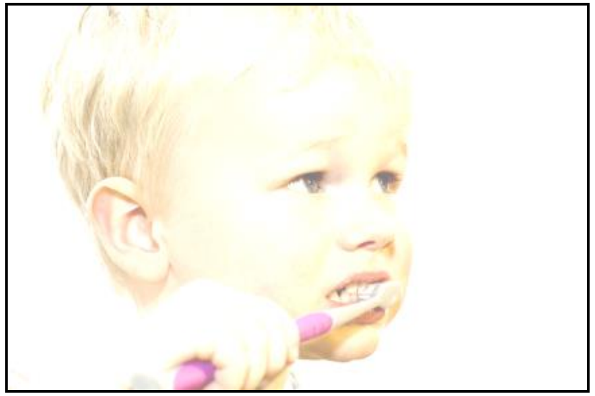


invert

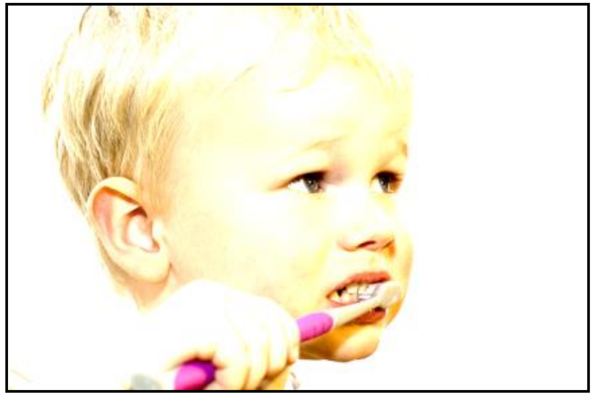


$255 - x$

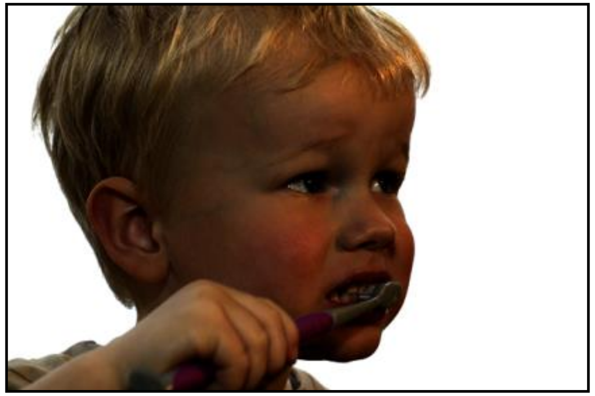
lighten



raise contrast

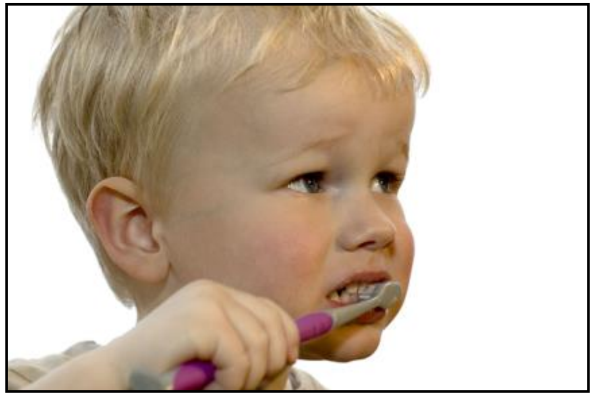


Gamma expansion



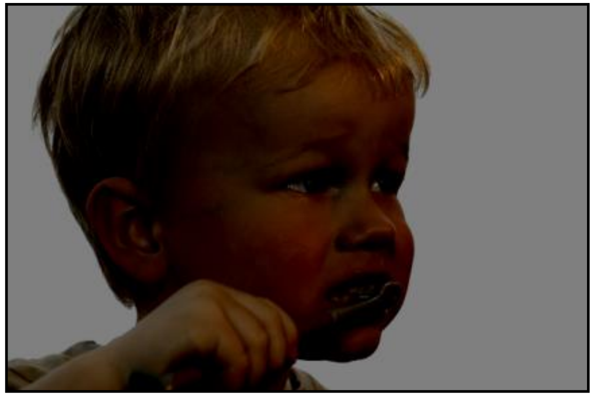
Pixel-wise operators

original



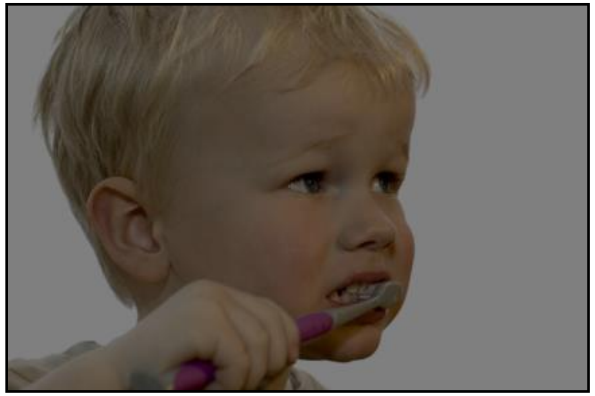
$$x$$

darken

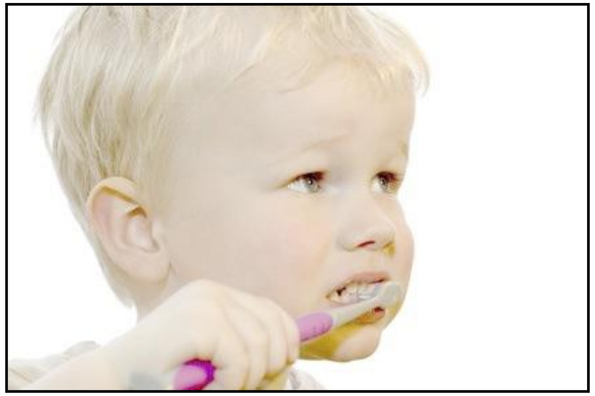


$$x - 128$$

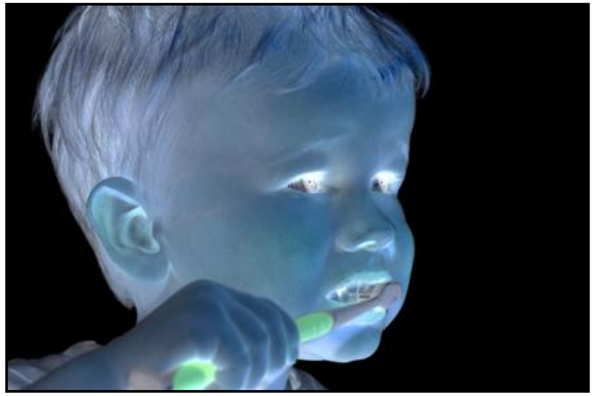
lower contrast



Gamma compression

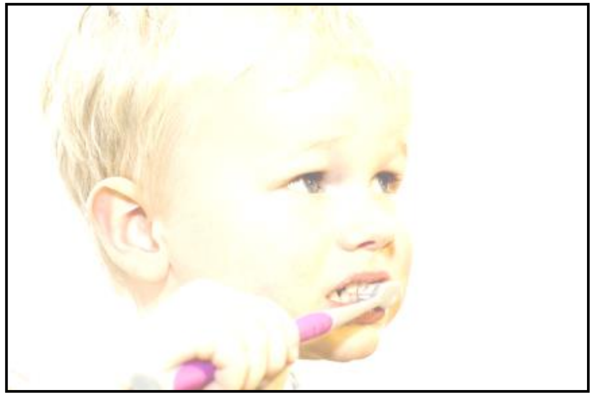


invert

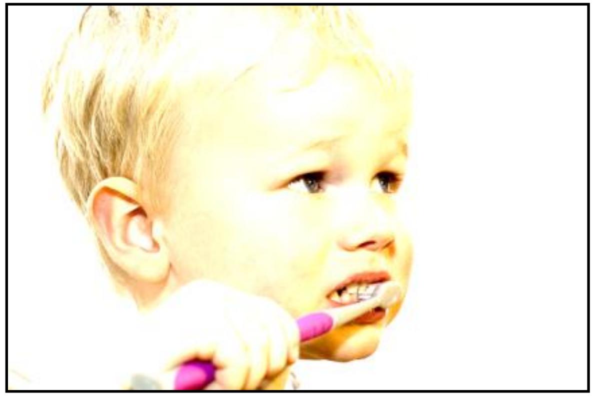


$$255 - x$$

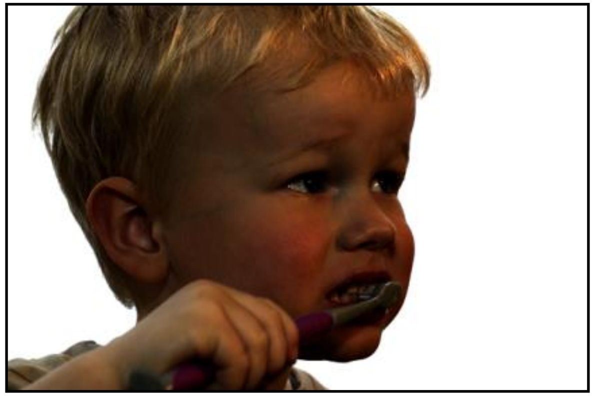
lighten



raise contrast

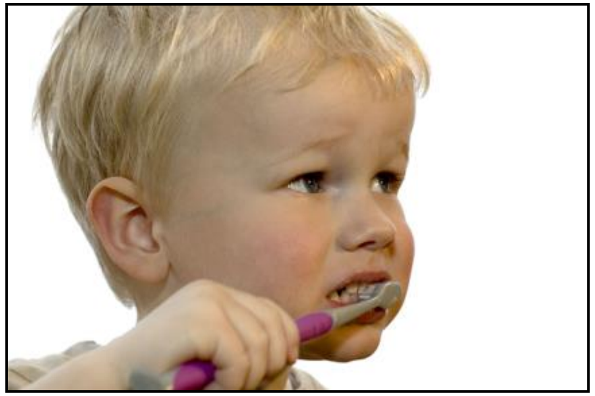


Gamma expansion



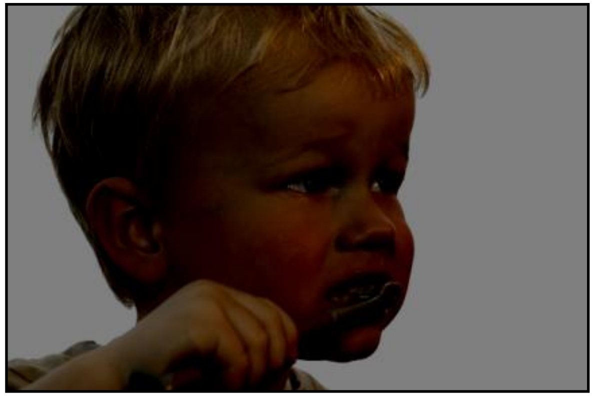
Pixel-wise operators

original



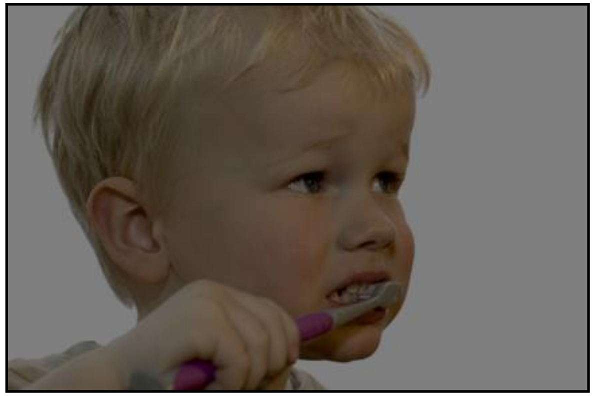
x

darken

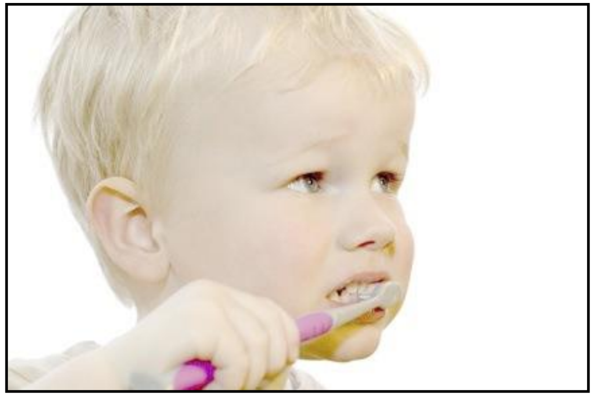


$x - 128$

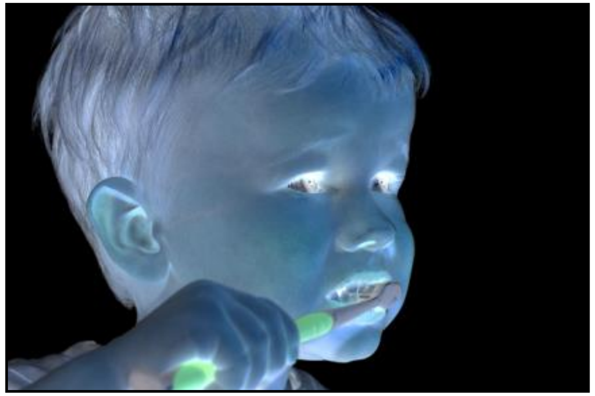
lower contrast



Gamma compression

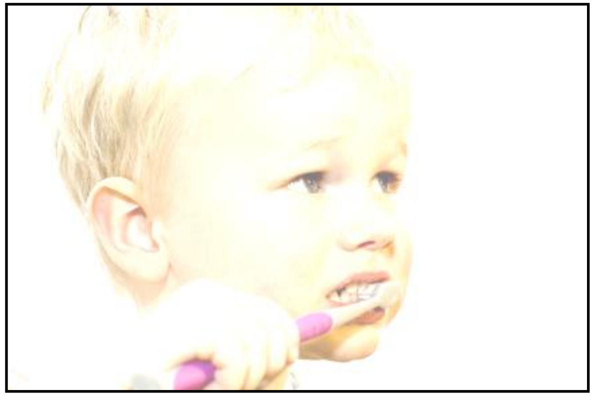


invert



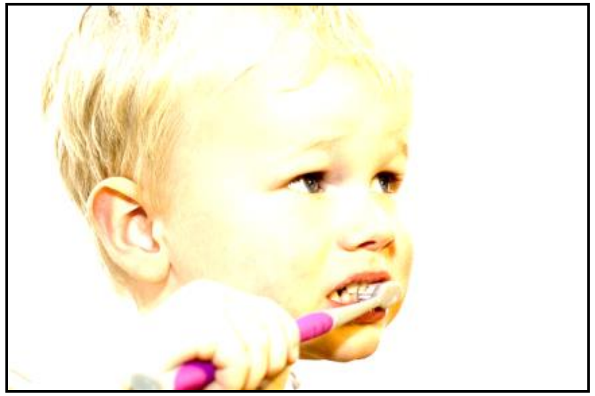
$255 - x$

lighten

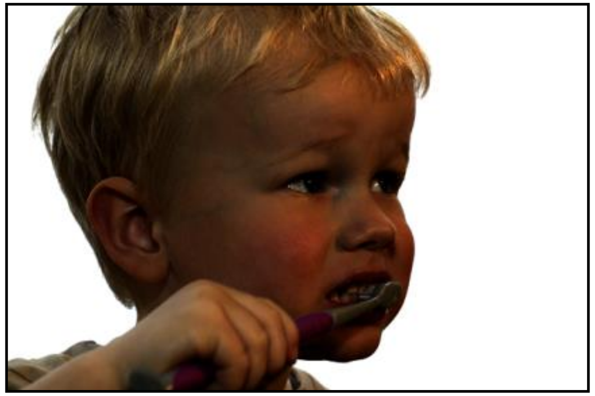


$x + 128$

raise contrast

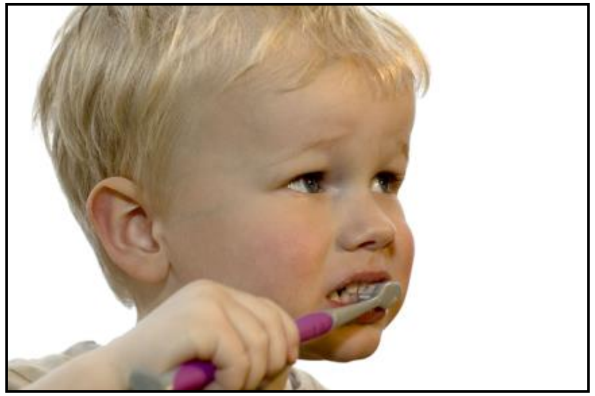


Gamma expansion



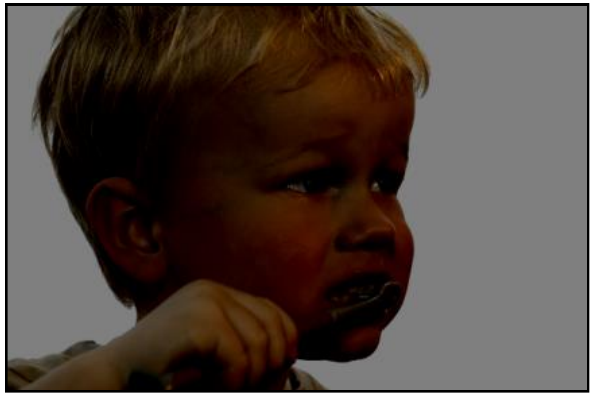
Pixel-wise operators

original



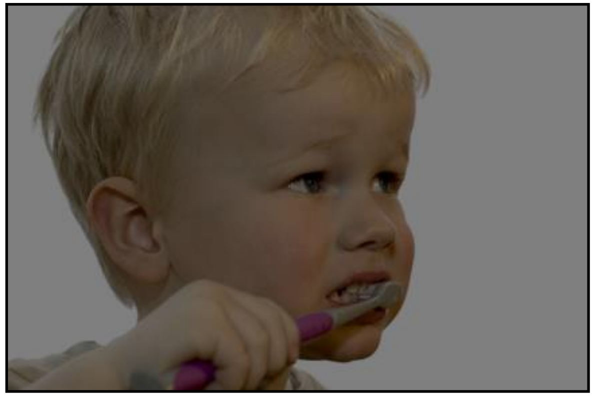
x

darken



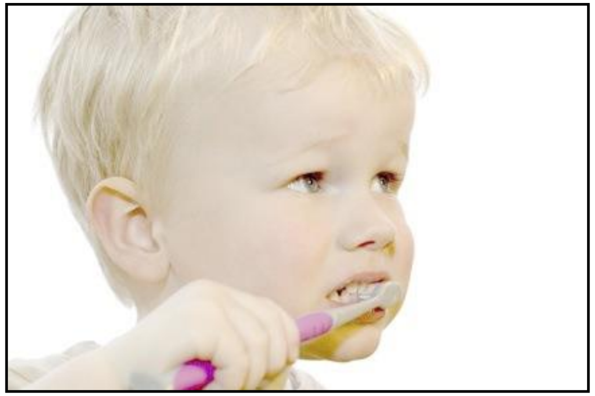
$x - 128$

lower contrast

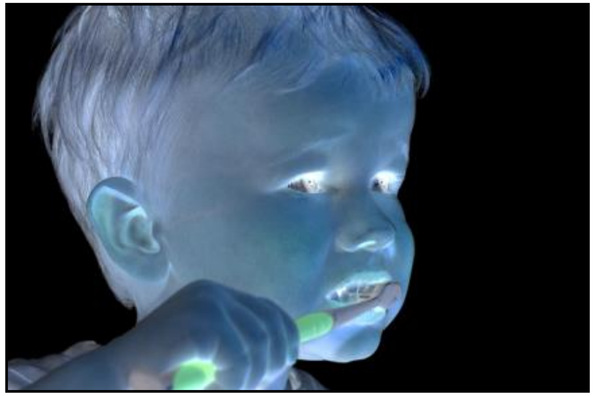


$\frac{x}{2}$

Gamma compression

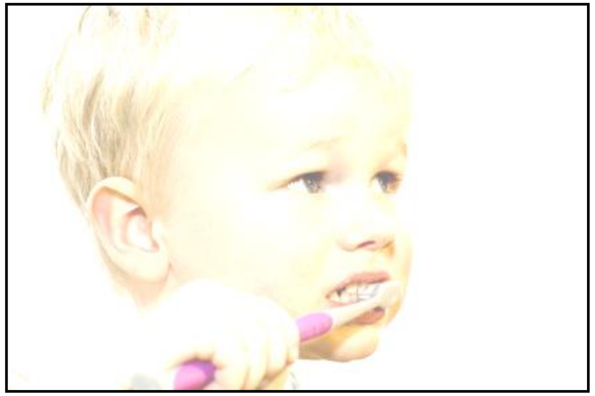


invert



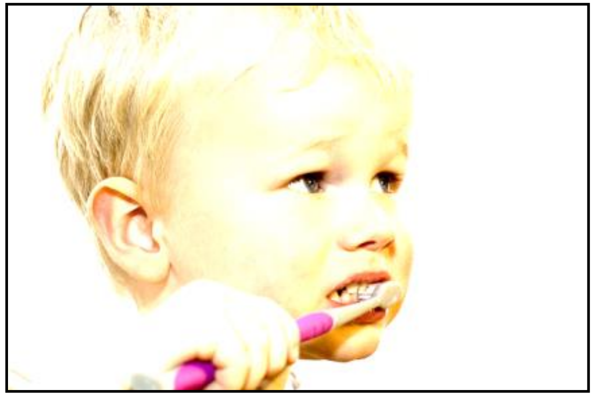
$255 - x$

lighten

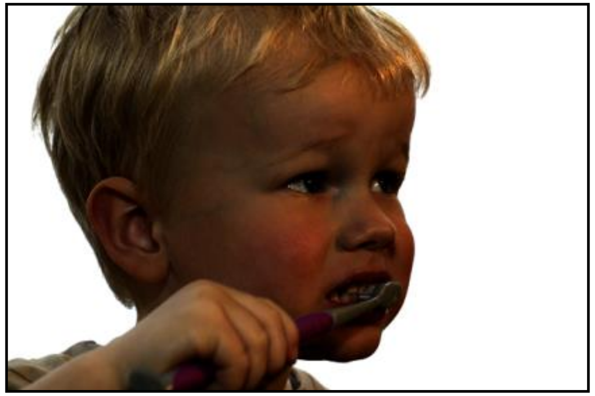


$x + 128$

raise contrast

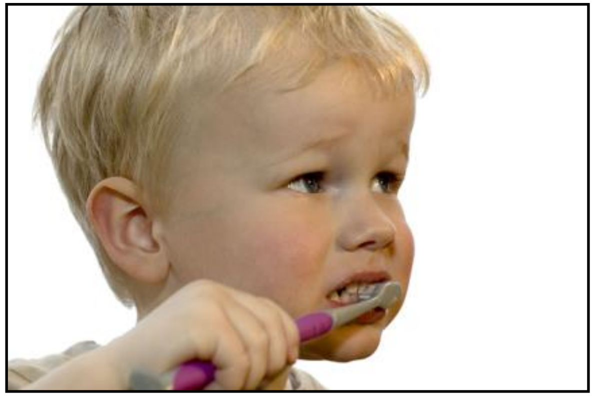


Gamma expansion



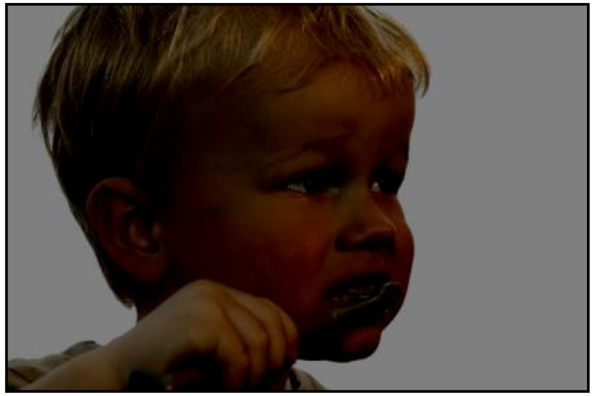
Pixel-wise operators

original



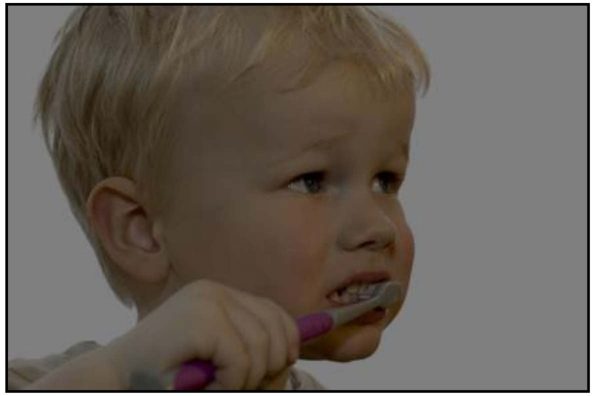
$$x$$

darken



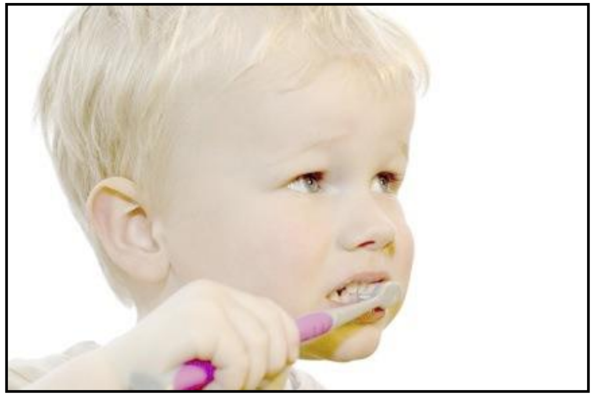
$$x - 128$$

lower contrast

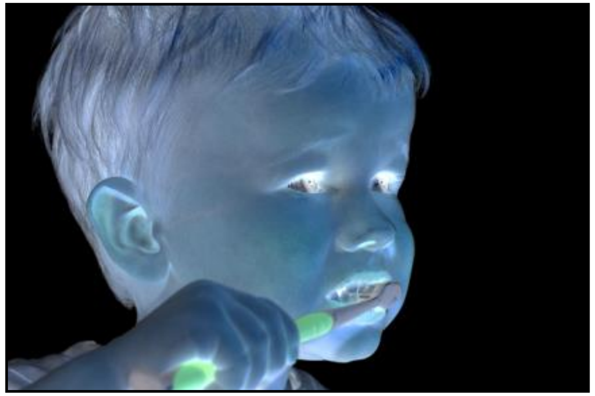


$$\frac{x}{2}$$

Gamma compression

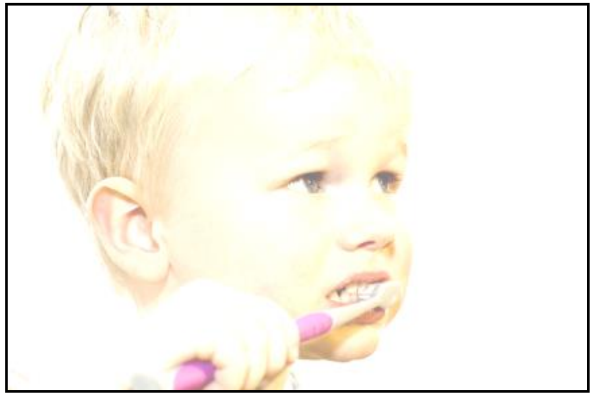


invert



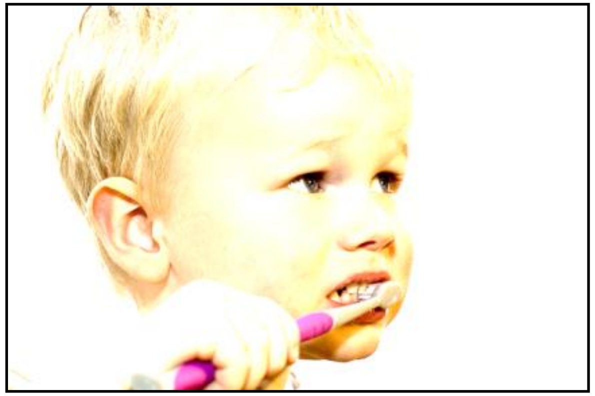
$$255 - x$$

lighten



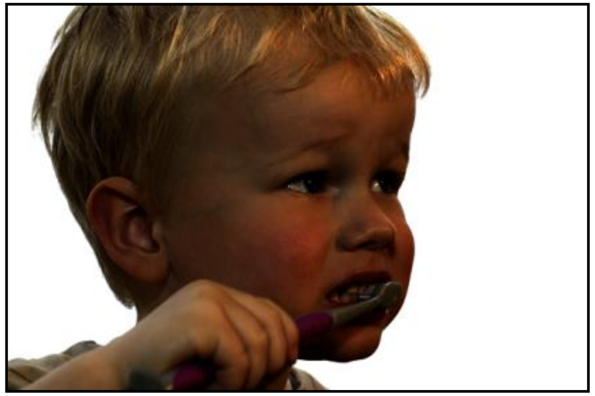
$$x + 128$$

raise contrast



$$x \times 2$$

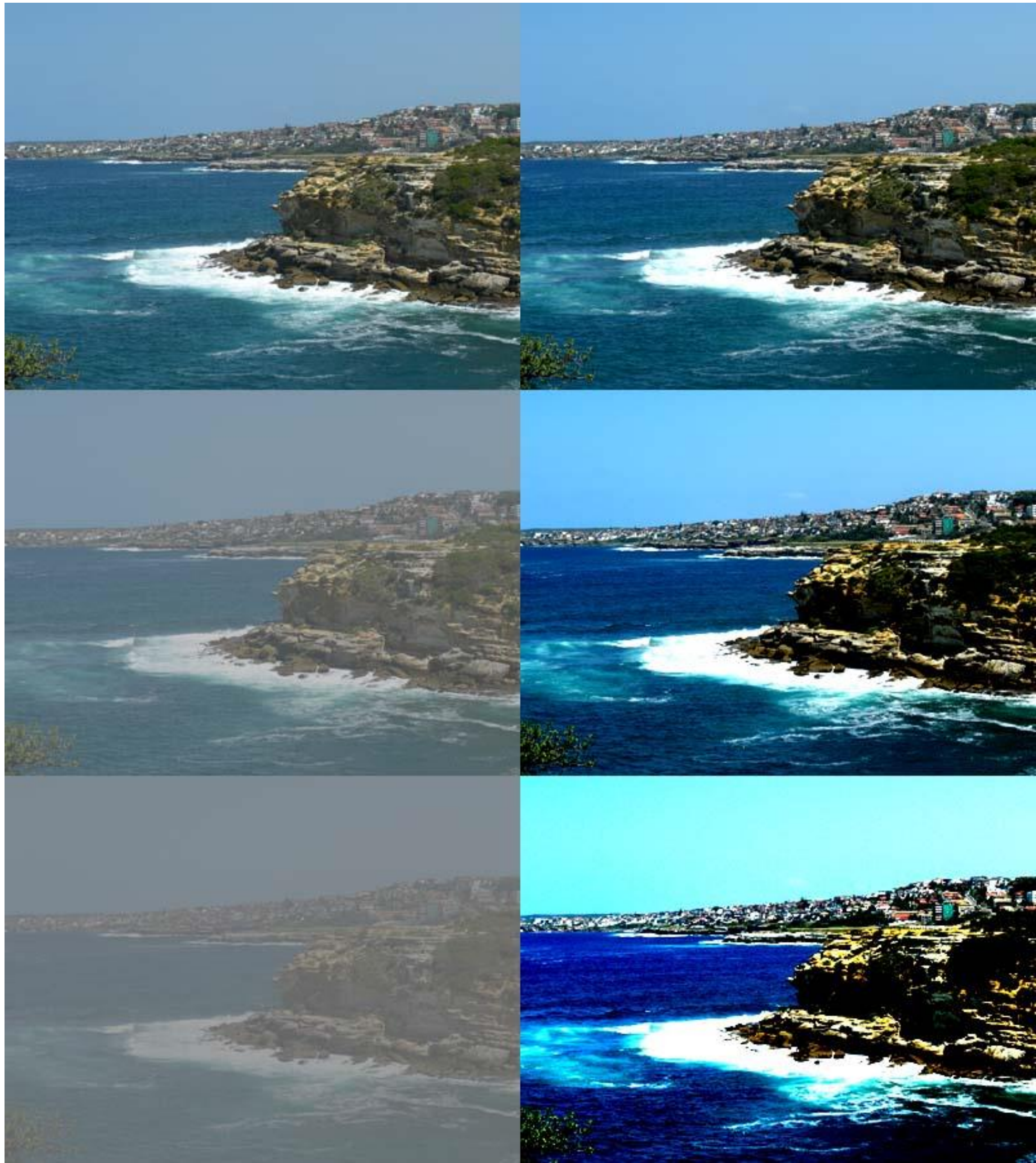
Gamma expansion



Contrast

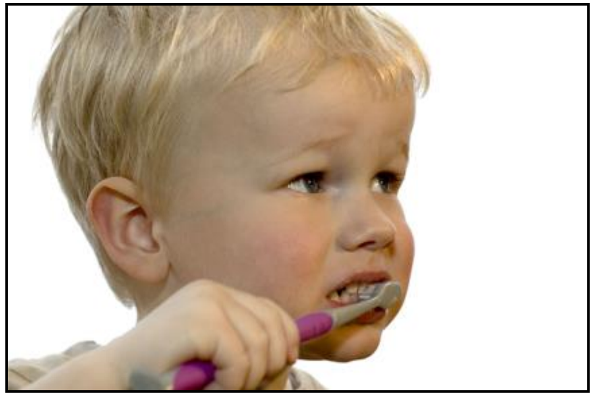
- **Contrast** in visual perception is the difference in appearance of two or more parts of a seen field.
- The human visual system is more sensitive to contrast than absolute luminance;
- **Contrast ratio**, or **dynamic range**, is the ratio between the largest and smallest values of the image or:

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



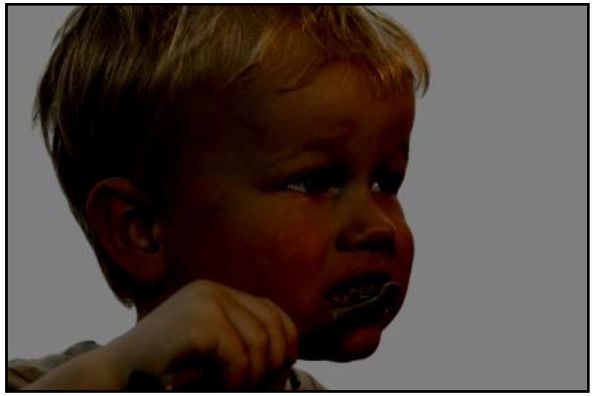
Pixel-wise operators

original



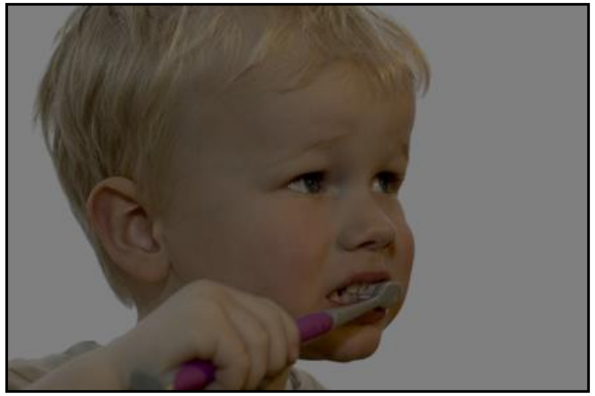
x

darken



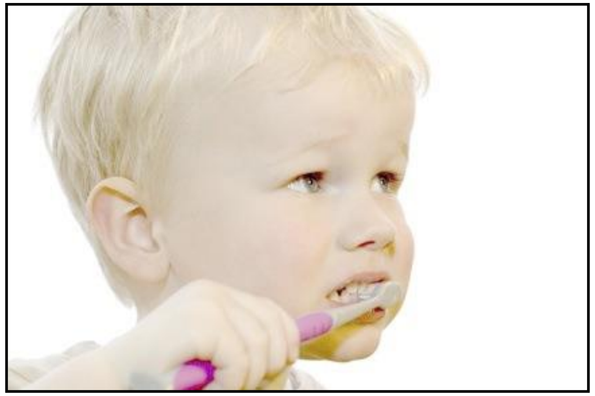
$x - 128$

lower contrast

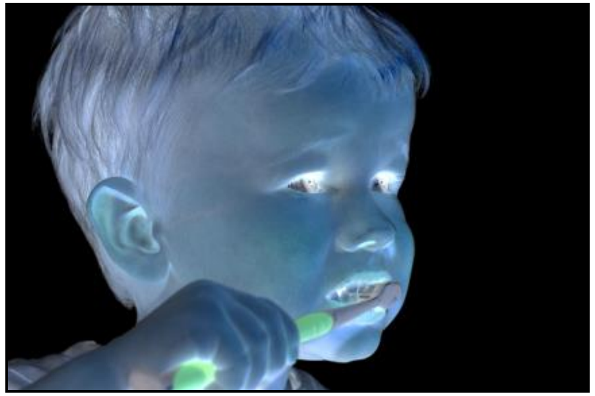


$\frac{x}{2}$

Gamma compression

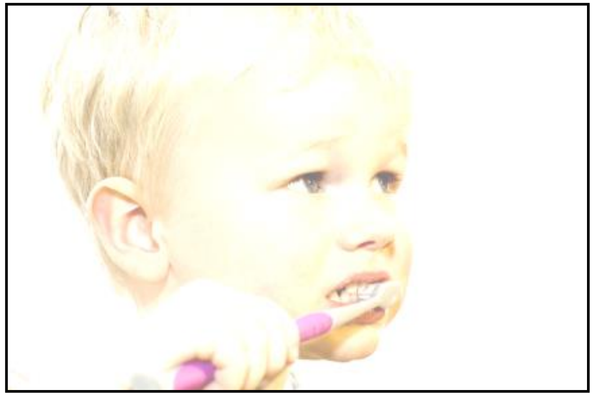


invert



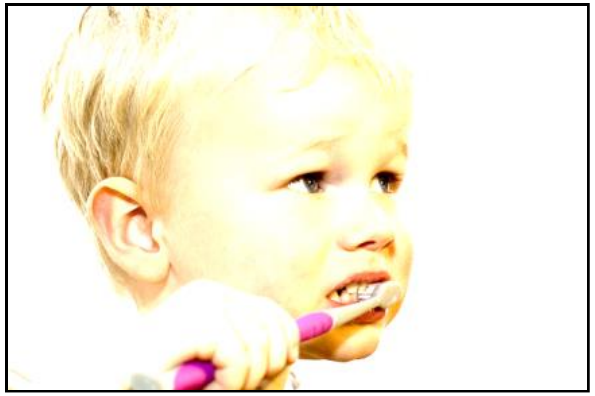
$255 - x$

lighten



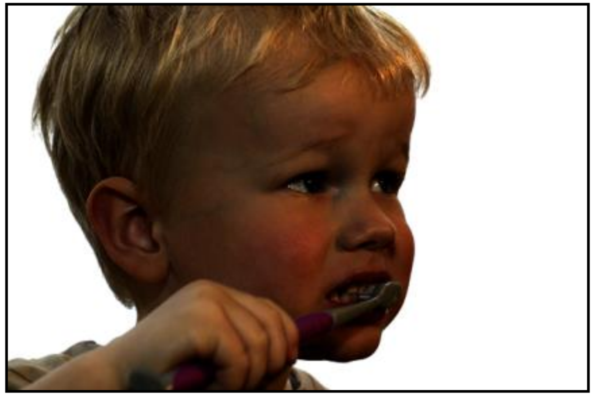
$x + 128$

raise contrast



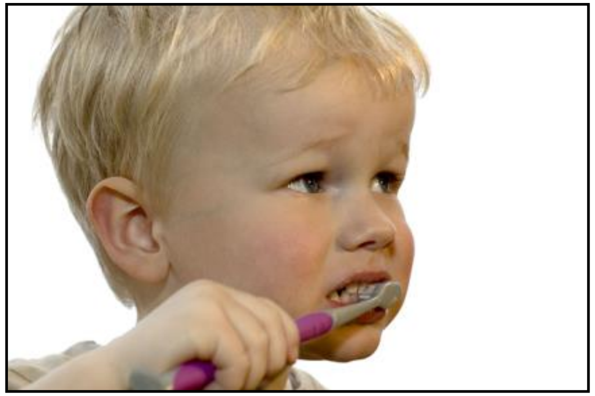
$x \times 2$

Gamma expansion



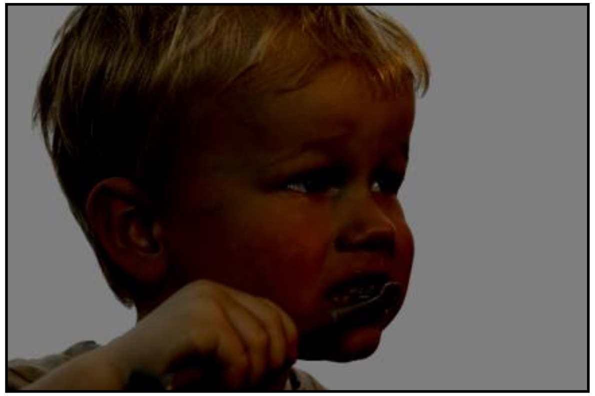
Pixel-wise operators

original



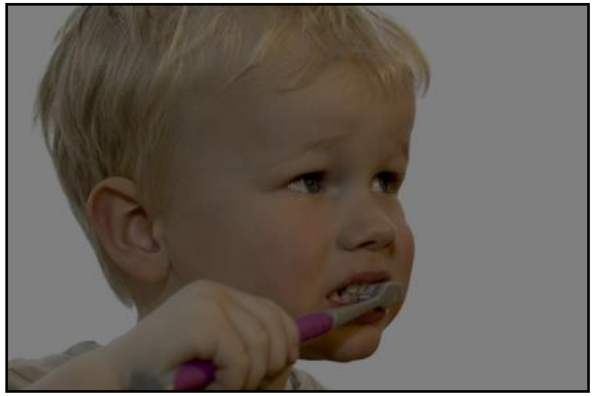
$$x$$

darken



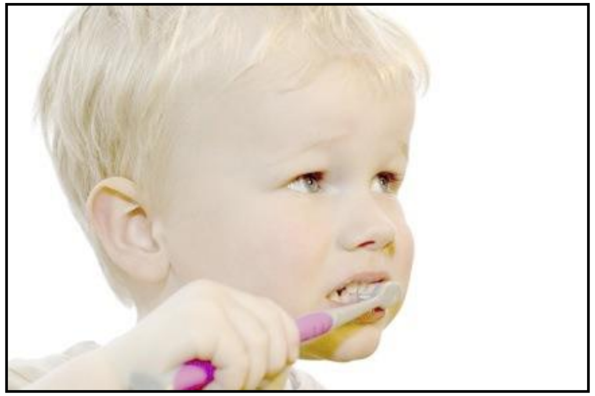
$$x - 128$$

lower contrast



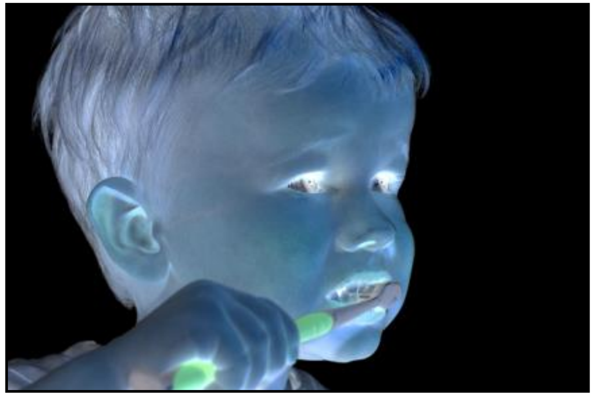
$$\frac{x}{2}$$

Gamma compression



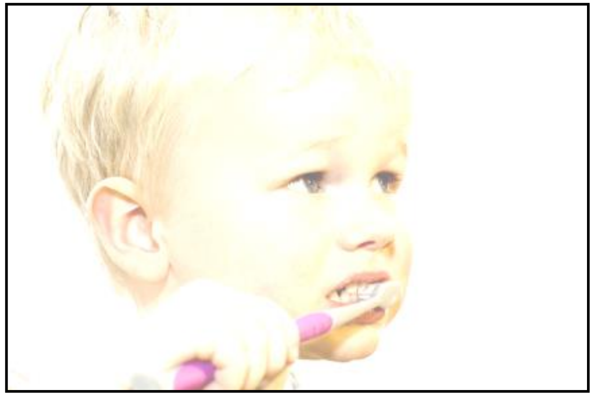
$$\left(\frac{x}{255}\right)^{1/3} \times 255$$

invert



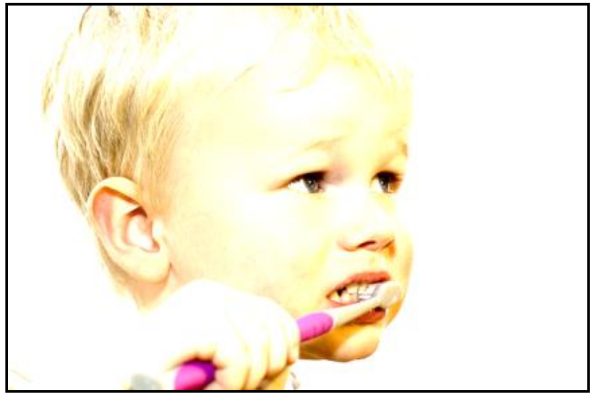
$$255 - x$$

lighten



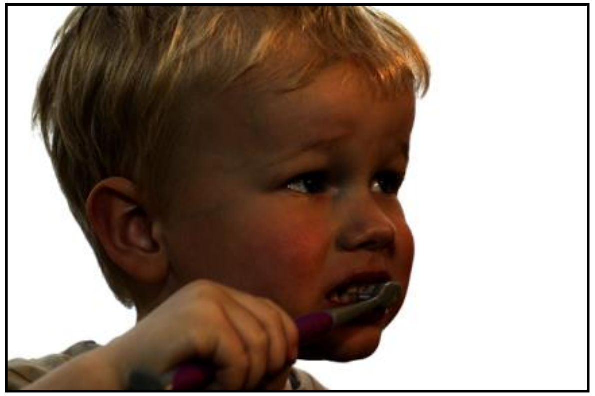
$$x + 128$$

raise contrast



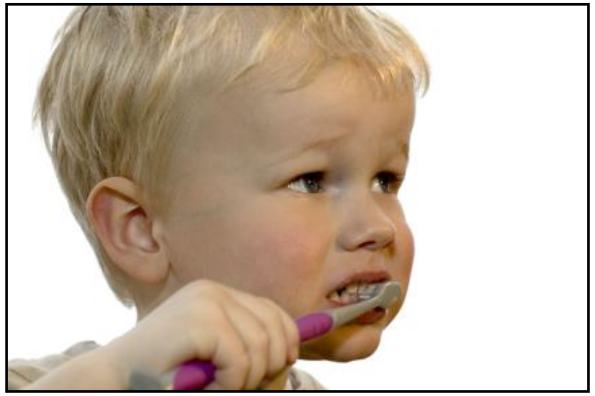
$$x \times 2$$

Gamma expansion



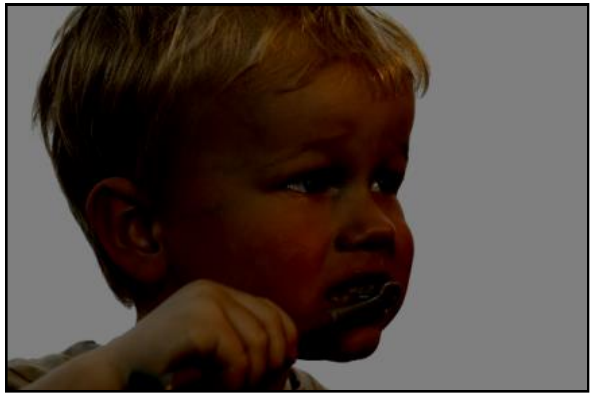
Pixel-wise operators

original



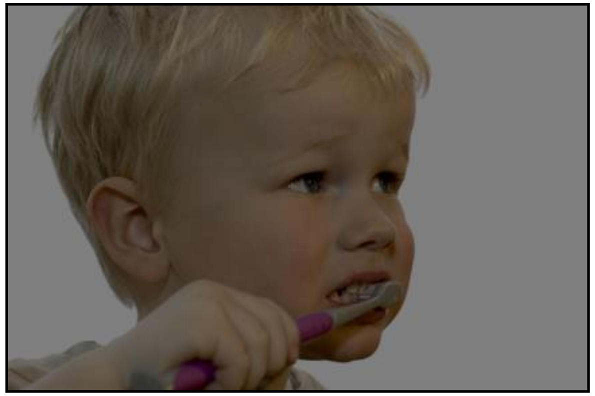
$$x$$

darken



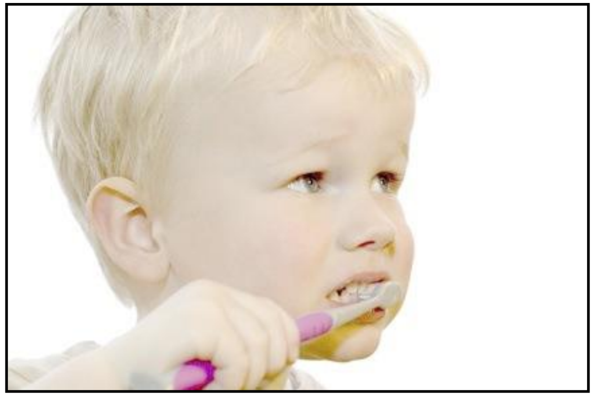
$$x - 128$$

lower contrast



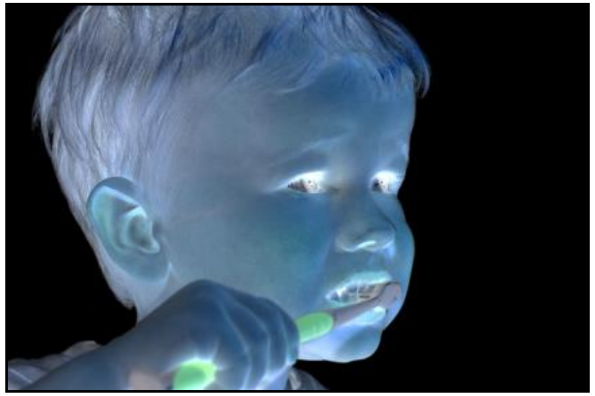
$$\frac{x}{2}$$

Gamma compression



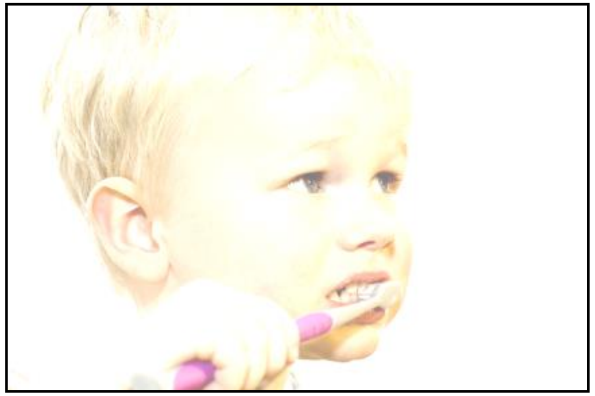
$$\left(\frac{x}{255}\right)^{1/3} \times 255$$

invert



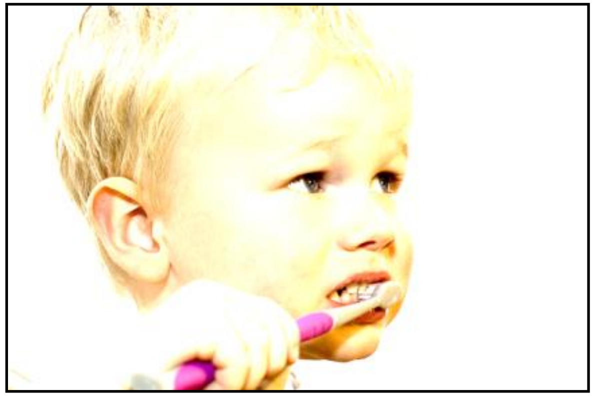
$$255 - x$$

lighten



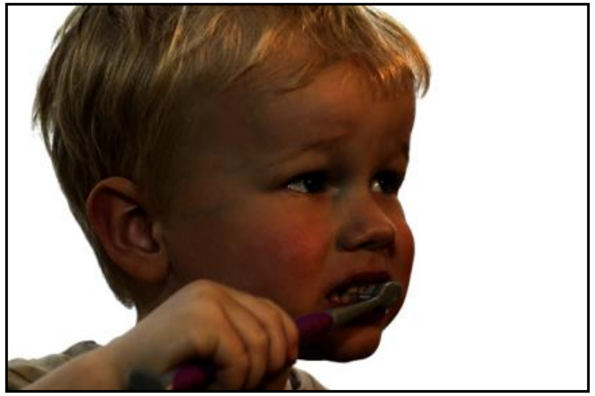
$$x + 128$$

raise contrast



$$x \times 2$$

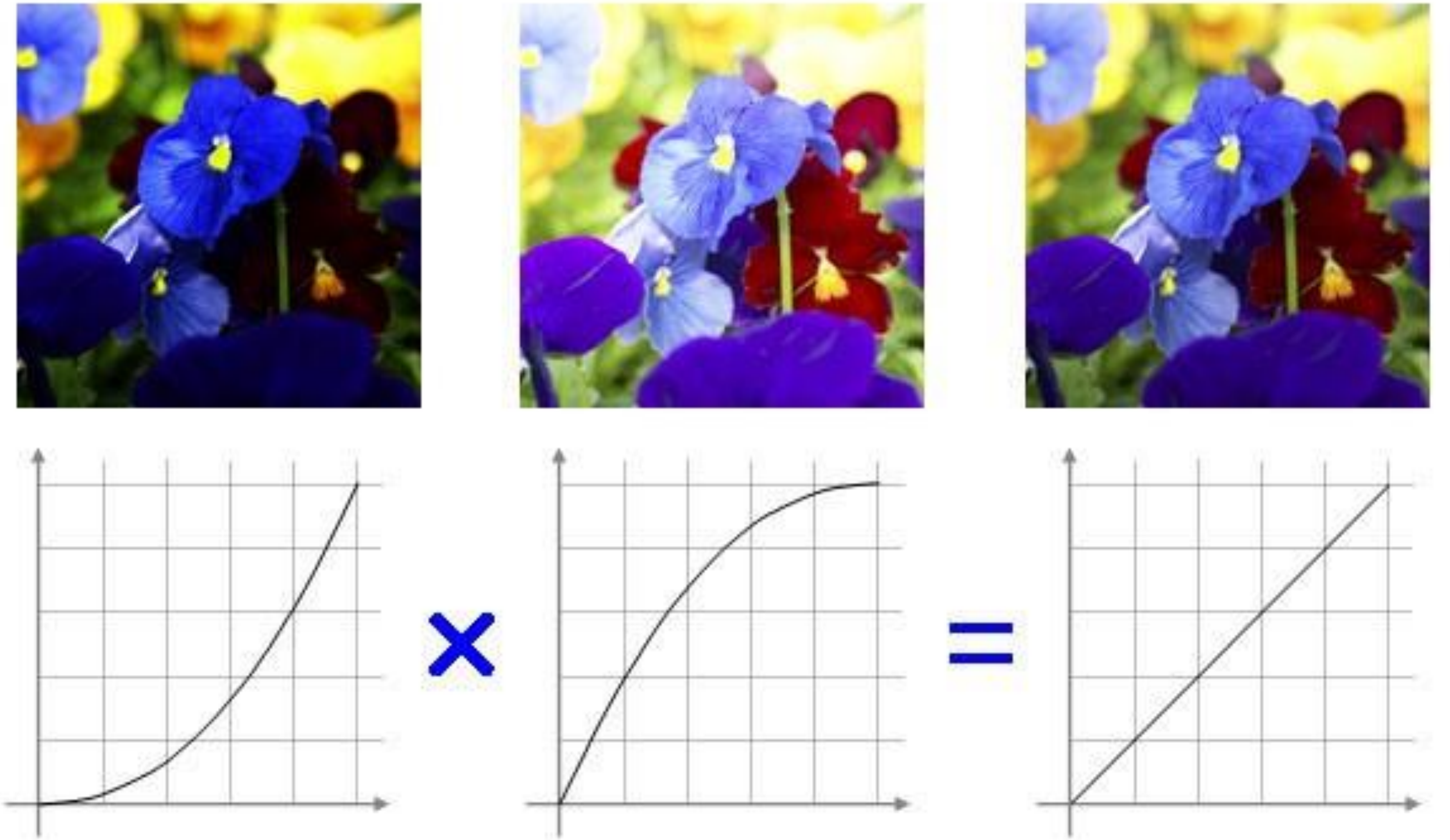
Gamma expansion



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

Gamma correction

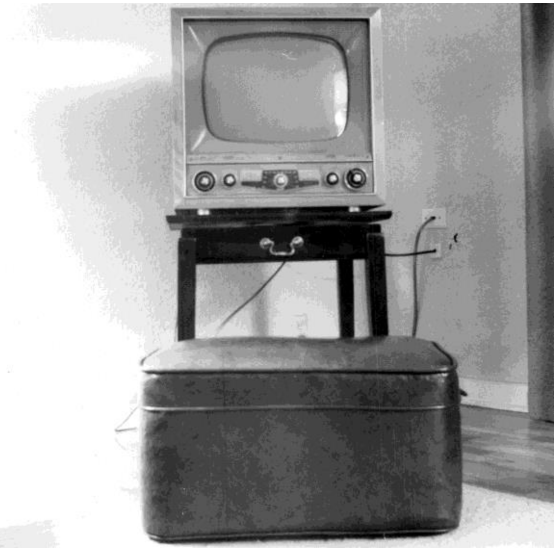
- Originally, Due to non-linearities in the old CRT televisions, intensities was seen different then they are.

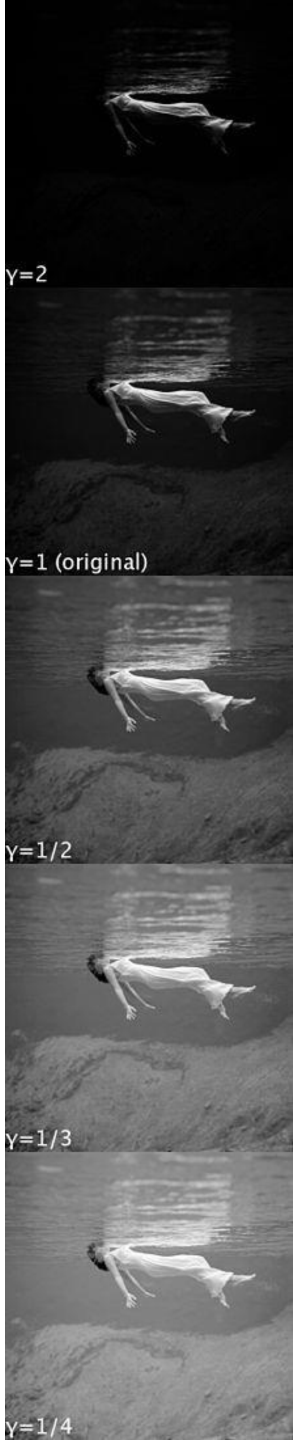


Gamma characteristics of monitors

Color information adjusted to match gamma characteristics

Color handling approaching the “y = x” idealcs



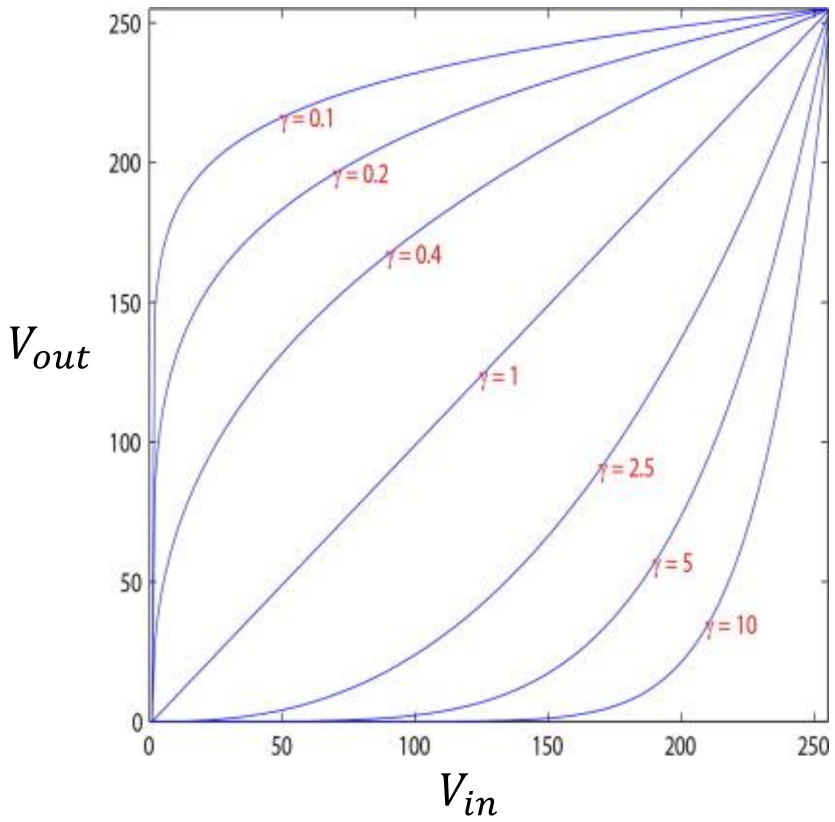


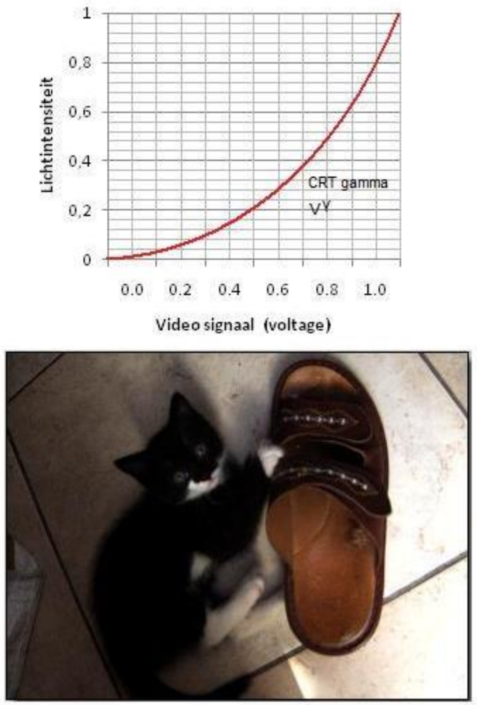
Gamma correction

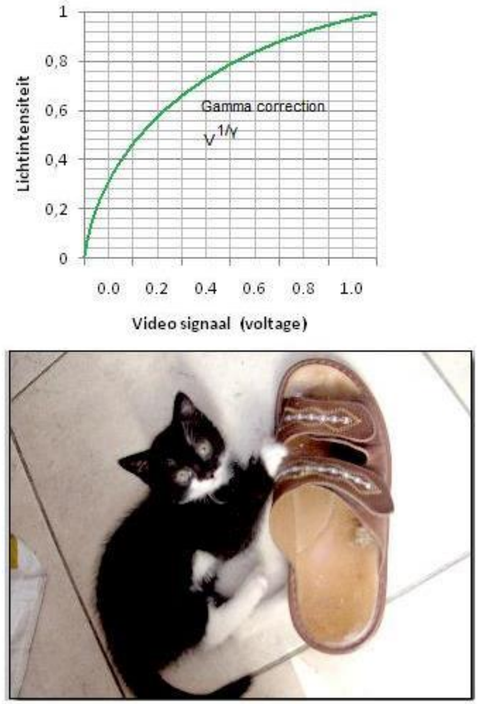
- To correct this non-linear transformation, gamma correction was done:

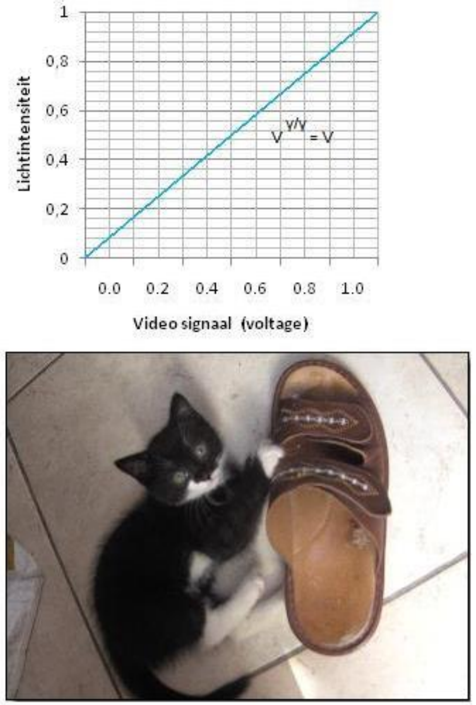
$$\boxed{V_{out} = \left(\frac{V_{in}}{255}\right)^{\gamma} \cdot 255} \qquad (V_{in}, V_{out} \in \{0,1,\dots,255\})$$

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









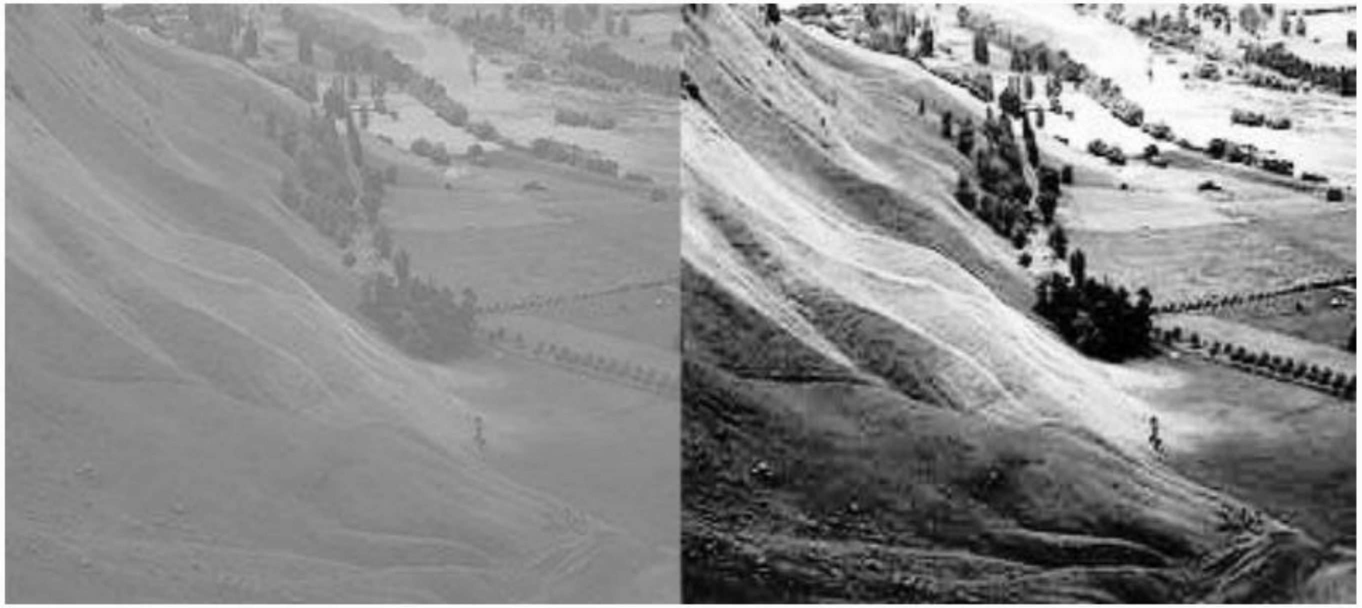
Some more point- wise operators

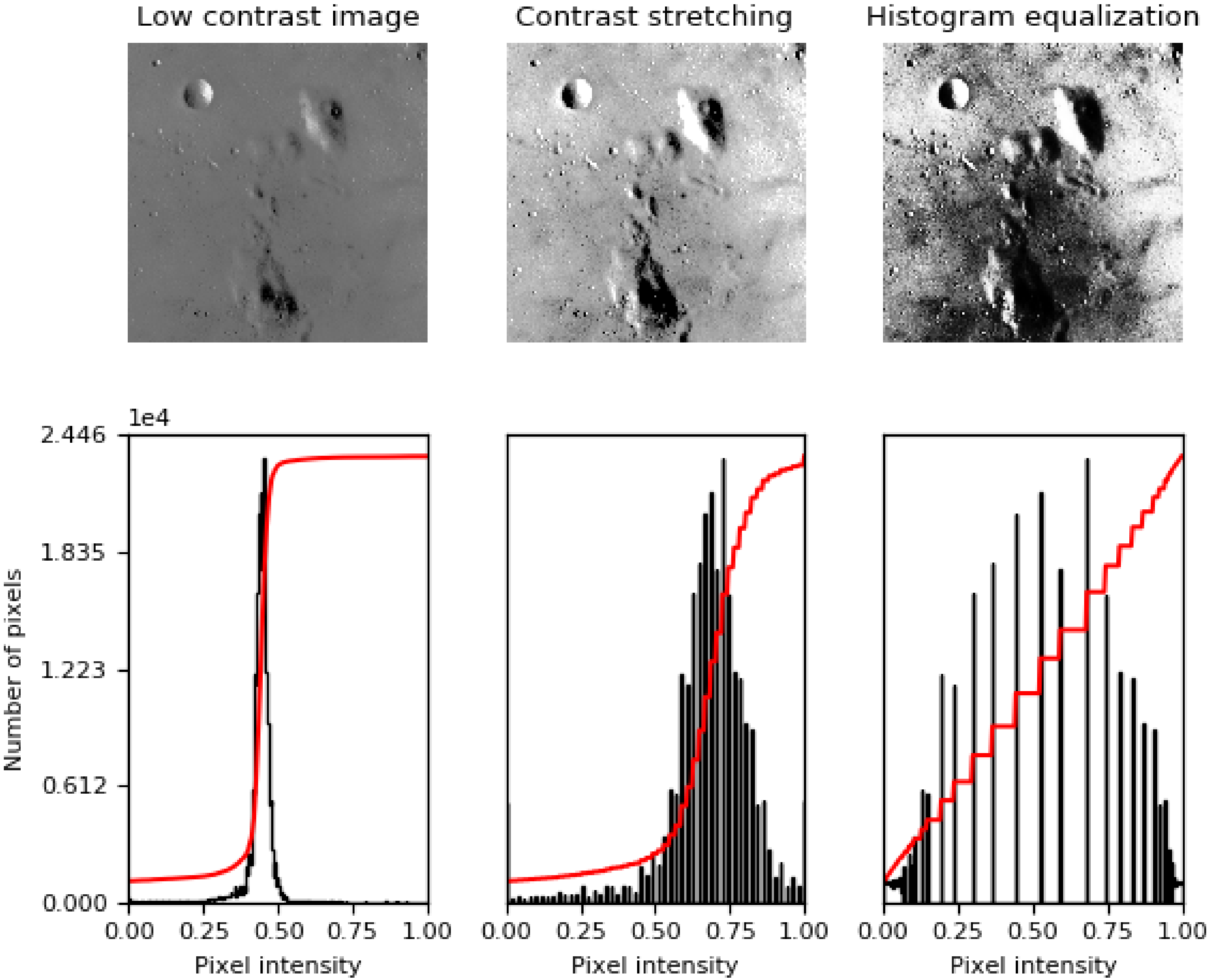
contents

- 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.**

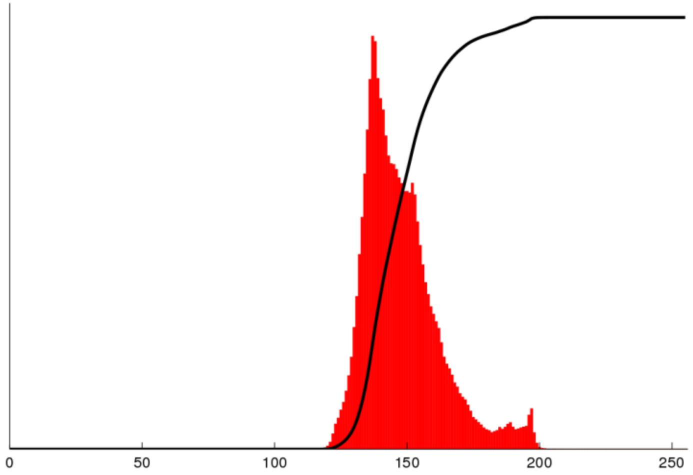




Low contrast image
Contrast stretching
Histogram equalization
1e4
2.446
1.835
1.223
0.612
0.000
Number of pixels
0.00
0.25
0.50
0.75
1.00
Pixel intensity
0.00
0.25
0.50
0.75
1.00
Pixel intensity
0.00
0.25
0.50
0.75
1.00
Pixel intensity

Histogram equalization

- A histogram is a discrete form representation of the distribution of numerical data.
- We will assume at first that our image is continues 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:
  - $.\lim_{x\to-\infty} F_X(x) = 0, \quad \lim_{x\to+\infty} F_X(x) = 1$
  - Monotonically non decreasing.

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

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

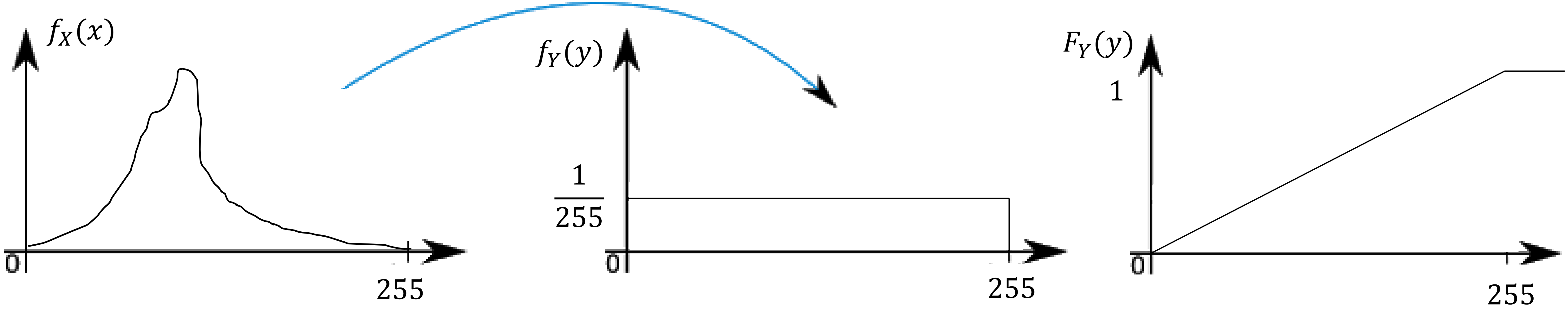
PDF definition- Wikipedia

- **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:

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



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]:

$$P(Y \leq y) = y$$
$$P(T(X) \leq y) = y$$
$$assuming\ T\ is\ invertible\text{:}\quad P\left(X \leq T^{-1}(y)\right) = y$$
$$change\ of\ variables\ z = T^{-1}(y)\text{:}\quad P(X \leq z) = \mathrm{T(z)}$$

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

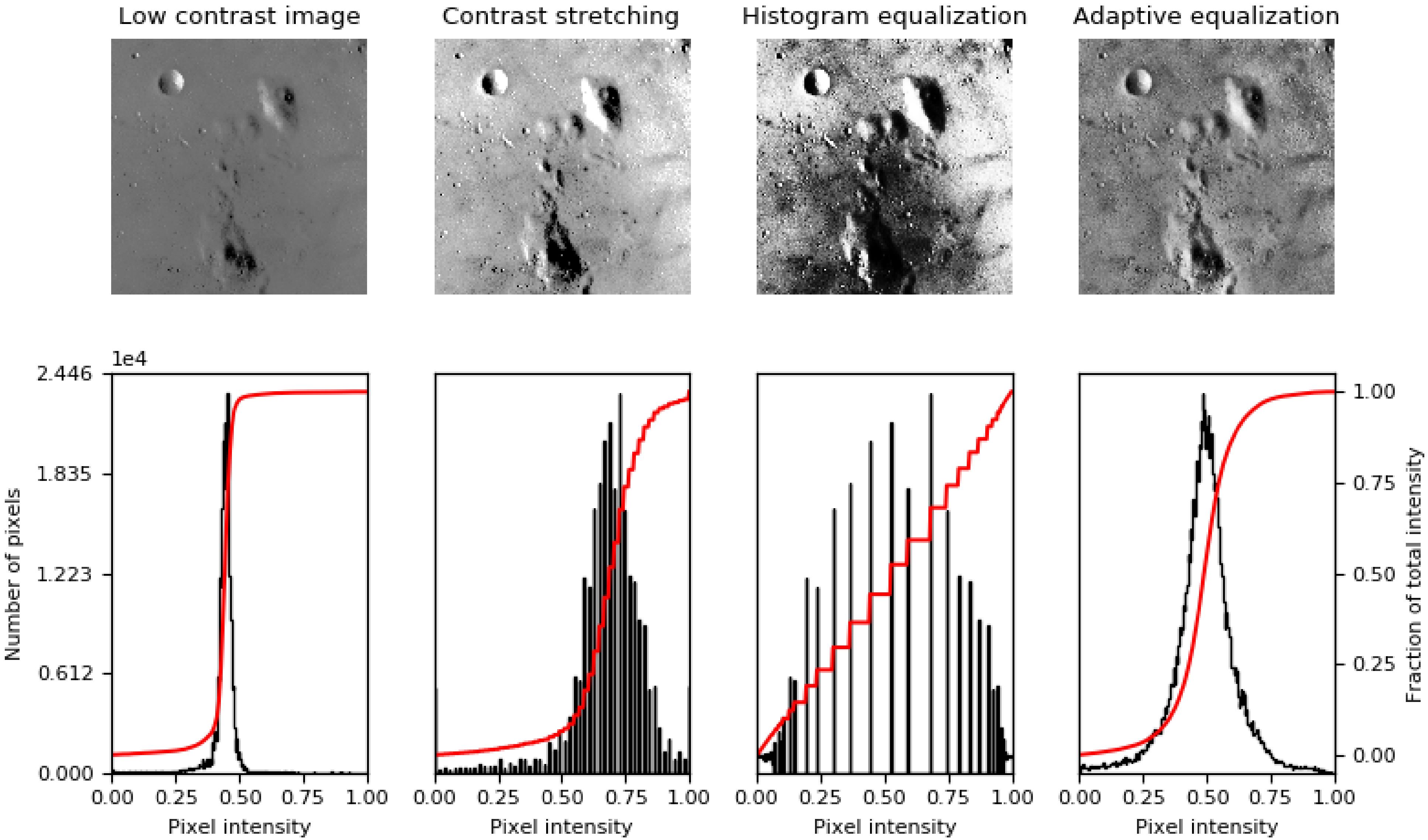
Back to histogram equalization

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

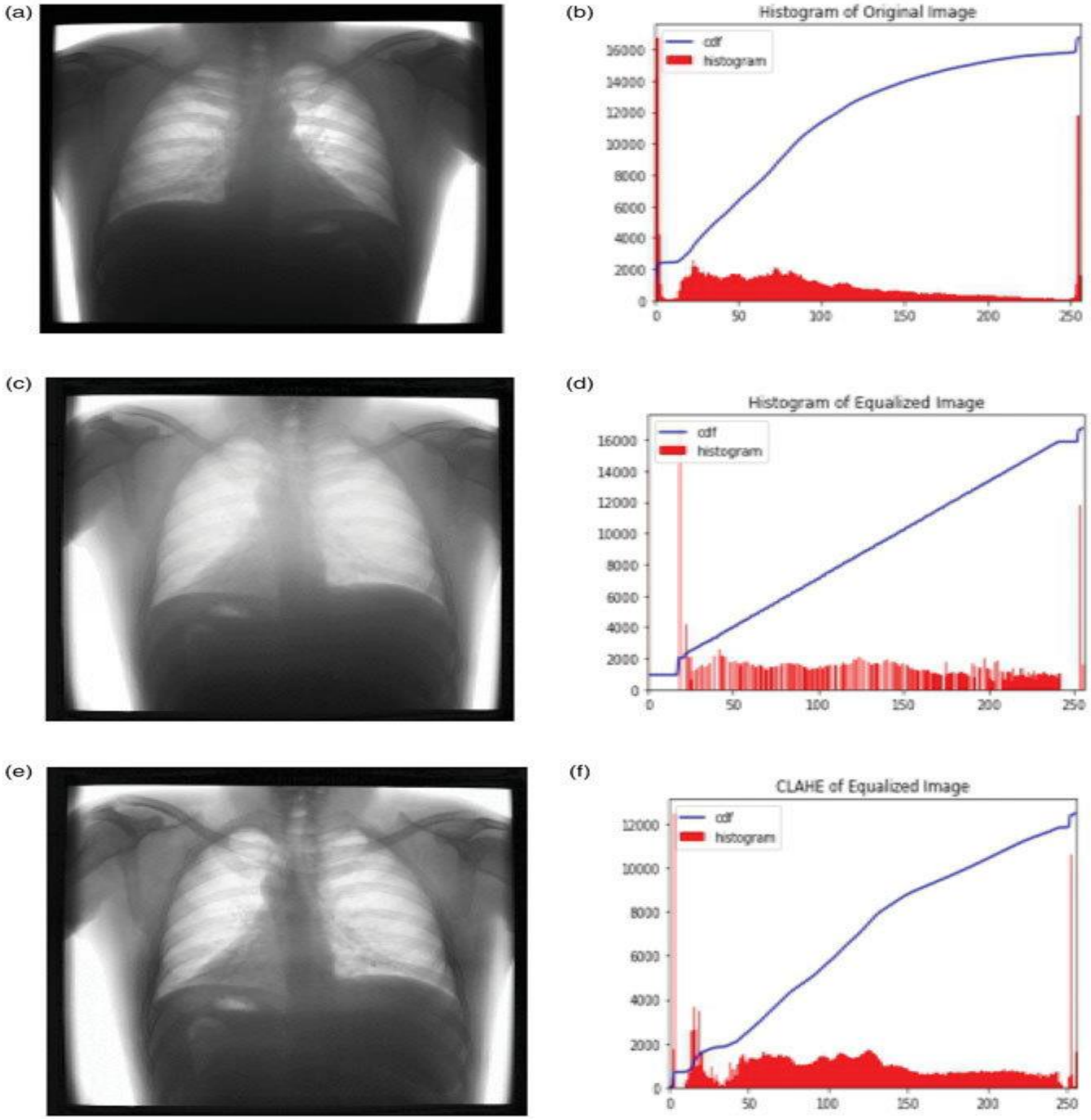
- Build histogram of a given image.
- To make the histogram act like a discrete pdf, we will normalize the PDF: divide each bin by the sum of all bins.
- Build the discrete CDF.
- Un-normalize the CDF and round the results back to uint8:

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

Other variants of histogram equalization



Other variants of histogram equalization

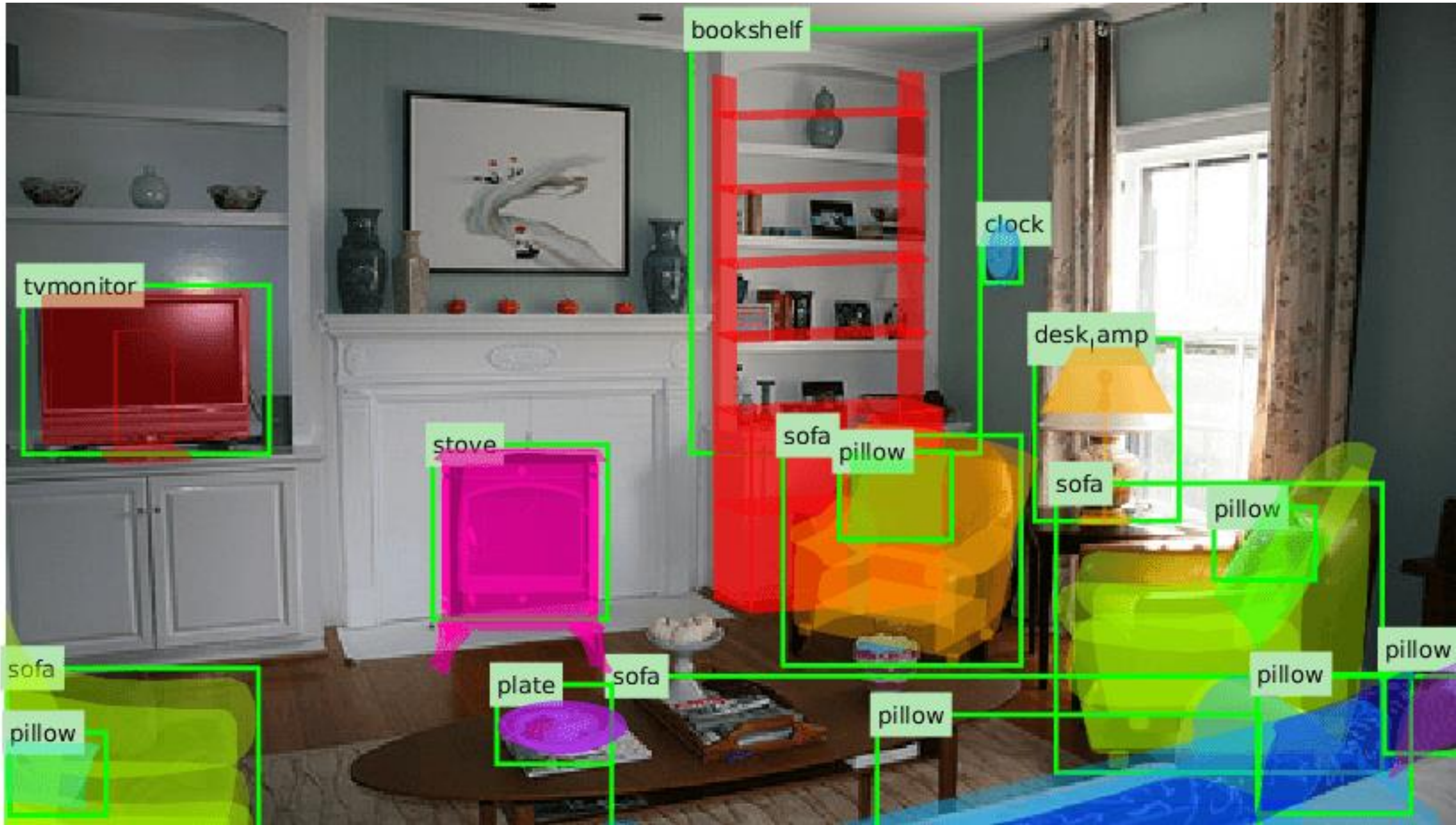


contents

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

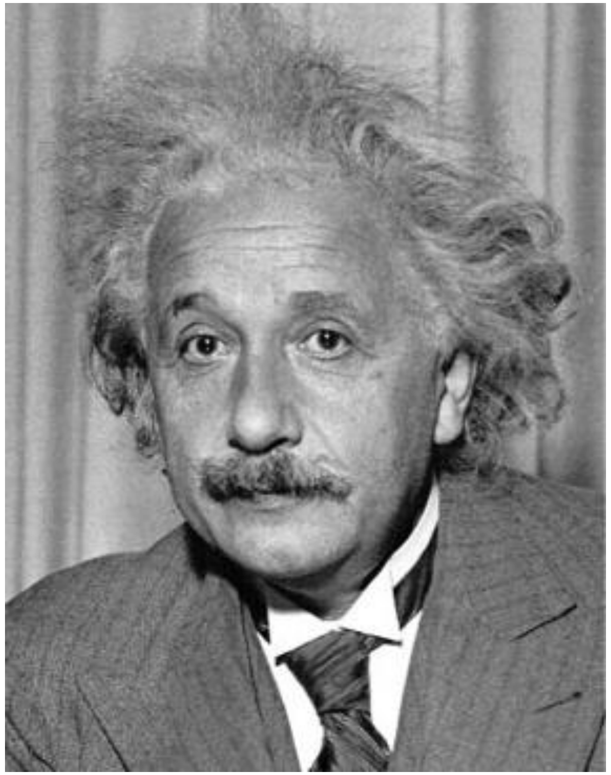
Template matching

- 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



CC – cross correlation

How do we detect the template in he following image?



filter

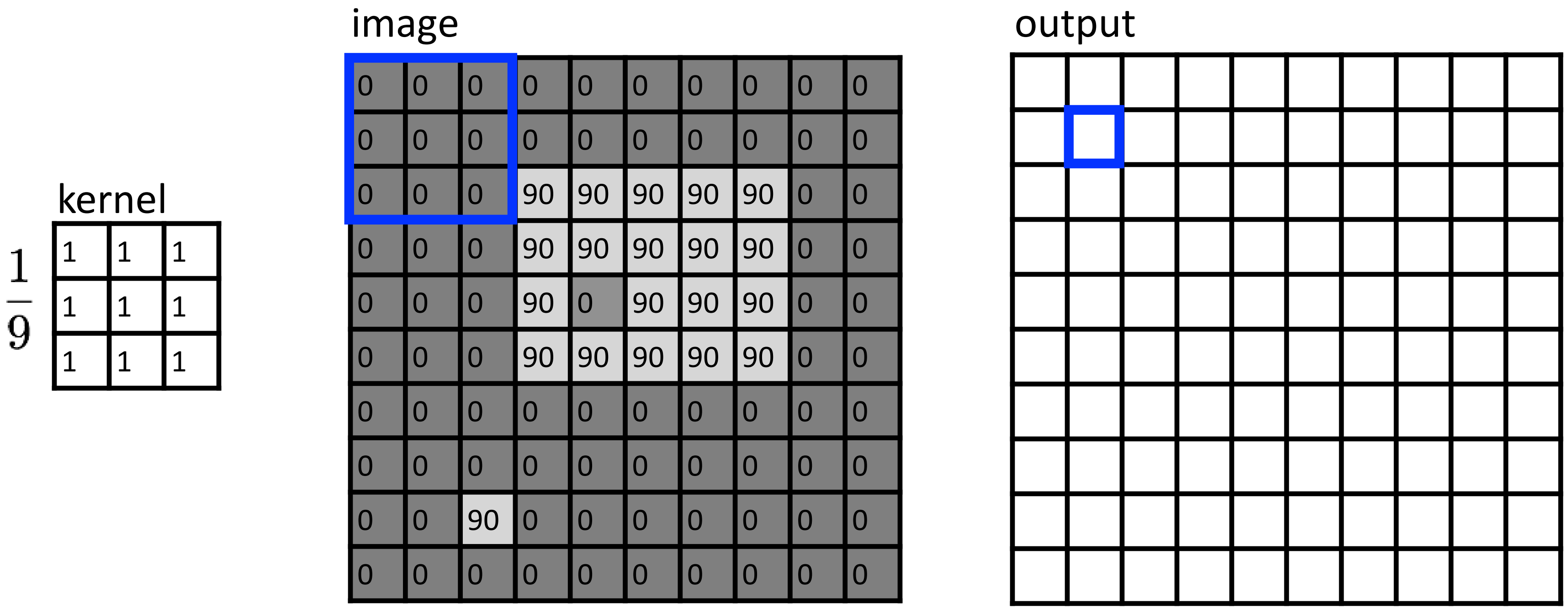
output

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

image

What will the output look like?

Run the filter



kernel

$\frac{1}{9}$

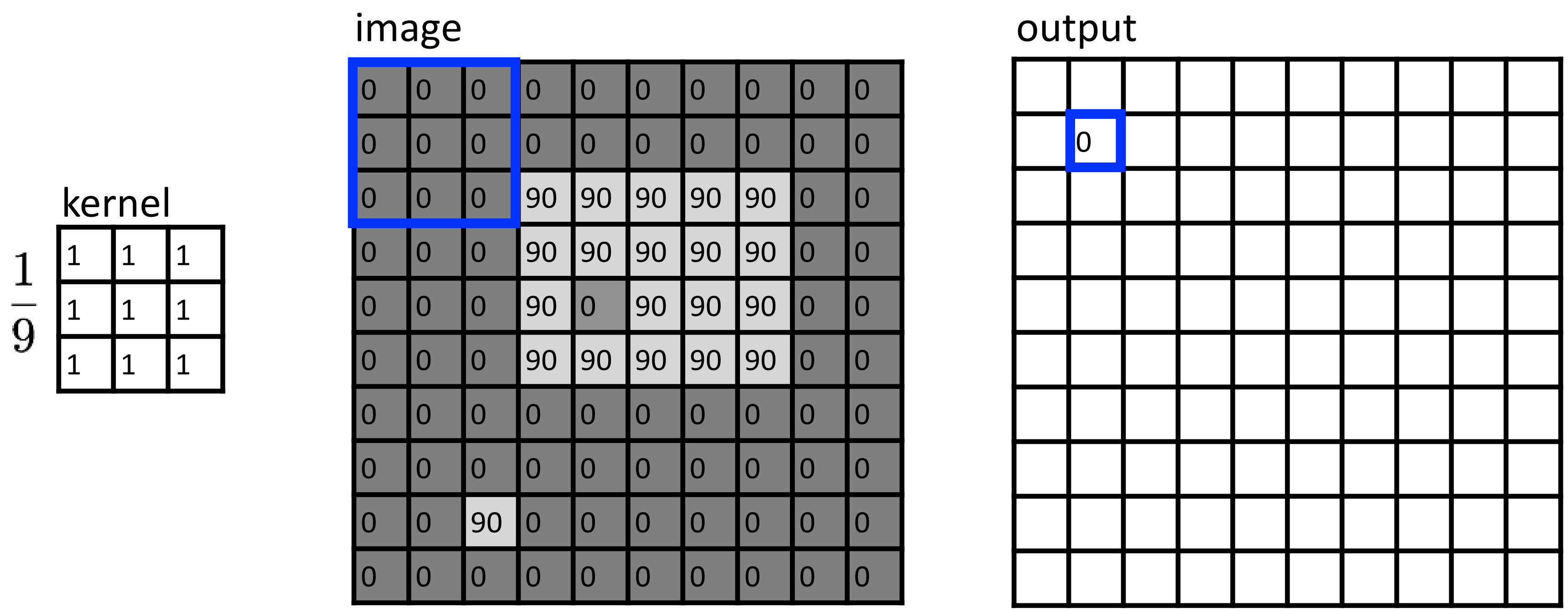
1	1	1
1	1	1
1	1	1

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

output

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

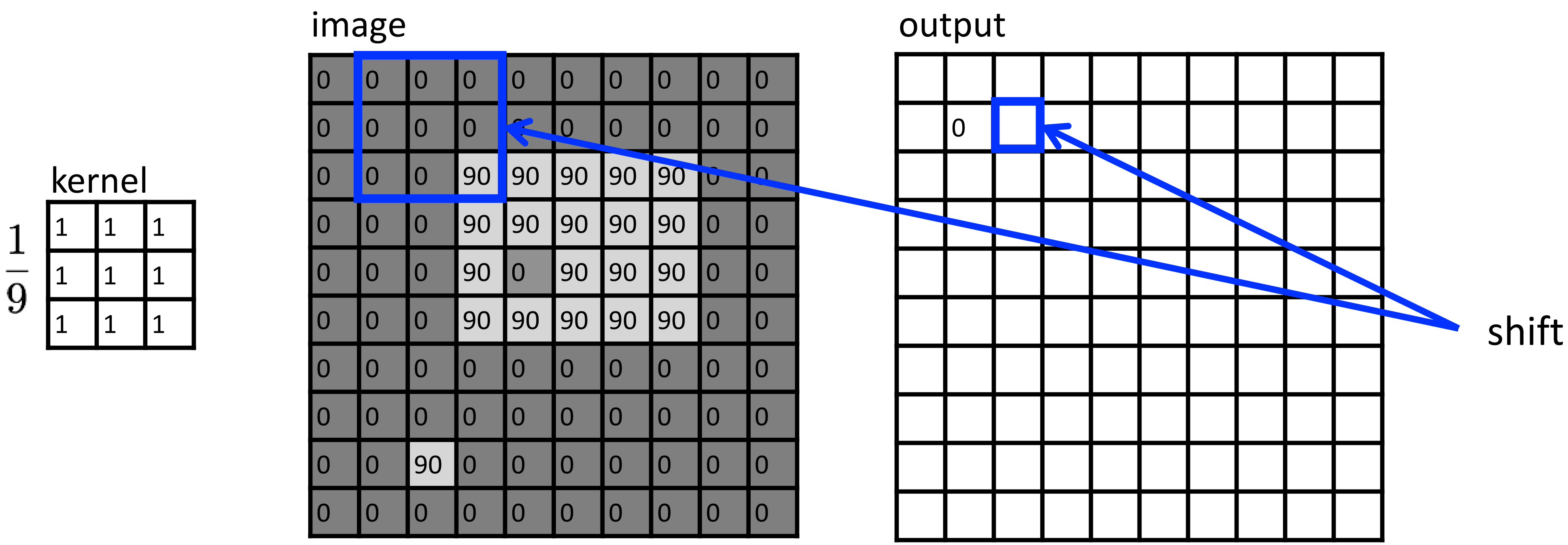
image

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

output

	0								

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

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

output

	0								

Run the filter

kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

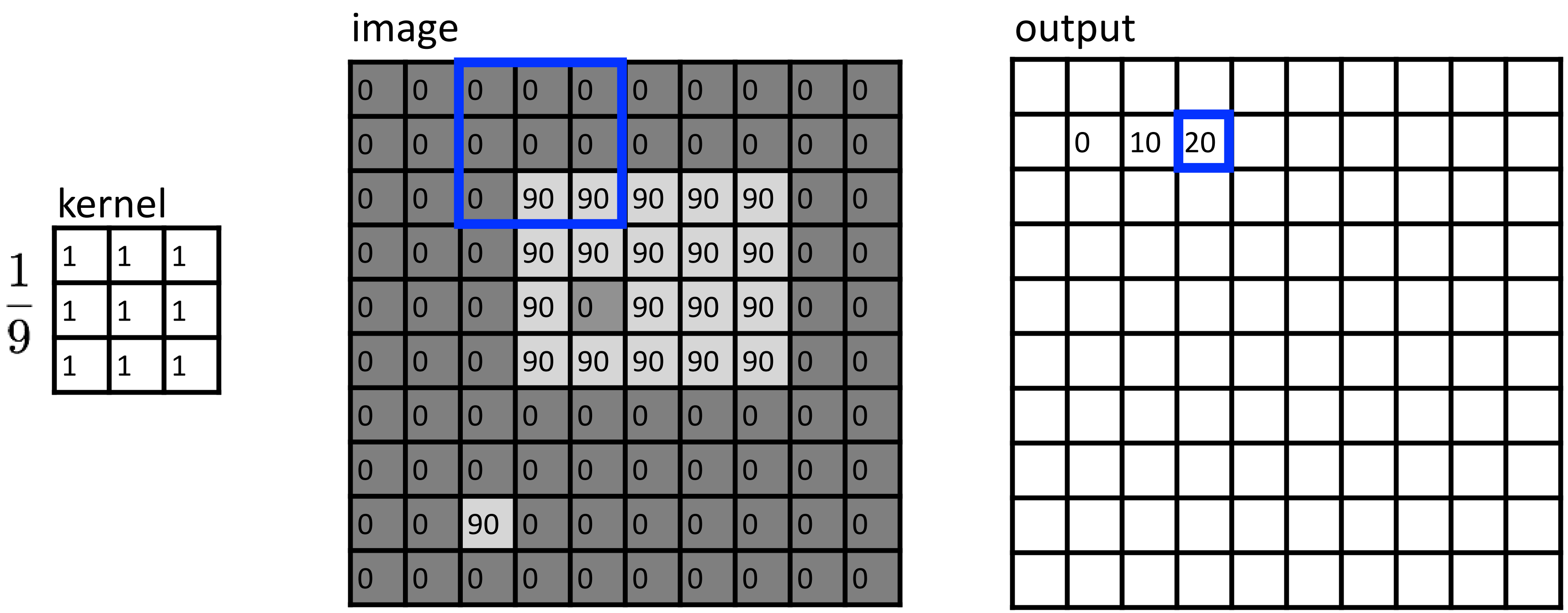
image

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

output

	0	10							

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

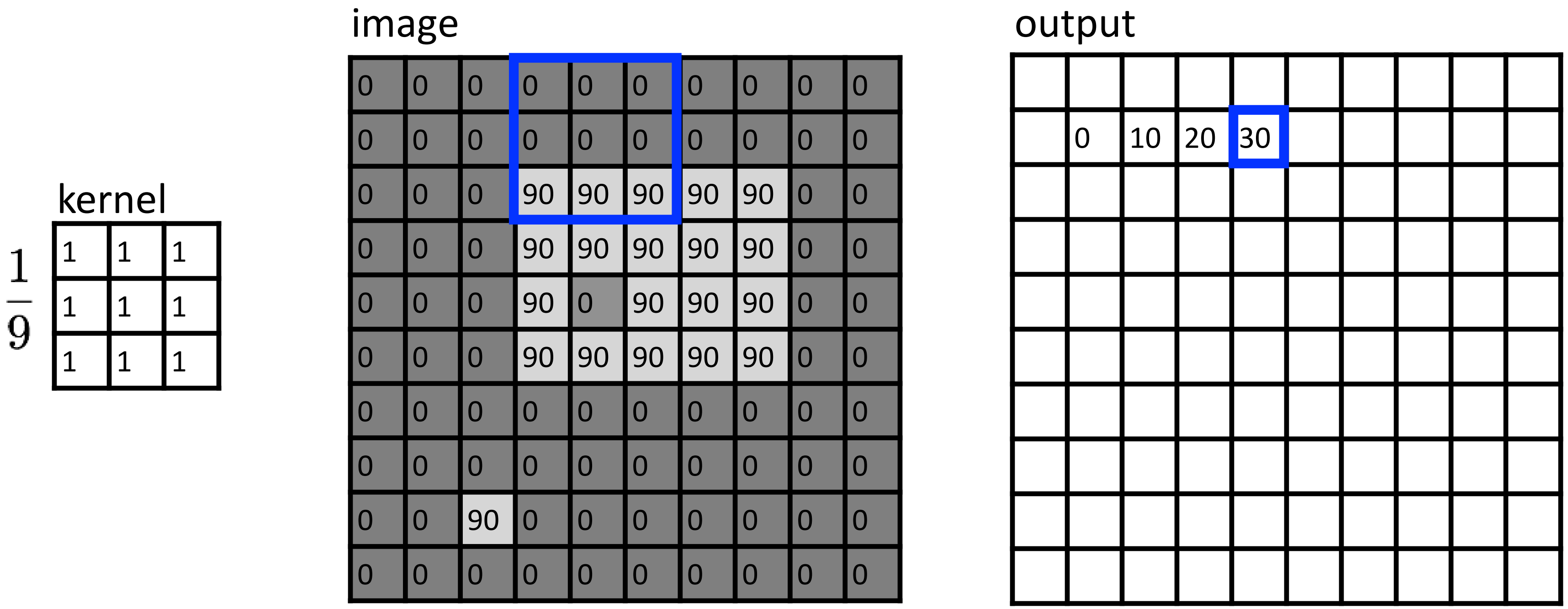
image

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

output

	0	10	20						

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

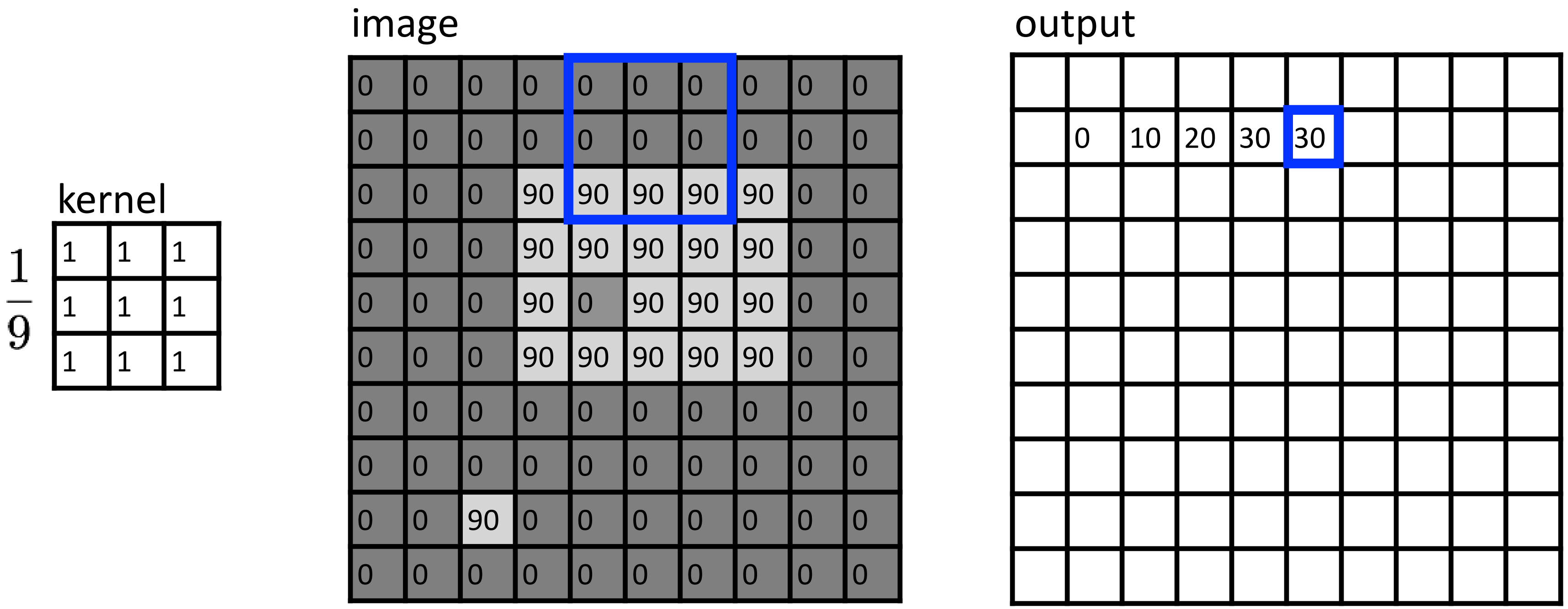
image

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

output

	0	10	20	30					

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

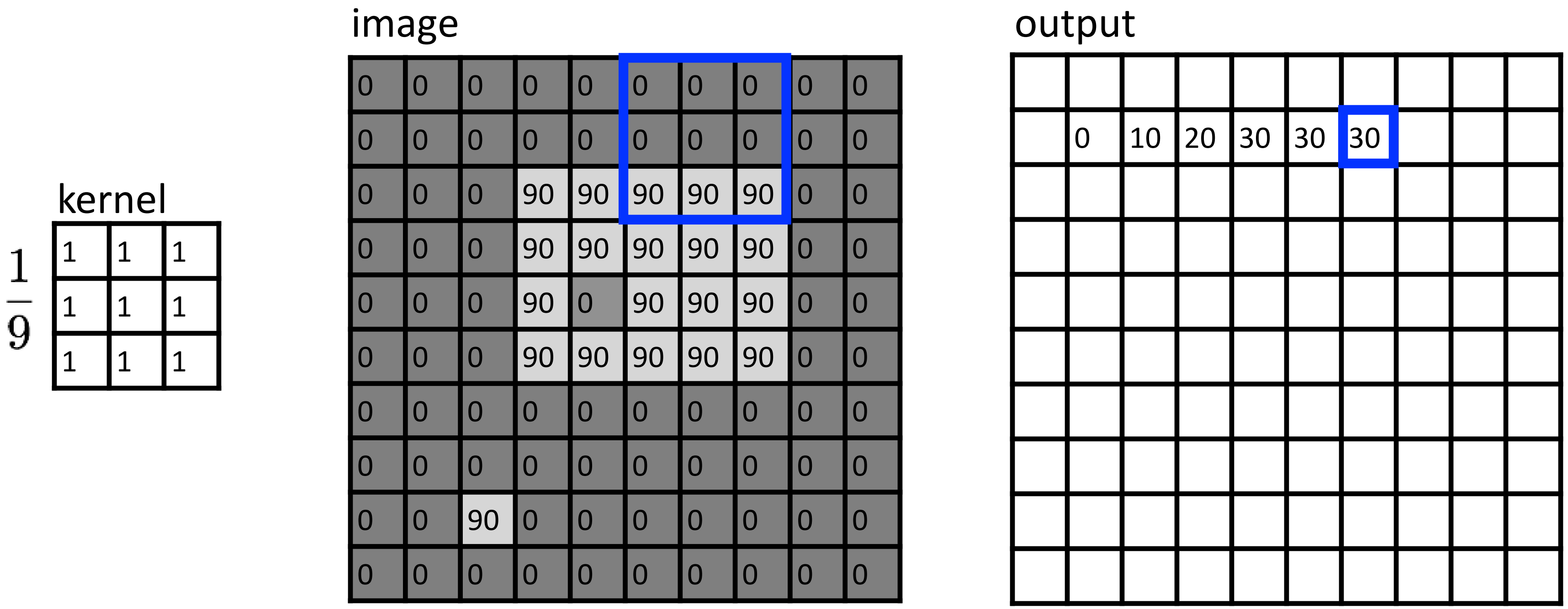
image

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

output

	0	10	20	30	30				

Run the filter



kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

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

output

	0	10	20	30	30	30			

… and the result is

kernel

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image

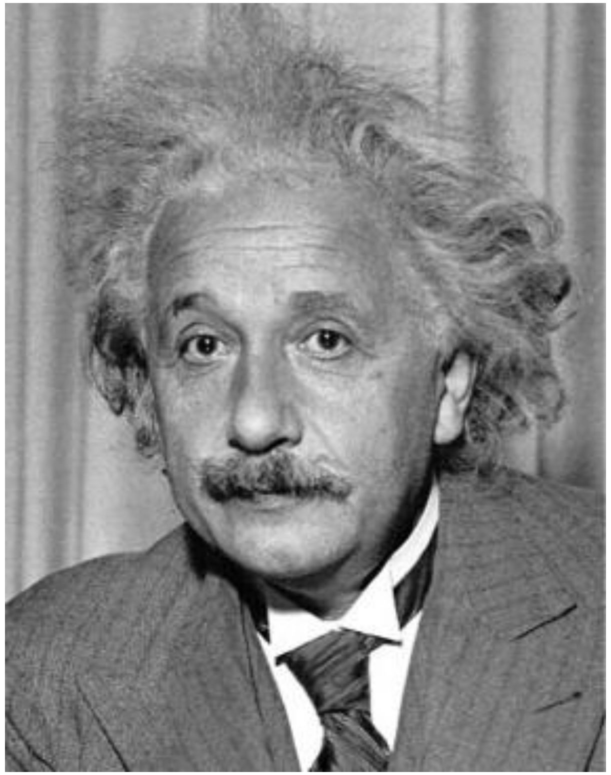
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

output

	0	10	20	30	30	30	20	10	
	0	20	40	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	

CC – cross correlation

How do we detect the template in he following image?



filter

output

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

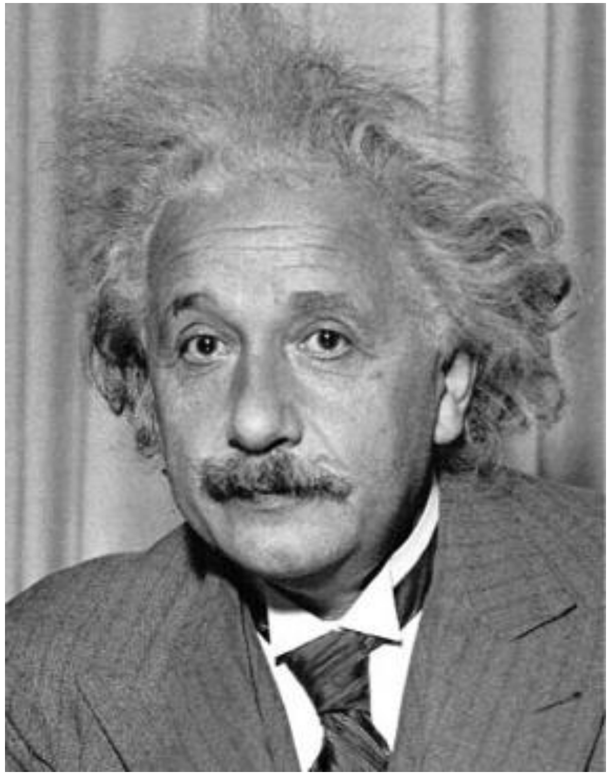
image

What will the output look like?

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

CC – cross correlation

How do we detect the template in he following image?

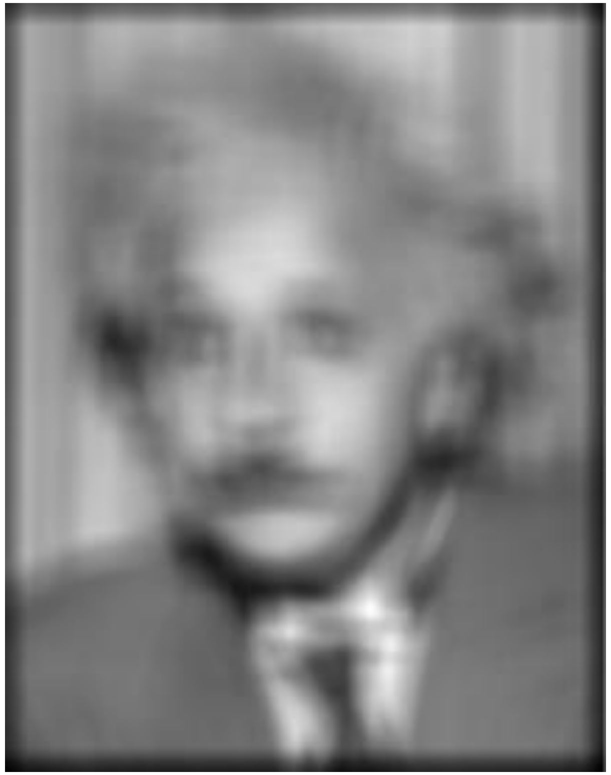


filter

output

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

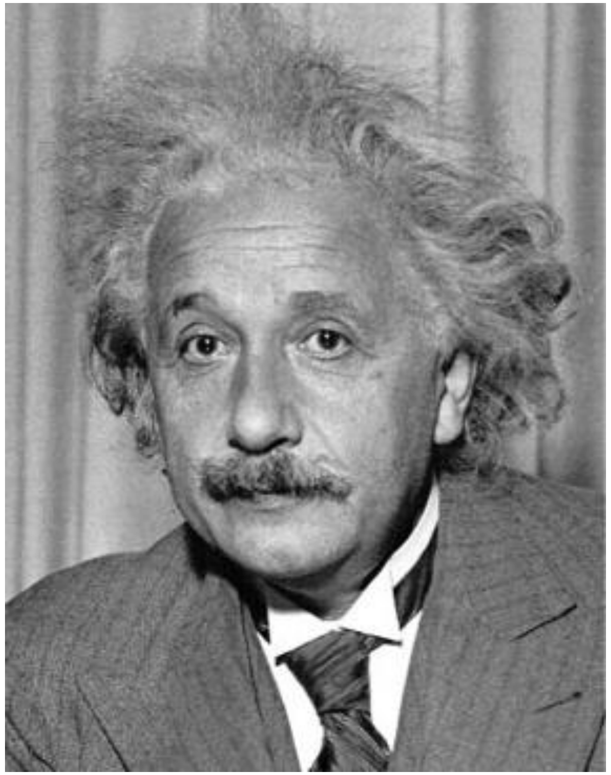
image



Is this good for template matching?

CC – cross correlation

How do we detect the template in he following image?

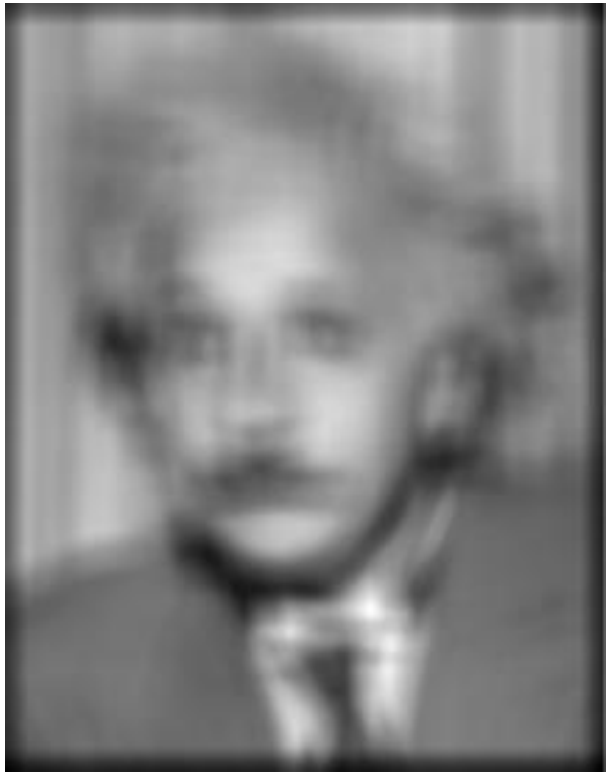


filter

output

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

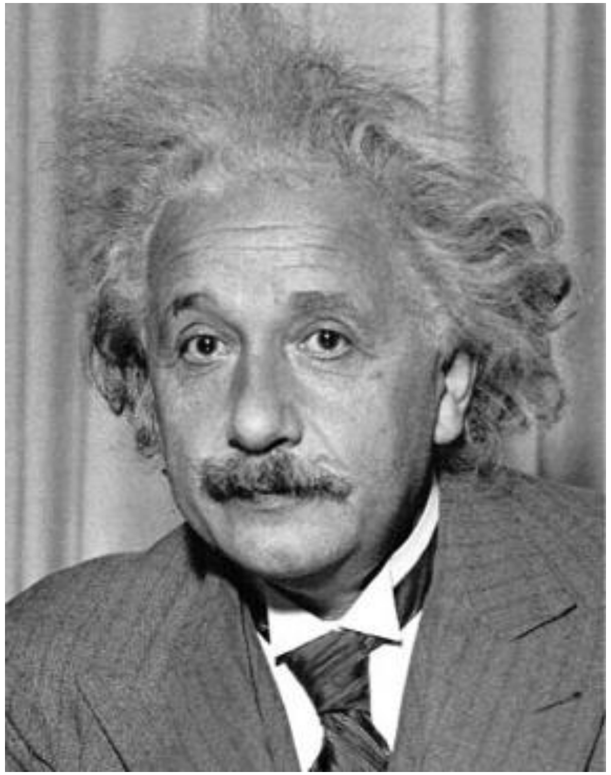
image



Increases for higher local intensities.

Zero mean cross correlation

How do we detect the template in he following image?



filter

template mean

output

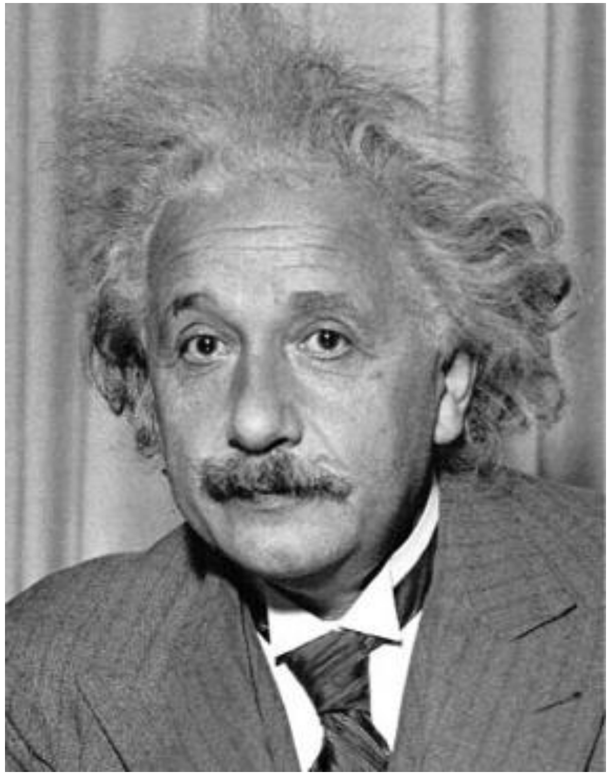
$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[m+k, n+l]$$

image

What will the output look like?

Zero mean cross correlation

How do we detect the template in he following image?



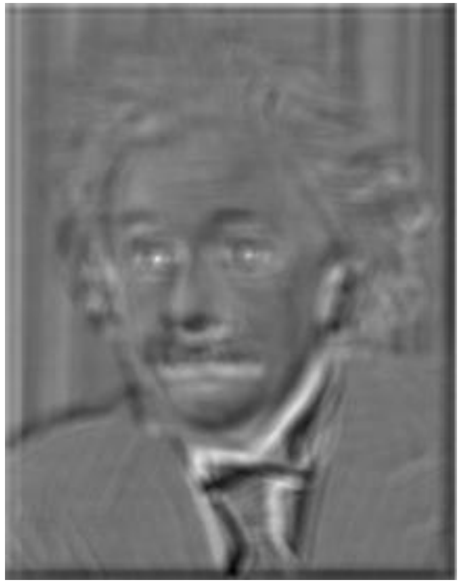
filter template mean

output

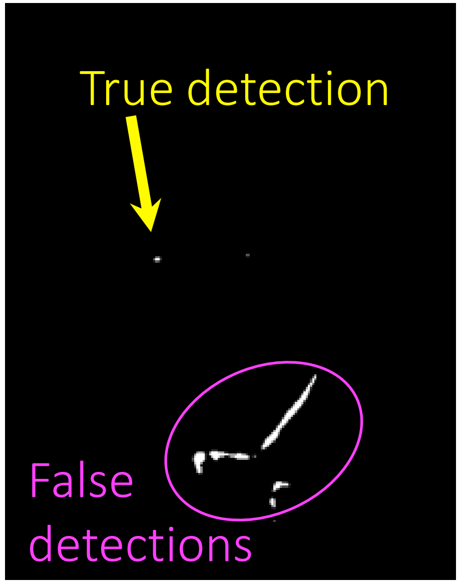
$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[m+k, n+l]$$

image

output



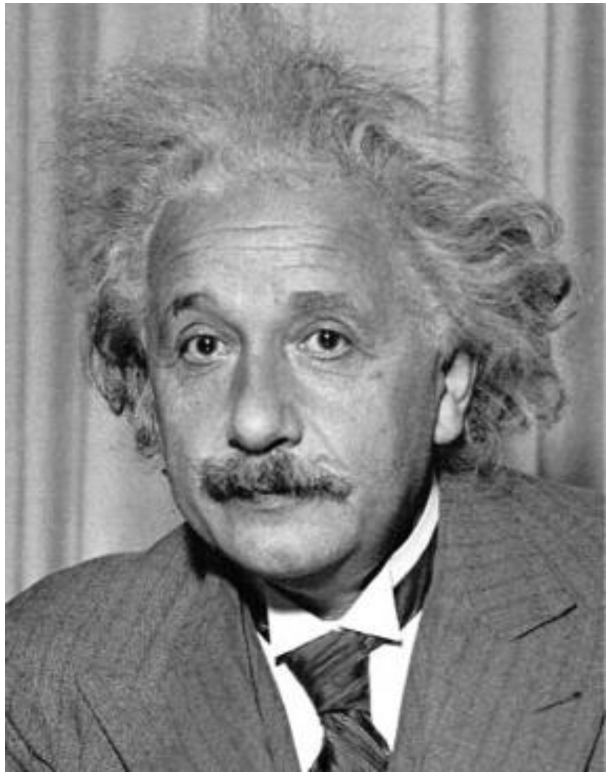
thresholding



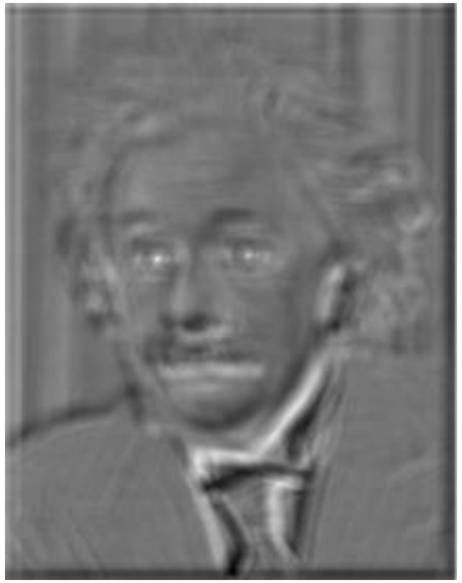
What went wrong?

Zero mean cross correlation

How do we detect the template in he following image?



output



filter

template mean

output

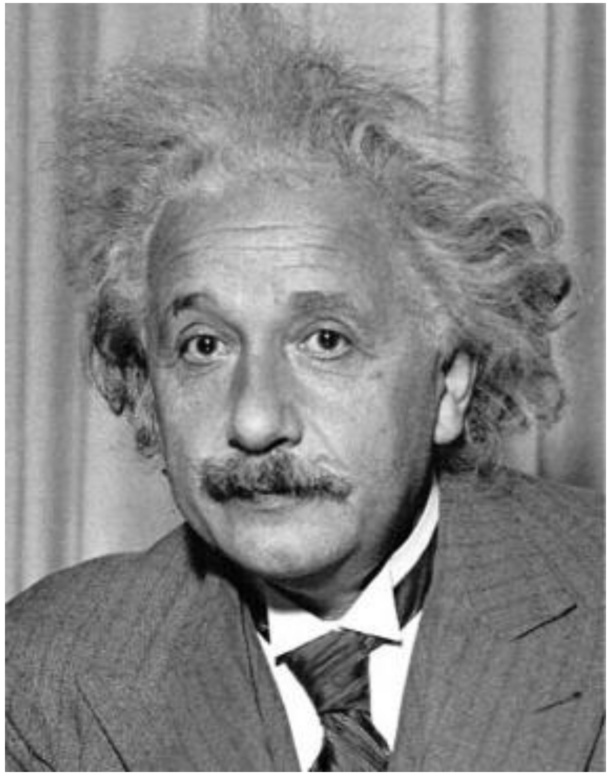
$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[m+k, n+l]$$

image

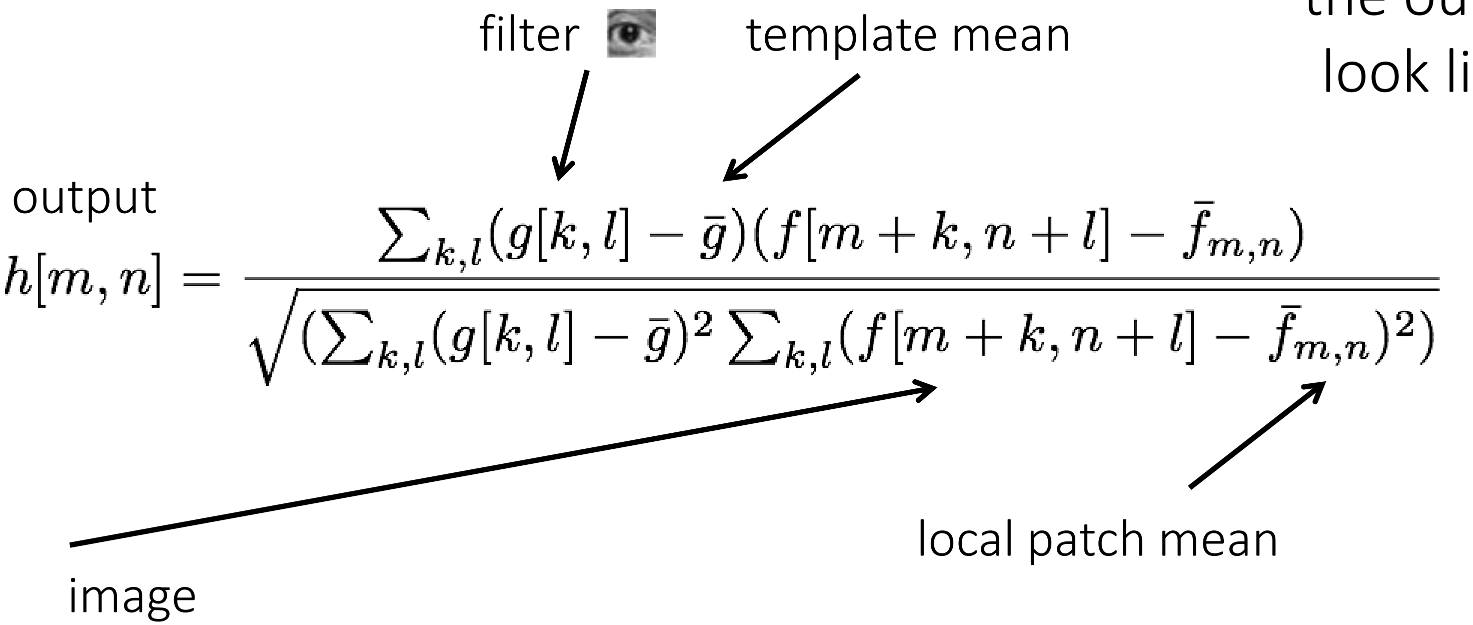
Not robust to high-contrast areas

ZNCC – zero mean normalized cross correlation

How do we detect the template in he following image?



What will the output look like?

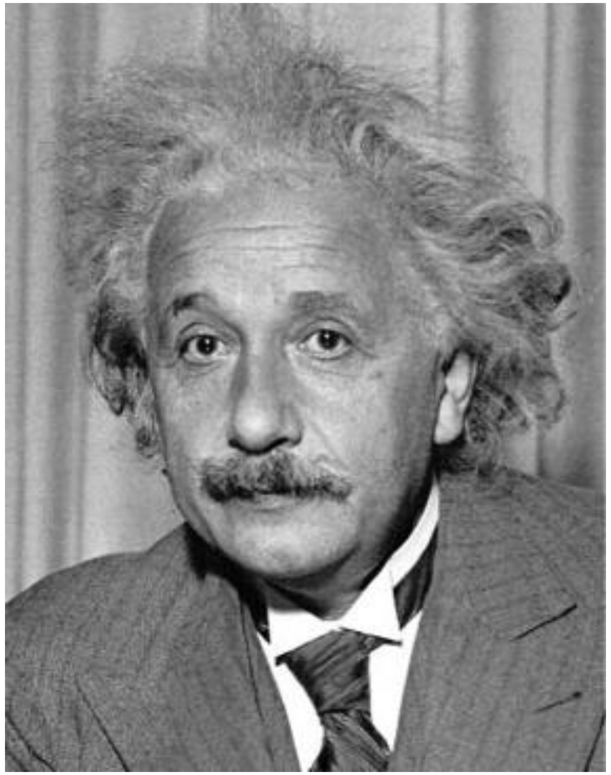


$$h[m,n] = \frac{\sum_{k,l}(g[k,l]-\bar{g})(f[m+k,n+l]-\bar{f}_{m,n})}{\sqrt{(\sum_{k,l}(g[k,l]-\bar{g})^2\sum_{k,l}(f[m+k,n+l]-\bar{f}_{m,n})^2)}}$$

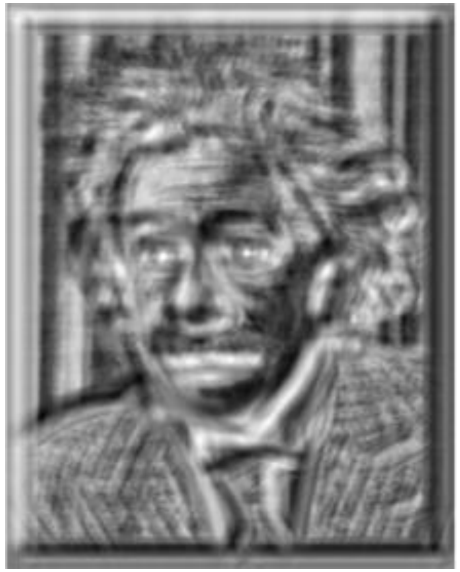
The below square root is actually the product of both template and patch STD.

ZNCC – zero mean normalized cross correlation

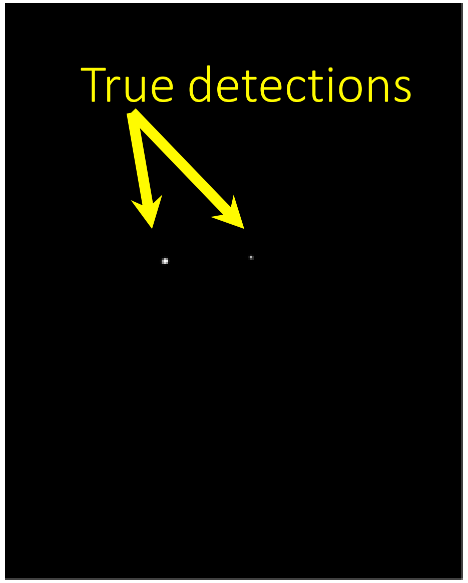
How do we detect the template in he following image?



output

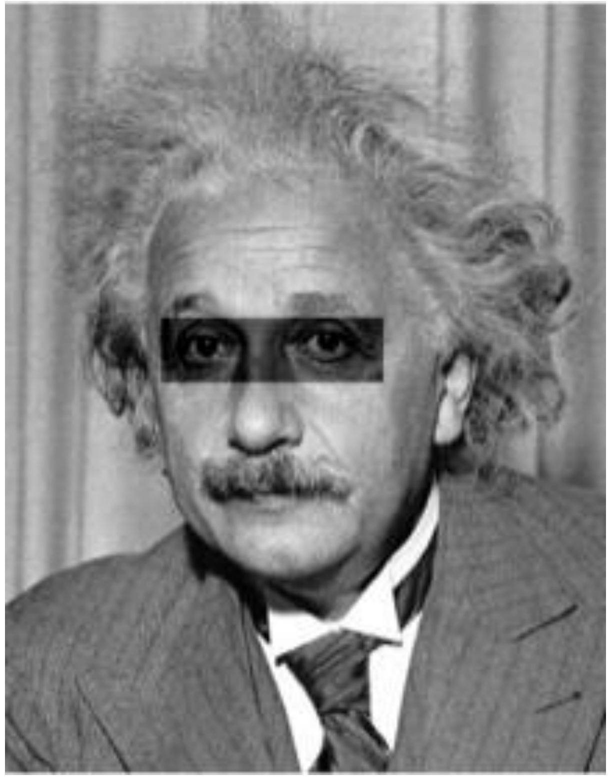


thresholding

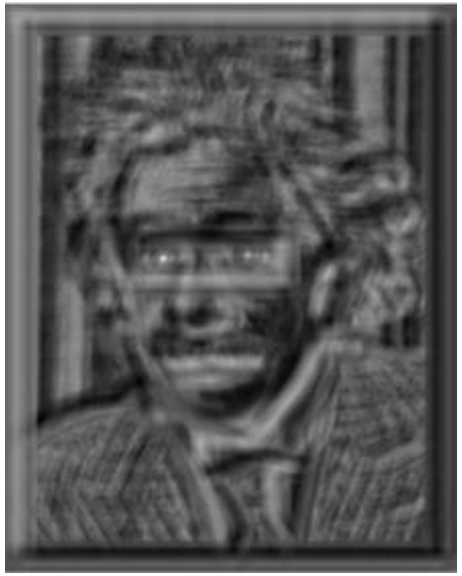


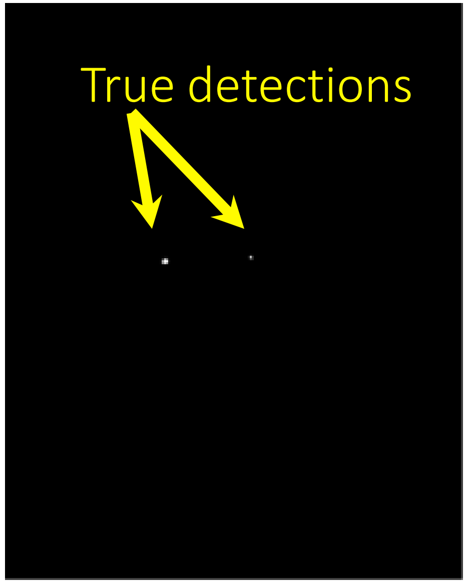
ZNCC – zero mean normalized cross correlation

How do we detect the template in he following image?



output



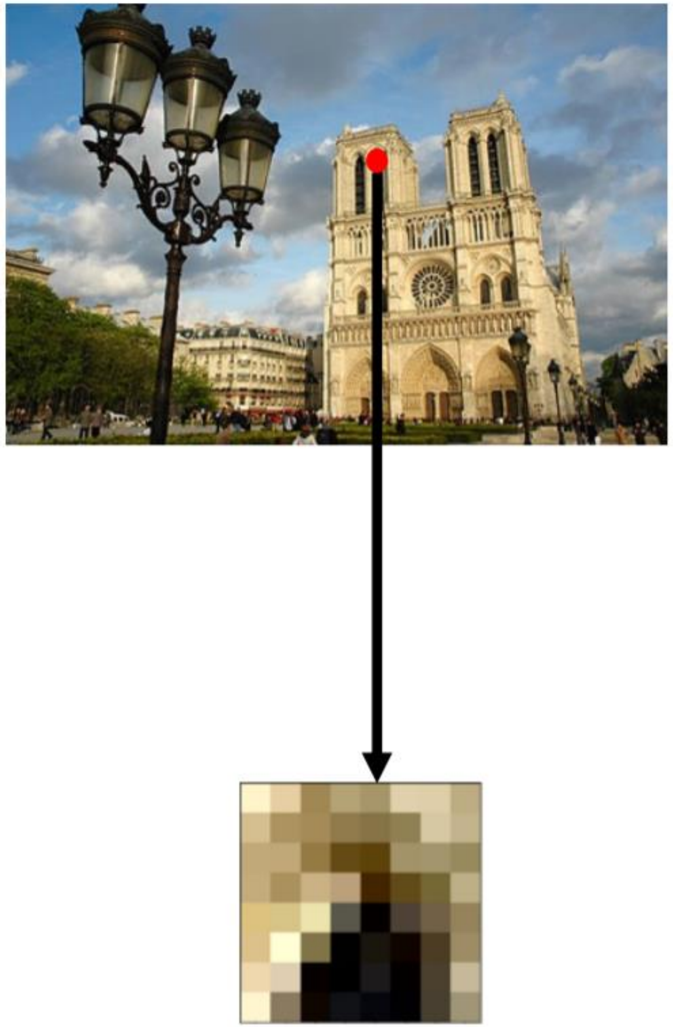


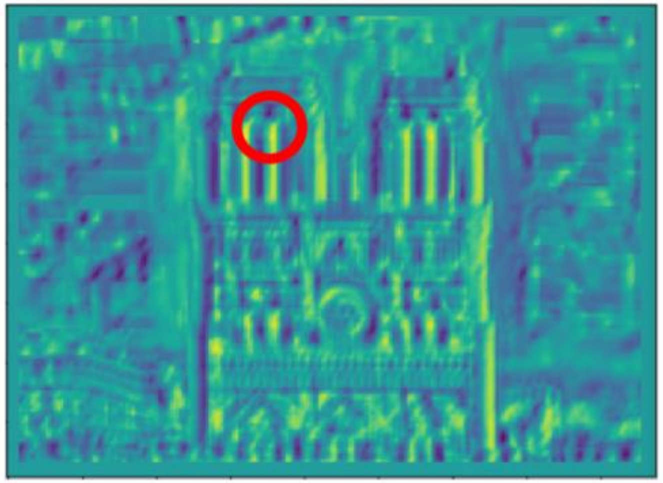
thresholding

robust to change in intensities

Template matching- ZNCC

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





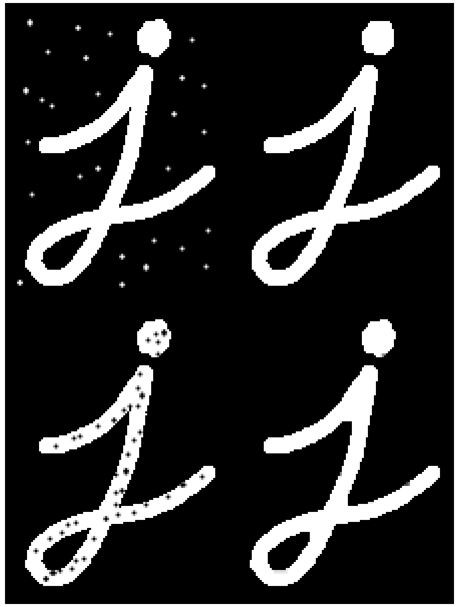
contents

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

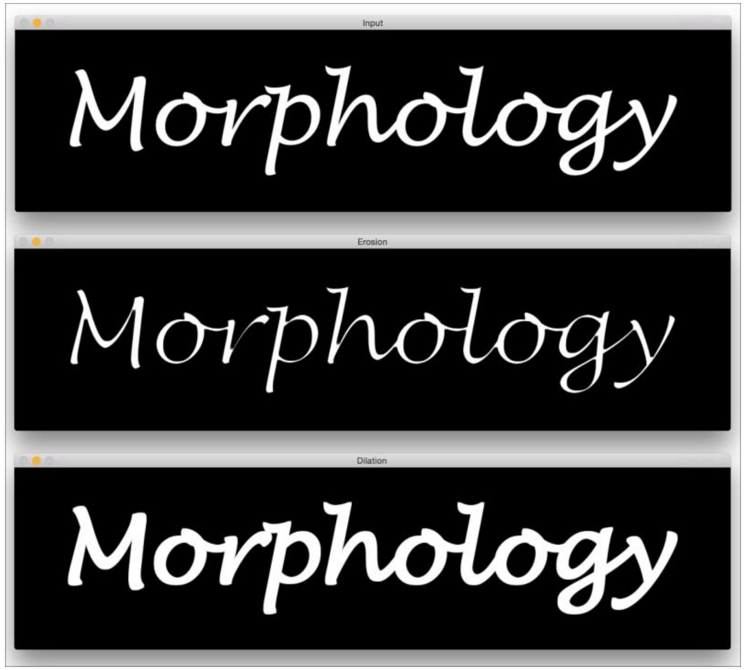
Morphology

Examples:

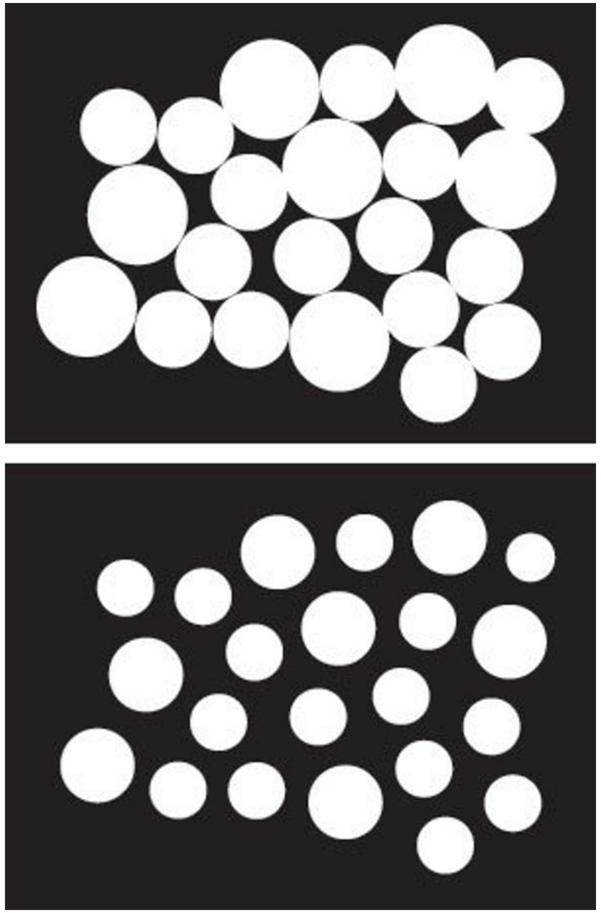
Image cleaning



Style

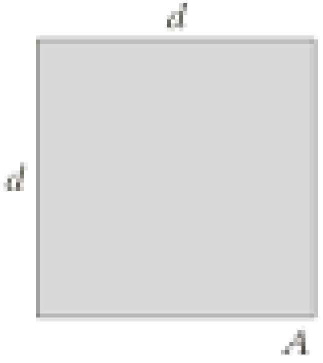


Coin counting
(using connected components)

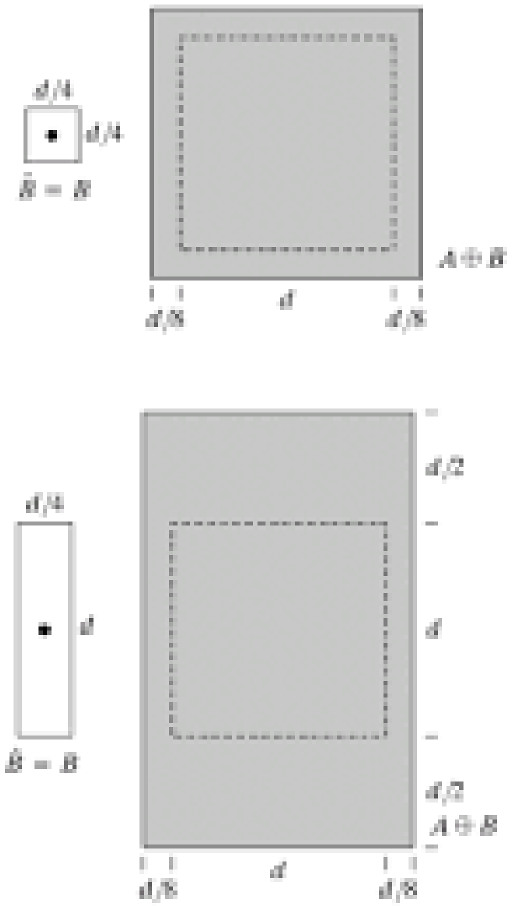


Morphology

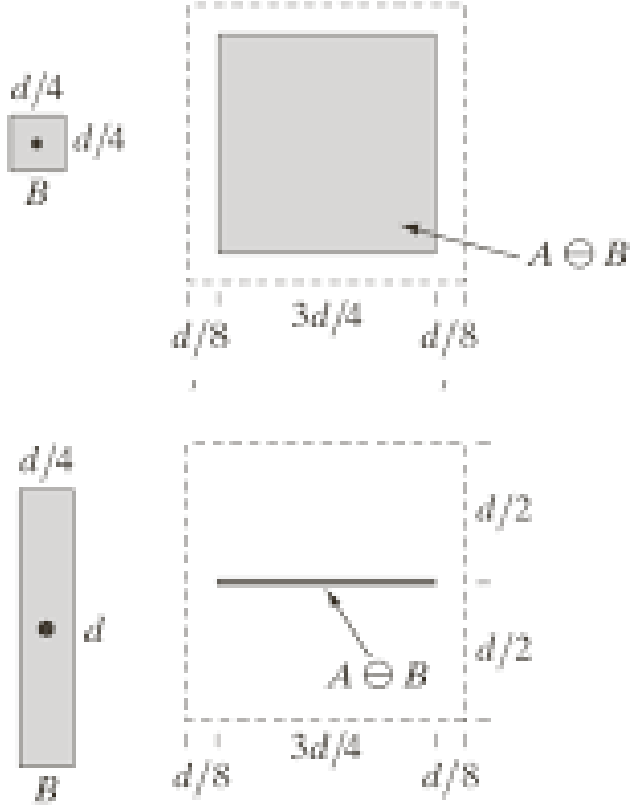
- Two basic morphology operators



Dilation: “inflating” a shape with a given kernel

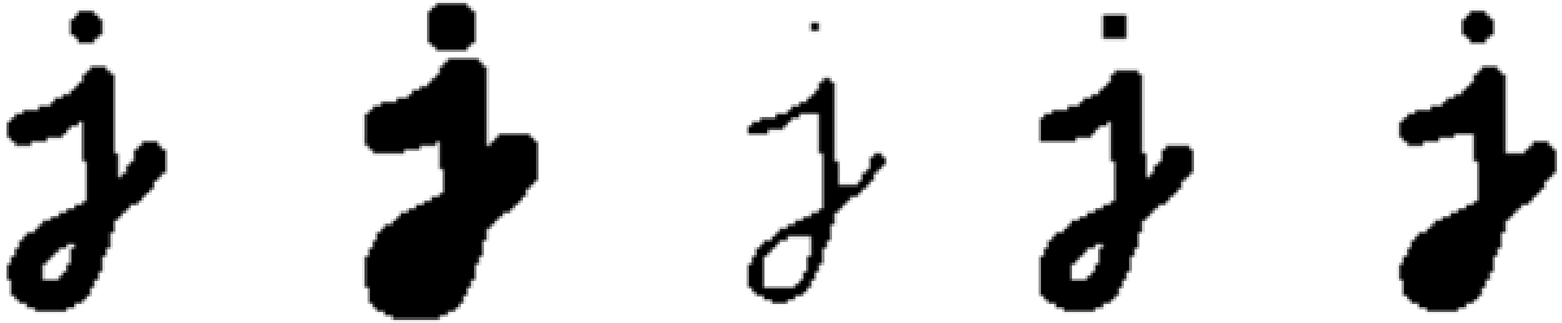


Erosion: “reducing” a shape with a given kernel



Morphology: geometric interpretation

- Each kernel 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**.



Morphology: algorithm

- 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: algorithm

- 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: algorithm

- 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

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

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

Morphology: algorithm

- 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

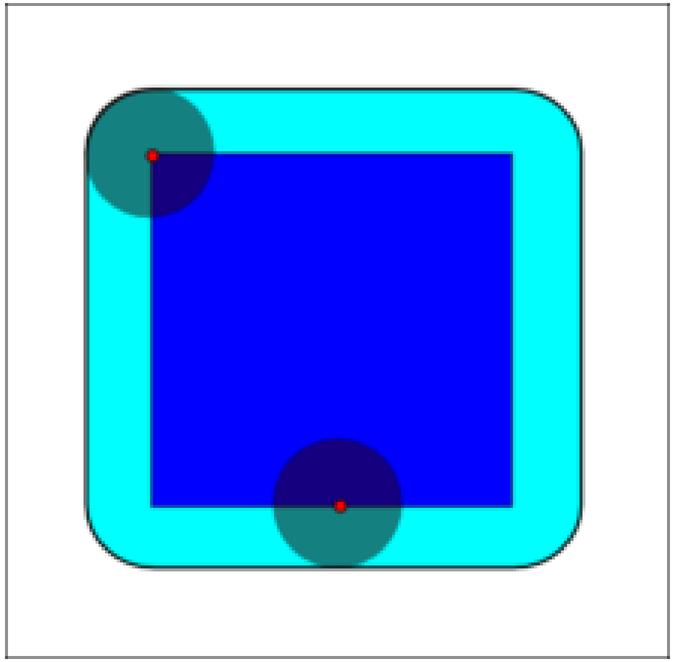
$$s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

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

- Overall morphologic operation should look like so:

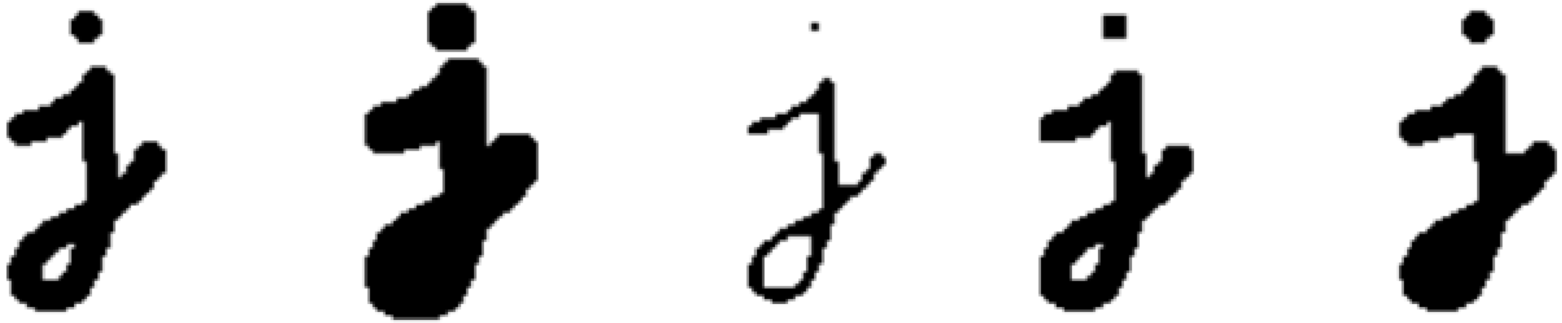
$$k = \theta_{TH}(f \star s, t)$$

Dilation



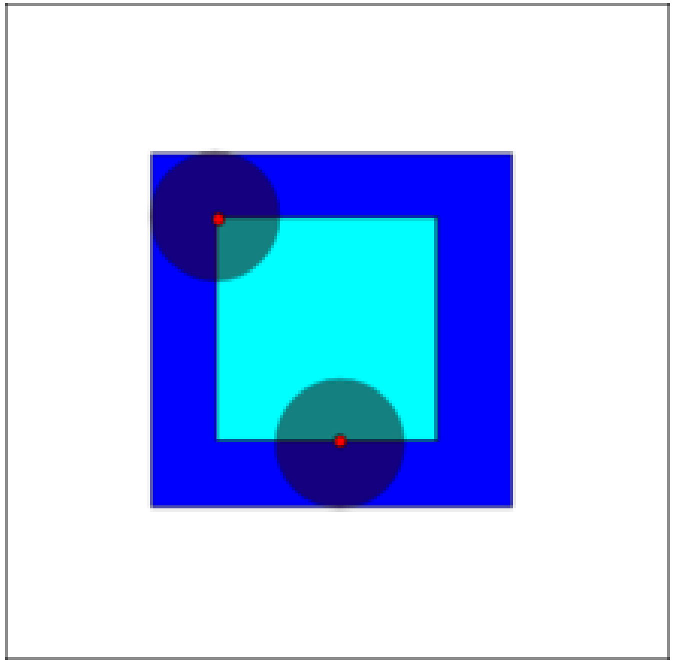
- $k = \theta_{TH}(f \star s, t)$

- Dilation: $t = 1$

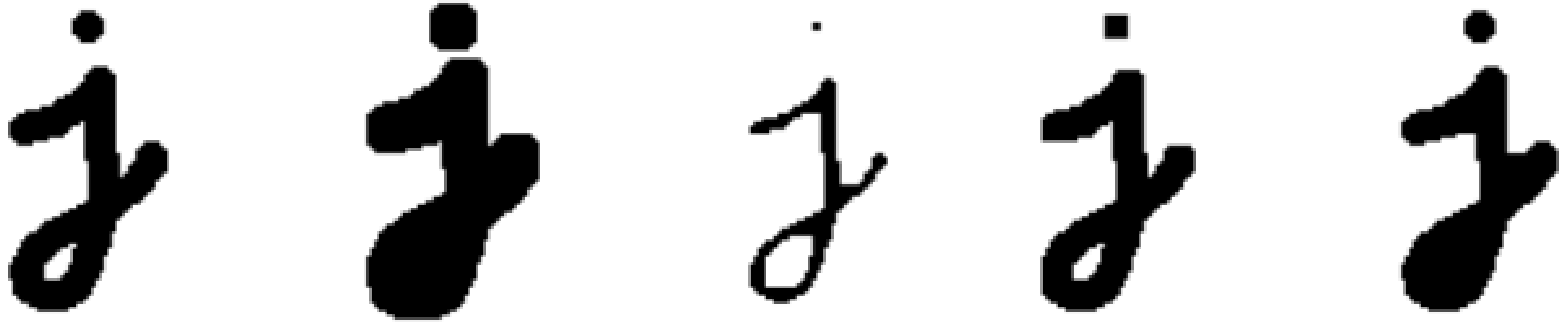


Erosion

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

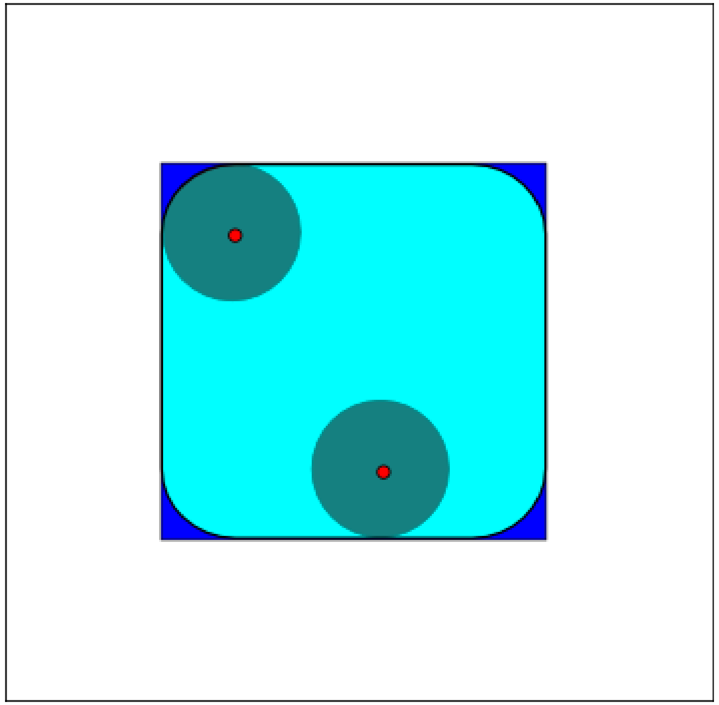


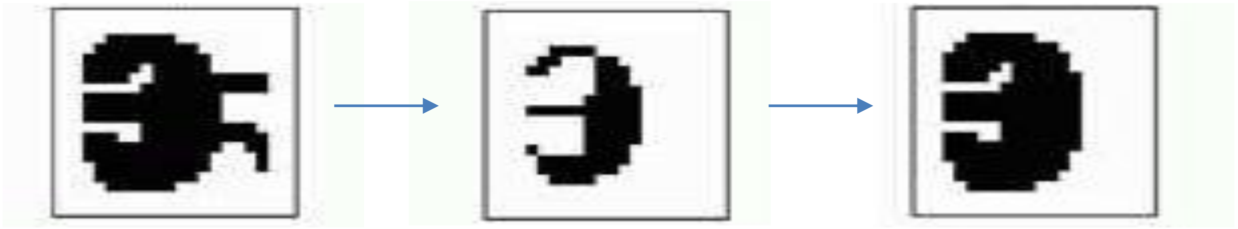
- Erosion: $t = sum(s)$



Opening

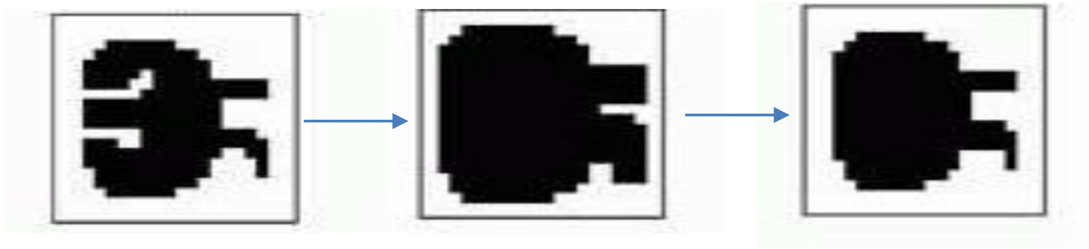
- Erosion followed by dilation.
  - The effect is of rounding off sharp edges.

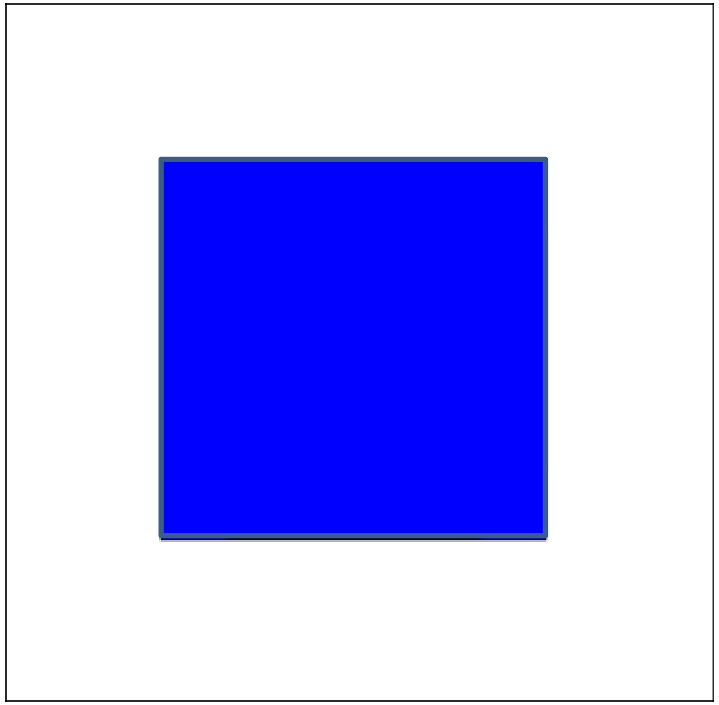


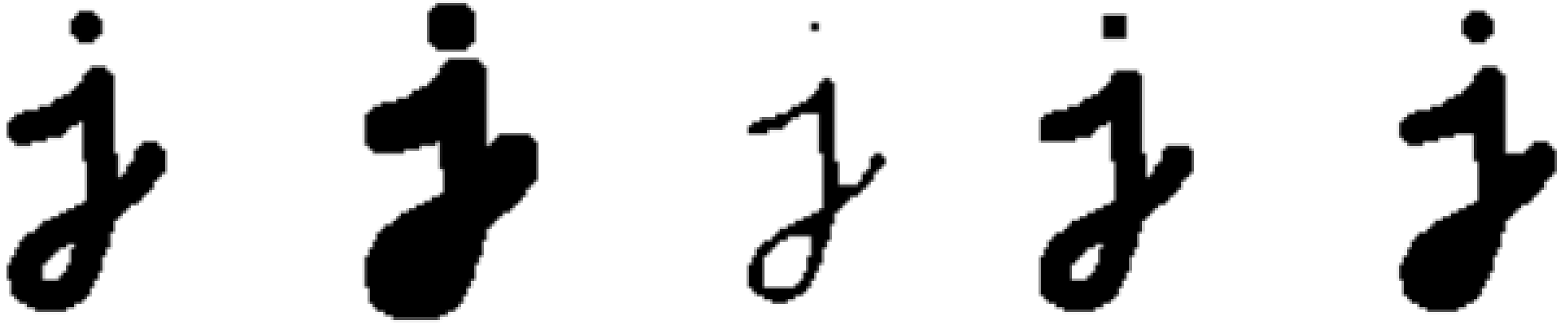


Closing

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





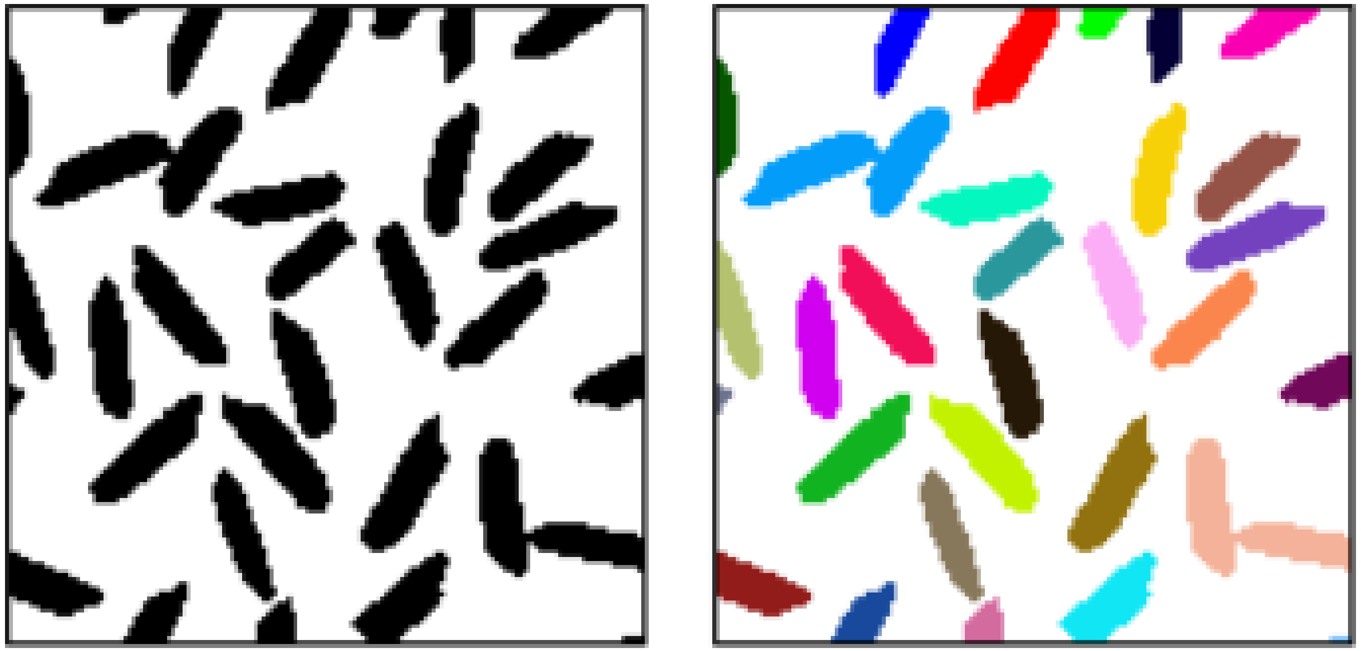


contents

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

Connected components

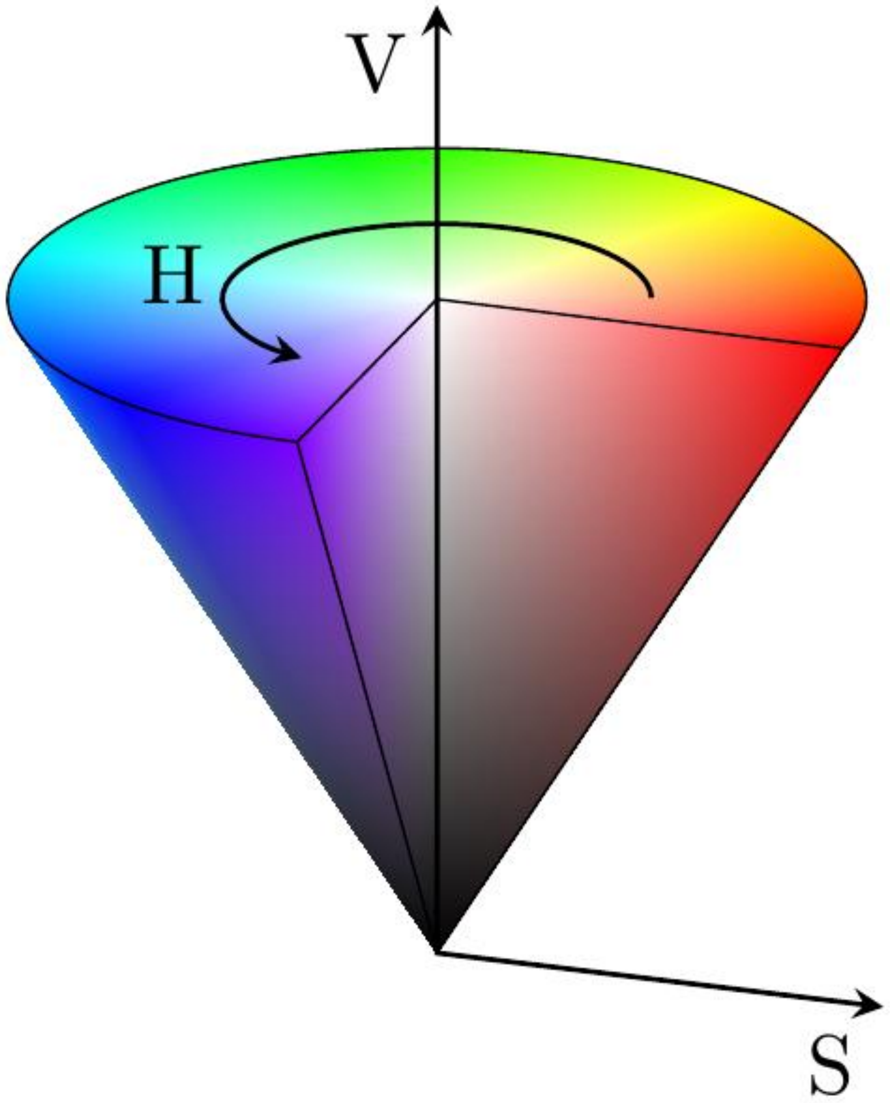
- Defined as regions of adjacent pixels that have the same value.
- Commonly used with binary images to find stand alone objects.
  - e.g.: letters in a document.



contents

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

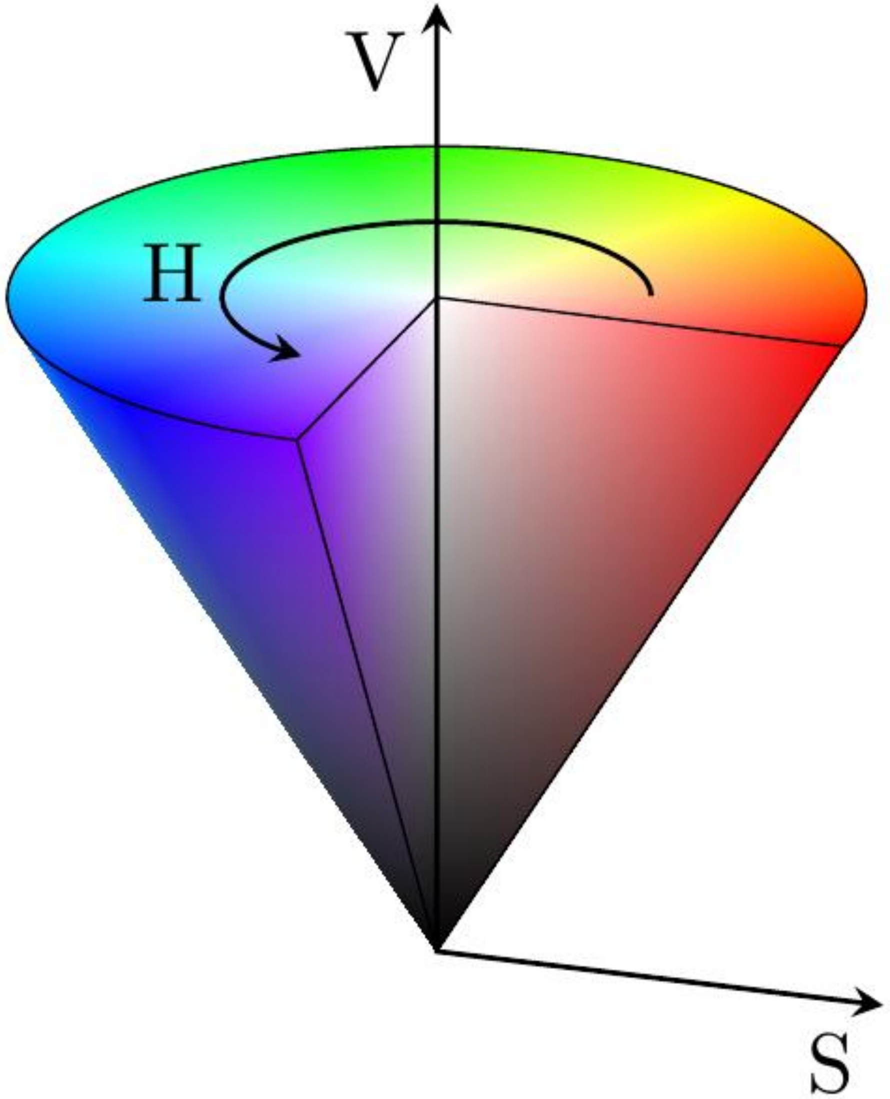
HSV

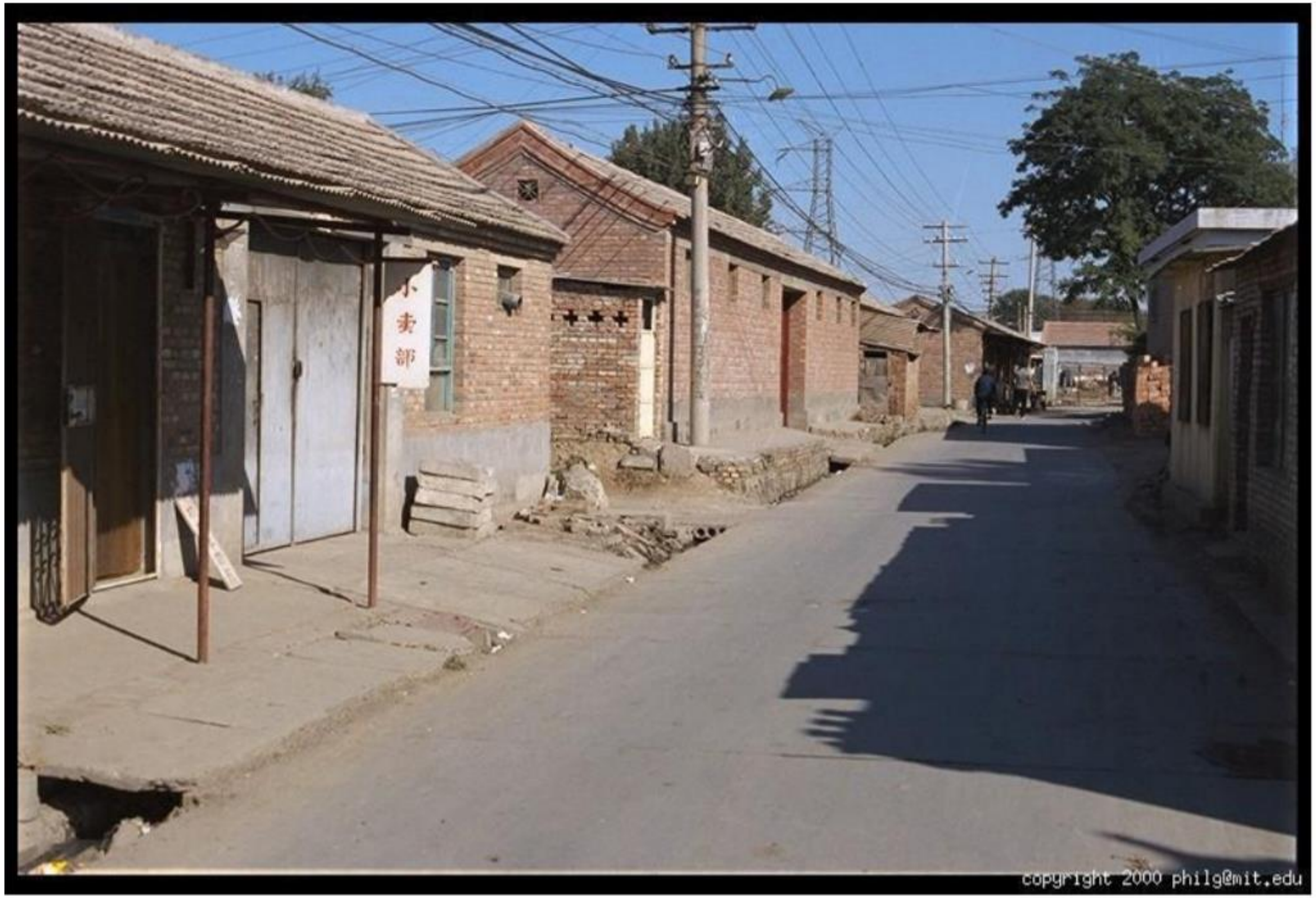


https://cs.brown.edu/courses/csci1430/lectures/2020Spring_08_CamerasLightAndColor.pdf

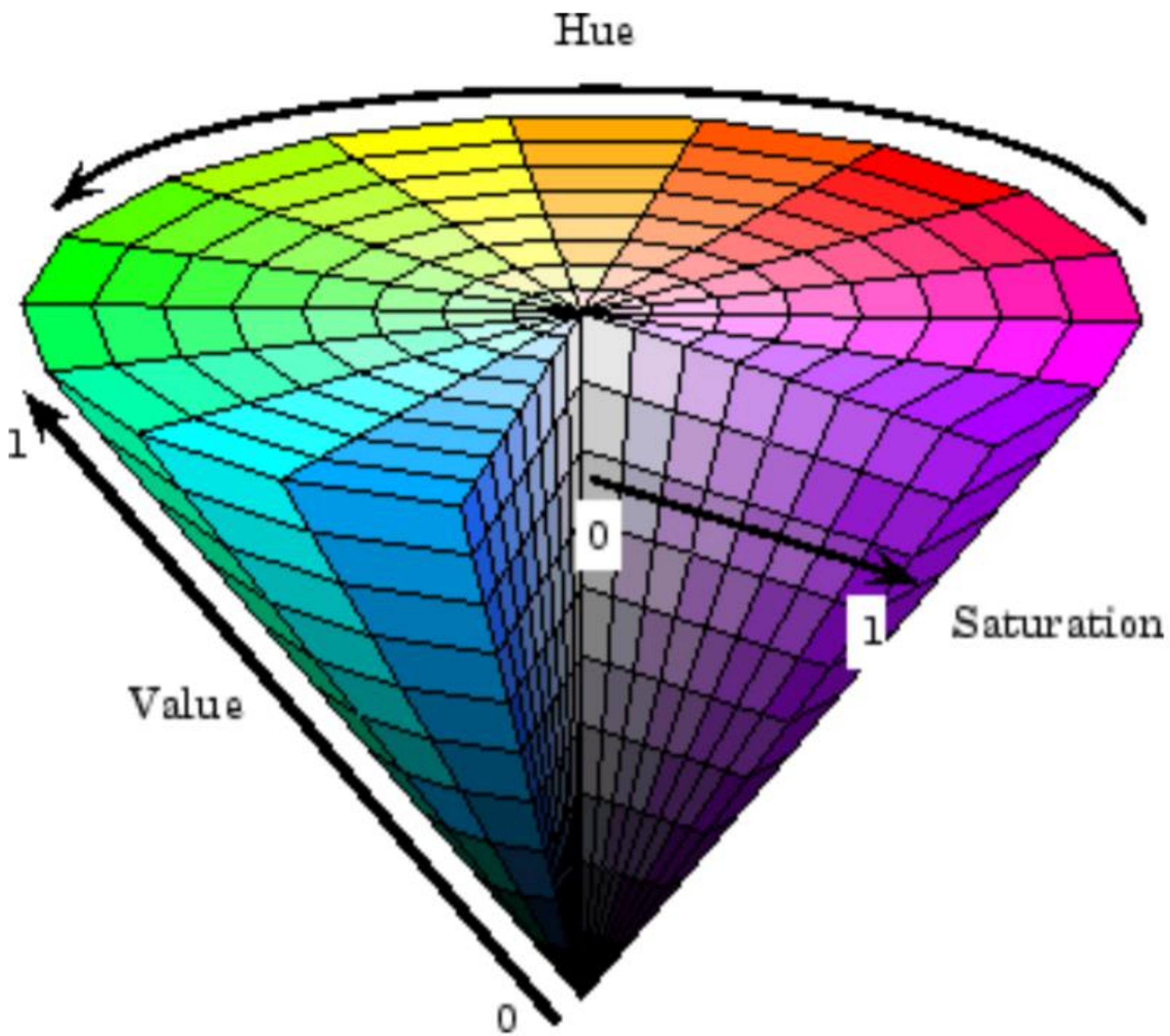
HSV

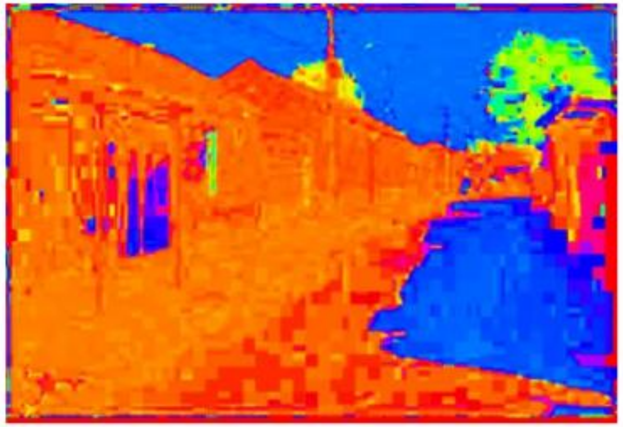
- **Hue**: The "attribute of a visual sensation according to which an area appears to be similar to one of the perceived colors: red, yellow, green, and blue, or to a combination of two of them"
- **Saturation**: The "colorfulness of a stimulus relative to its own brightness"
- **Value**: The "brightness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



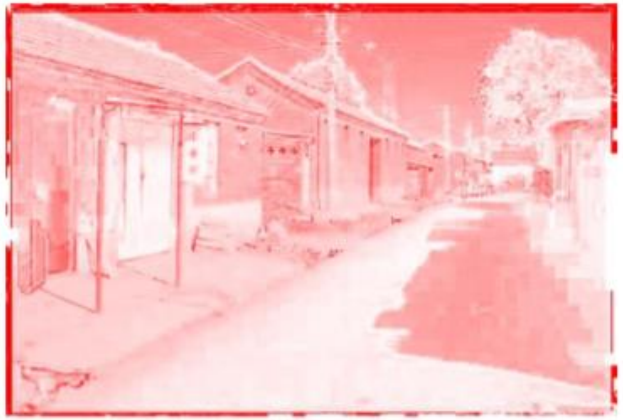


Original image

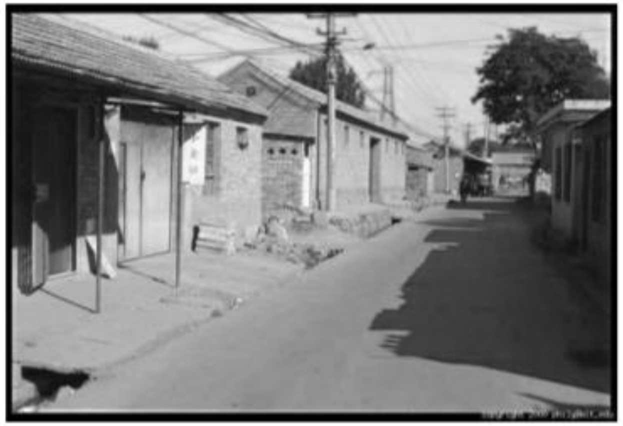




H
(S=1,V=1)



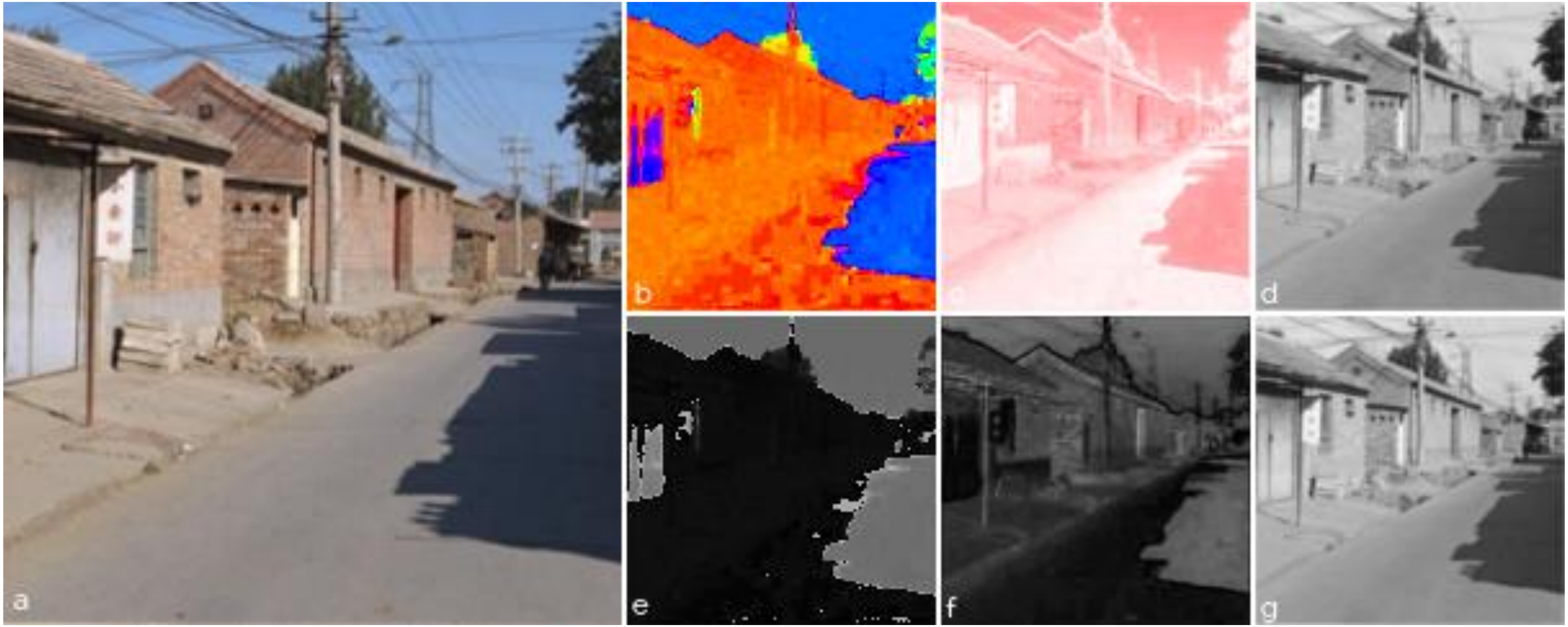
S
(H=1,V=1)



V
(H=1,S=0)

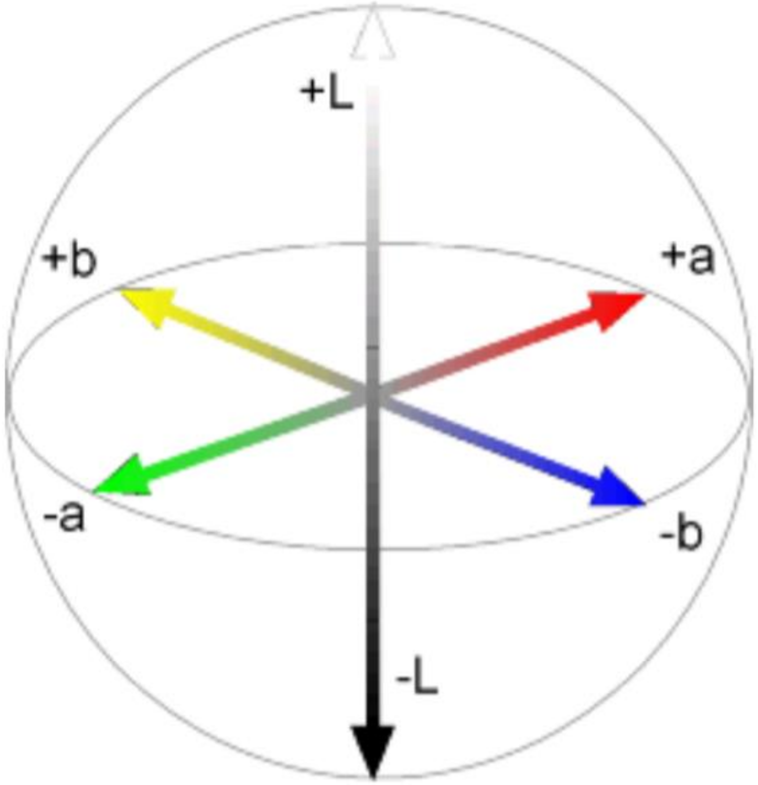
HSV

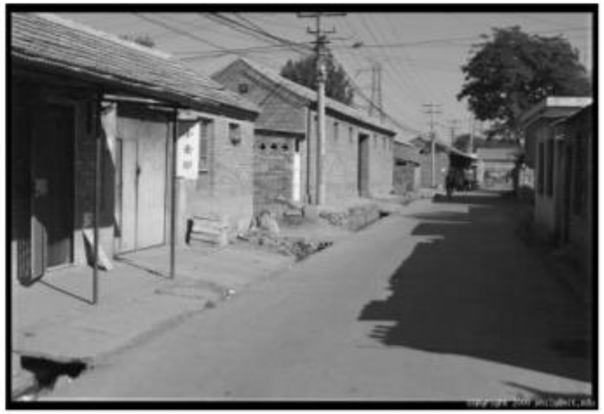
- In e, f, g: single channel image representation.
- Conclusion: people are much more responsive to intensity then chroma.



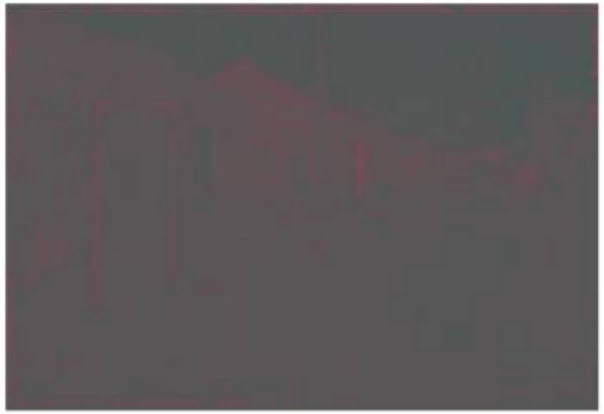
More color spaces: LAB

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

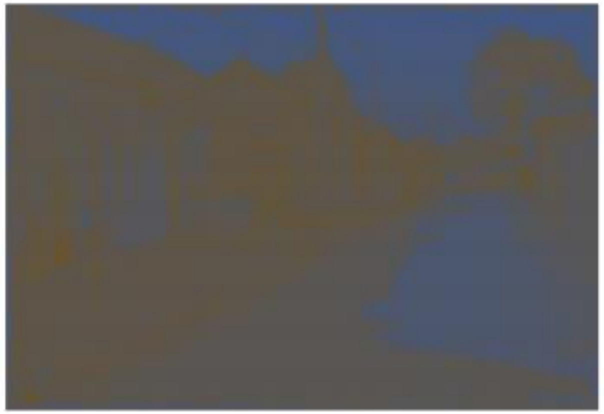




L
(a=0,b=0)



a
(L=65,b=0)

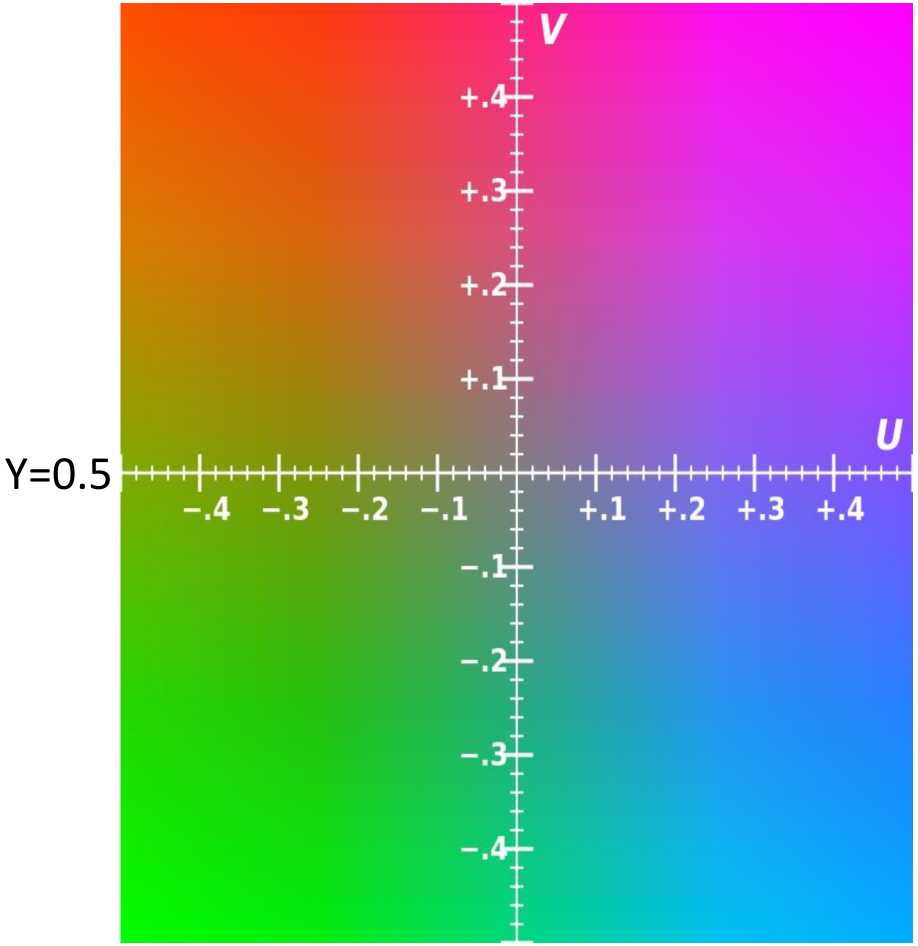


b
(L=65,a=0)

More color spaces: YUV

- Y: brightness/ intensity.
- U: blue projection.
- V: red projection.

- [Similar to YCbCr]


V
+.4
+.3
+.2
+.1
U
Y=0.5
−.4
−.3
−.2
−.1
+.1
+.2
+.3
+.4
−.1
−.2
−.3
−.4