Image Representation and Image Processing

CS 3630

Image Processing …

Alt Text:
In the 60s, Marvin Minsky assigned a couple of undergrads to spend the summer programming a computer to use a camera to identify objects in a scene. He figured they'd have the problem solved by the end of the summer. Half a century later, we're still working on it.

Source: https://xkcd.com/1425/ (2014)

Cameras are the primary sensor for most robotic platforms

Common camera types

For now we'll
focus here

Single view
RGB image
(2D data)

Stereo
RGB-D image
(3D data)

Structured light or time of flight
RGB-D image
(3D data)

Images

- An image is basically a 2D array of intensity/color values
- Image types:

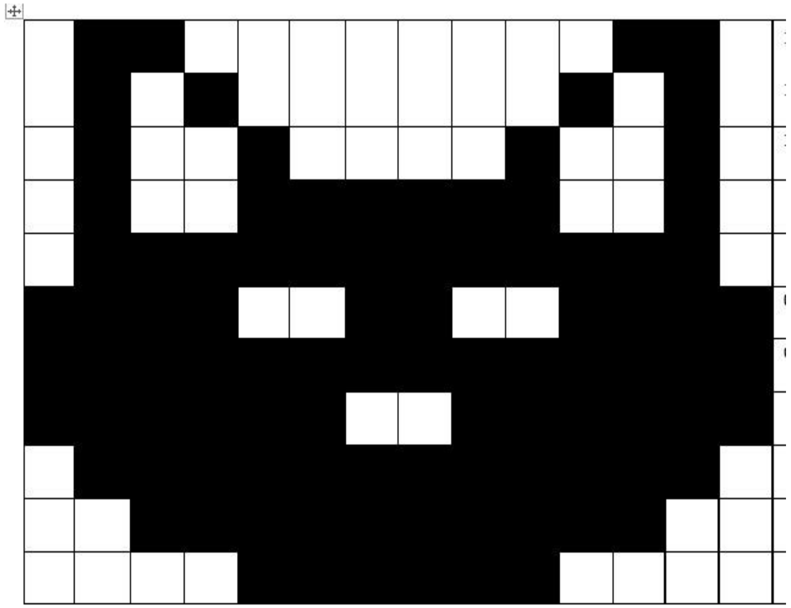
Color

Grayscale

B-W

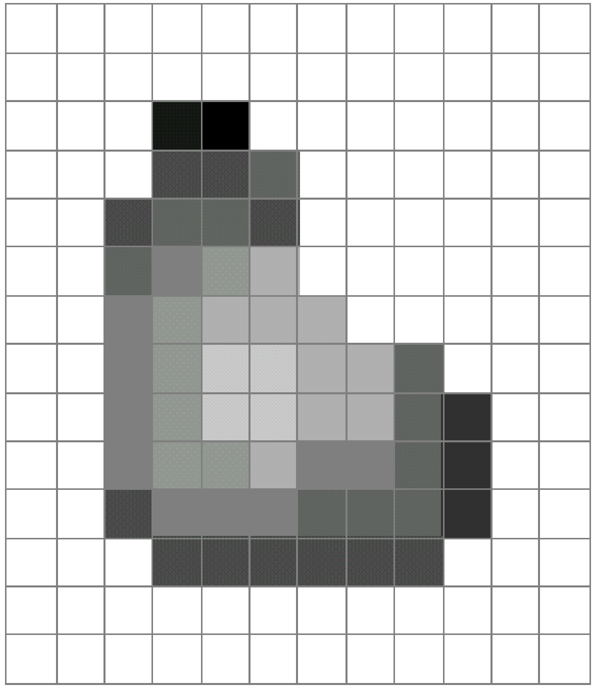
Black and White Images

- A grid (matrix) of 1’s and 0’s



Grayscale Images

A grid (matrix) of intensity values



=

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

(common to use one byte per value: 0 = black, 255 = white)

Color Images

Three grids (matrices) of intensity values - [R,G,B]

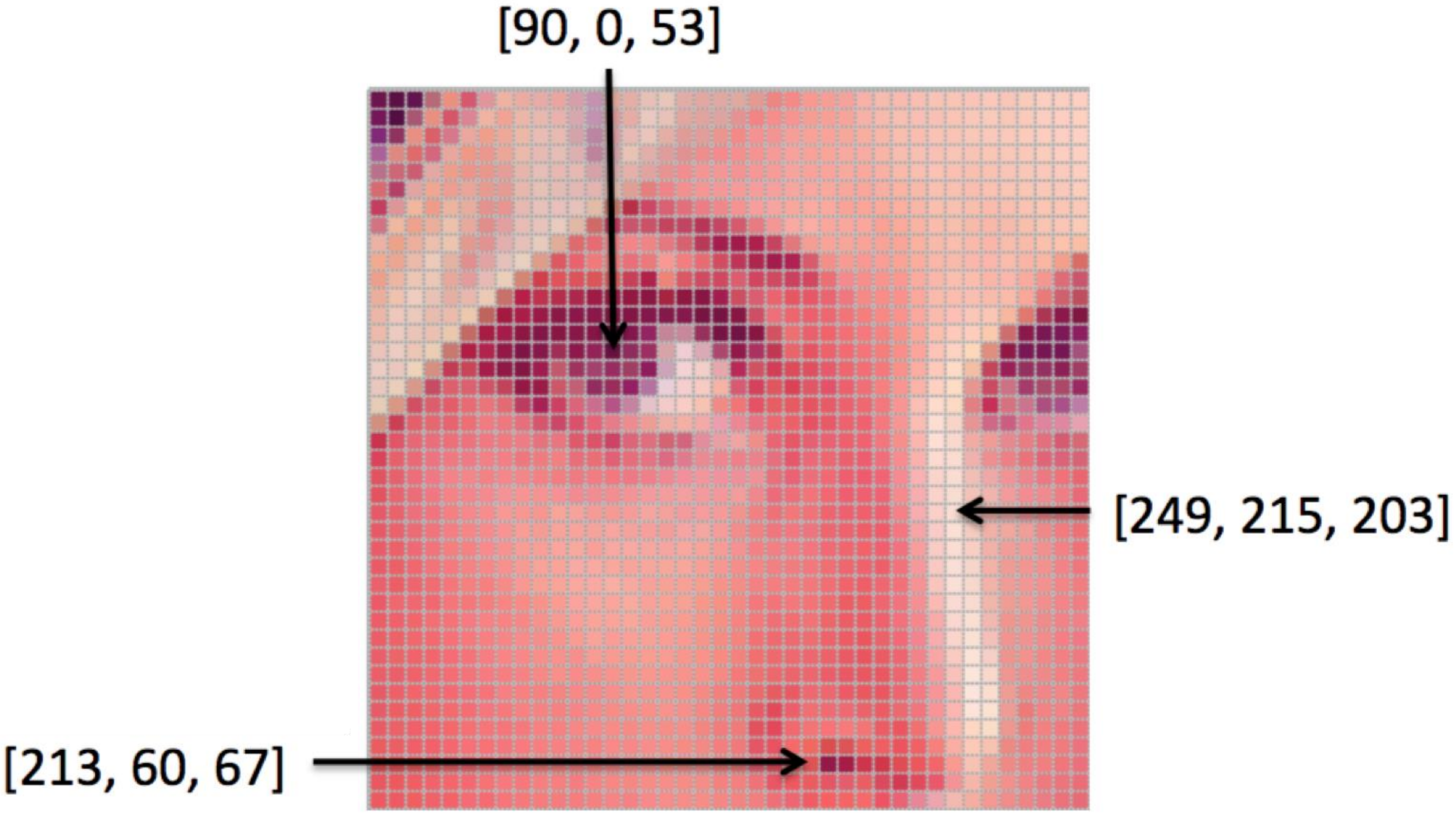
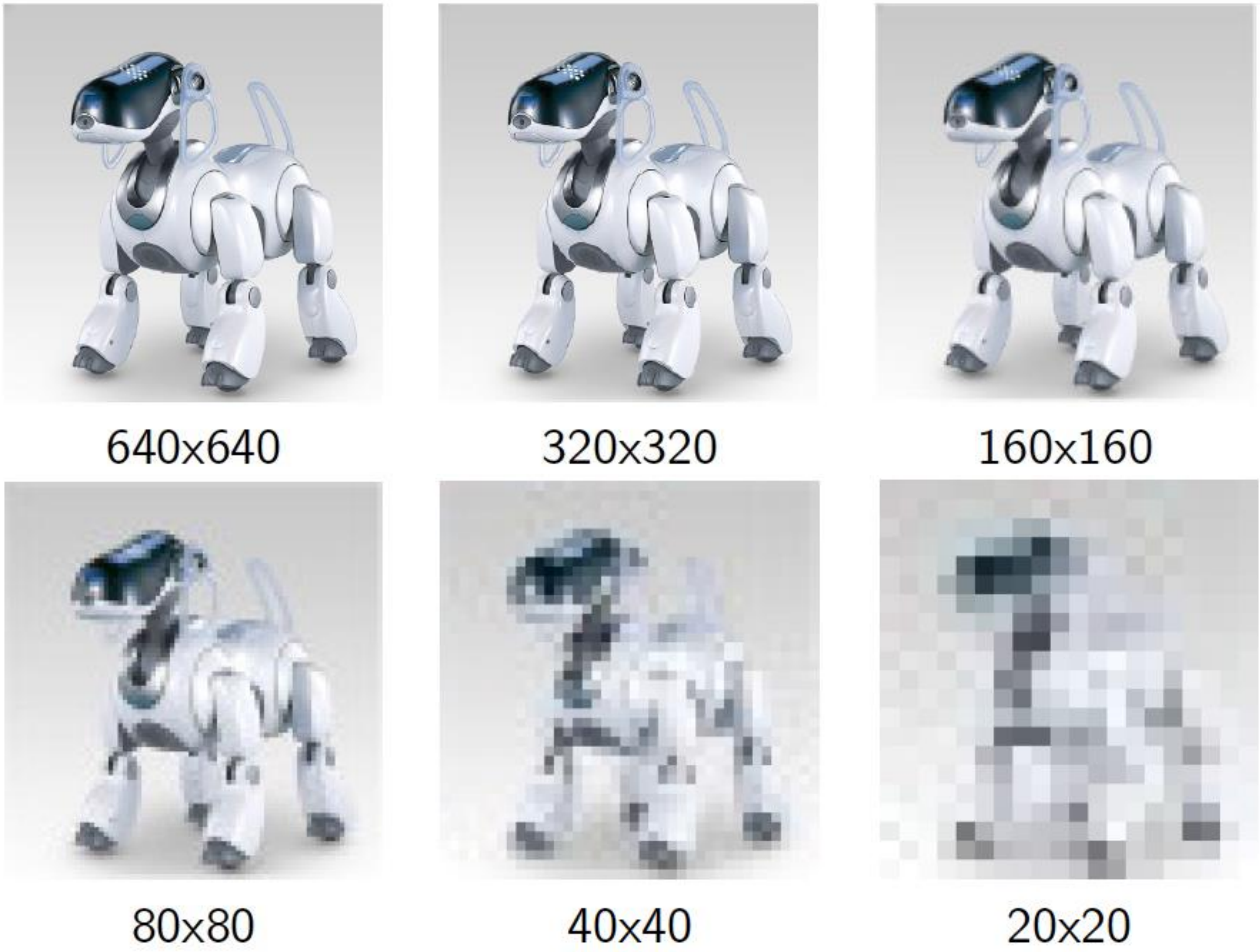


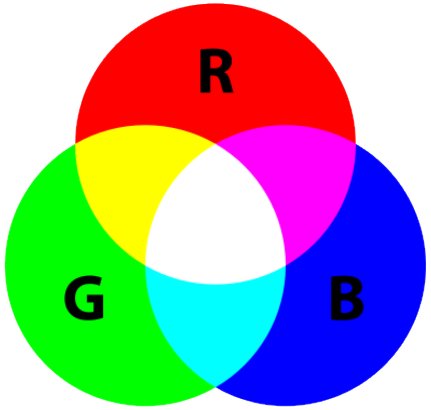
Image Adjustments…

- Scaling
- Color manipulation (color space, color->grayscale, etc.)
- Contrast
- Exposure
- …

Image sampling

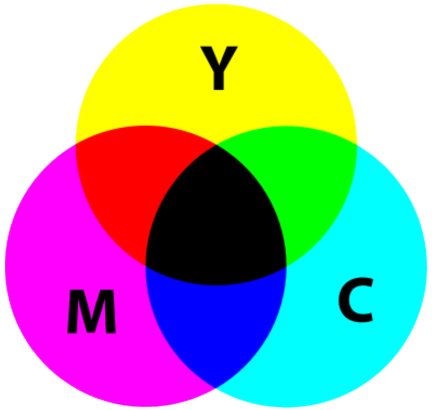


Color Models

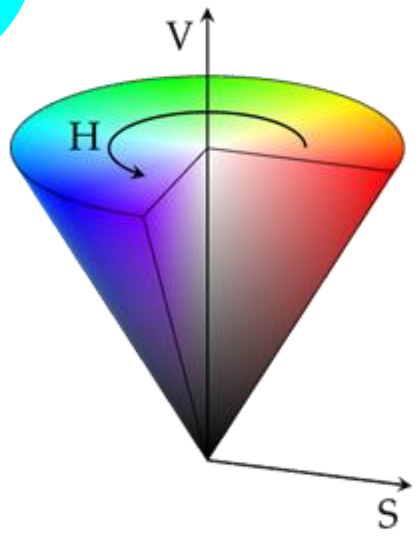


A color model is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components.

- RGB – Red, green, blue
- CMYK – cyan, magenta, yellow, black
- HSV – hue, saturation, value



- No one color model is always "better" than another.
- For specific applications, one model might be more suitable than another.
  - e.g., HSV would likely work better when looking for a dark object on top of light background



Explore color models: http://colorizer.org/

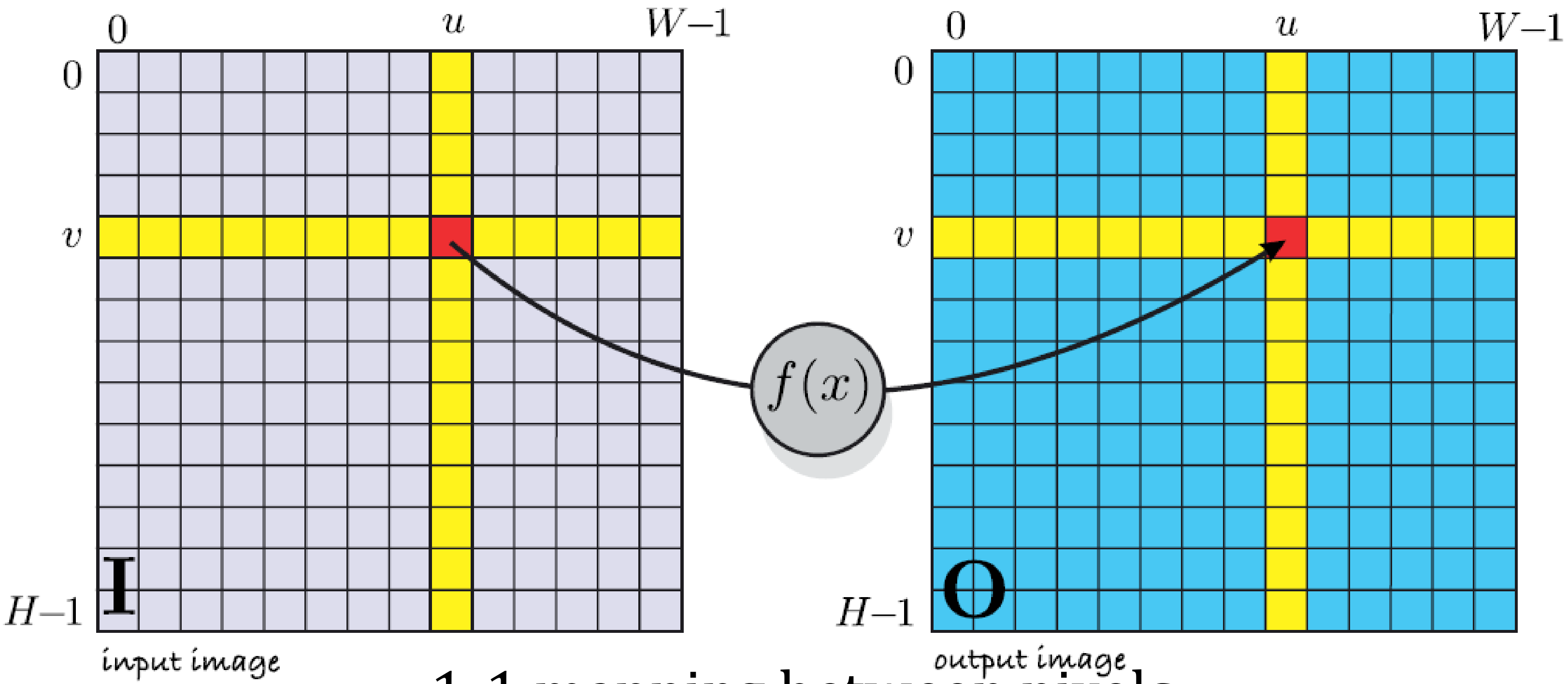
Filtering

- Types of filtering:
  - Noise removal
  - Edge detection
  - Texture description
  - Multi-scale algorithms
  - Feature detection
  - Matched filters
  - ...

- We will only focus on a few specific methods useful for our application

Monadic operators for filtering

- Operations that take a single pixel as input and output, and do not consider neighboring pixel values



1-1 mapping between pixels

Example: Sepia Tone

$$\text{outRed} = (\text{inRed} * .393) + (\text{inGreen} * .769) + (\text{inBlue} * .189)$$

$$\text{outGreen} = (\text{inRed} * .349) + (\text{inGreen} * .686) + (\text{inBlue} * .168)$$

$$\text{outBlue} = (\text{inRed} * .272) + (\text{inGreen} * .534) + (\text{inBlue} * .131)$$

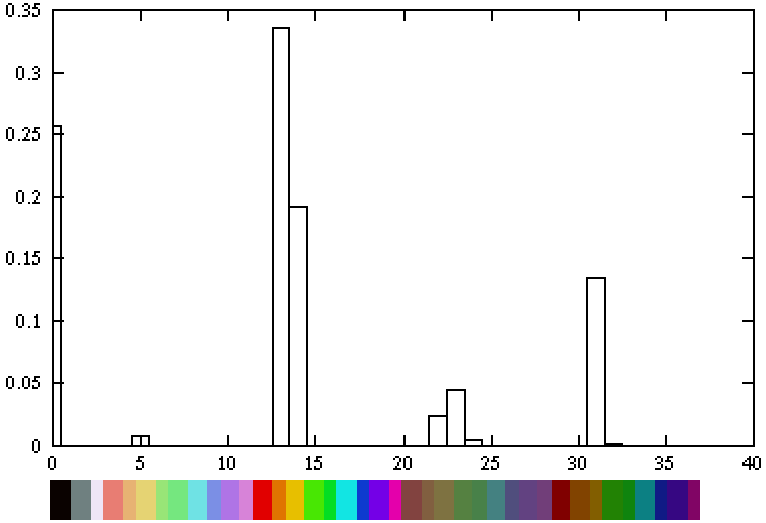
Example: Histogram Normalization

Input Image

Output Image

What is an image histogram?

For each color c_i, $\mathcal{H}_{ci}(img)$ provides the number of pixels of color c_i in *img*.



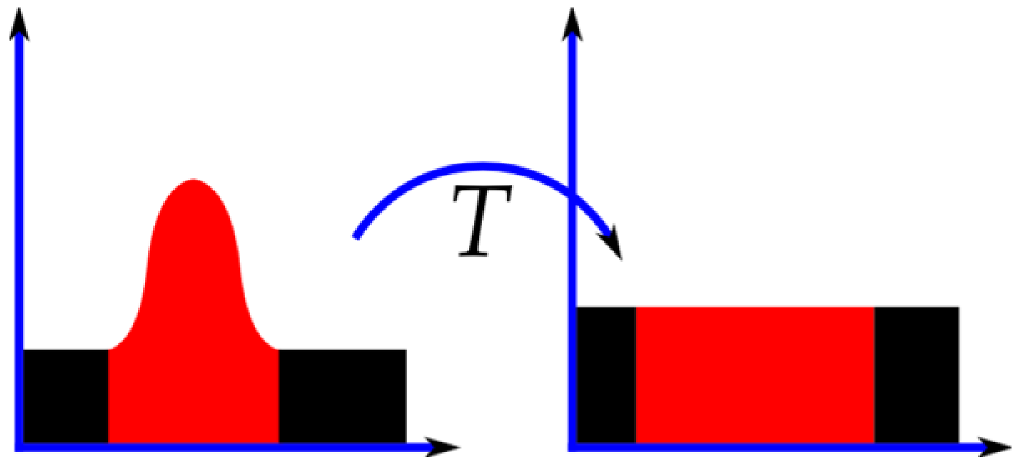
More generally, a color histogram represents the <u>distribution of various colors</u> (or *intensities* if grayscale) in the image.

Histogram filter

- A histogram can be used to adjust or redistribute the intensity or color distribution of an image

- Usually increases the contrast of an image by effectively spreading out the most frequent intensity values

- Useful in images with backgrounds and foregrounds that are both bright or both dark



Histograms of an image before and after equalization.

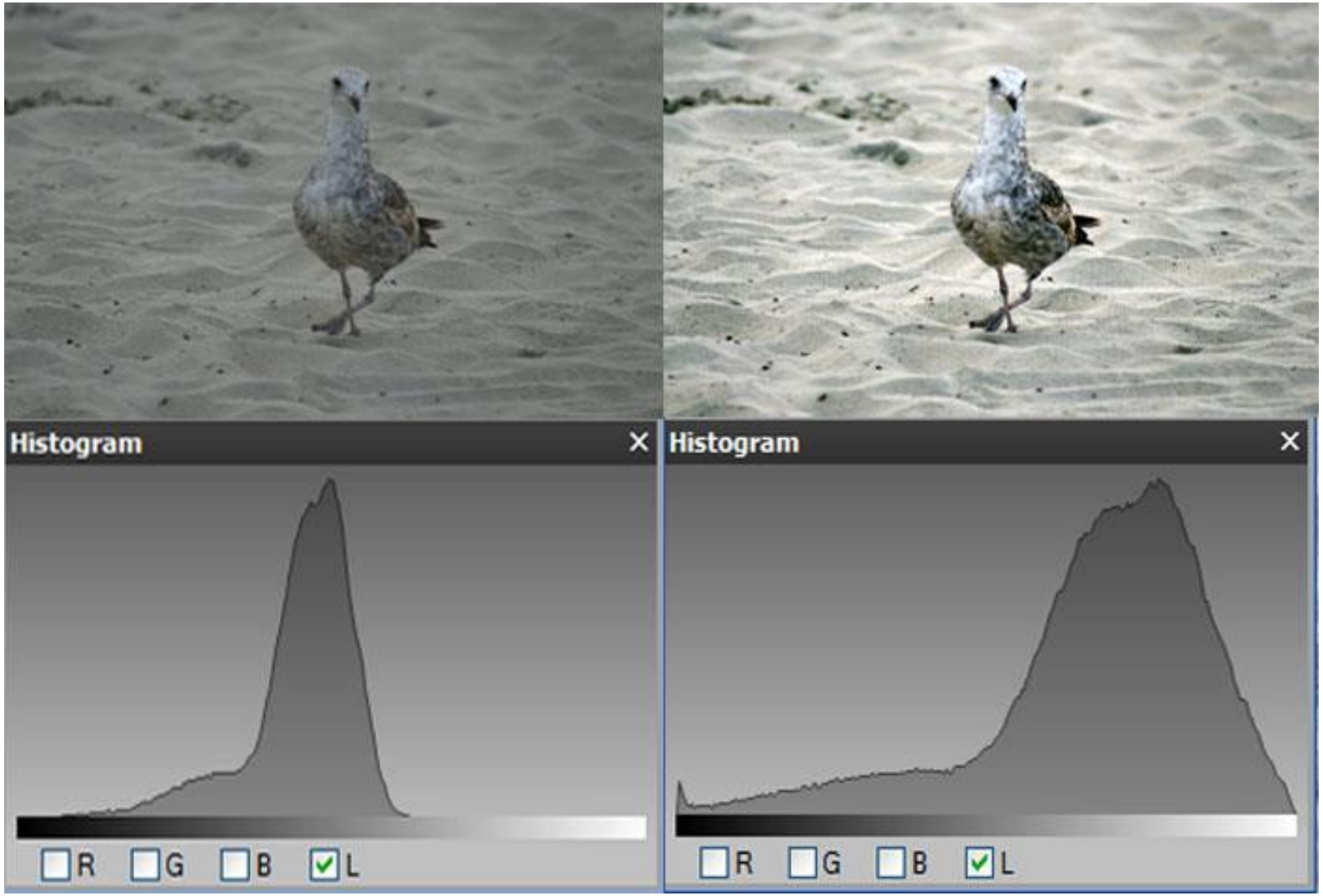
Image Contrast

Contrast is the difference between maximum and minimum pixel intensity in a given region

Low contrast

High contrast

Low contrast High contrast

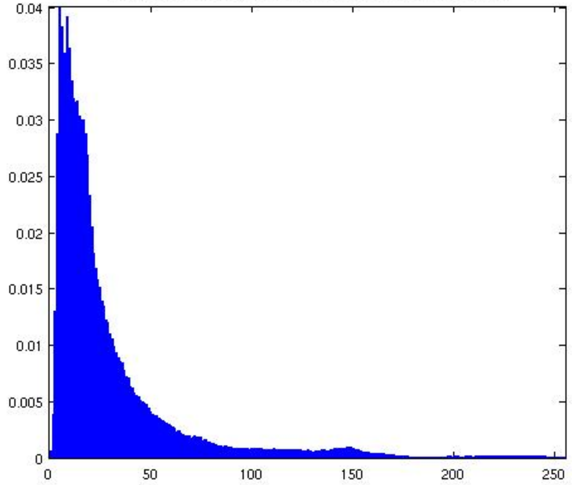


Narrow Wide

Example: Histogram Normalization

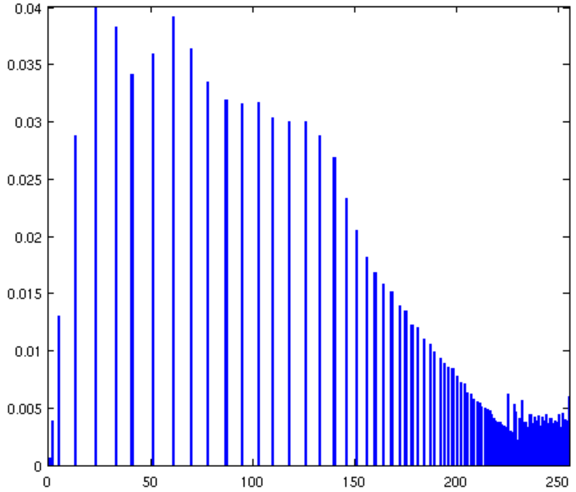
Input Image

Original Histogram before equalization



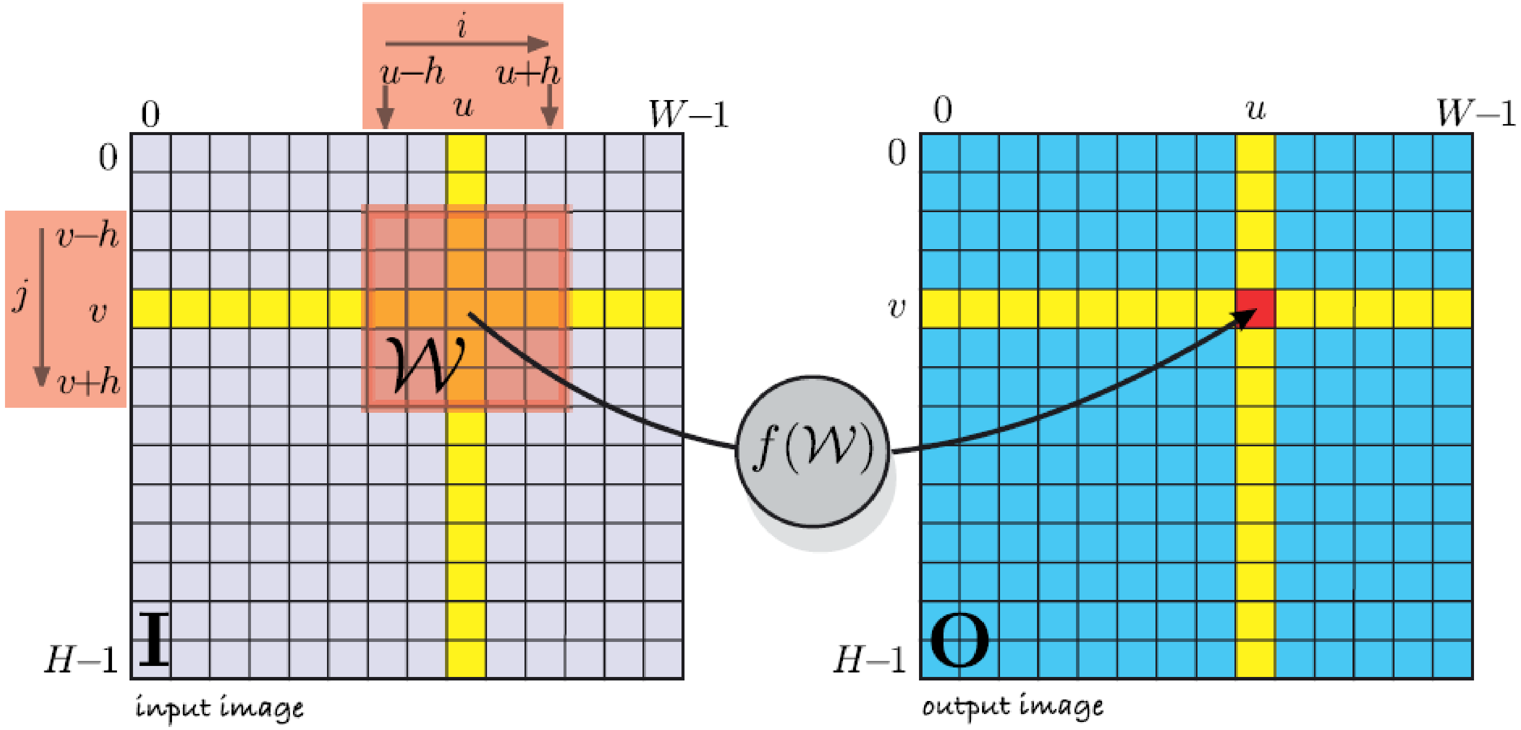
Output Image

Equalized histogram



Local operators

Image processing operations that use a region of pixels in the input image to determine the value of a single pixel in the output image



many-to-one mapping between pixels

Example: Linear averaging filter

- Replace each pixel by a linear combination of its neighbors
- The matrix of the linear combination is called the "kernel," "mask", or "filter"

$$\underbrace{\begin{bmatrix}10 & 5 & 3\\ 4 & 6 & 1\\ 1 & 1 & 8\end{bmatrix}}_{\text{Local image data}} \longrightarrow \underbrace{\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}}_{\text{kernel}} \longrightarrow \underbrace{\begin{bmatrix} & & \\ & 4 & \\ & & \end{bmatrix}}_{\text{Output image}}$$

What do you think will happen?

Smoothing and blur!

Source: L. Zhang

Example: Linear averaging filter

Let F be the image, H be the kernel (of size $2k + 1$ by $2k + 1$), and G be the output image

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

10	5	3
4	6	1
1	1	8

Local image data

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

kernel

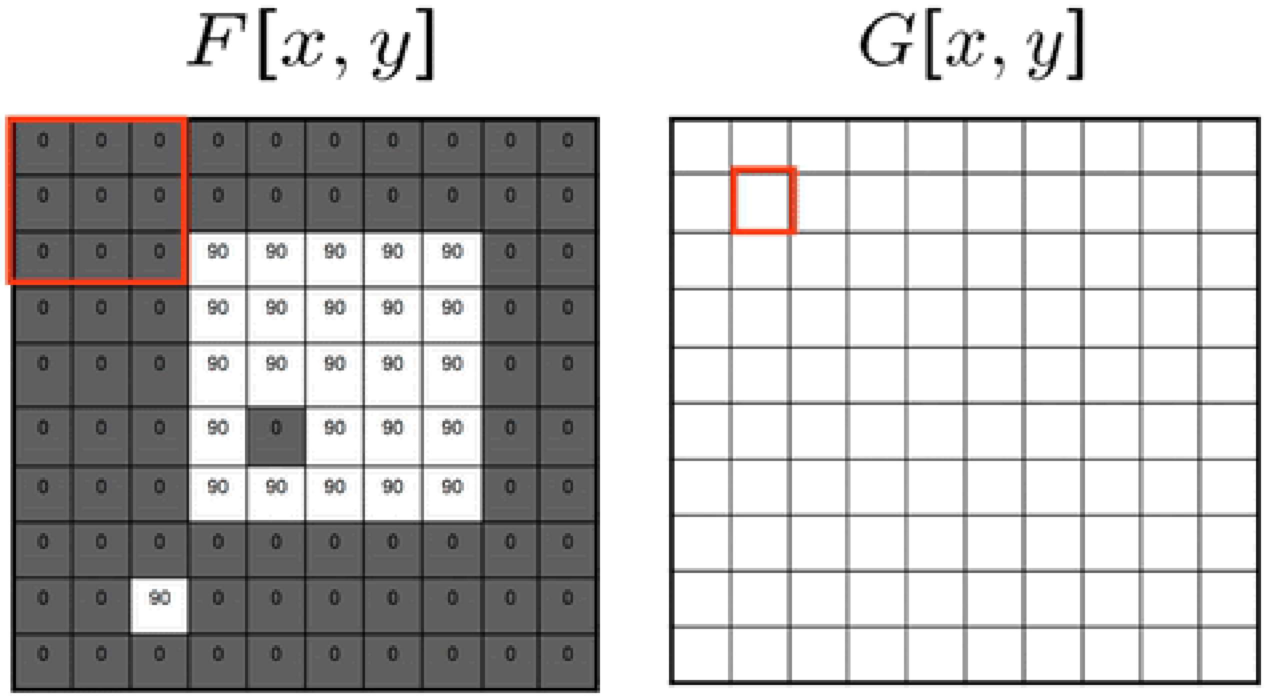
	4	

Output image

Source: L. Zhang

Convolution

- Applying a kernel to an image in this way is called **convolution**.
- NOTE: Convolution can produce unwanted artifacts along the edges of the image.
- Techniques for addressing this include zero padding, edge value replication, mirror extension, and others.



Linear filters: examples

Symbol for convolution

Original

$\otimes$

0	0	0
0	1	0
0	0	0

$=$

Identical image

Source: D. Lowe

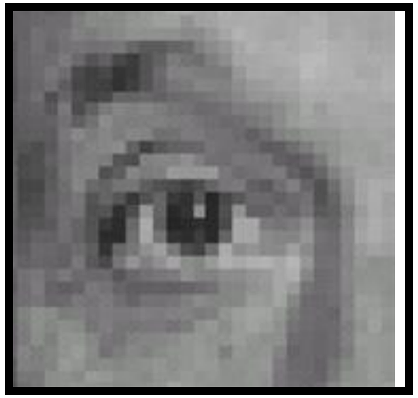
Linear filters: examples

Original

$\otimes$

0	0	0
0	0	1
0	0	0

$=$



Shifted left
By 1 pixel

Source: D. Lowe

Linear filters: examples

Original

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

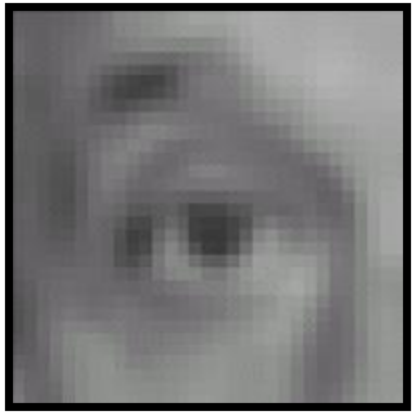
Shifted <u>right</u>
By 1 pixel

Source: D. Lowe

Linear filters: examples

Original

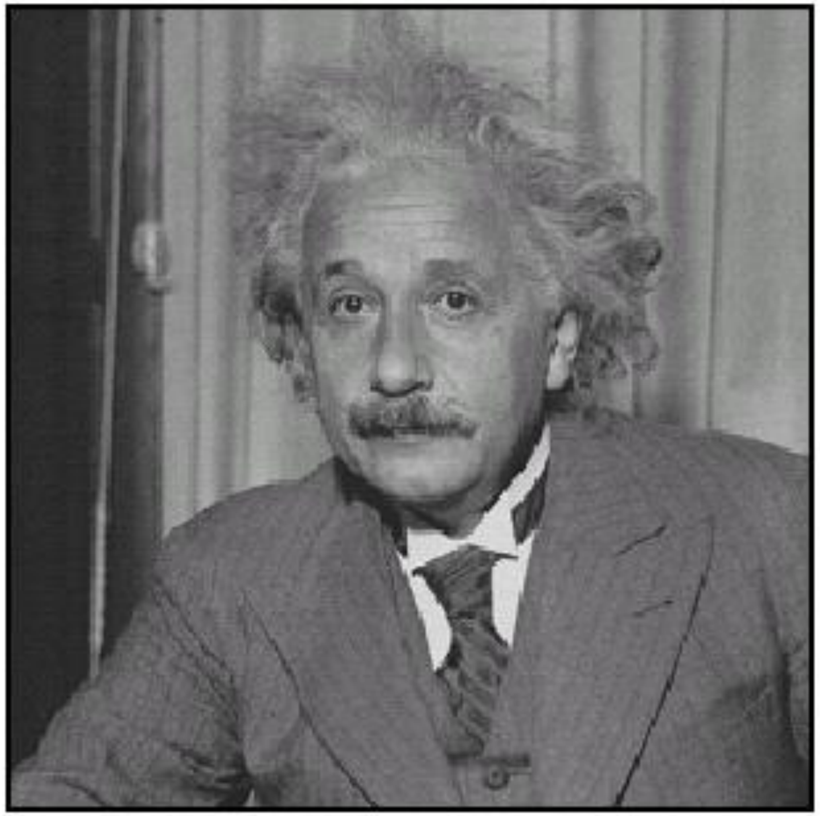
$$\otimes \; \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \; =$$



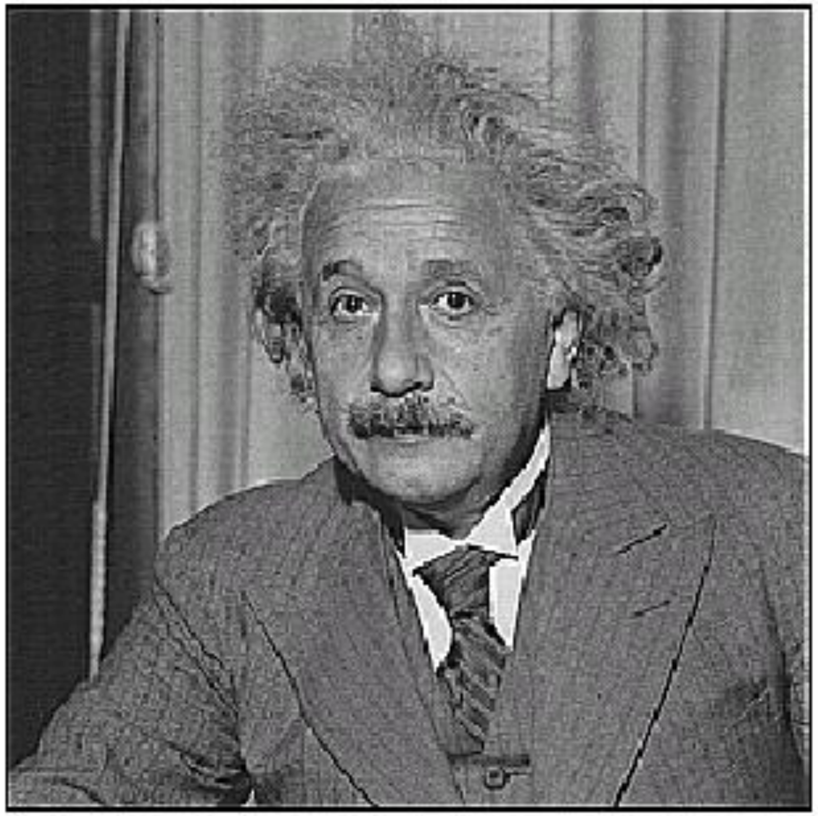
Mean filter (blurring)

Source: D. Lowe

Sharpening



before



after

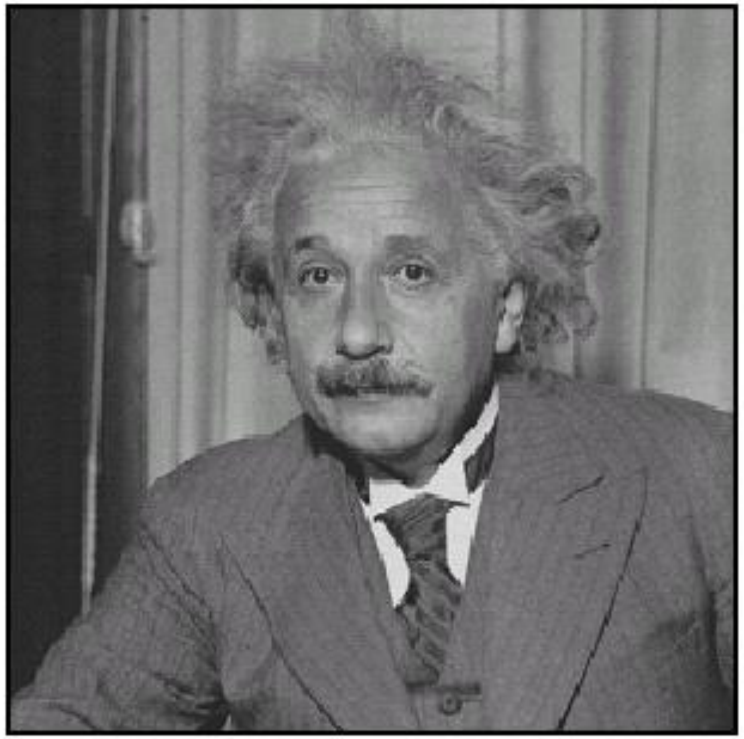
Source: D. Lowe

Sharpening

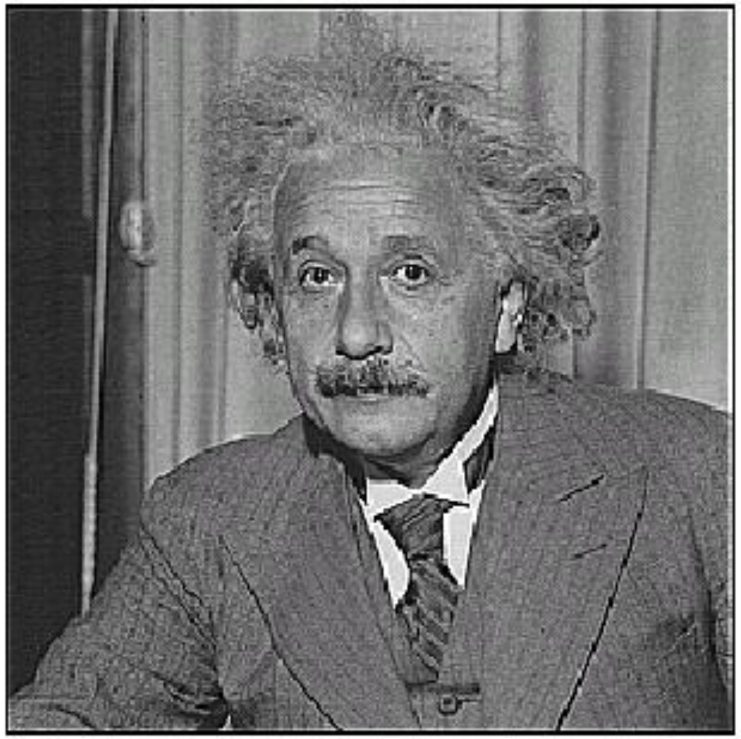
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Original image

(Original image) – (average of neighbors) = highlights



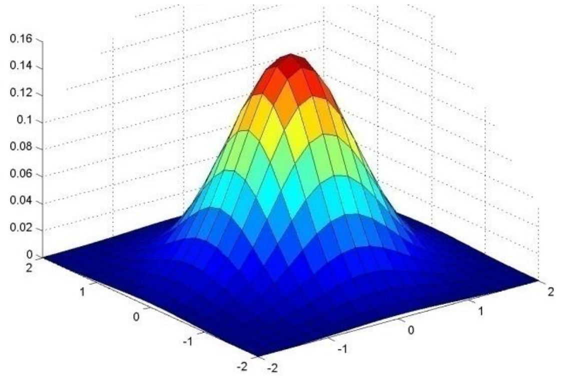
before

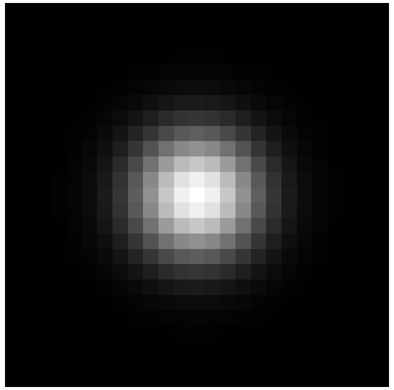


after

Source: D. Lowe

Gaussian Kernel



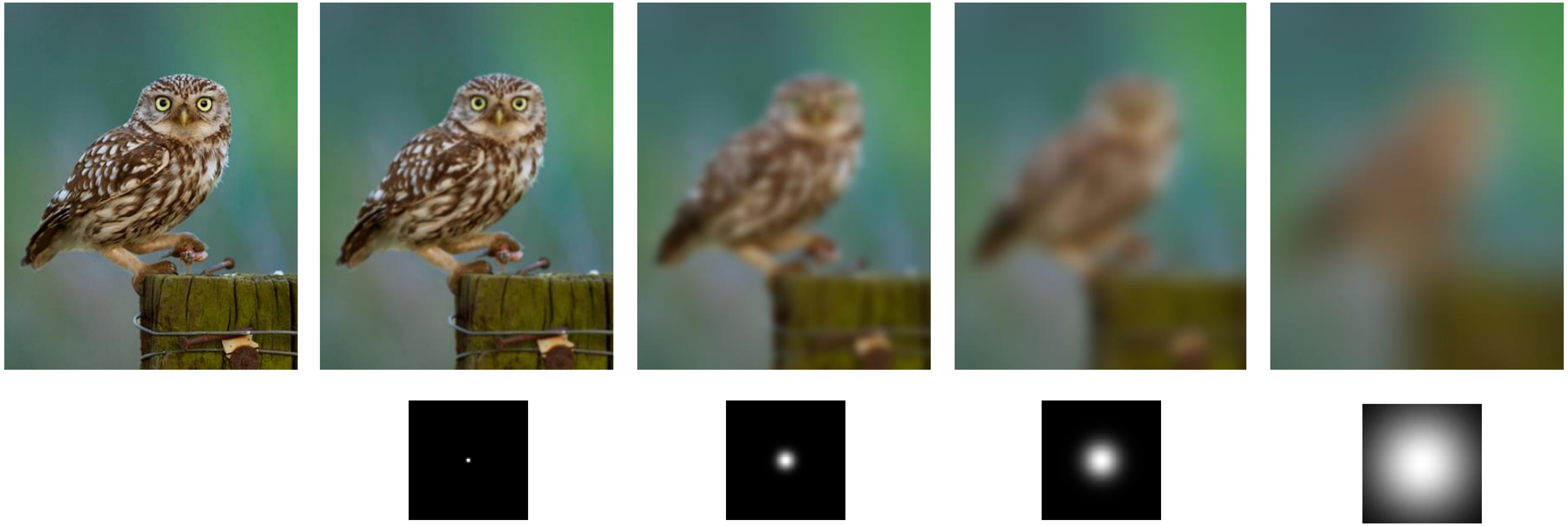


Approximated by: $\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

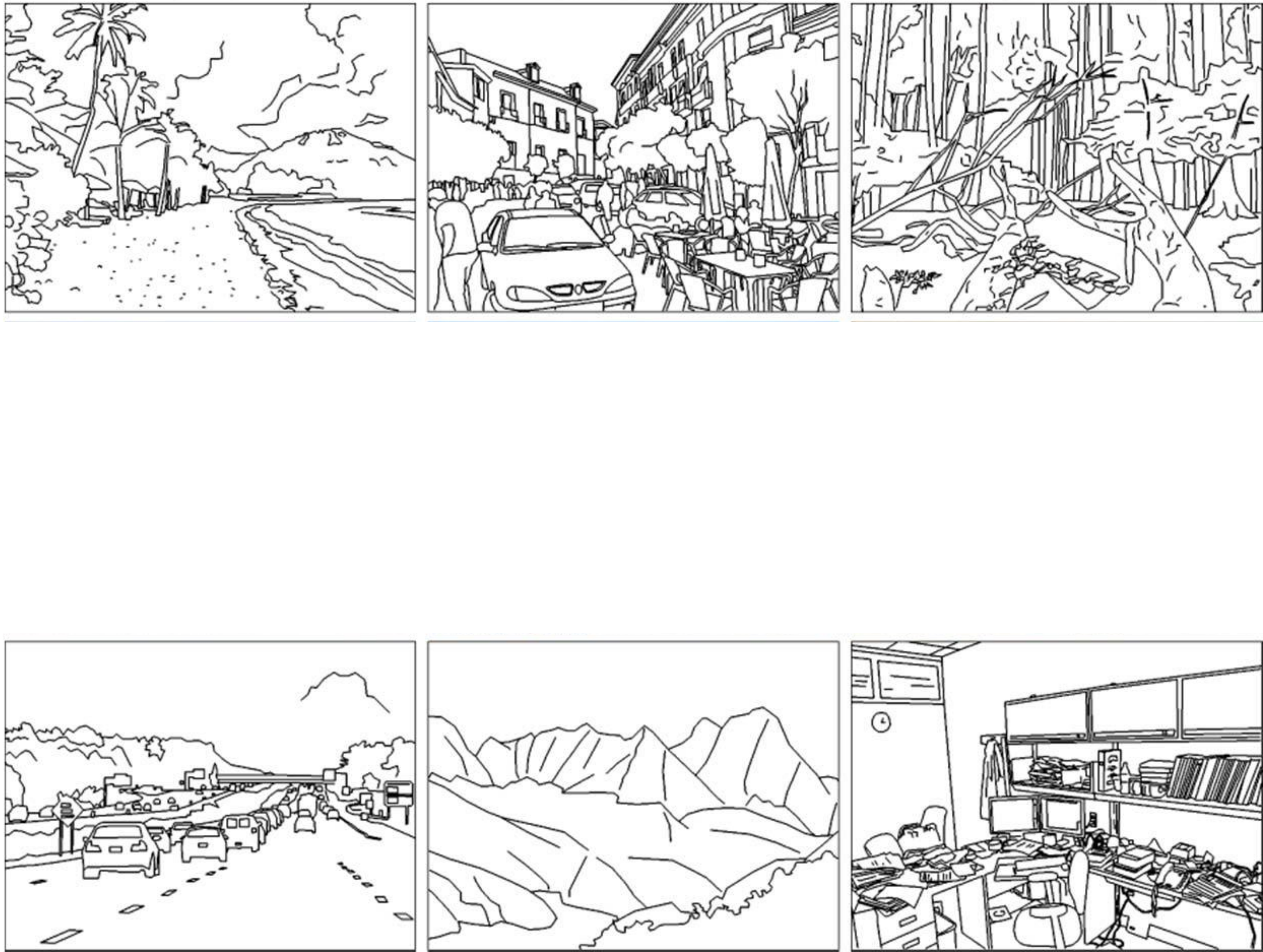
Source: C. Rasmussen

Gaussian filter



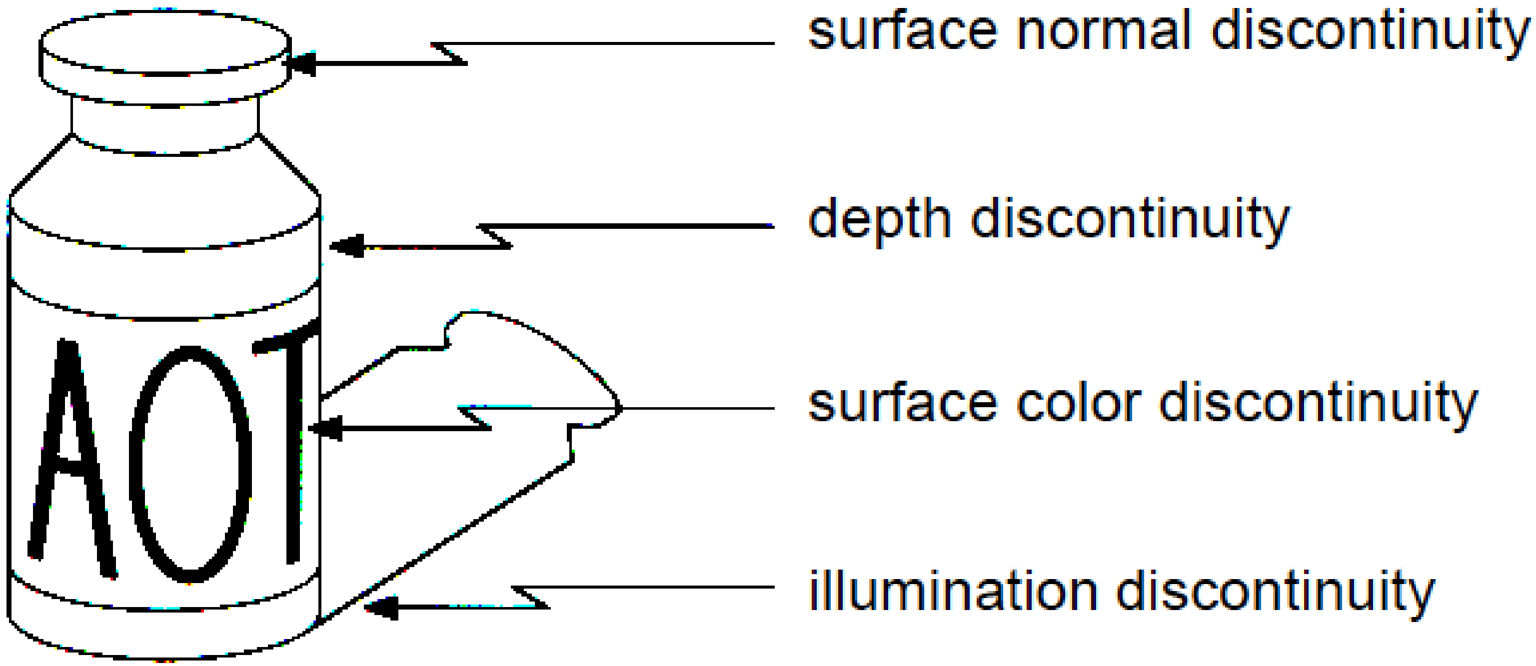
σ = 1 pixel σ = 5 pixels σ = 10 pixels σ = 30 pixels

Edge Detection



Edge Detection

Origin of Edges



Edges

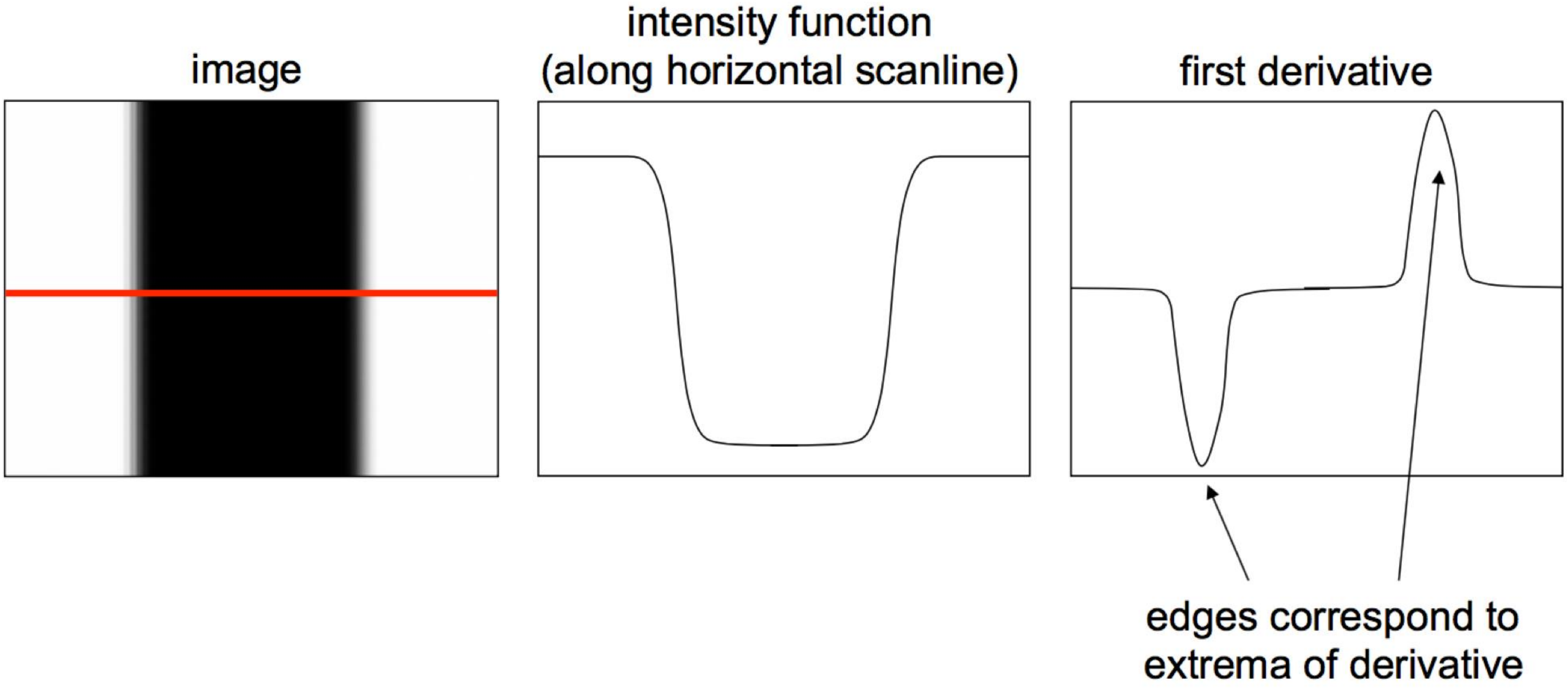


image
intensity function
(along horizontal scanline)
first derivative
edges correspond to
extrema of derivative

Edge Detection

Through Convolution

Sobel:

-1	0	1
-2	0	2
-1	0	1

Prewitt:

-1	0	1
-1	0	1
-1	0	1

Canny:

more complex multi-stage algorithm that uses a Gaussian filter and the intensity gradient in an image. One of the most widely-used techniques.

Basic Image Processing

- An image is an array of pixels, which can be binary, grey-value, or color.
- Image filtering takes an image as input and performs basic computation at each pixel to construct an output image.
- Image filters can be used for:
  - Increasing contrast
  - Removing noise
  - Finding edges
- Basic image processing is a precursor for image understanding (aka perception).