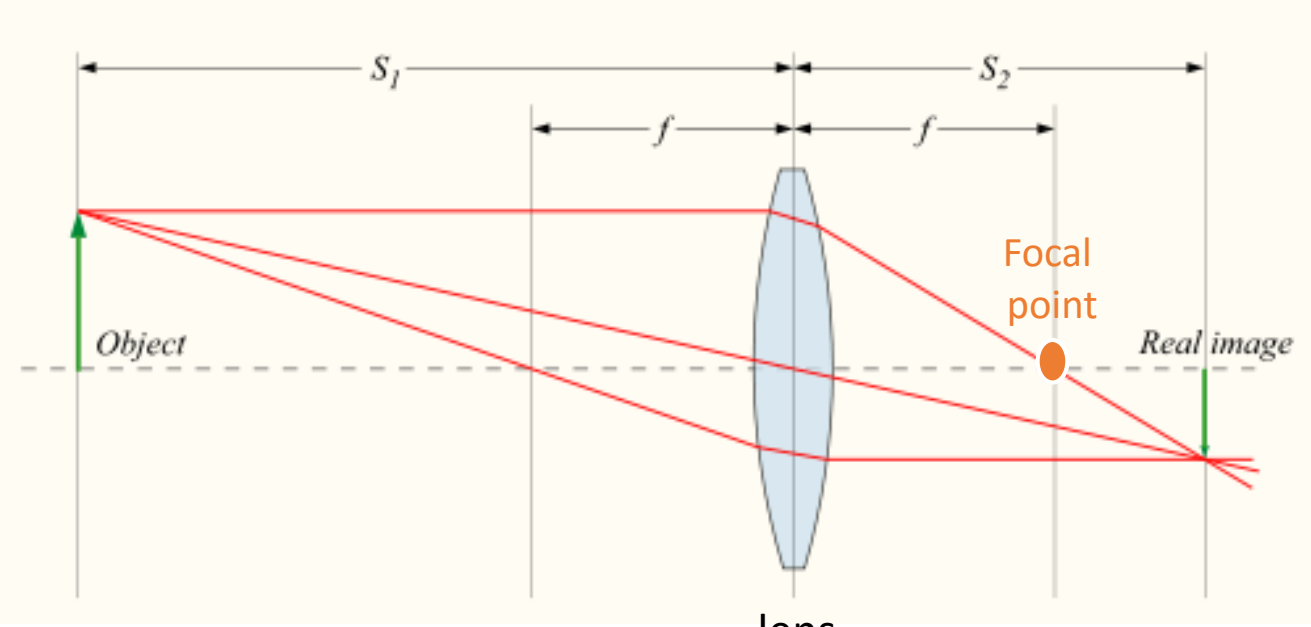
Mid-term Review

CPS592 – Visual Computing and Mixed Reality

Outline

- Imaging
- Color
- Image Operators
- Image Filtering
- Bilateral Filtering
- Color Transfer
- Visual Saliency
- Saliency Prediction Computational Models

Focal length

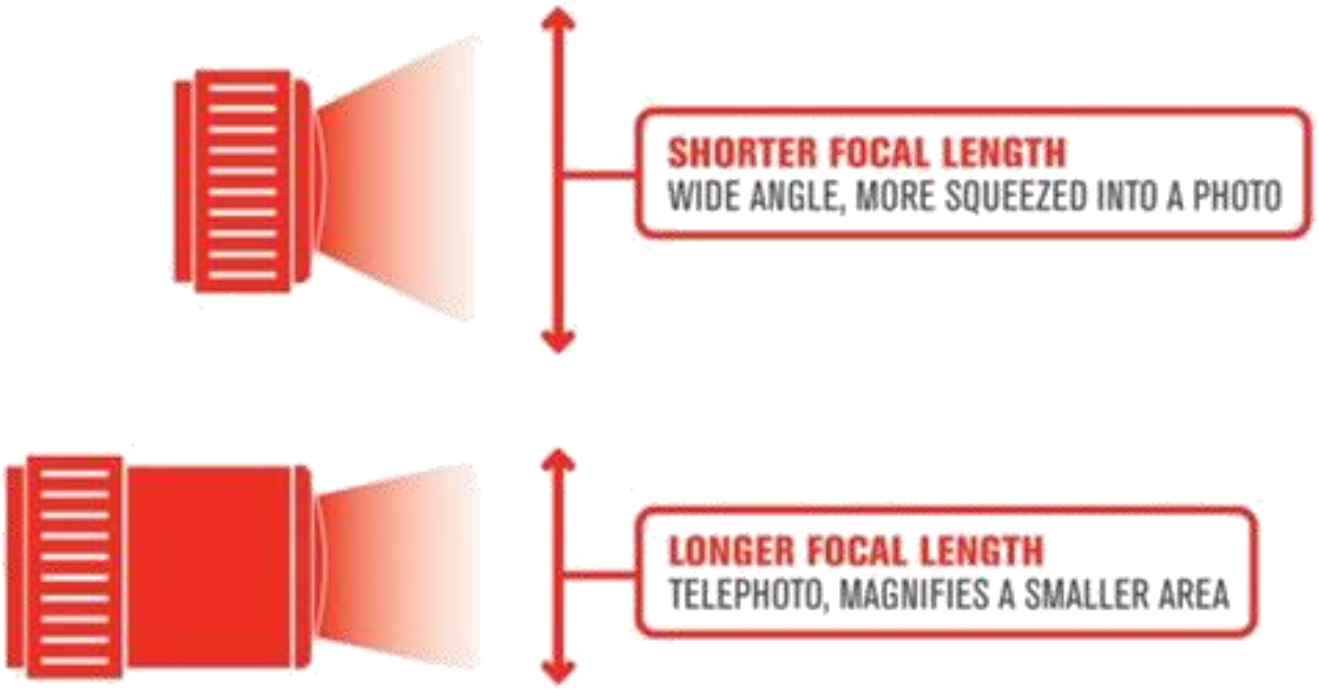


Thin lens equation:

$$\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$$

Any object point satisfying this equation is in focus

Focal length



Exposure

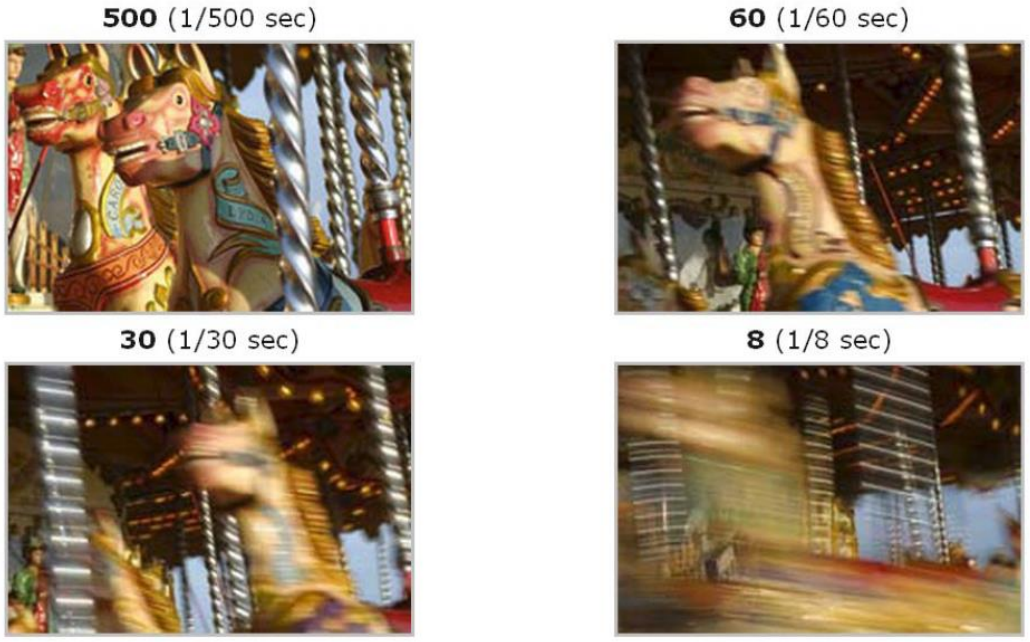
- Exposure controls how much light hits the camera sensor
- Two ways to control this:
  - **Aperture**: the “hole” in the optical path for the light
  - **Shutter speed**: the time the “hole” is opened

Aperture

Controllable Shutter

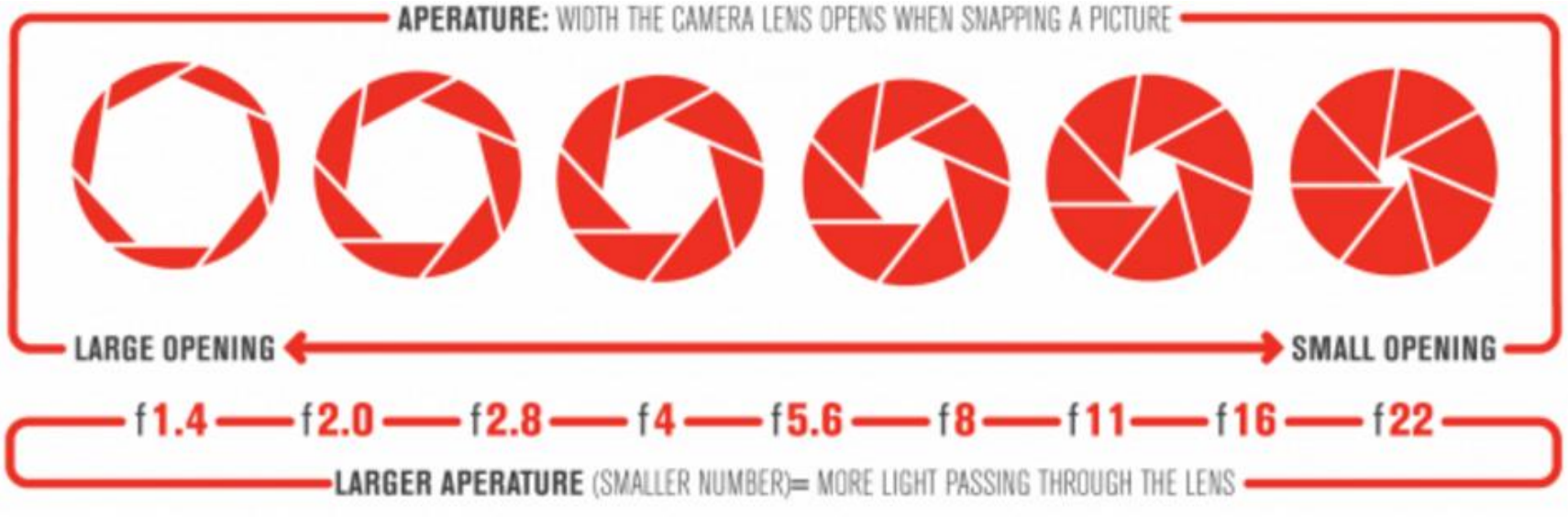
Shutter speed

- Shutter speed
  - Expressed in fraction of a second:
  - 1/30, 1/60, 1/125, 1/250, 1/500
  - (in reality, 1/32, 1/64, 1/128, 1/256, . . .)



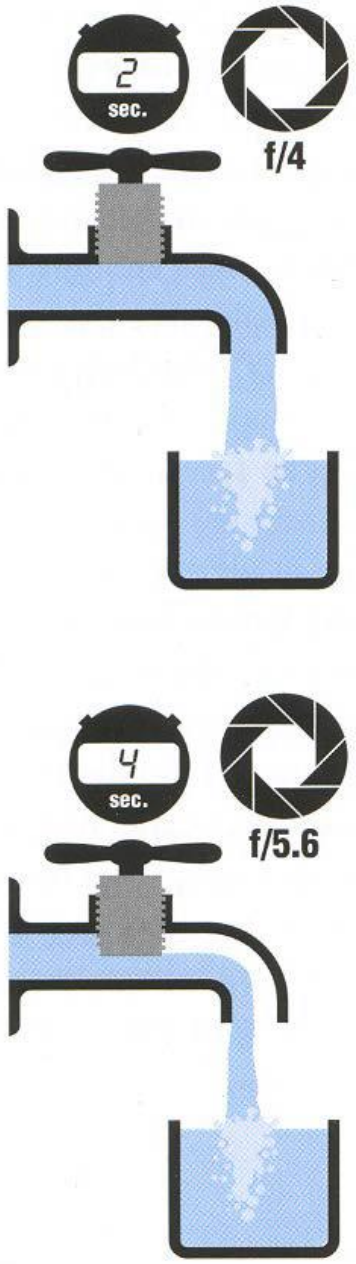
Aperture

- Aperture
  - Expressed as ratio of aperture size to focal length (f-stop)
  - f/2.0, f/2.8, f/4, f/5.6, f/8, f/11, f/16, f/22, f/32
  - f/***X***, means focal length is ***X*** times bigger than the aperture
  - Each f-stop reduces the area of the aperture by half
  - So, the larger the f-stop, the smaller the aperture



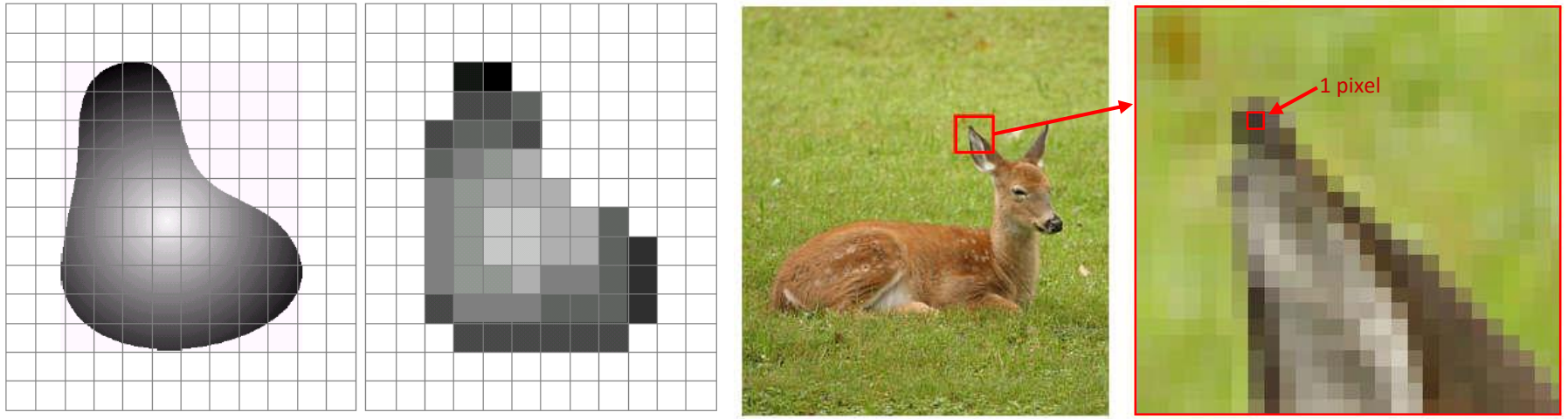
Exposure

- The play between f-stop and shutter:
  - Aperture (in f stop)
  - Shutter speed (in fraction of a second)
- Reciprocity
  - The same exposure is obtained with a shutter speed twice as long and an aperture area half as big



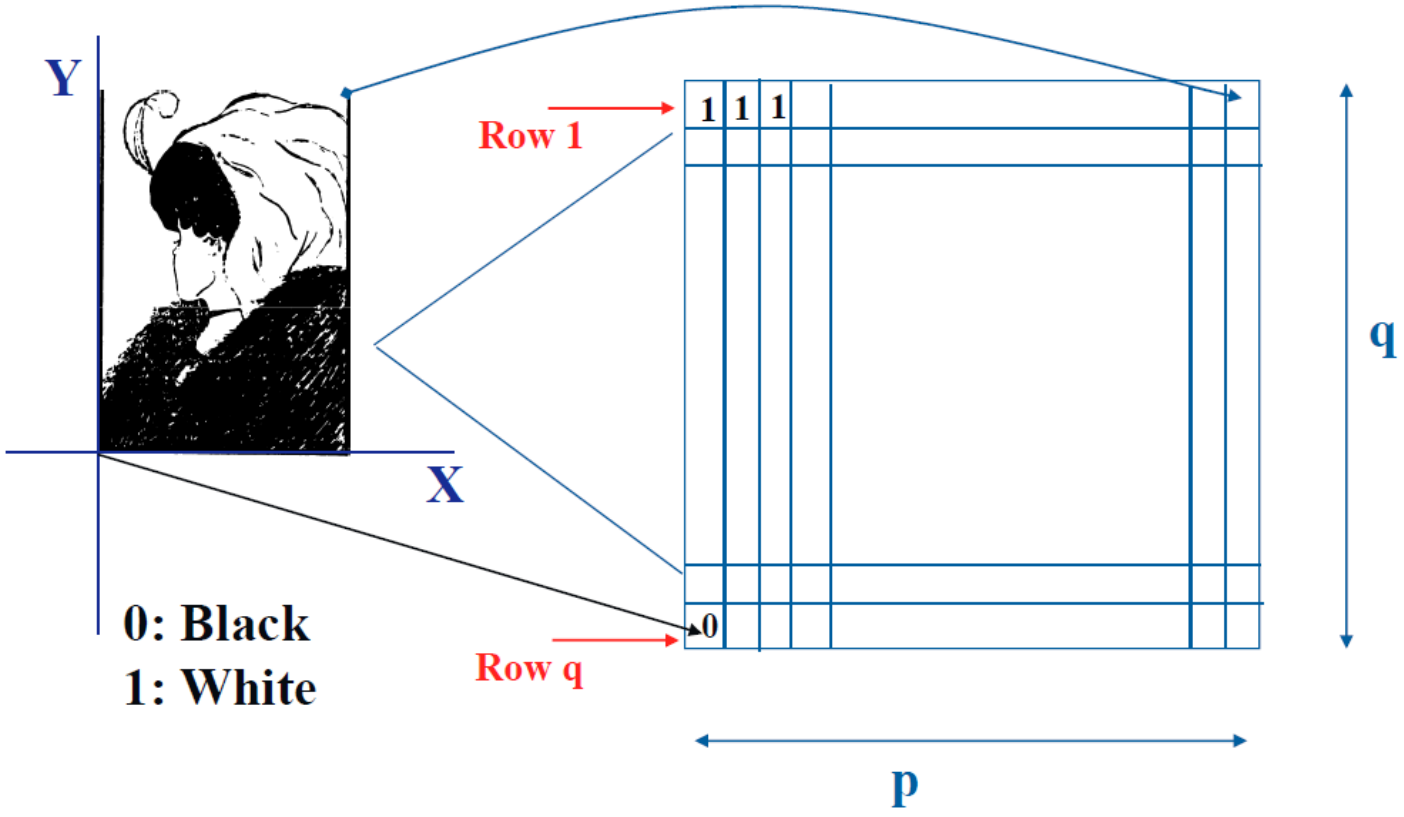
Digital Images

- A **digital image** is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels.
- We can think of an **image** as a function, *f*, from R^2 to R:
  - *f*(*x, y*) gives the **intensity** at position (*x, y*)

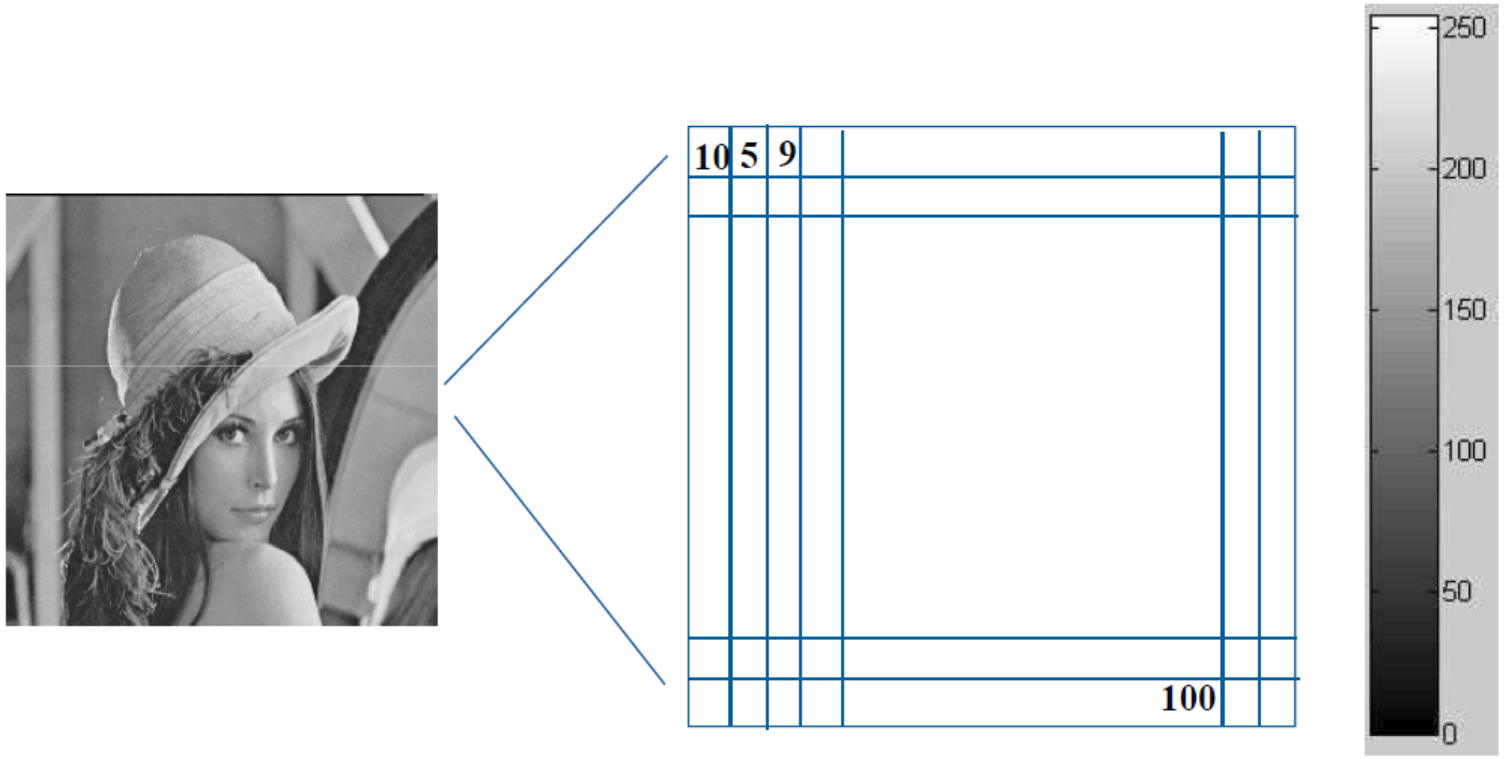


Digitization implies that a digital image is an *approximation* of a real scene.

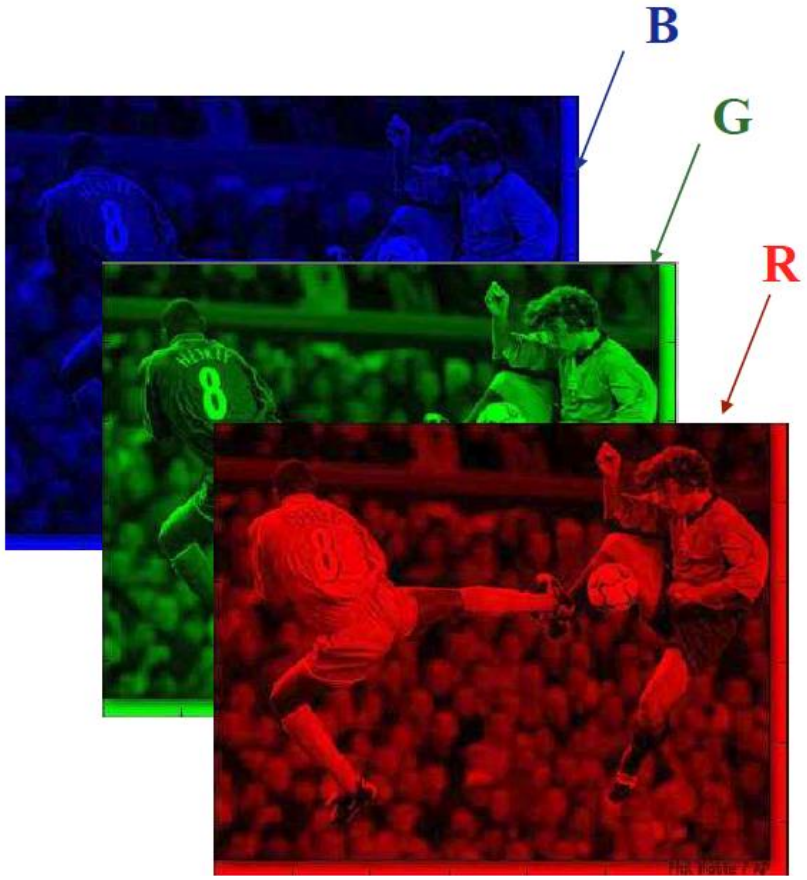
Binary Image



Grayscale Image

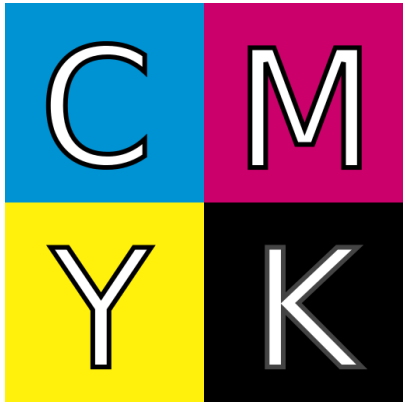


Color Image (RGB)



3-Component Color

- The de facto representation of color on screen display is RGB. (additive color)
- Some printers use CMY(K), (subtractive color)



Main Color Spaces

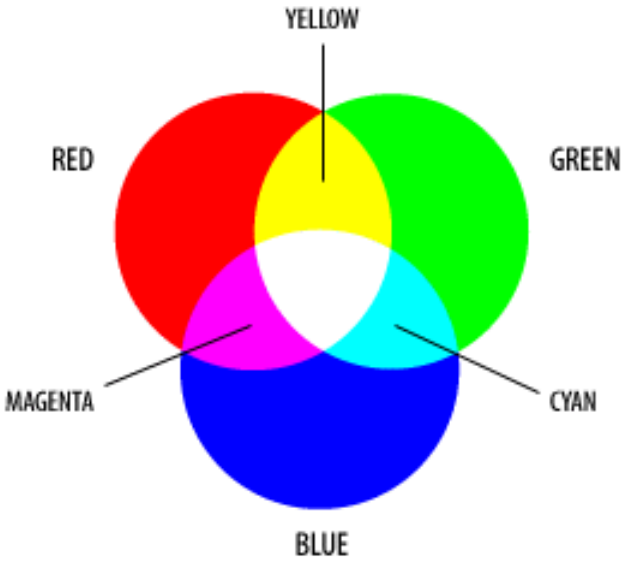
- RGB, CMYK
- HSV (Munsell, HSL, IHS)
- Lab, UVW, YUV, YCrCb, Luv,

Differences in Color Spaces

- What is the use?
  - Display,
  - Editing
  - Computation
  - Compression
- Several key (very often conflicting) features may be sought after:
  - Additive (RGB) or subtractive (CMYK)
  - Separation of luminance and chromaticity
  - Equal distance between colors are equally perceivable

RGB

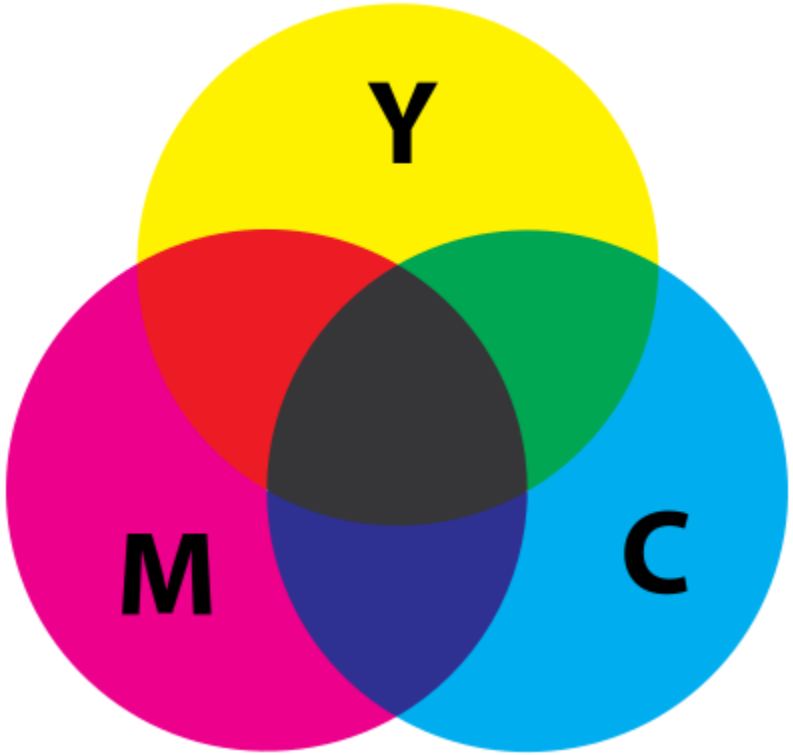
- The de facto standard



$$\mathbf{C} = r\mathbf{R} + g\mathbf{G} + b\mathbf{B}$$

CMY(K): printing

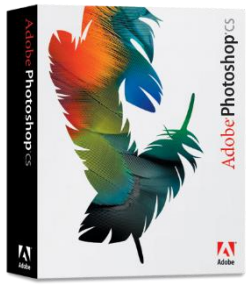
- Cyan, Magenta, Yellow (Black) – CMY(K)
- A subtractive color model



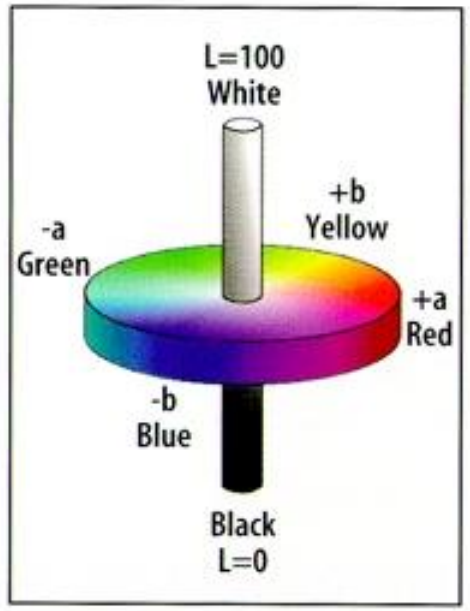
HSV

- This color model is based on polar coordinates, not Cartesian coordinates.
- HSV is a non-linearly transformed (skewed) version of RGB cube
  - Hue: quantity that distinguishes color family, say red from yellow, green from blue
  - Saturation (Chroma): color intensity (strong to weak). Intensity of distinctive hue, or degree of color sensation from that of white or grey
  - Value (luminance): light color or dark color

Lab: photoshop



- Photoshop uses this model to get more control over color
- Luminance: L
- Chrominance: a – ranges from green to red and b ranges from blue to yellow



Lab model

Yuv and YCrCb: digital video

- Initially, for PAL analog video, it is now also used in CCIR 601 standard for digital video
- Y (luminance) is the CIE Y primary.
  $Y = 0.299R + 0.587G + 0.114B$
- *Chrominance* is defined as the difference between a color and a reference white at the same luminance. It can be represented by U and V -- the *color differences*.
  $U = B - Y; V = R - Y$
- YCrCb is a scaled and shifted version of YUV and used in JPEG and MPEG (all components are positive)

  $Cb = (B - Y) / 1.772 + 0.5; Cr = (R - Y) / 1.402 + 0.5$

Convert RGB to grayscale

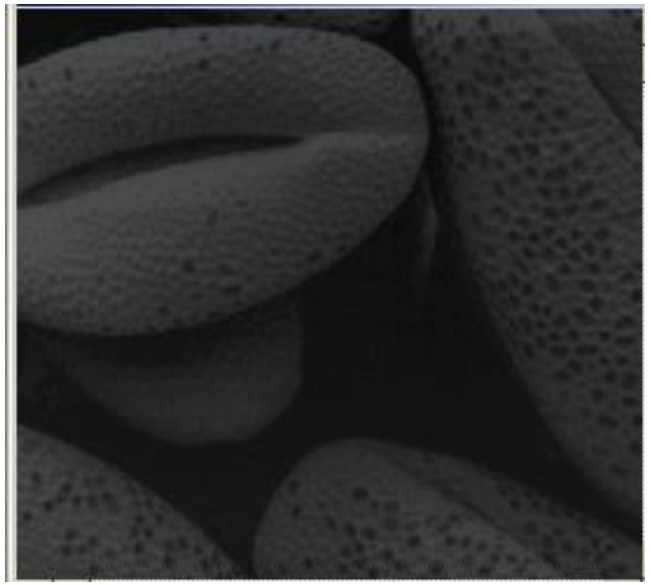
- intensity = 0.2989*R + 0.5870*G + 0.1140*B

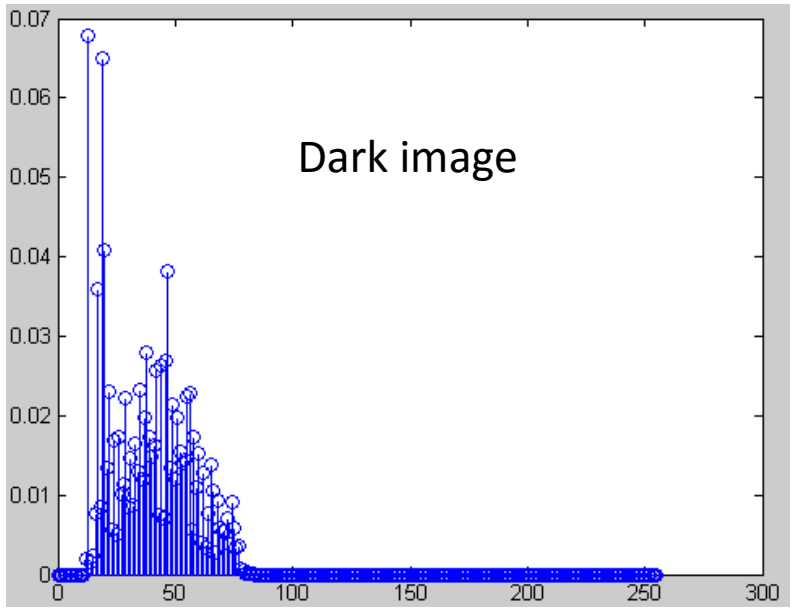
Original RGB

Grayscale Conversion

Image Histogram

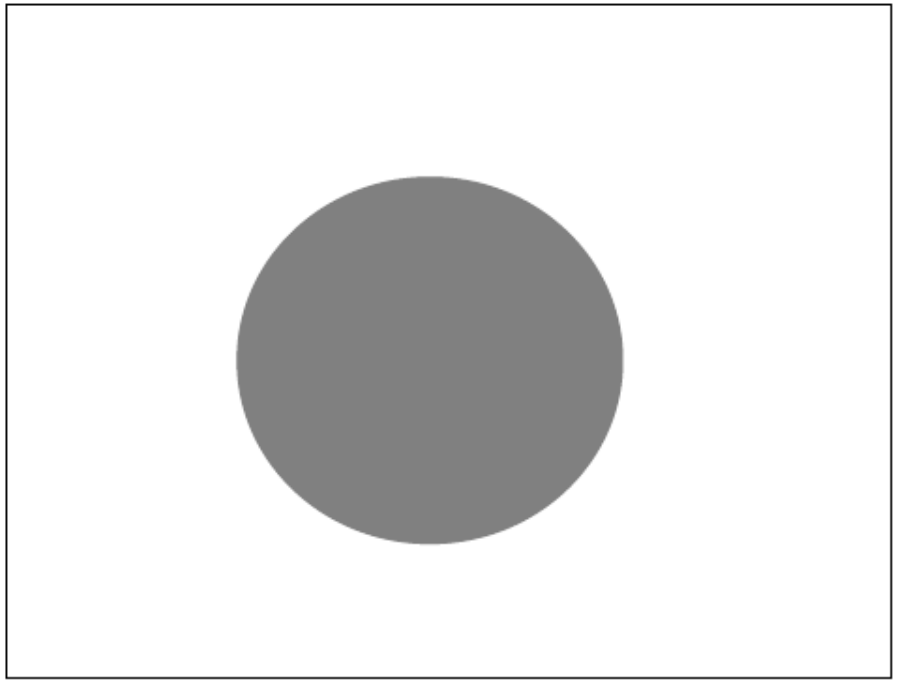
- The shape of a histogram provides useful information for contrast enhancement.

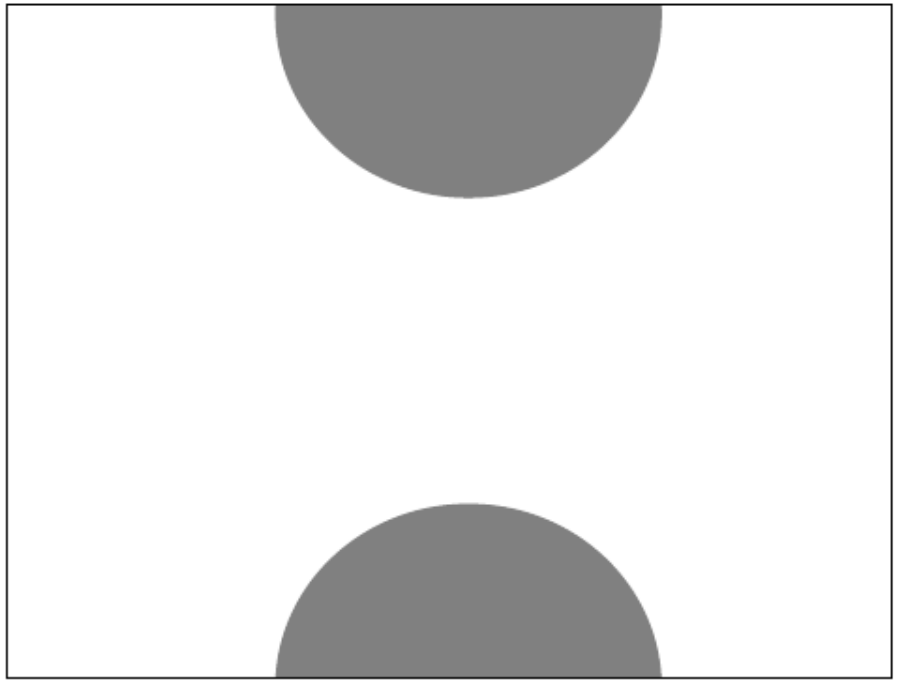





```
>> clear;
[ix,map]=imread('Fig3_15a.jpg');
imshow(ix)
figure;
ix=double(ix);
h=histogram(ix);
figure
stem(0:255,h);
```

Image Histogram

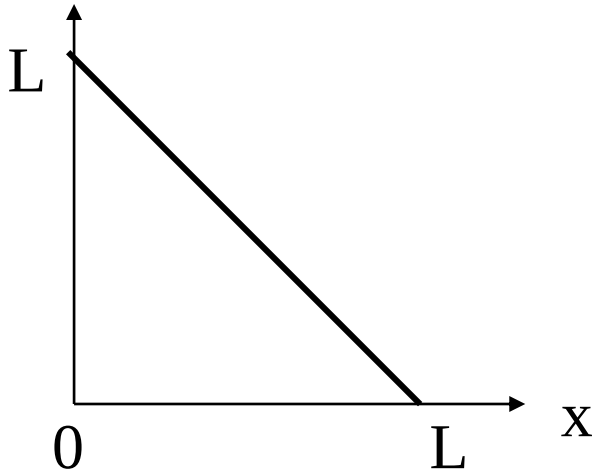




- Keep in mind that histograms are not unique

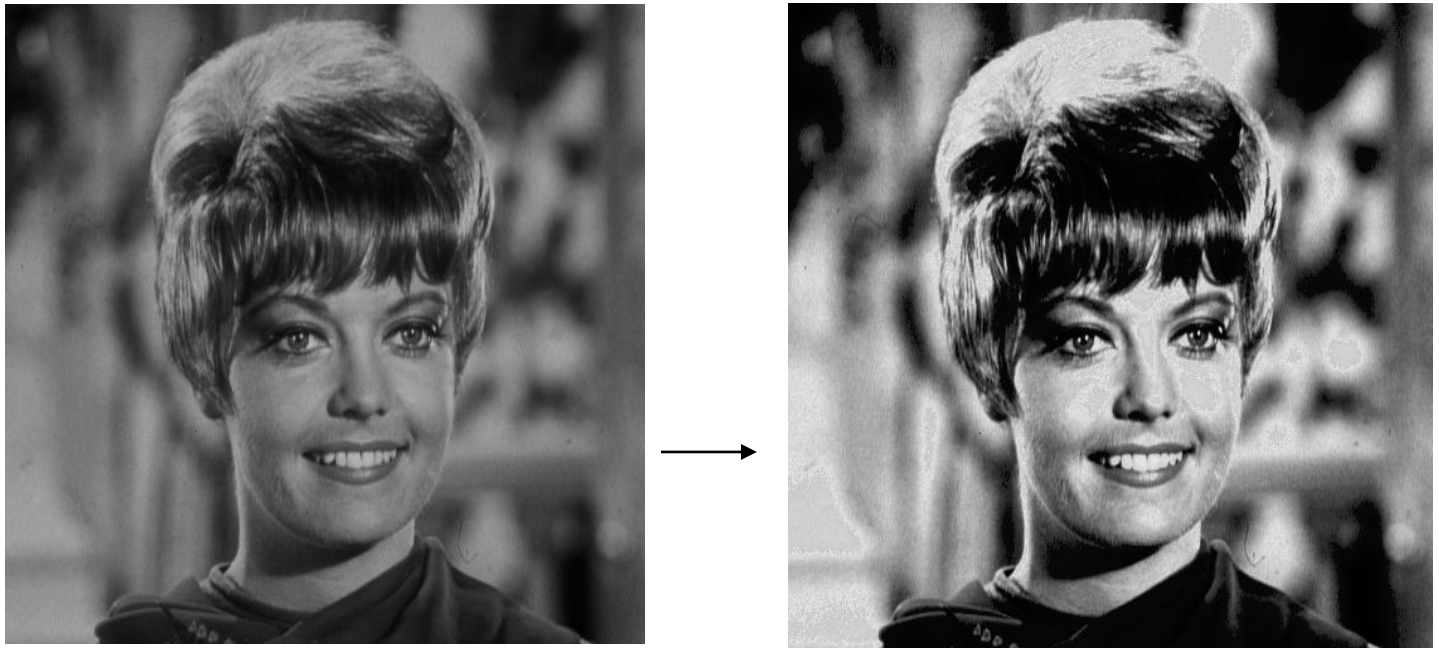
Digital Negative

$$y = L - x$$



Histogram Stretching

- Stretch the histogram to increase the contrast.



$[50,150] -> [30,200]$

Histogram Equalization

- Objective: we want an image with equally many pixels at every gray level
  - This makes the image look nice
  - Also maximizes pixel resources
  - We would say this has an “equal histogram”

Example

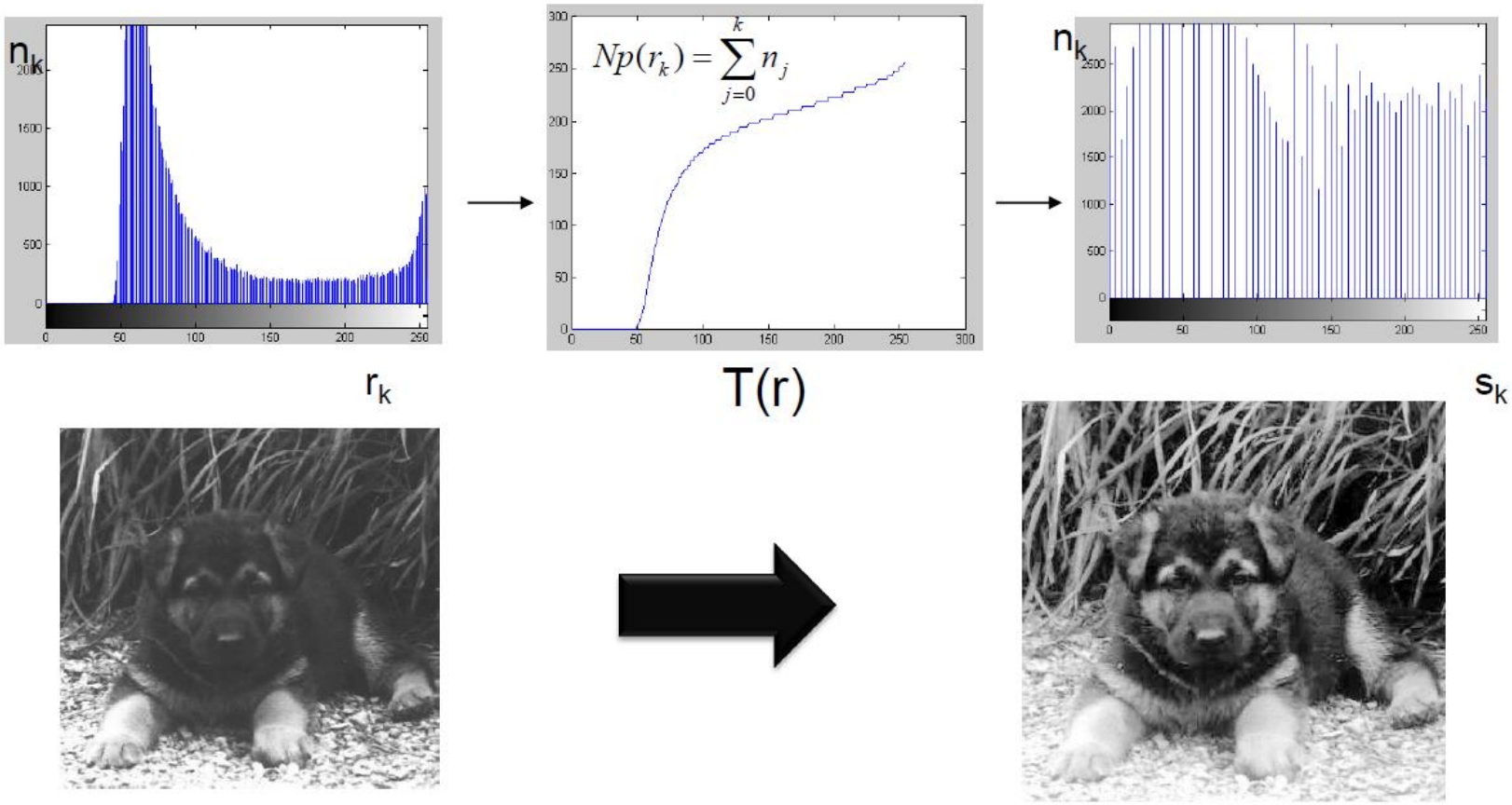
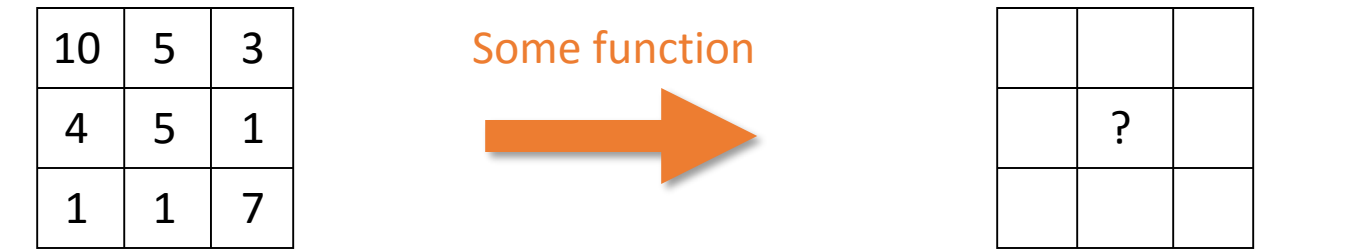


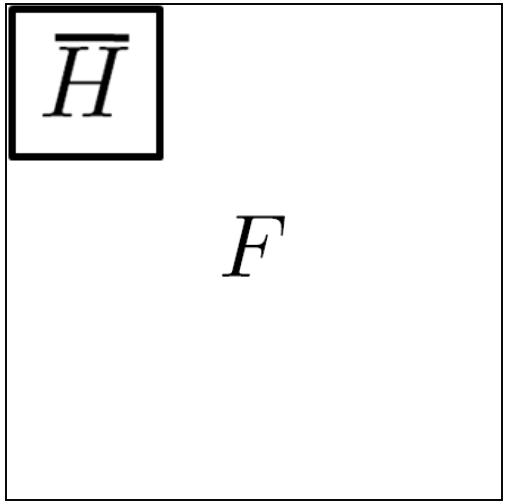
Image Filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel



Source: L. Zhang

Example



Adapted from F. Durand

Mean filtering

H

$*$

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

F

$=$

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

G

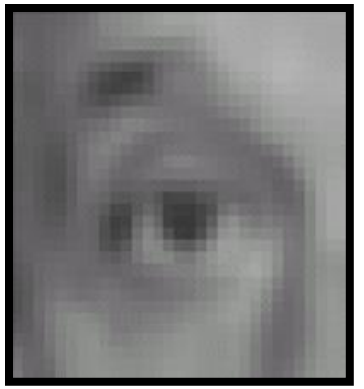
$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$*$

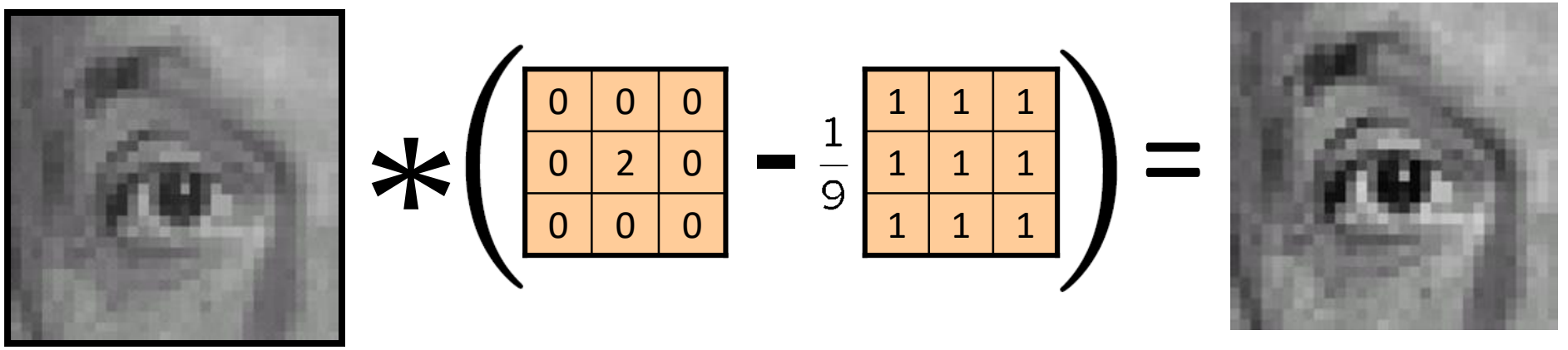
Original

$=$



Blur (with a mean filter)

Linear filters: examples

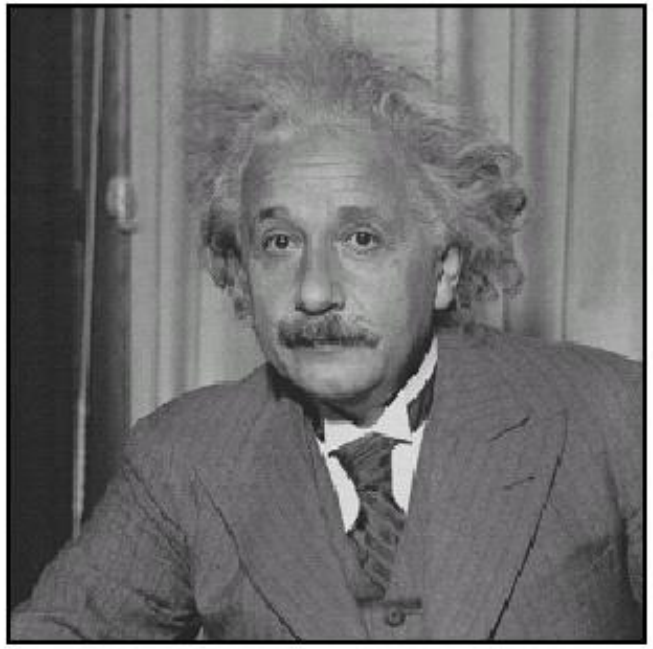


Original

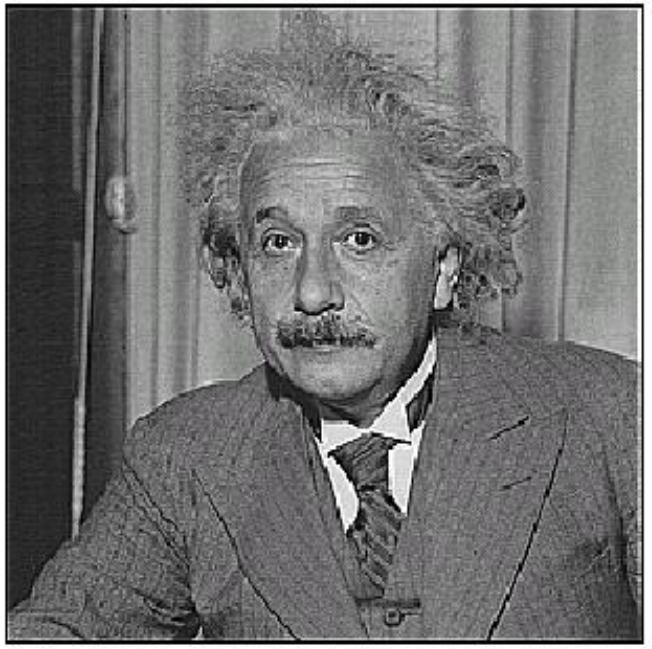
Sharpening filter
(accentuates edges)

Source: D. Lowe

Sharpening



before



after

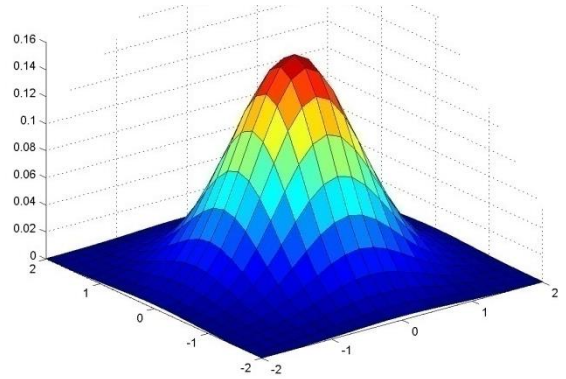
Source: D. Lowe

Gaussian filters

- A Gaussian kernel gives less weight to pixels further from the center of the window

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

$F[x, y]$



$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

Rank Filters: Median Filter

- One of the most popular non-linear filter
- Find the median of the window
- Preserves edges
- Removes impulse noise, avoids excessive smoothing

2	3	8
3	**4**	10
4	2	9

pixel values about (x,y) window 3x3

neighbor sort = {2,2,3,3,4,4,8,9,10}

f(x,y) = median

Rank Filters: Min/Max Filter

- Find the min or max of the neighborhood
- Not as “mainstream” as median filter
- Has various uses, will talk about these more later.

2	3	8
3	4	10
4	2	9

pixel values about (x,y) window 3x3

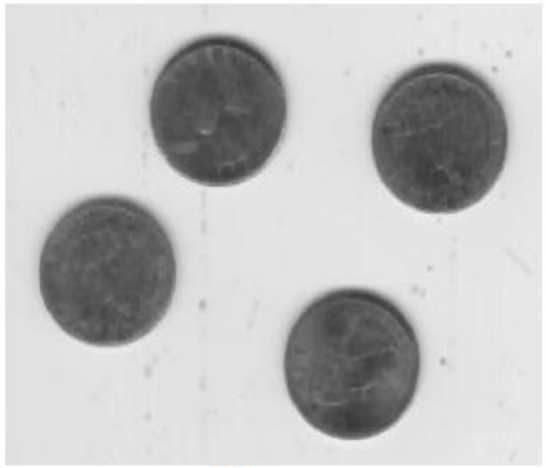
neighbor sort = {2,2,3,3,4,4,8,9,10}

f(x,y) = min **f(x,y) = max**

Examples

Original Image

Median



Min

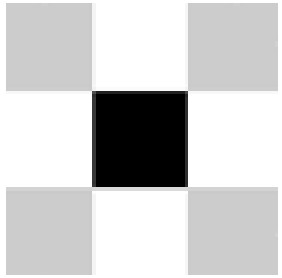
Max

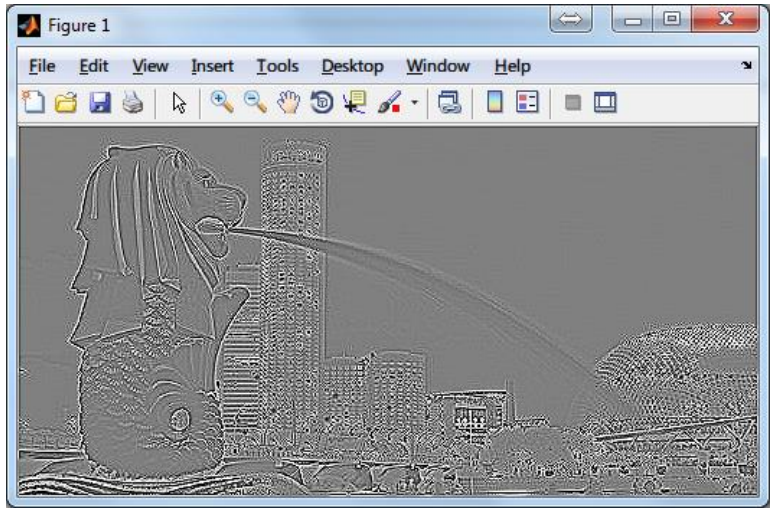
Laplacian Filter

kernel = [0 1 0;

1 -4 1;

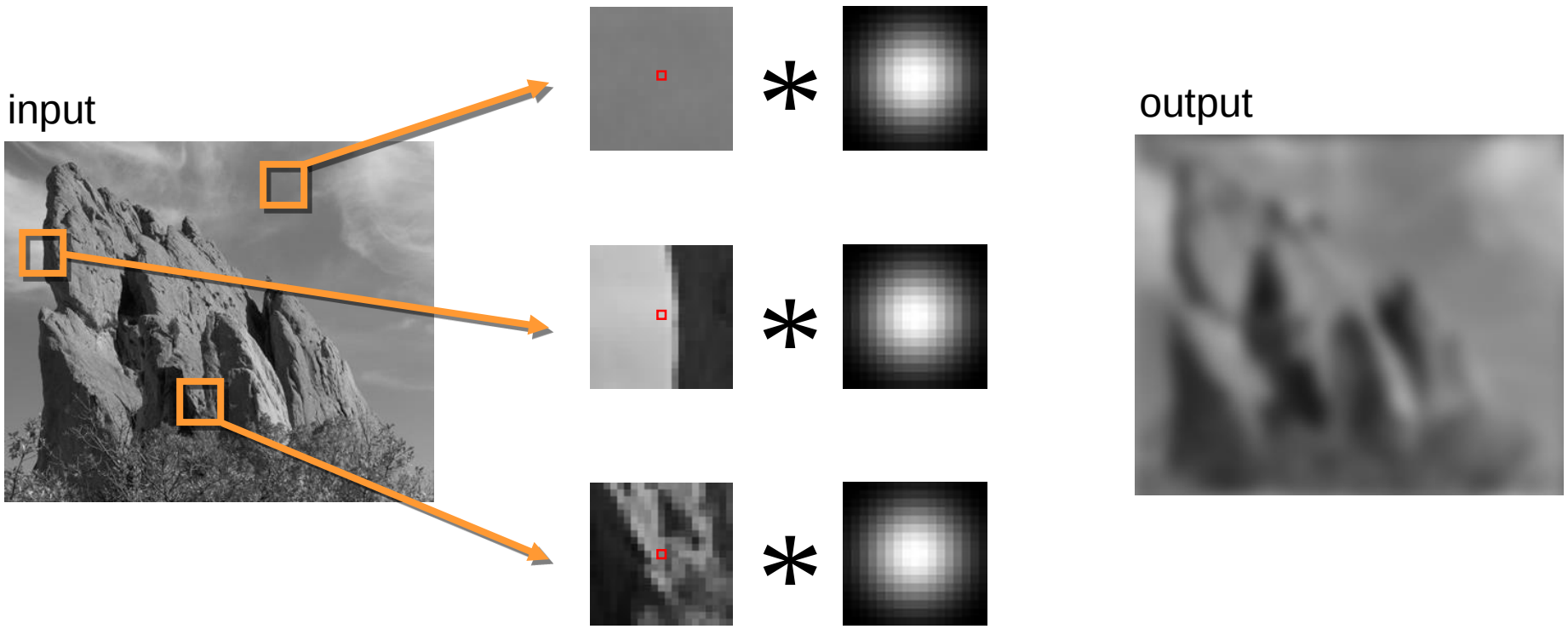
0 1 0];





Slides taken from Sylvain Paris, Siggraph 2007

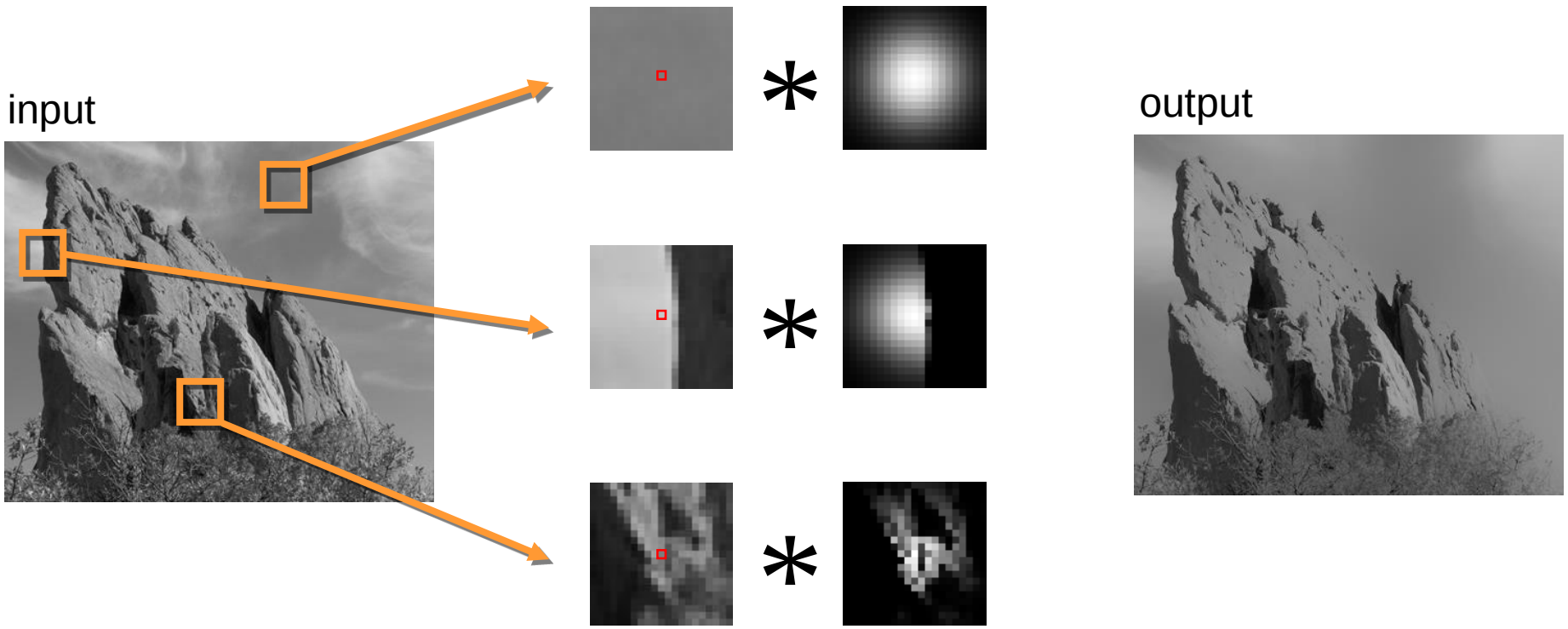
Gaussian Smoothing



Same Gaussian kernel everywhere
Averages across edges ⇒ blur

Slides taken from Sylvain Paris, Siggraph 2007

Bilateral Filtering



Kernel shape depends on image content
Avoids averaging across edges

Bilateral Filter Definition: an Additional Edge Term

Idea: **weighted average of pixels**.

new

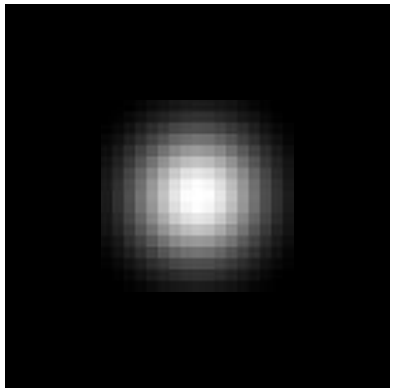
not new

new

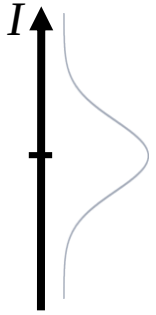
$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|) G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) I_{\mathbf{q}}$$

normalization factor

space weight



range weight



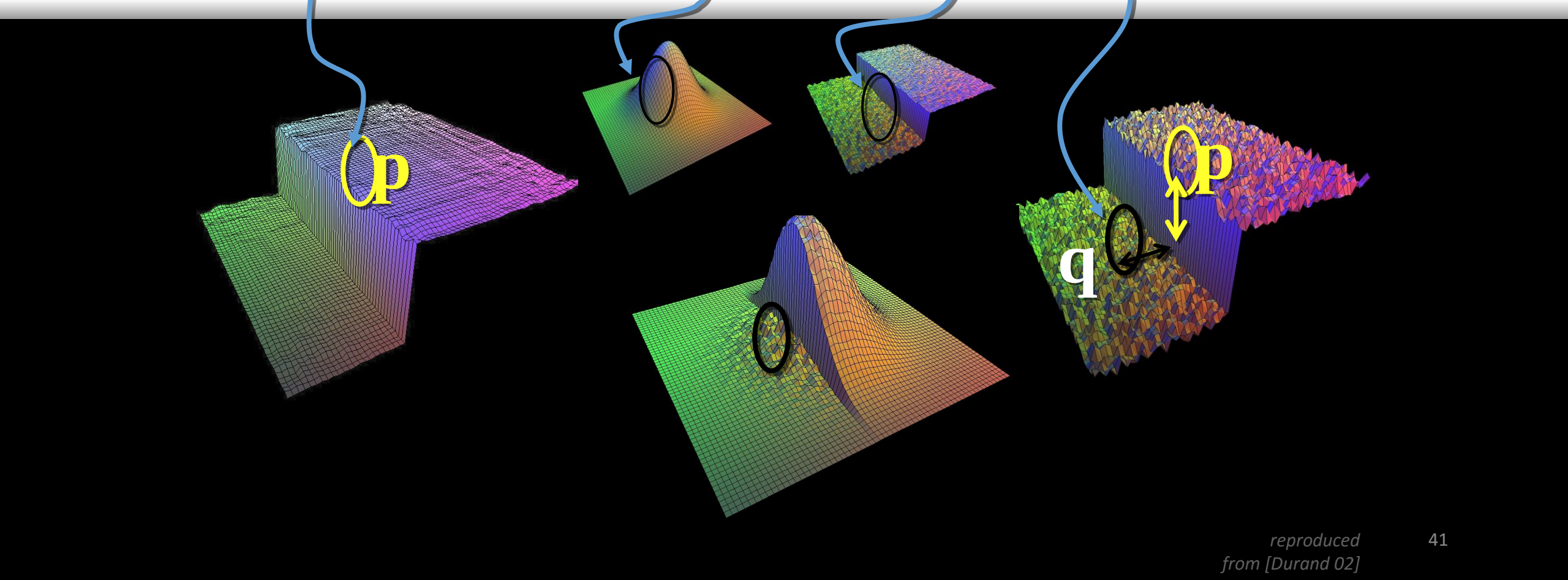
$$G_{\sigma_s}(||p-q||) = \frac{1}{2\pi\sigma^2} e^{-\frac{||p-q||^2}{\sigma^2}}$$

$$h(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

$$W_p = \sum_{q \in S} G_{\sigma_s}(||p-q||) G_{\sigma_r}(|I_p - I_q|)$$

Bilateral Filter on a Height Field

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$



reproduced from [Durand 02]

Implementation

- Brute-force Implementation
- Separable Kernel [Pham and Van Vliet 05]
- Box Kernel [Weiss 06]

Brute-force Implementation

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$

For each pixel $\mathbf{p}$

For each pixel $\mathbf{q}$

Compute $G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$

VERY SLOW!

Separable Kernel

[Pham and Van Vliet 05]

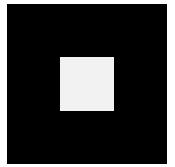
- Strategy: filter the rows then the columns

- Two “cheap” 1D filters
  instead of an “expensive” 2D filter

Box Kernel [Weiss 06]

- Bilateral filter with a square box window [Yarovlasky 85]

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|) G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) I_{\mathbf{q}}$$



box window

restrict the sum

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in B_{\sigma_s}} G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) I_{\mathbf{q}}$$

independent of position $\mathbf{q}$

- The bilateral filter can be computed only from the list of pixels in a square neighborhood.

Color Transfer

- Make one image look like another

target image source image result image

Method

- For both images:
  - Transfer to new color space
  - Compute mean and standard deviation along each color axis

- Linear color transfer: scale and shift
  - Shift and scale the source image to have same statistics as the target image

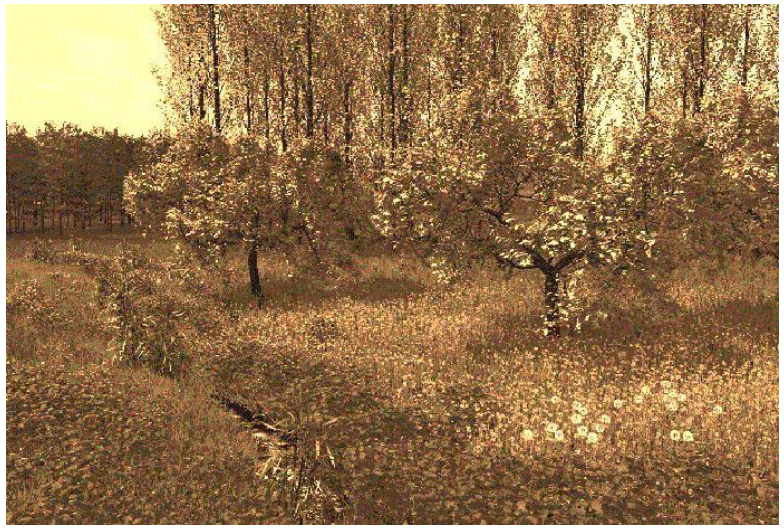
Color Transfer Implementation

$$C' = \frac{\sigma_t}{\sigma_s}(C_s - \mu_s) + \mu_t$$

- Where
  - C' = new color
  - C_s = old color
  - σ_t = SD of target image
  - σ_s = SD of source image
  - μ_t = mean of target image
  - μ_s = mean of source image

Color Transfer Example

Color Transfer Example



What is Visual Saliency?

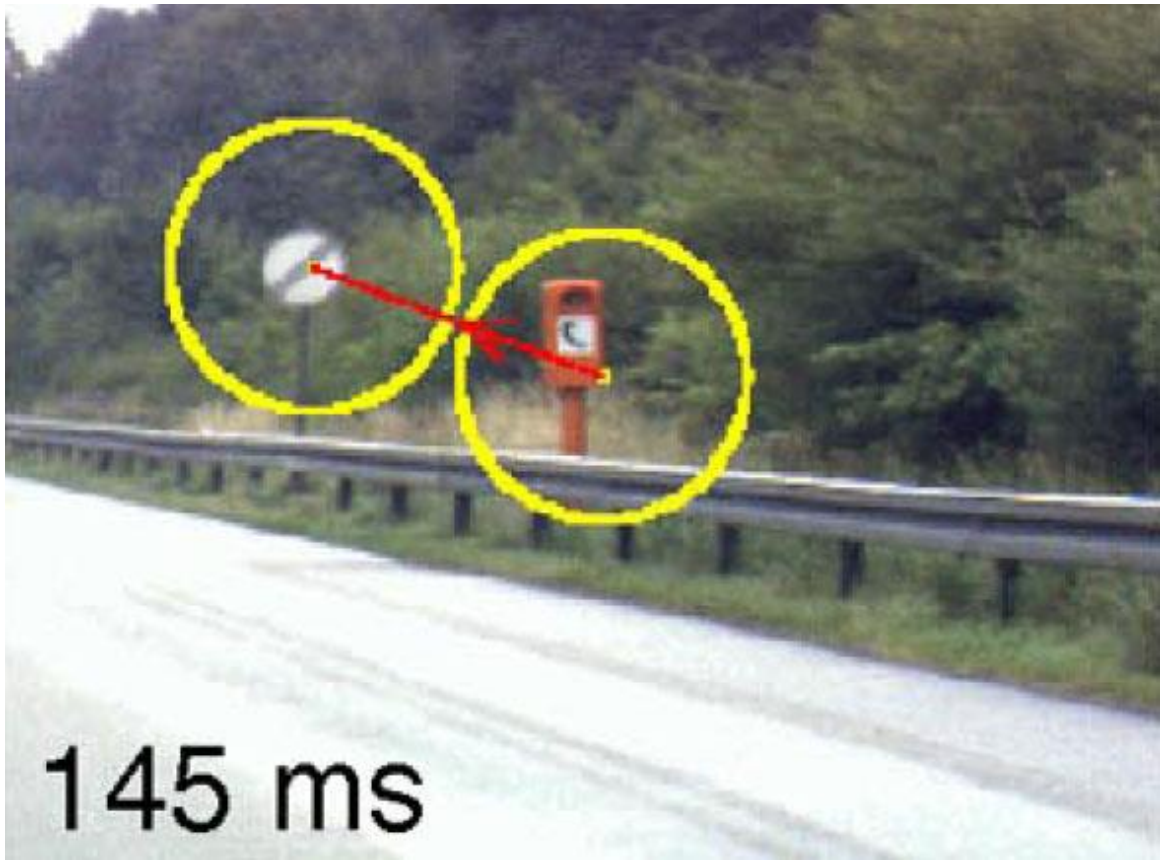
- Something is said to be **salient** if it **stands out**
- **E.g. road signs should have high saliency**

Introduction

- Trying to model visual attention
- Find locations of **Focus of Attention** in an image
- Use the idea of saliency as a basis for their model
- For primates focus of attention directed from:
  - **Bottom-up: rapid, saliency driven, task-independent**
  - Top-down: slower, task dependent

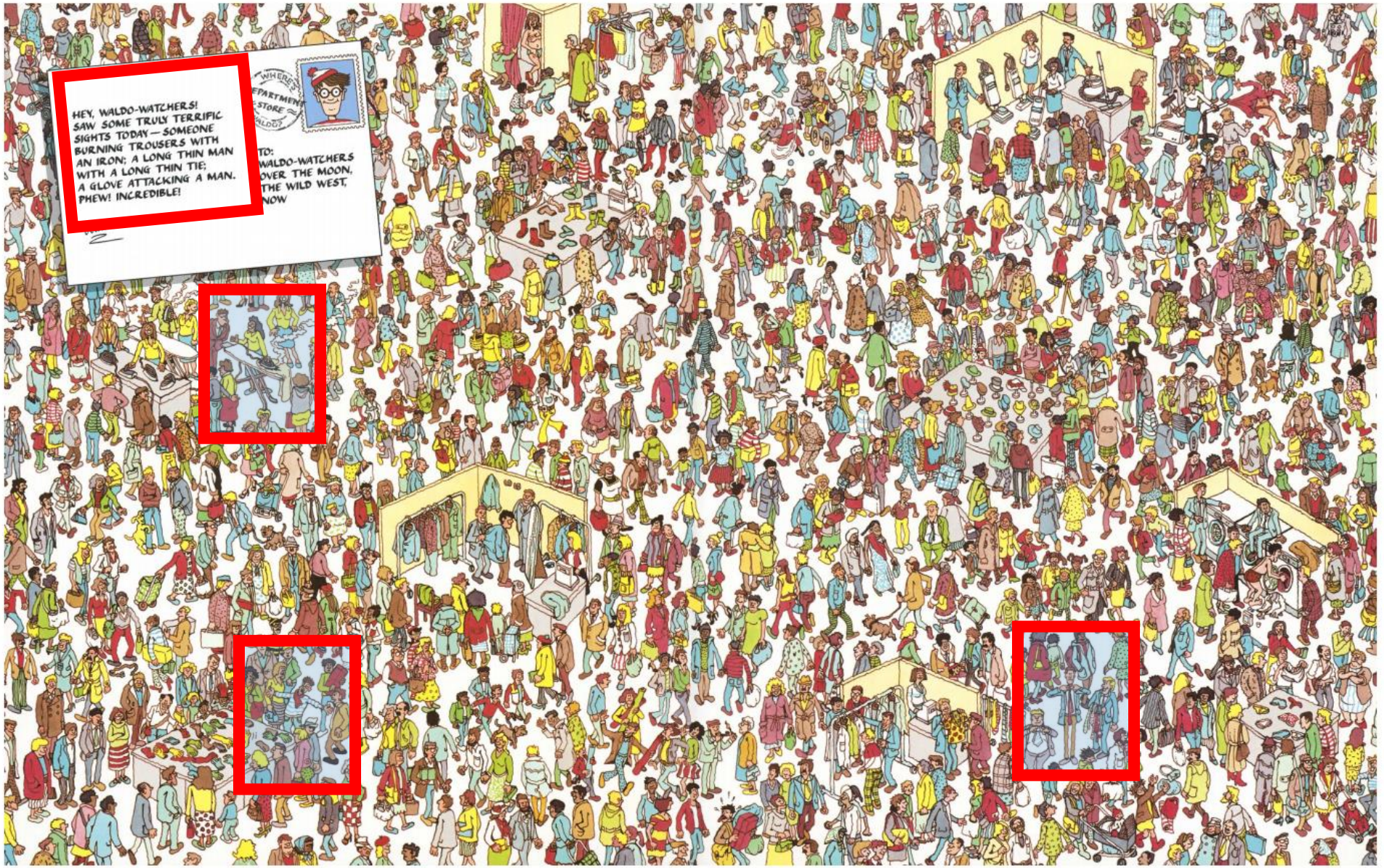
Bottom-up

- Task-independent



Top-down (Task dependent)

SOMEONE BURNING TROUSERS WITH AN IRON
A LONG THIN MAN WITH A LONG THIN TIE
A GLOVE ATTACKING A MAN



Purpose of visual saliency models

- Warning (animals, flashes, sudden motion)
- Exploration (find objects, verification)
- Inspection

Motivation - application

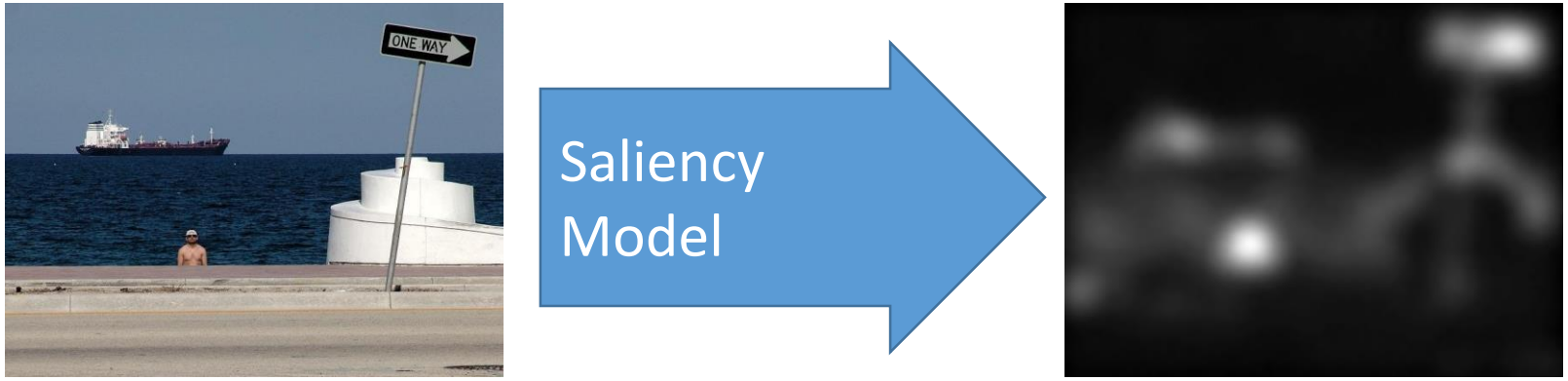
Image mosaicking: the salient details are preserved, with the use of smaller building blocks.

Input

Mosaic

Computational saliency models

Researchers create computational models to predict where people look.



A Model of Saliency-Based Visual Attention for Rapid Scene Analysis

Laurent Itti, Christof Koch, and Ernst Niebur

Itti's model

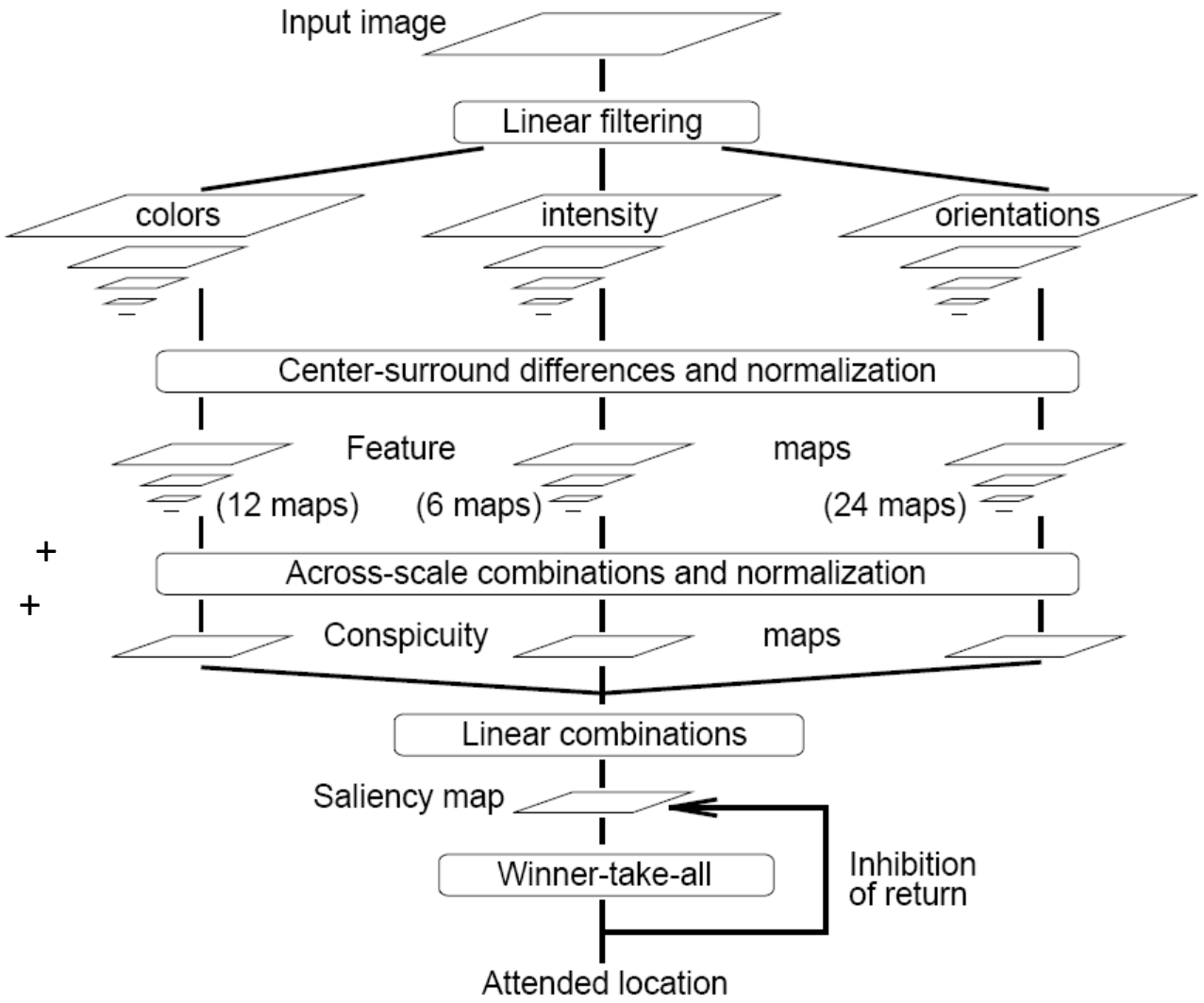
Gabor Pyramid +
Orientation Filters

Subtract low-res (3-4 octaves) from higher res

Normalize (0..1) +
map * (1 – max_{ave})^2 +
add maps

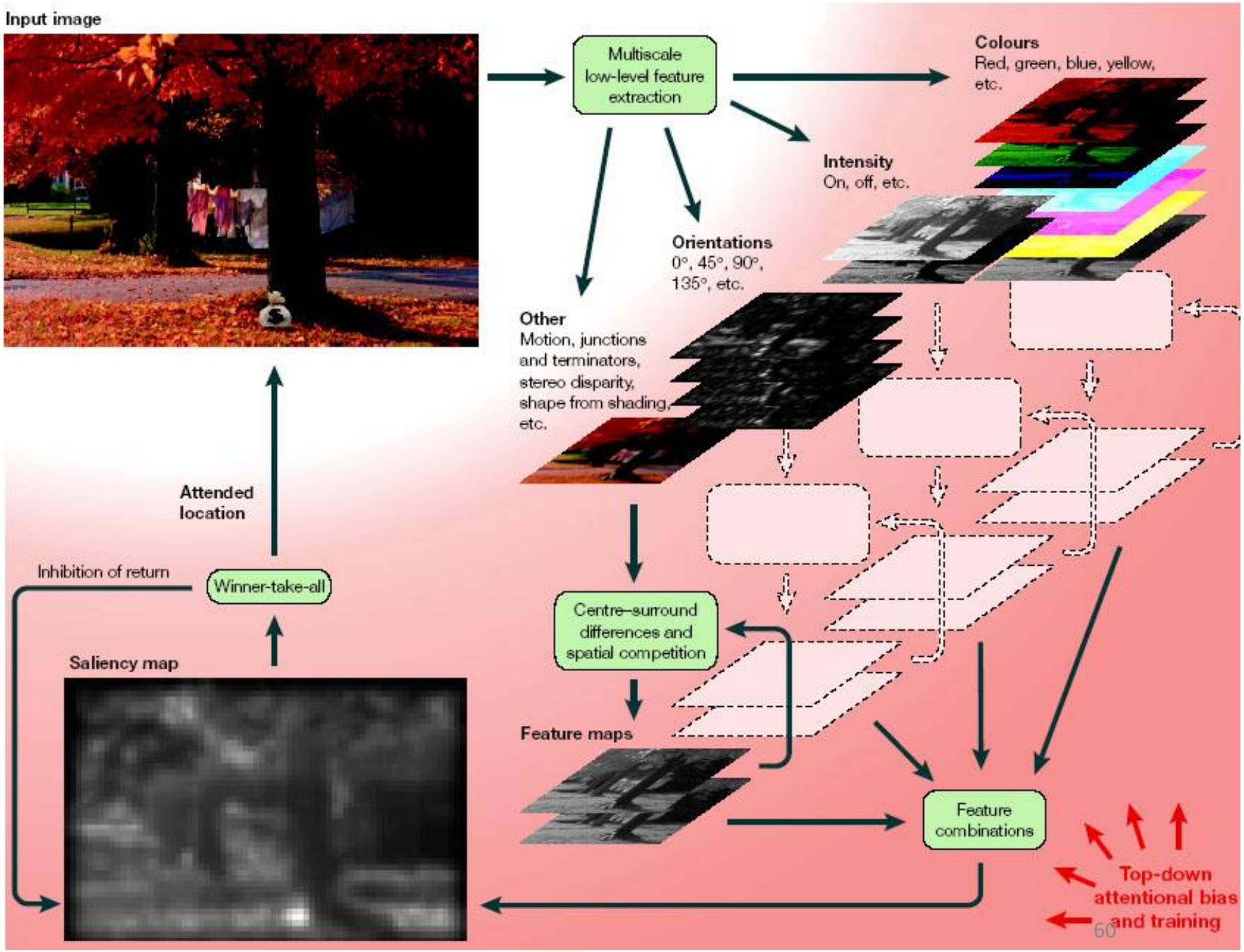
Average Maps

Inhibition + Excitation



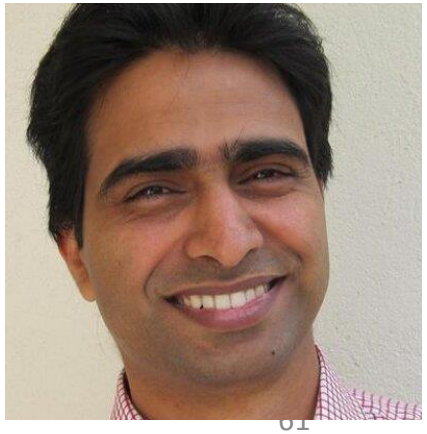
[Itti Koch Niebur 1998]

How it works

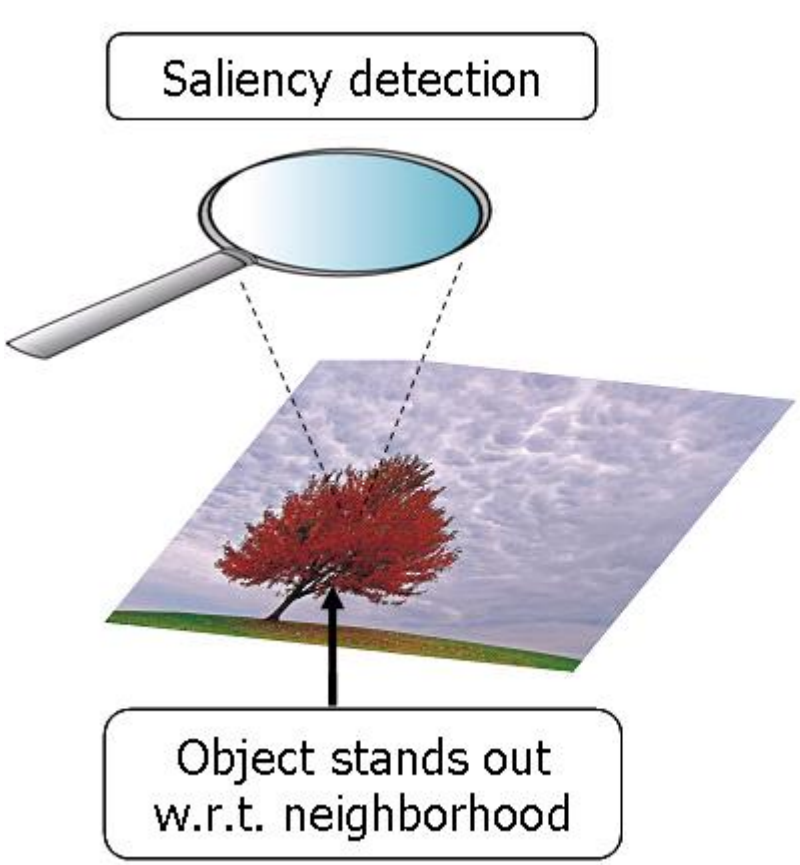

Input image
Multiscale low-level feature extraction
Colours
Red, green, blue, yellow, etc.
Intensity
On, off, etc.
Orientations
0°, 45°, 90°, 135°, etc.
Other
Motion, junctions and terminators, stereo disparity, shape from shading, etc.
Attended location
Inhibition of return
Winner-take-all
Saliency map
Centre–surround differences and spatial competition
Feature maps
Feature combinations
Top-down attentional bias and training

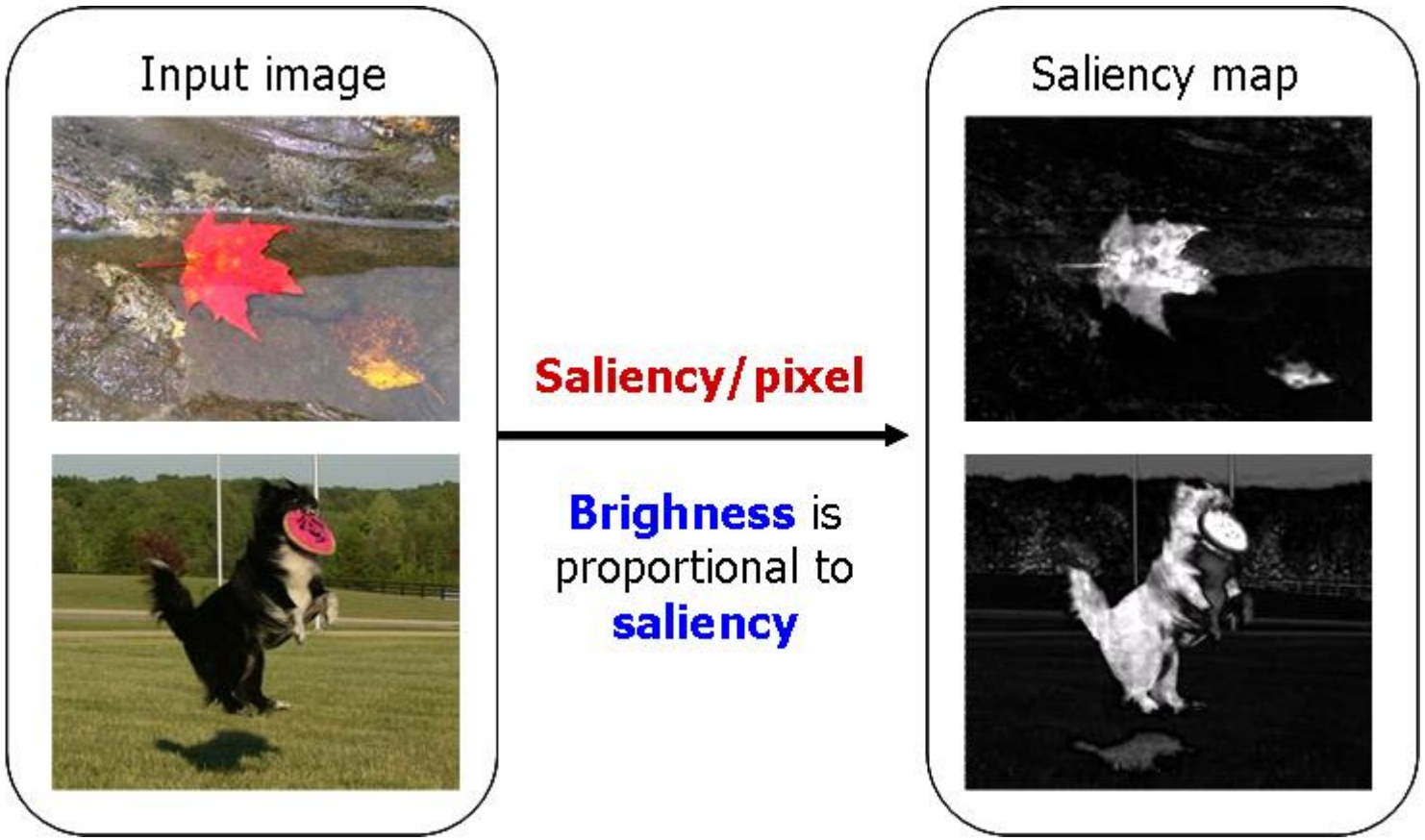
Frequency-tuned Salient Region Detection

Radhakrishna Achanta[†], Sheila Hemami[‡], Francisco Estrada[†], and Sabine Süsstrunk[†]
[†]School of Computer and Communication Sciences (IC)
Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015, Switzerland.
[radhakrishna.achanta,francisco.estrada,sabine.susstrunk]@epfl.ch
[‡]School of Electrical and Computer Engineering
Cornell University, Ithaca, NY 14853, U.S.A.
hemami@ece.cornell.edu

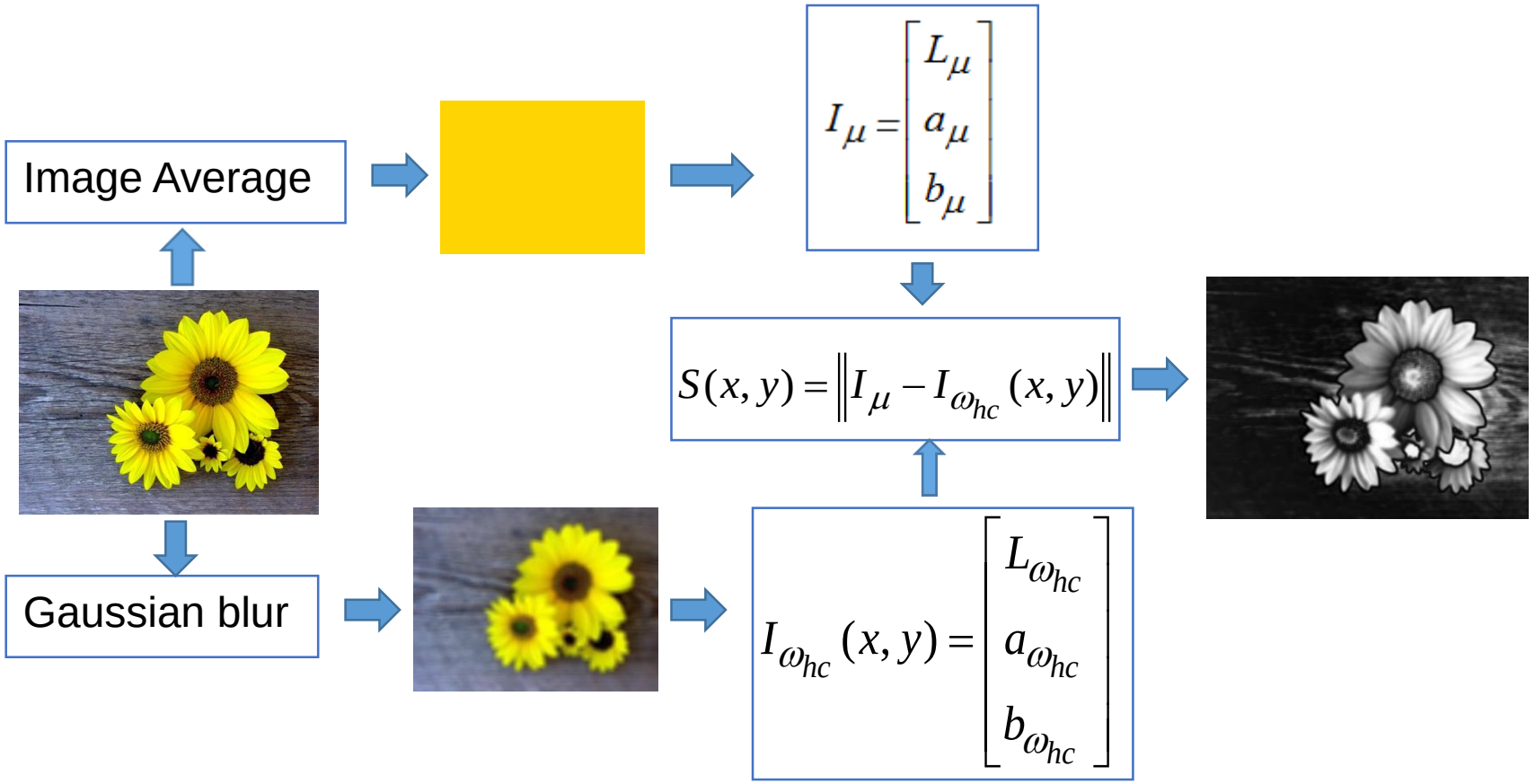


Idea


Saliency detection
Object stands out
w.r.t. neighborhood


Input image
Saliency/pixel
Brighness is
proportional to
saliency
Saliency map

Frequency-tuned



Example

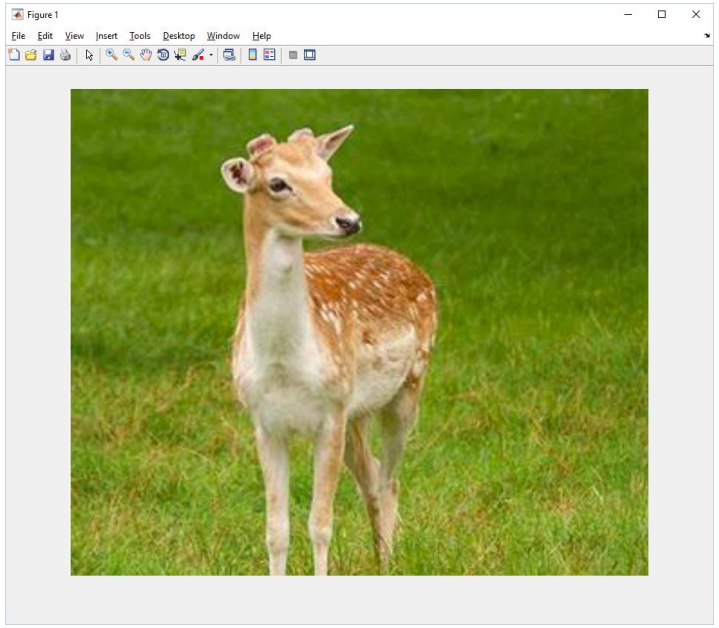


Figure 1
File Edit View Insert Tools Desktop Window Help

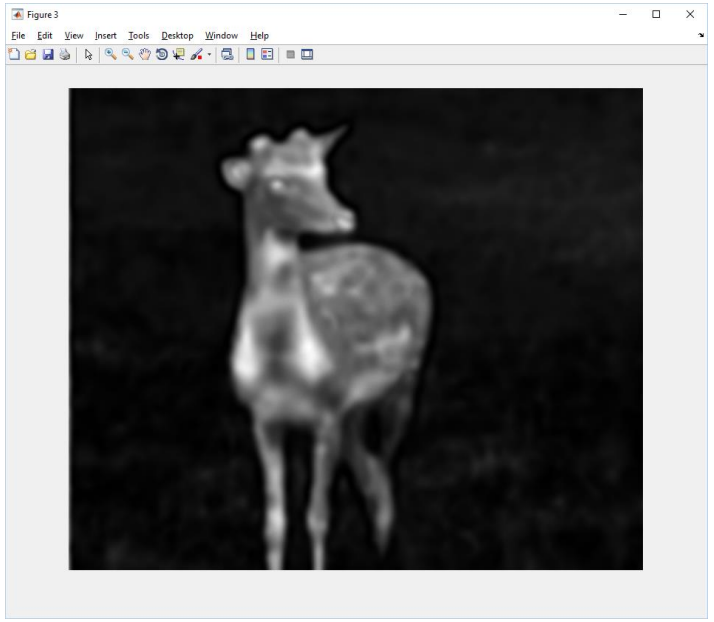


Figure 3
File Edit View Insert Tools Desktop Window Help

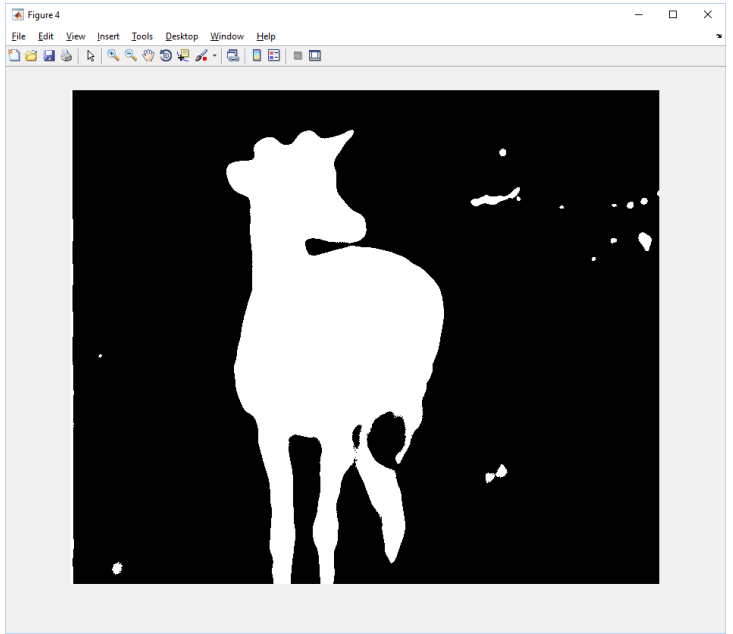


Figure 4
File Edit View Insert Tools Desktop Window Help

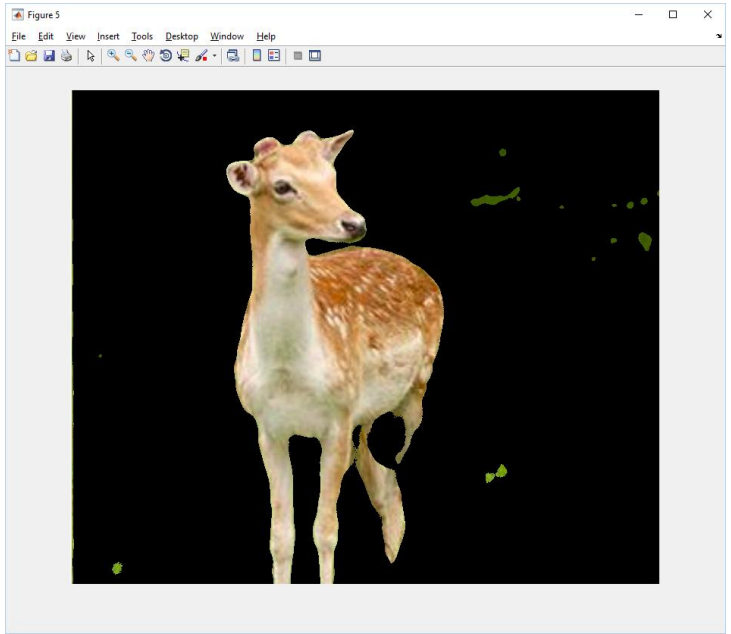


Figure 5
File Edit View Insert Tools Desktop Window Help

Context-Aware Saliency Detection

Stas Goferman
Technion
stasix@gmail.com

Lihi Zelnik-Manor
Technion
lihi@ee.technion.ac.il

Ayellet Tal
Technion
ayellet@ee.technion.ac.il

Frequency-tuned results

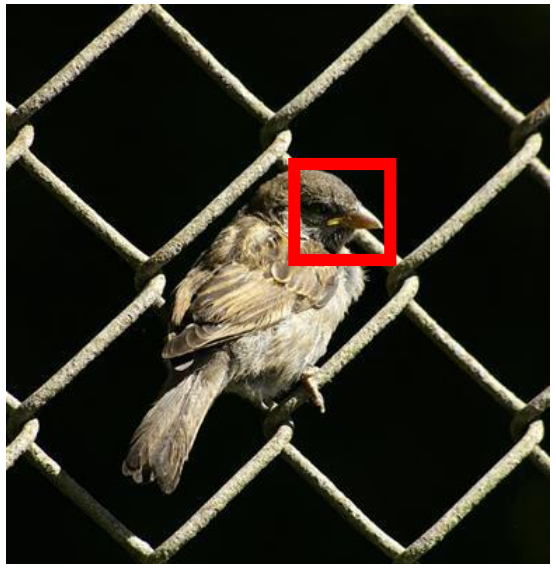
Input image

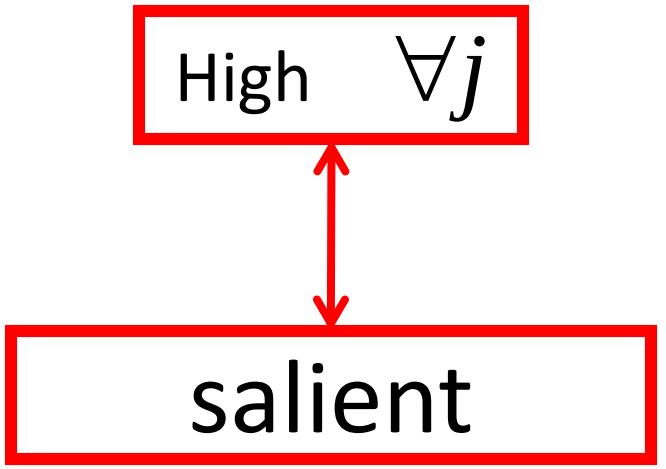
FT saliency map

Context-Aware

- Distance between a pair of patches:

$$d(p_i, p_j) = \frac{d_{color}(p_i, p_j)}{1 + c \cdot d_{position}(p_i, p_j)}$$





Q&A