Deep Learning for Computer Vision

Spring 2018

http://vllab.ee.ntu.edu.tw/dlcv.html (primary)

https://ceiba.ntu.edu.tw/1062DLCV (grade, etc.)

FB: DLCV Spring 2018

Yu-Chiang Frank Wang 王鈺強, Associate Professor

Dept. Electrical Engineering, National Taiwan University

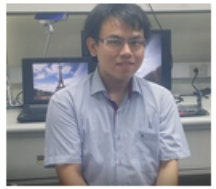
2018/03/21

About BONUS in HW1

- "Given a d x d symmetric matrix"
  → "Given a d x d positive semi-definite symmetric matrix"
- We still assume that this matrix has distinct eigenvalues.
- And, you need to explain why adding this statement matters.

- Deadline for BONUS submission (3/23 Friday 1:00am)
- No late day for the BONUS.

TA Hours

Mon	Tue	Wed	Thu	Fri
 Yu-Jhe Li 11:00-12:00	 Yan-Bo Lin 14:20-15:10	 Alex Liu 13:20-14:20	 Chi-Hsin Lo 10:00-11:00	 曾耕森 13:20-14:20
 Huai-Jin Peng 13:20-14:20	 Hsuan-I Ho 15:30-17:20	 楊耀程 14:30-15:30	 Cheng-Yen Yang 13:30-14:30	
 Jia-Wei Yan 15:10-16:10	 Frank Liao 18:30-19:30	 Jason Kuo 18:00-19:00	 Yu-Ying Yeh 14:30-15:30	
			 Boris Chen 15:30-16:30	

- FB: **DLCV Spring 2018**
- Feedback are welcome (to TA/me).

About Grading & Final Project

- HW assignments (60%)
  - HW 1 (5%), HW 2 (10%), HWs 3~5 (15% each)
- Final project + report + poster presentation (35%)
- Participation (5%)
  - Mostly 課堂/課後/facebook群組互動、Q&A等

- About Final Project
  - **3 (preferable) or 4 (max) people per group**; plz team up early!
  - Details (e.g., computing resource, topics, prize, etc.): TBA in April

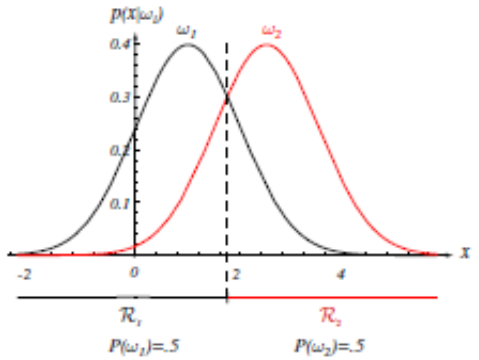
- Bonus Points (5%)
  - Turn your final project/report with sufficient quality into contests or major/top-tier conference submissions.

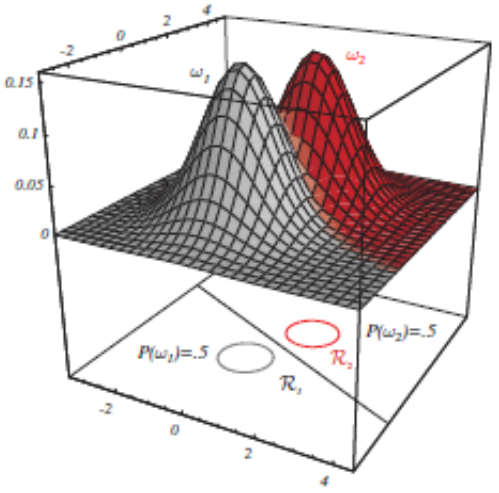
Tight (yet tentative) Schedule

Week	Date	Topic	Remarks
0	3/7	Course Logistics + Intro to Computer Vision	
1	3/14	Machine Learning 101	HW #1 out (due in 1 week)
2	3/21	**Image Representation**	**HW #1 due** **(Bonus due 3/23 Fri. 1am)**
3	3/28	Interest Point: From Recognition to Tracking	HW #2 out
4	4/4	N/A	Spring Break
5	4/11	Intro to Neural Networks + CNN	HW #2 due
6	4/18	Detection & Segmentation	HW #3 out
7	4/25	Generative Models	
8	5/2	Visualization and Understanding NNs	HW #3 due / HW #4 out
9	5/9	Recurrent NNs and Seq-to-Seq Models (I)	Project team
10	5/16	Recurrent NNs and Seq-to-Seq Models (II)	HW #4 due / HW #5 out
11	5/23	Deep Reinforcement Learning for Visual Apps	Project proposal
12	5/30	Learning Beyond Images (2D/3D, depth, etc.)	
13	6/6	Transfer Learning for Visual Analysis	HW #5 due
14	6/13	Vision & Language / Guest Lectures	Project midterm progress
15	6/20	Industrial Visit (e.g., Microsoft @ TPE)	CVPR week
16	6/27 (?)	Final Project Presentation	

What We Covered Last Week…

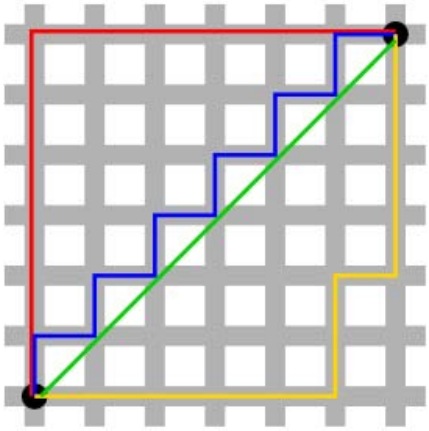
- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification





Revisit of Clustering

- Similarity:
  - A key component/measure to perform data clustering
  - Inversely proportional to distance
  - Example distance metrics:
    - Euclidean distance (L2 norm): $d(x, z) = \|x - z\|_2 = \sqrt{\sum_{i=1}^{D}(x_i - z_i)^2}$
    - Manhattan distance (L1 norm): $d(x, z) = \|x - z\|_1 = \sum_{i=1}^{D}|x_i - z_i|$



Revisit of Clustering (cont'd)

- Similarity:
  - A key component/measure to perform data clustering
  - Inversely proportional to distance
  - Example distance metrics:
    - Kernelized (non-linear) distance:

$$d(x,z) = \|\Phi(x) - \Phi(z)\|_2^2 = \|\Phi(x)\|_2^2 + \|\Phi(z)\|_2^2 - 2\Phi(x)^T\Phi(z)$$

    - No need to know explicit Φ, just the way to calculate $\Phi(x)^T\Phi(z)$.
    - E.g., Gaussian kernel: $K(x,z) = \Phi(x)^T\Phi(z) = exp\left(-\frac{\|x-z\|_2^2}{2\sigma^2}\right)$, we have

$$0 < \Phi(x)^T\Phi(z) < 1$$

    - Distance is more sensitive to larger/smaller σ?
    - E.g., Given the same data pair x and z, $\Phi(x)^T\Phi(z) \to 0$ if a very large/small σ.

K-Means Clustering

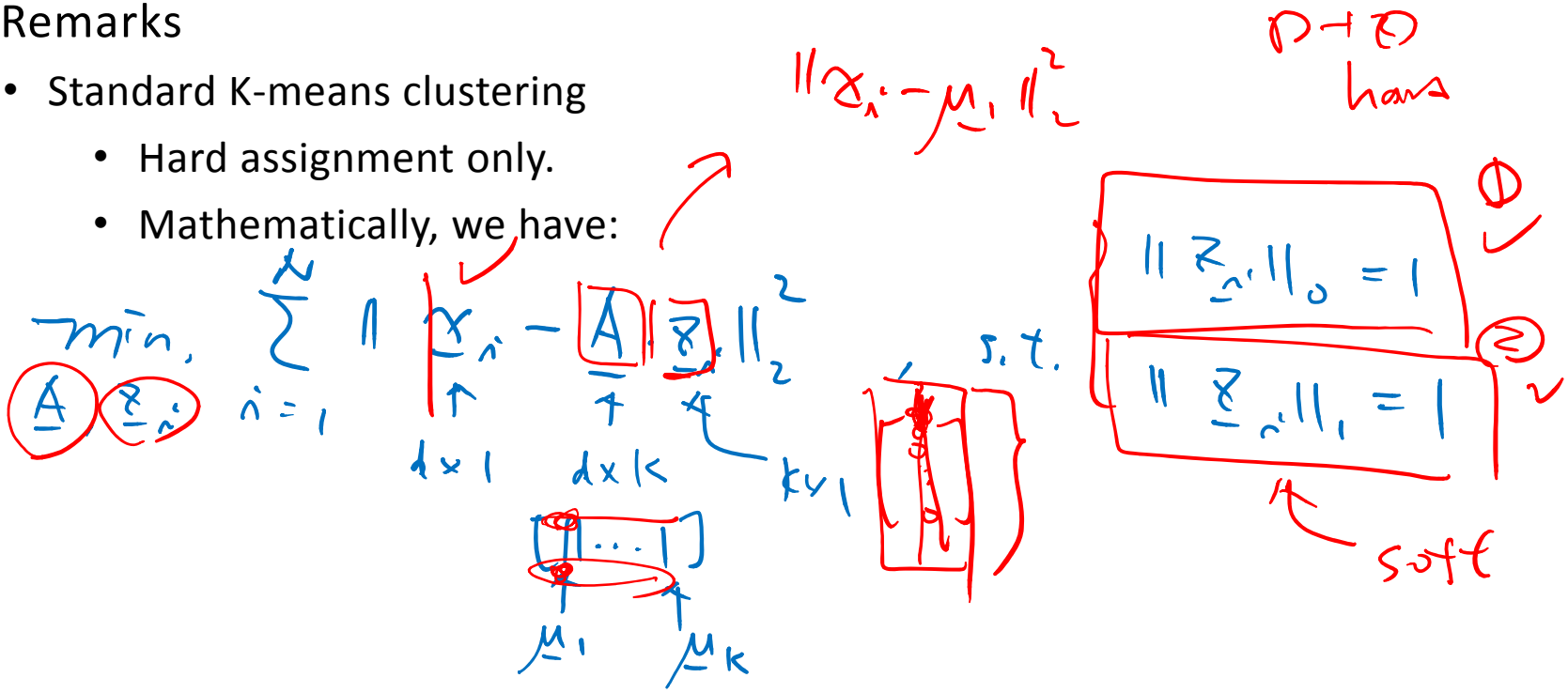
- Input: N examples $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ ($\boldsymbol{x}_n \in \mathrm{R}^D$); number of partitions K
- Initialize: K cluster centers $\mu_1, \ldots, \mu_K$. Several initialization options:
  - Randomly initialize $\mu_1, \ldots, \mu_K$ anywhere in R^D
  - Or, simply choose any K examples as the cluster centers
- Iterate:
  - Assign each of example $\boldsymbol{x}_n$ to its closest cluster center
  - Recompute the new cluster centers μ_k (mean/centroid of the set C_k)
  - Repeat while not converge
- Possible convergence criteria:
  - Cluster centers do not change anymore
  - Max. number of iterations reached
- Output:
  - K clusters (with centers/means of each cluster)

K-Means Clustering (cont'd)

- Proof in 1-D case (if time permits).

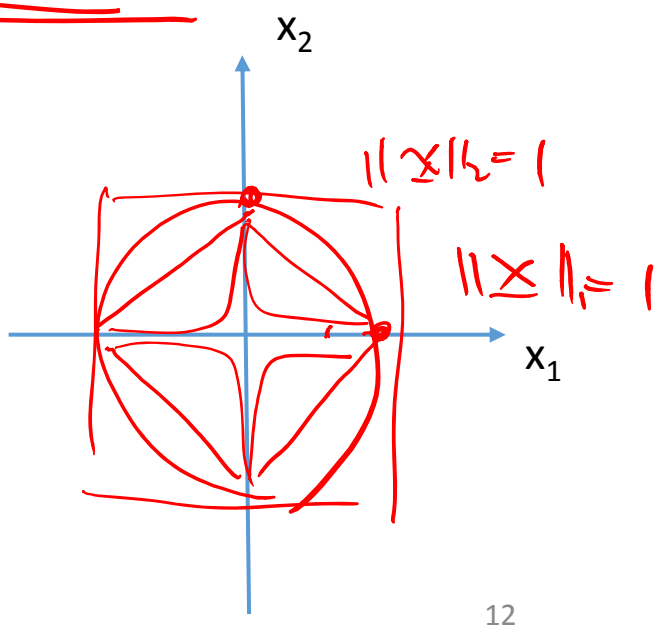
K-Means Clustering (cont'd)

- Remarks
  - Standard K-means clustering
    - Hard assignment only.
    - Mathematically, we have:



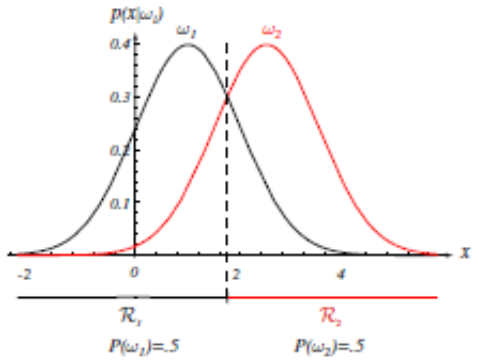
K-Means Clustering (cont'd)

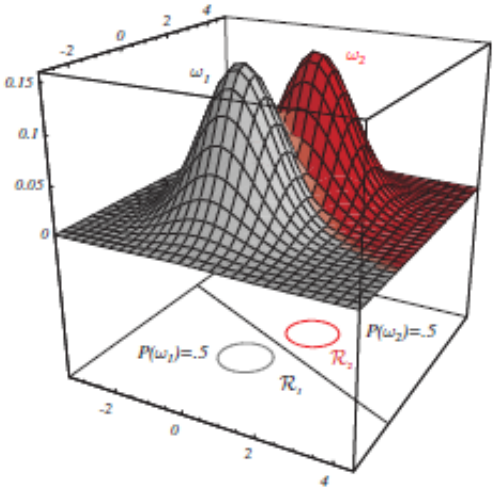
- Remarks
  - Recall that, we have $\|x\|_p = \left(\sum_{i=1}^{d} x_i^{\,p}\right)^{1/p}$
    - L_2/Euclidean norm $\|x\|_2$
    - L_1 norm $\|x\|_1 = \sum_{j=1}^{d} |x_j|$
    - Maximum norm $\|x\|_\infty = \max\{|x_1|, \ldots, |x_d|\}$
    - L_0 norm: $\|x\|_0$



What We Covered Last Week…

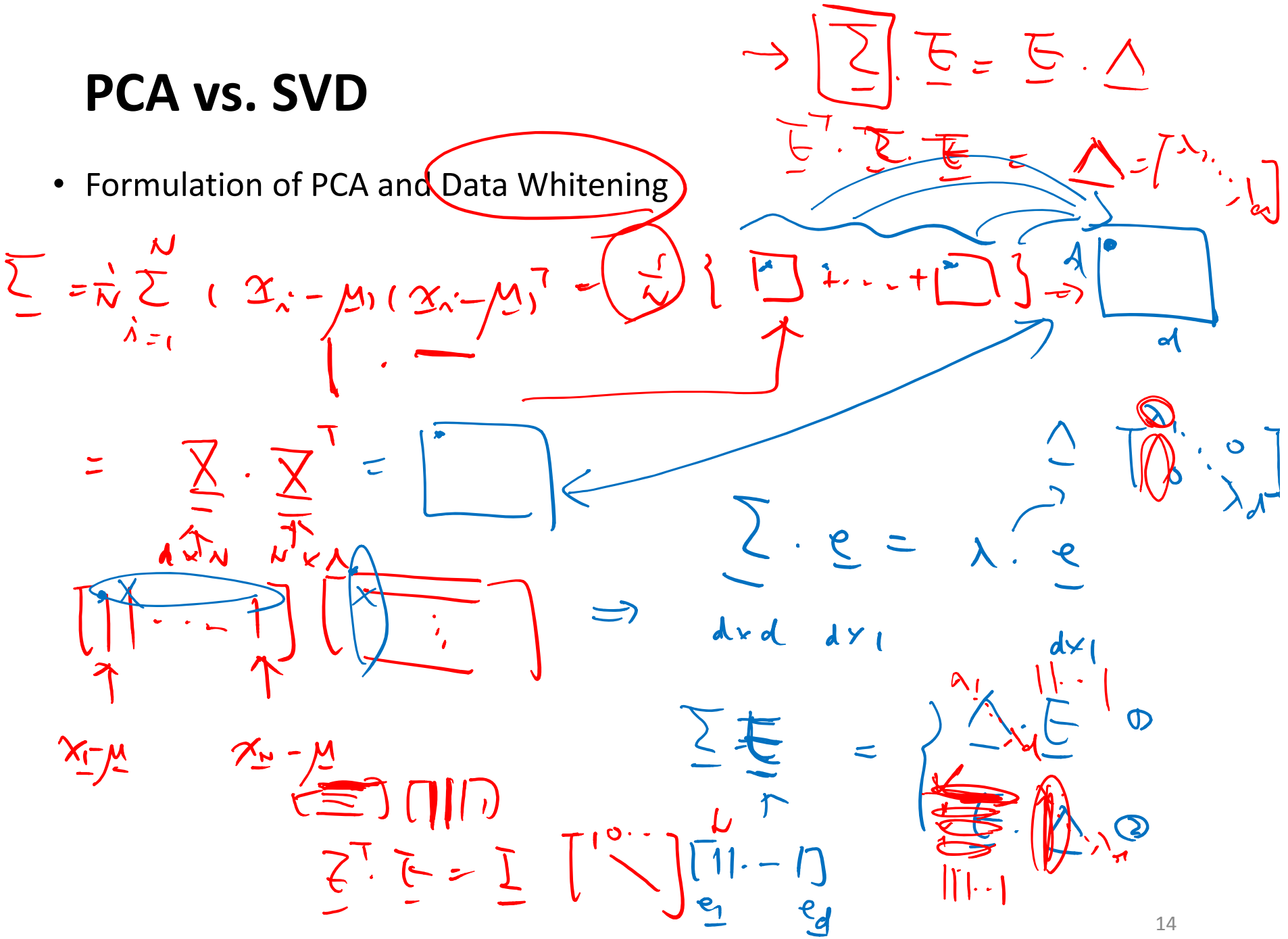
- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification





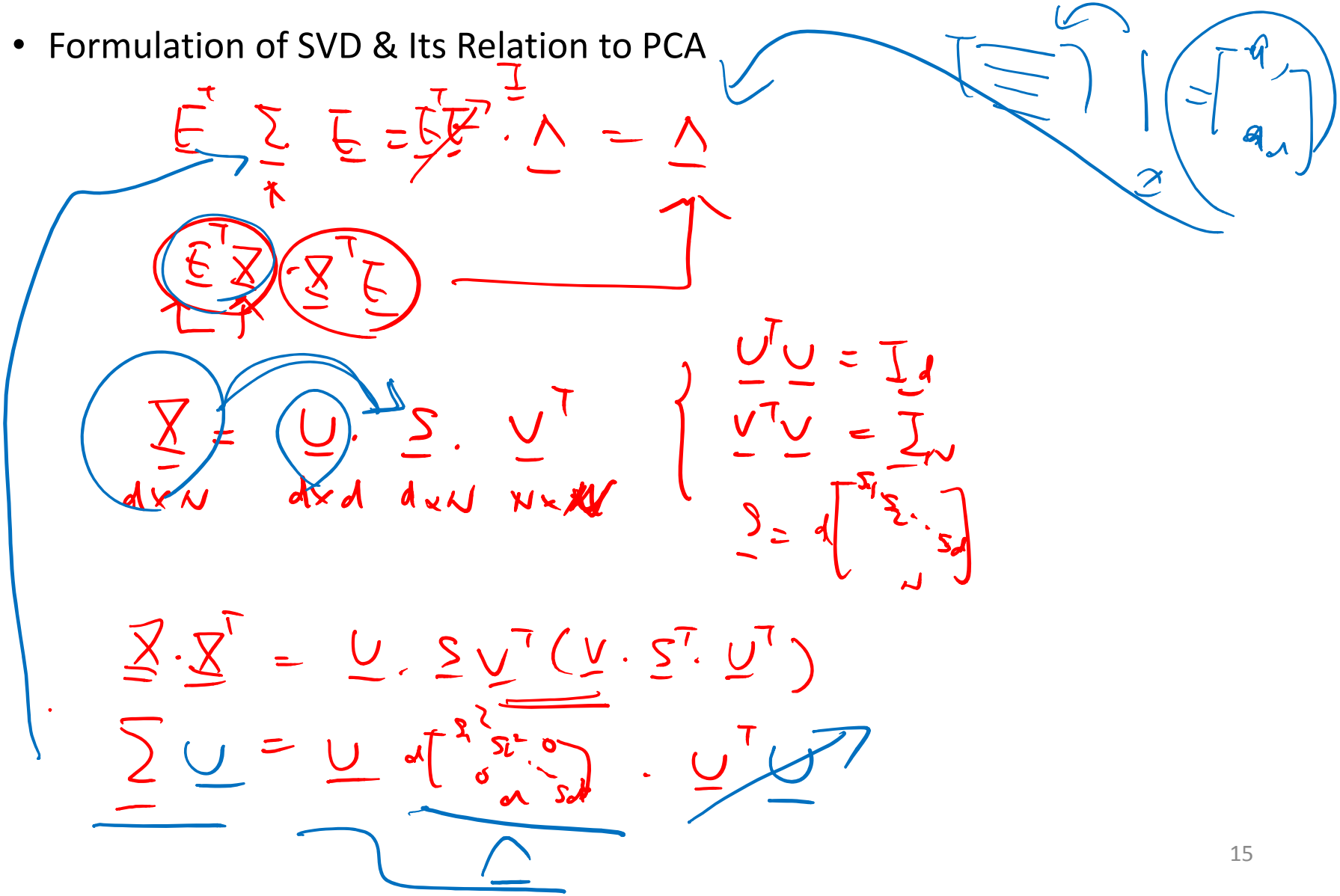
PCA vs. SVD

- Formulation of PCA and Data Whitening



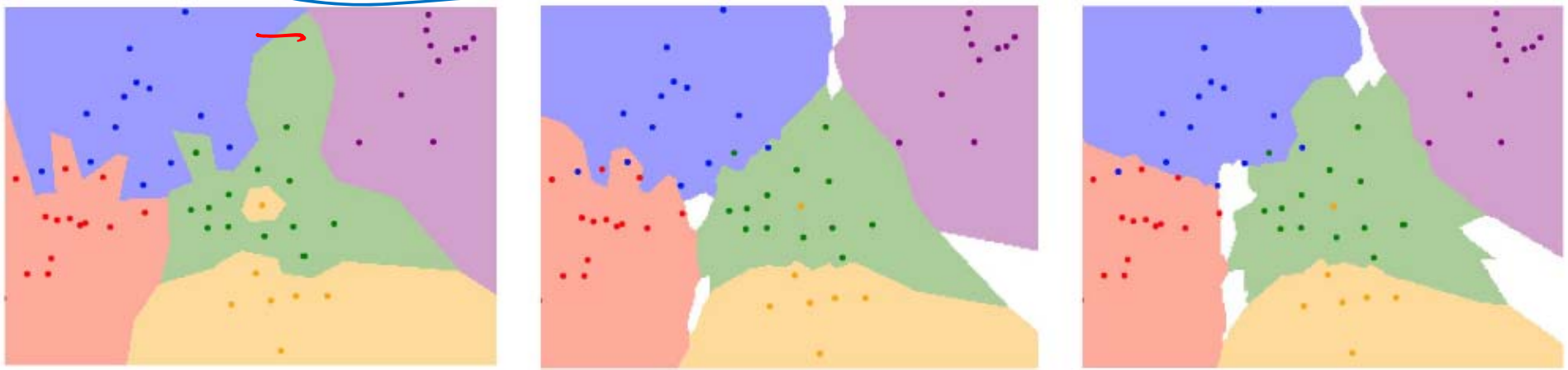
PCA vs. SVD (cont'd)

- Formulation of SVD & Its Relation to PCA



Remarks on Nearest-Neighbor Classification

- How do we classify? For example…
  - Given a test face input, project into the same 3D space (by the same 3 eigenvectors).
  - The resulting vector in the 3D space is the feature for this test input.
  - We can do a simple Nearest Neighbor (NN) classification with Euclidean distance, which calculates the distance to all the projected training data in this space.
  - If NN, then the label of the closest training instance determines the classification output.
  - If k-nearest neighbors (k-NN), then k-nearest neighbors need to vote for the decision.



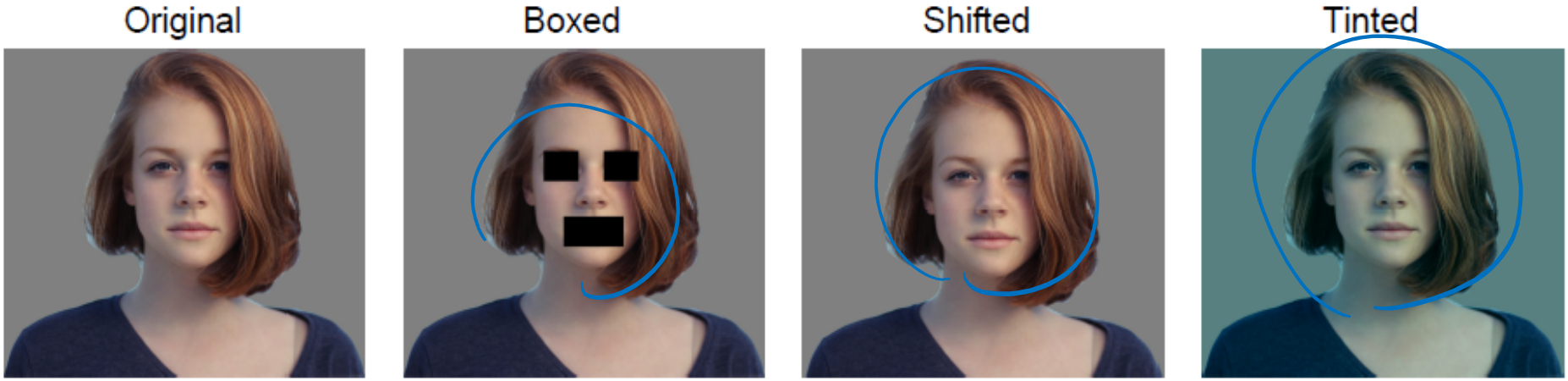
k = 1 k = 3 k = 5

Demo available at http://vision.stanford.edu/teaching/cs231n-demos/knn/

Image credit: Stanford CS231n

Minor Remarks on NN-based Methods

- In fact, k-NN (or even NN) is not of much interest in practice. Why?
  - Choice of distance metrics might be an issue. See example below.
  - Measuring distances in high-dimensional spaces might not be a good idea.
  - Moreover, NN-based methods require lots of data and storage !
    (That is why NN-based methods are viewed as *data-driven* approaches.)

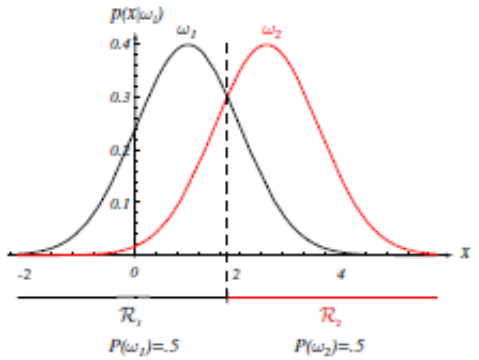


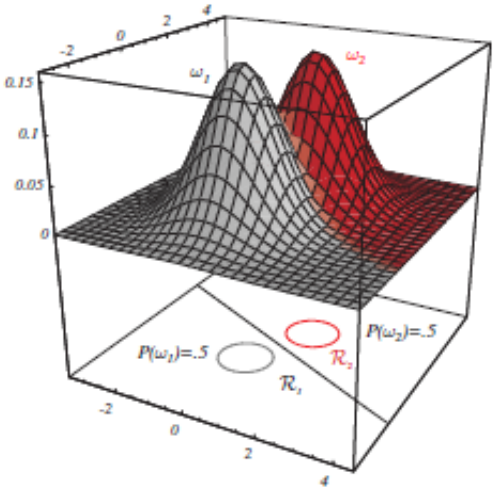
All three images have the same Euclidean distance to the original one.

Image credit: Stanford CS231n

What We Covered Last Week…

- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification





Linear Classification

- Linear Classifier
  - Can be viewed as a parametric approach, as discussed later.
  - Assuming that we need to recognize 10 object categories of interest
  - E.g., CIFAR10 with 50K training & 10K test images of 10 categories.
    And, each image is of size 32 x 32 x 3 pixels.

Linear Classification (cont'd)

- Linear Classifier
  - Can be viewed as a parametric approach. Why?
  - Assuming that we need to recognize 10 object categories of interest (e.g., CIFAR10).
  - Let's take the input image as **x**, and the linear classifier as **W**. We hope to see that **y** = **Wx** + **b** as a 10-dimensional output indicating the score for each class.

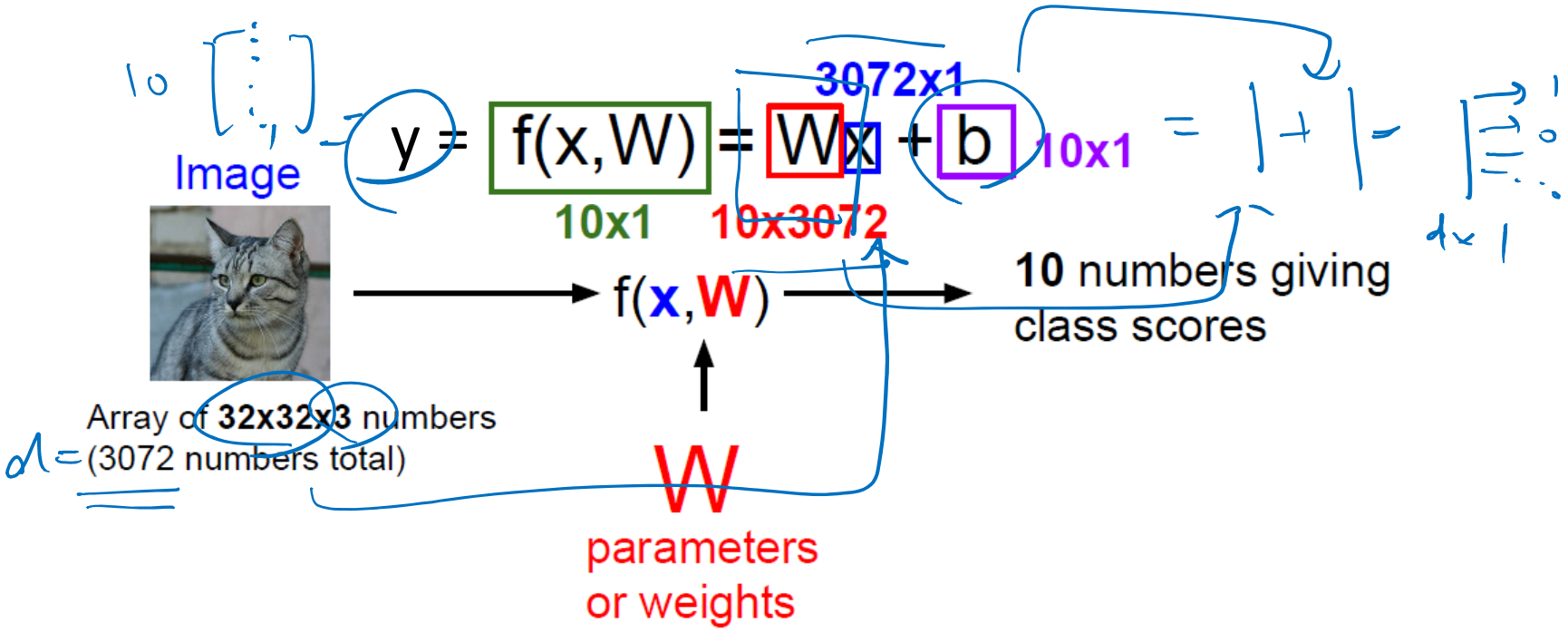


Image credit: Stanford CS231n

Linear Classification (cont'd)

- Linear Classifier
  - Can be viewed as a parametric approach. Why?
  - Assuming that we need to recognize 10 object categories of interest (e.g., CIFAR10).
  - Let's take the input image as **x**, and the linear classifier as **W**. We hope to see that **y** = **Wx** + **b** as a 10-dimensional output indicating the score for each class.
  - Take an image with 2 x 2 pixels & 3 classes of interest as example:
    we need to learn linear transformation/classifier **W** and bias **b**,
    so that desirable outputs **y** = **Wx** + **b** can be expected.

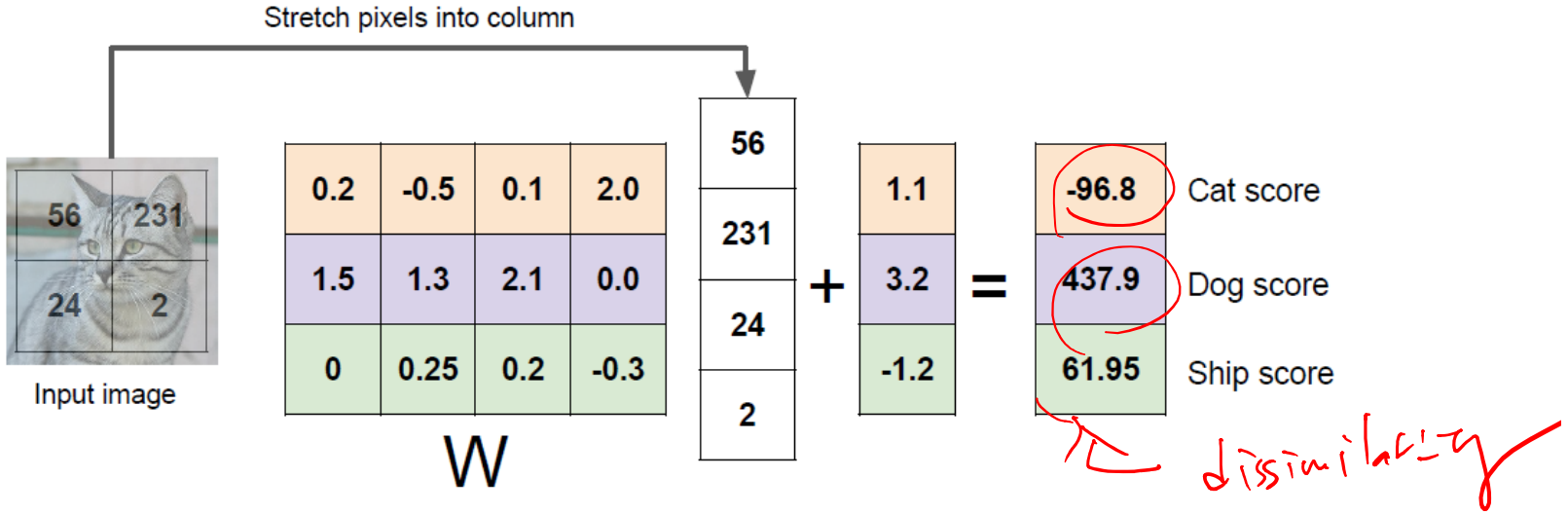
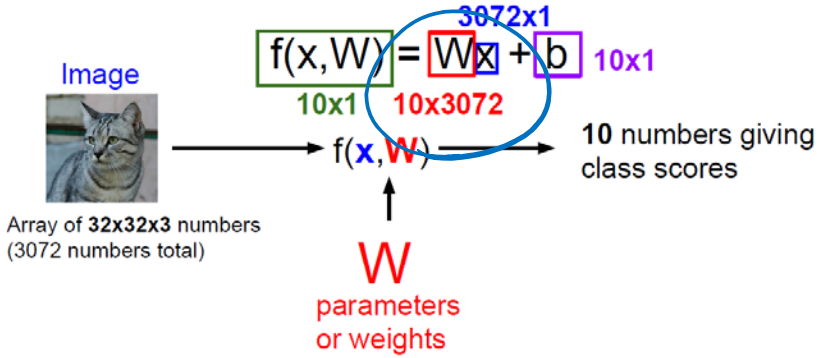
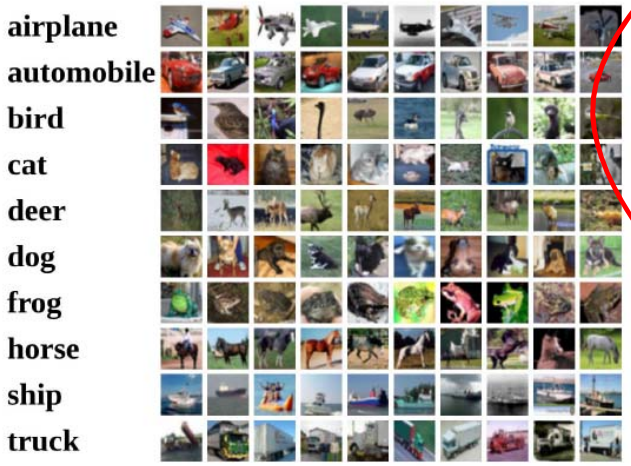


Image credit: Stanford CS231n

Some Remarks



- Interpreting **y** = **Wx** + **b**
  - What can we say about the learned **W**?
  - The weights in **W** are trained by observing training data **X** and their ground truth **Y**.
  - Each column in **W** can be viewed as an exemplar of the corresponding class.
  - Thus, **Wx** basically performs inner product (or correlation) between the input **x** and the exemplar of each class. (Signal & Systems!)



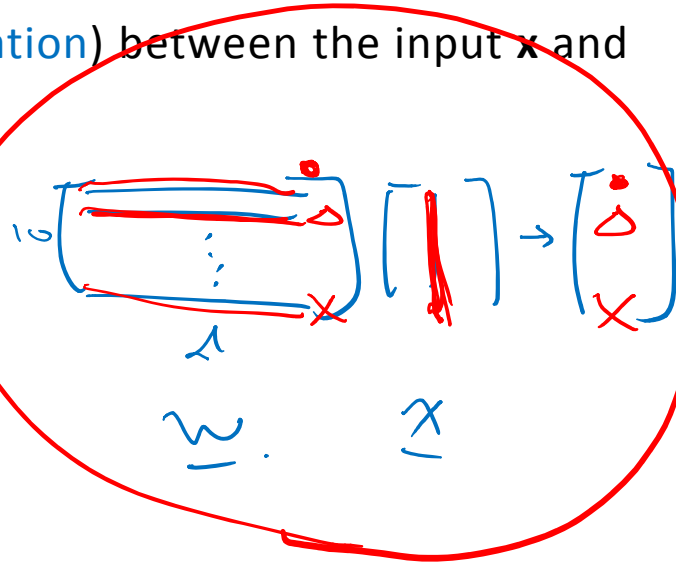
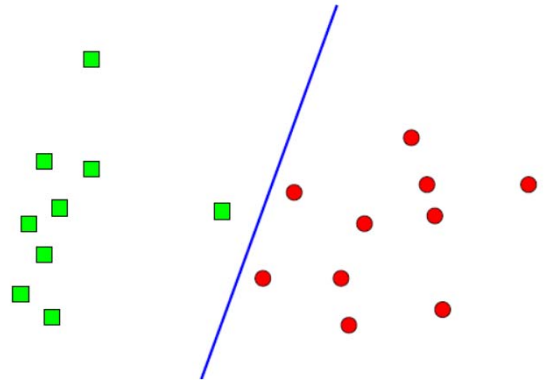


Image credit: Stanford CS231n

From Binary to Multi-Class Classification
(if time permits)



e.g., Support Vector Machine (SVM)

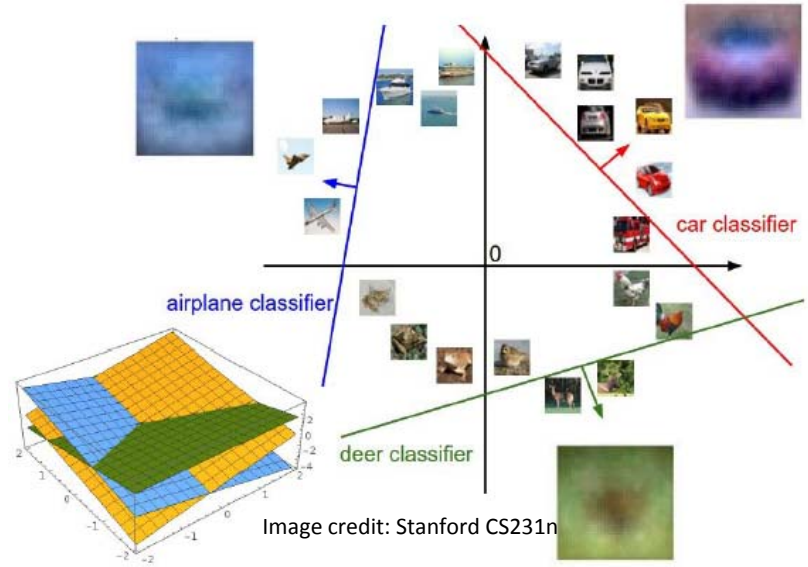
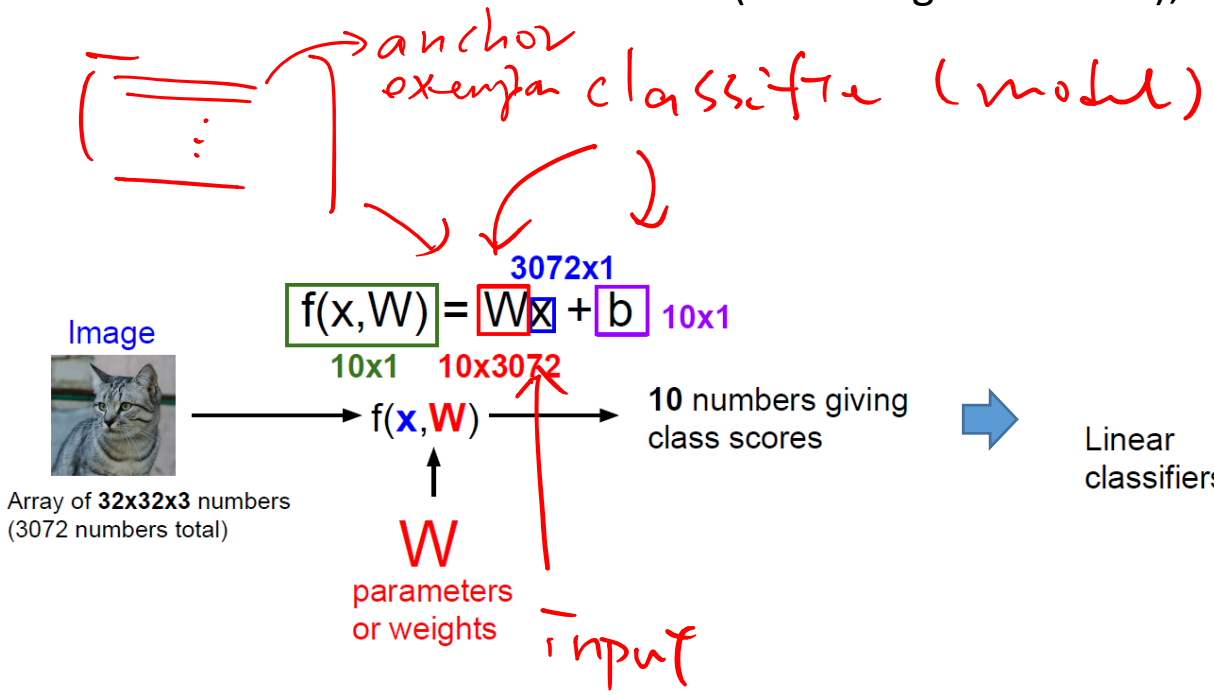


Image credit: Stanford CS231n

- How to extend binary classifiers for solving multi-class classification?
  - 1-vs.-1 (1-against-1) vs. 1-vs.-all (1-against-rest)

Summary for Linear Classification

- Remarks
  - Starting points for many multi-class or complex/nonlinear classifier
  - How to determine a proper loss function for matching **y** and **Wx**+**b**, and thus how to learn the model **W** (including the bias **b**), are the keys!



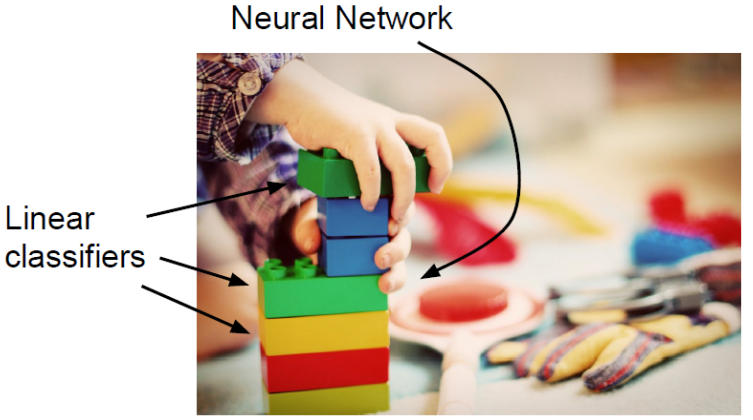
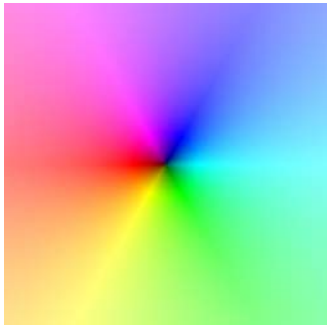
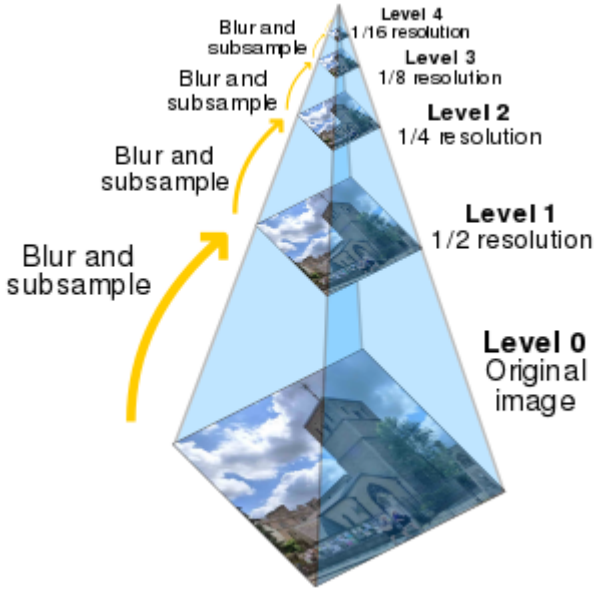


Image credit: Stanford CS231n

What's to Be Covered Today…

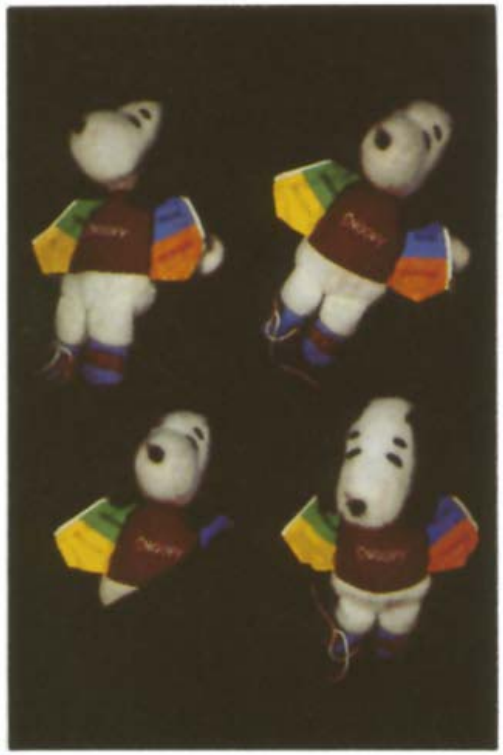
- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid

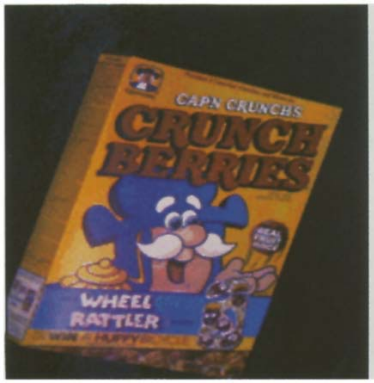


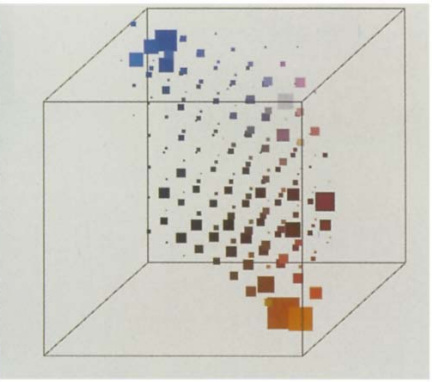


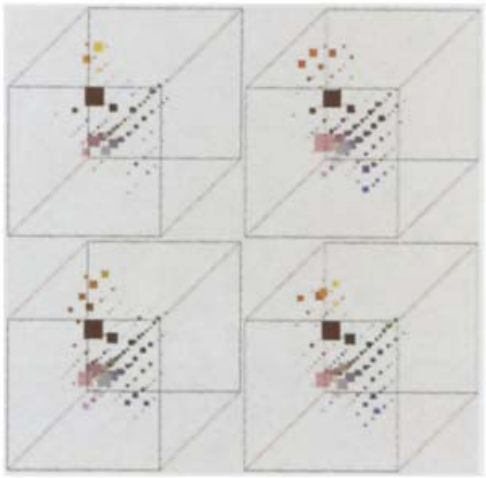
Uses of Color in Computer Vision

- Color histograms for indexing and retrieval





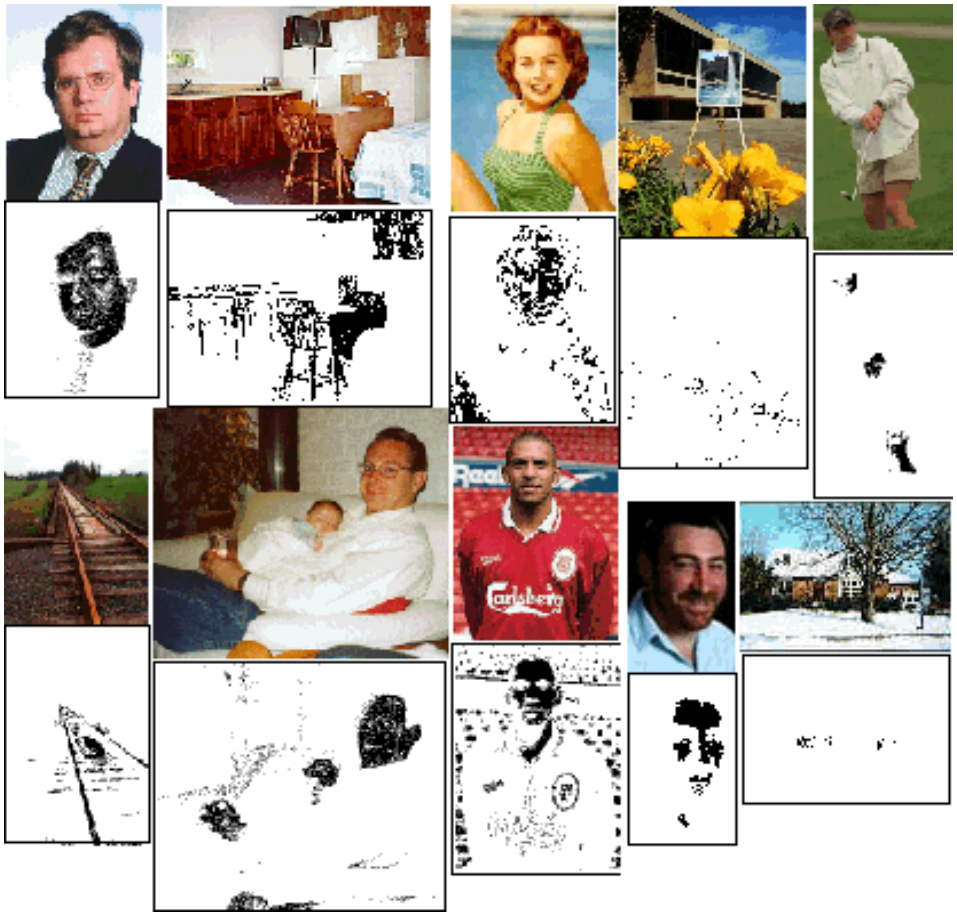




Swain and Ballard, Color Indexing, IJCV 1991

Uses of Color in Computer Vision

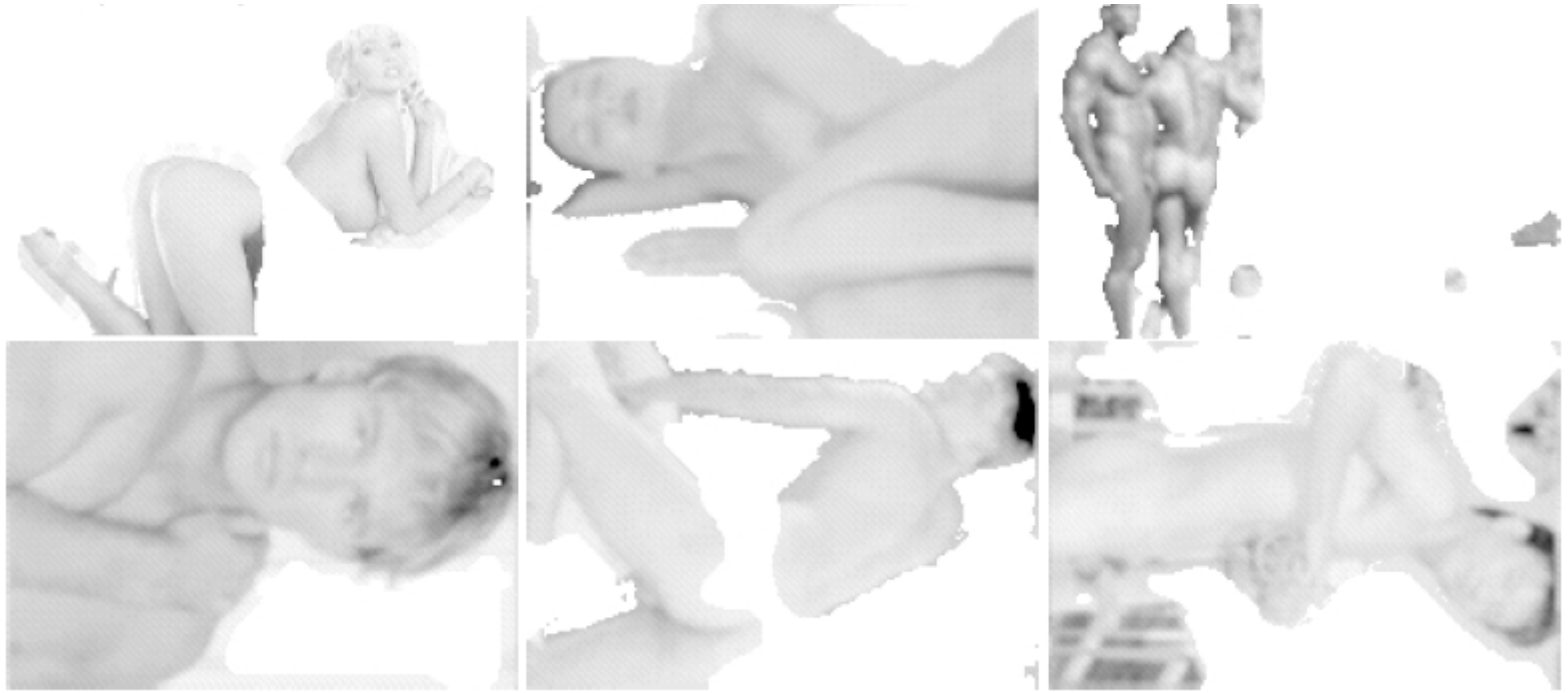
- Skin Detection



M. Jones and J. Rehg, Statistical Color Models with Application to Skin Detection, IJCV 2002.

Uses of Color in Computer Vision

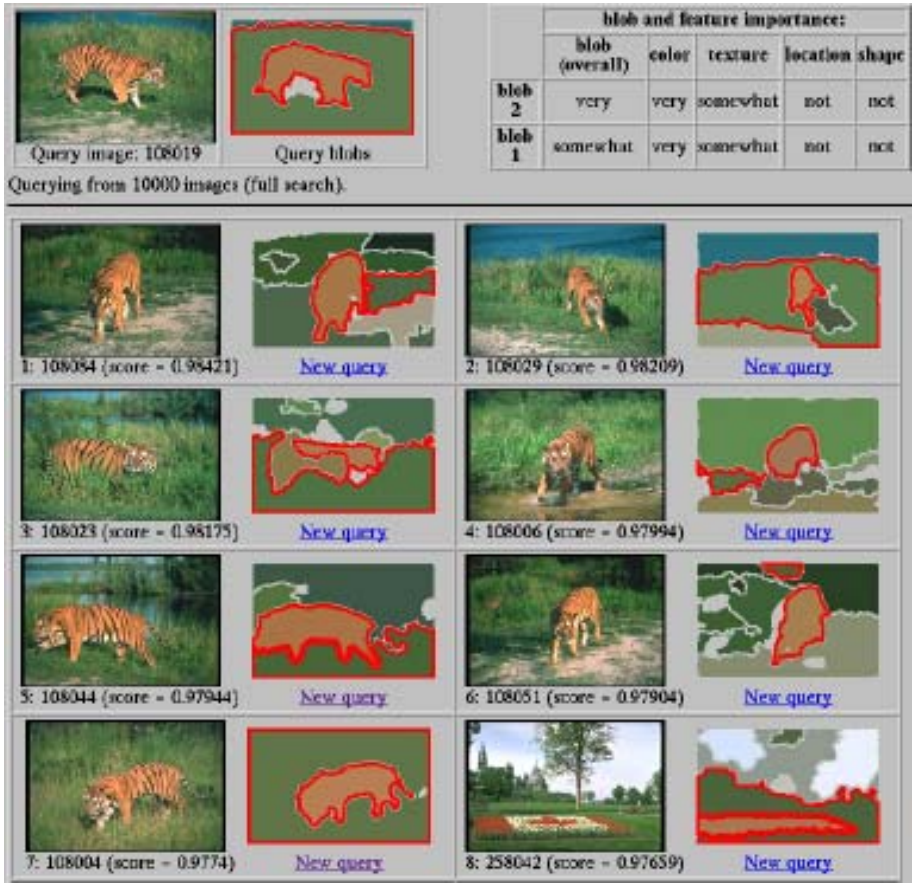
- Nudity detection



Forsyth and Fleck, ``Automatic Detection of Human Nudes,'' I*JCV*, 1999

Uses of Color in Computer Vision

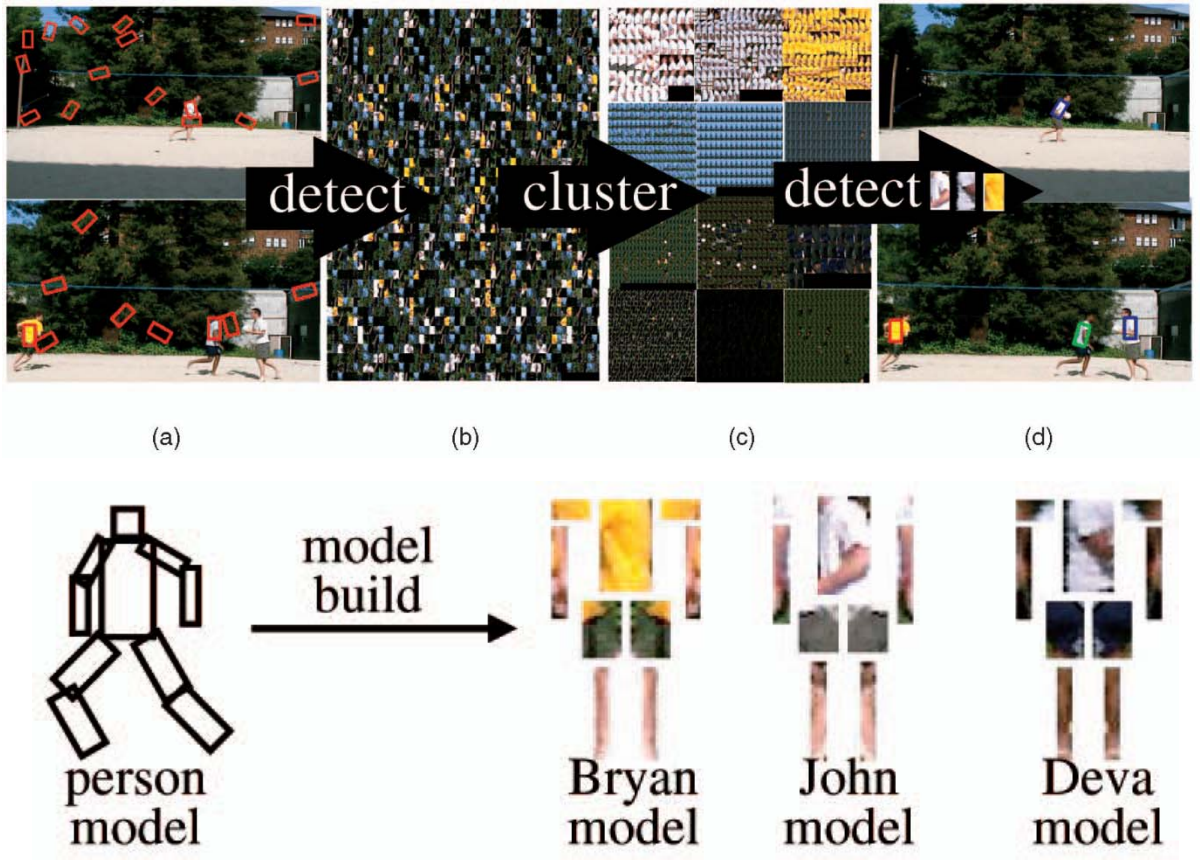
- Image Segmentation (for Detection, Retrieval, etc.)



C. Carson, S. Belongie, H. Greenspan, and Ji. Malik, Blobworld: Image segmentation using Expectation-Maximization and its application to image querying, ICVIS 1999.

Uses of Color in Computer Vision

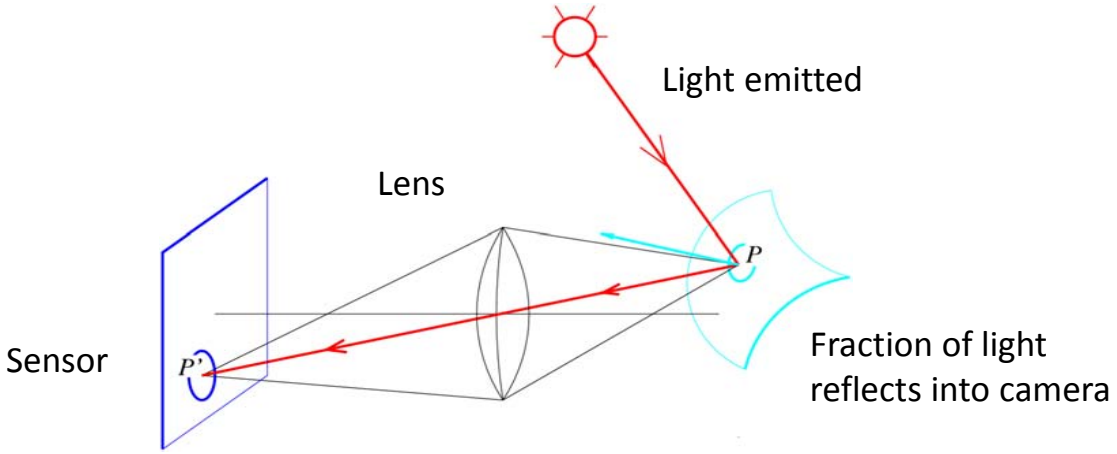
- Building Appearance Models for Tracking, etc.



D. Ramanan, D. Forsyth, and A. Zisserman. Tracking People by Learning their Appearance. PAMI 2007.

Image Representation

- Note that, since we focus on learning-based vision, we will not cover geometry, projection, camera models, etc. in this course.



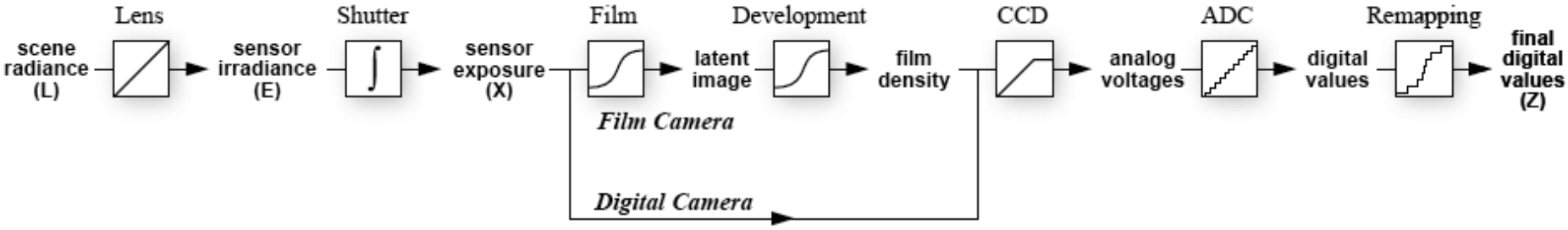
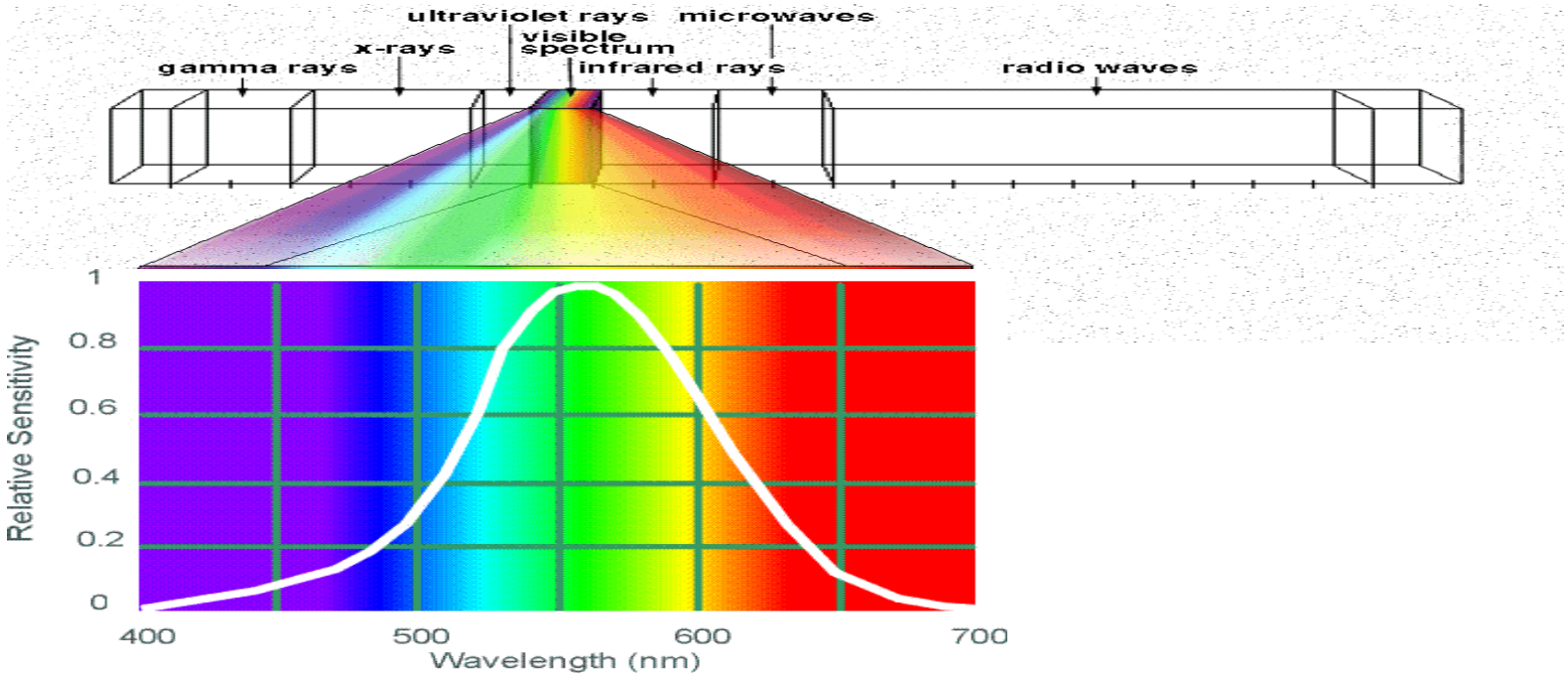


Image credit: Rob Fergus

The Physics of Light

- Light
  - Electromagnetic energy whose wavelength is between 400 nm and 700 nm. (1 nm = 10^{-9} meter)

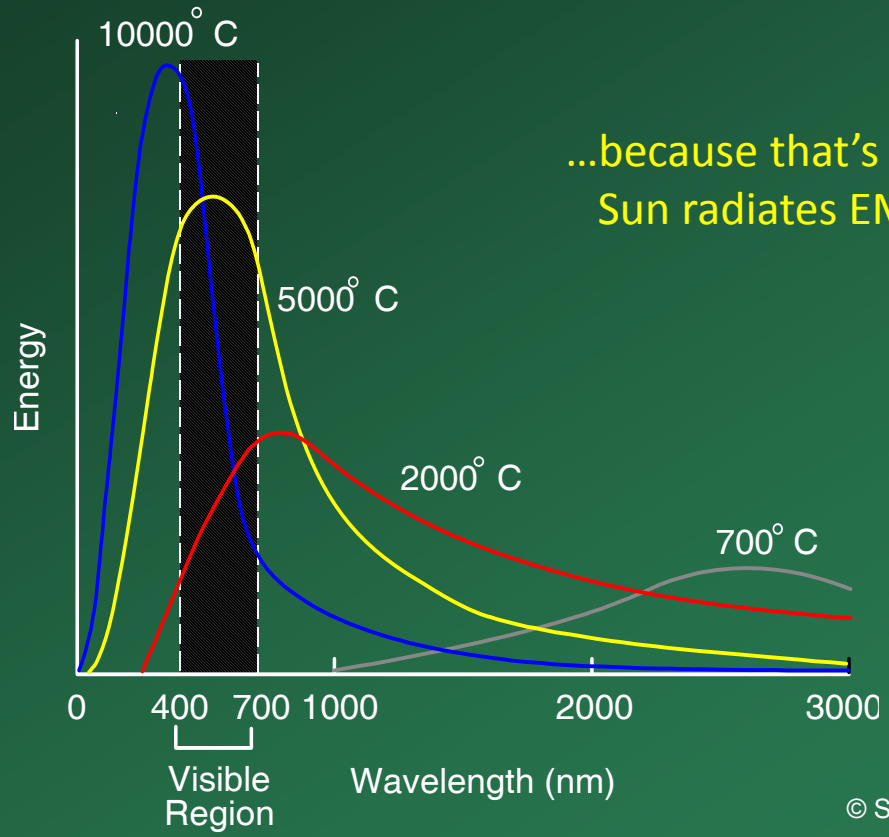


Human Luminance Sensitivity Function

Slide credit: Efros

Visible Light

Why do we see light of these wavelengths?

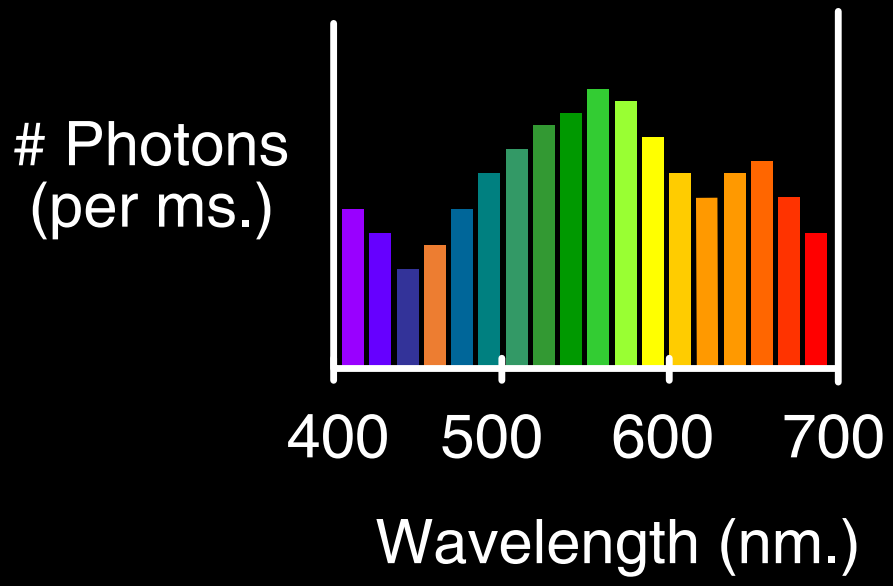


…because that's where the Sun radiates EM energy

© Stephen E. Palmer, 2002

The Physics of Light

Any patch of light can be completely described physically by its spectrum: the number of photons (per time unit) at each wavelength 400 - 700 nm.

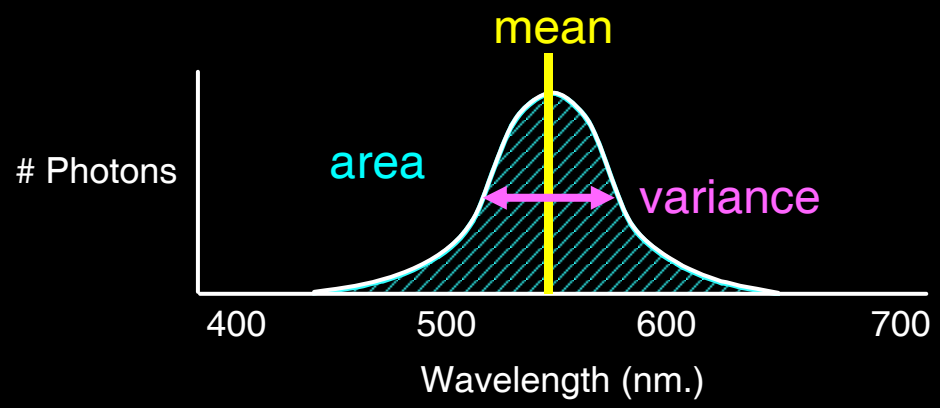


© Stephen E. Palmer, 2002

The Psychophysical Correspondence

There is no simple functional description for the perceived color of all lights under all viewing conditions, but ……

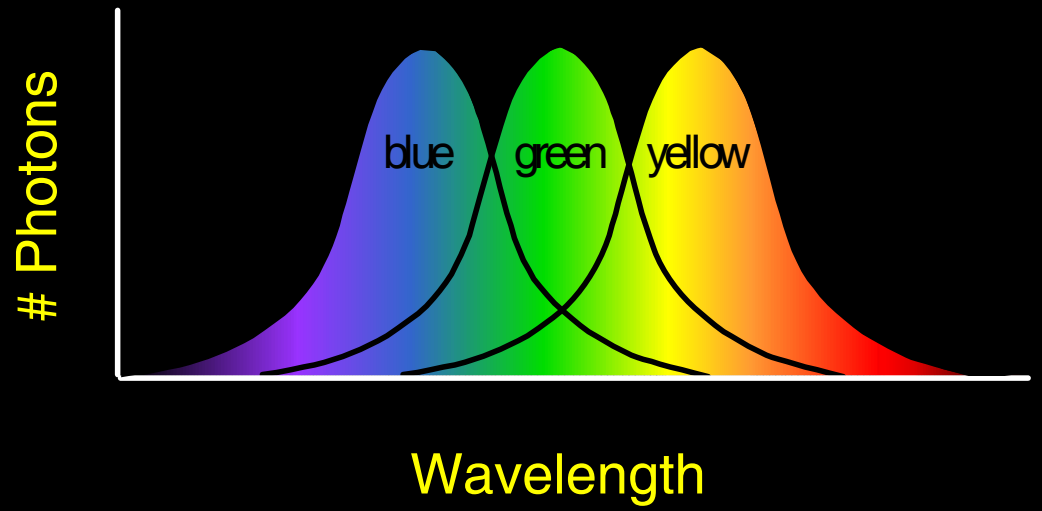
A helpful constraint:
Consider only physical spectra with normal distributions



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

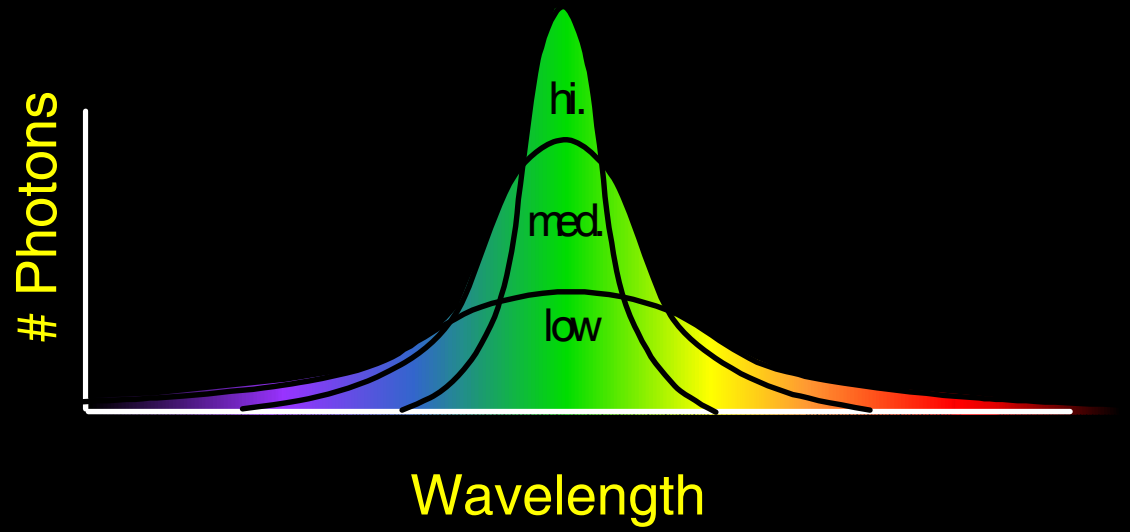
Mean ⟷ Hue



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

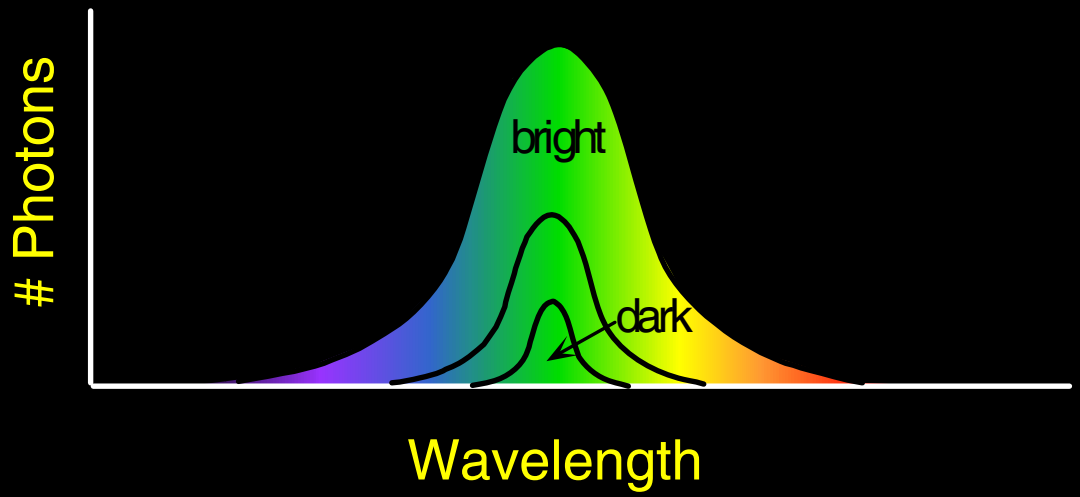
Variance ⟷ Saturation



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

Area ⟷ Brightness

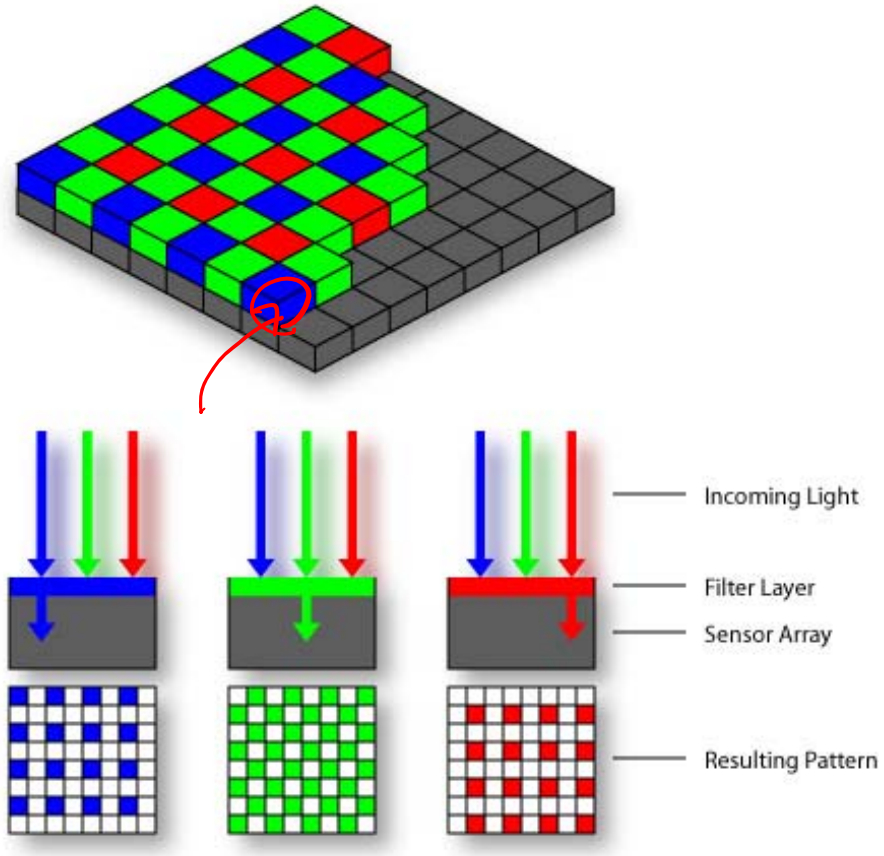


© Stephen E. Palmer, 2002

Practical Color Sensing

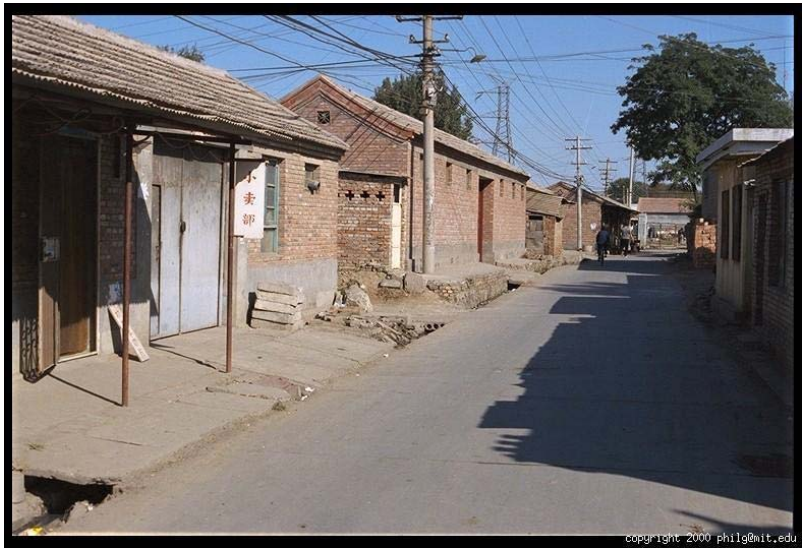
- Bayer Grid/Pattern

- Estimate RGB at 'G' cells from neighboring values

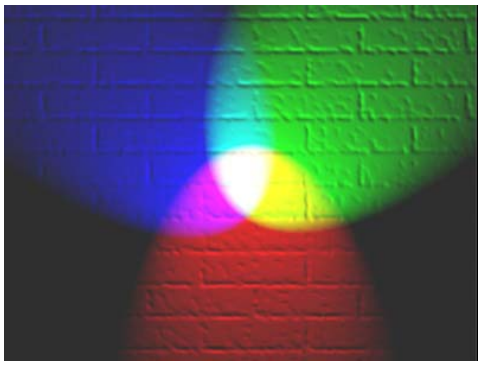


Slide credit: Steve Seitz

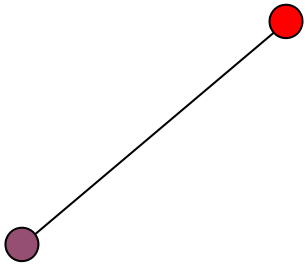
Example Color Image



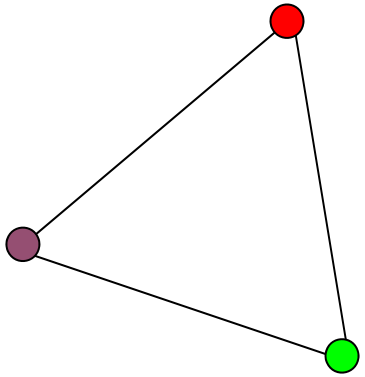
Color Space



- How can we represent color?
- Linear color spaces
  - Defined by a choice of three primaries
  - The coordinates of a color are given by the weights of each primary.



Mixing two lights produces colors that lie along a straight line in color space.

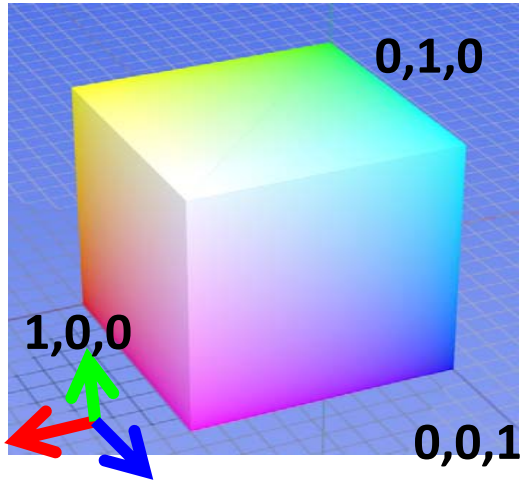


Mixing three lights produces colors that lie within the triangle defined in the color space.

Slide credit: Rob Fergus

Example Color Space: RGB

- Default Color Space



R
(G=0,B=0)

G
(R=0,B=0)

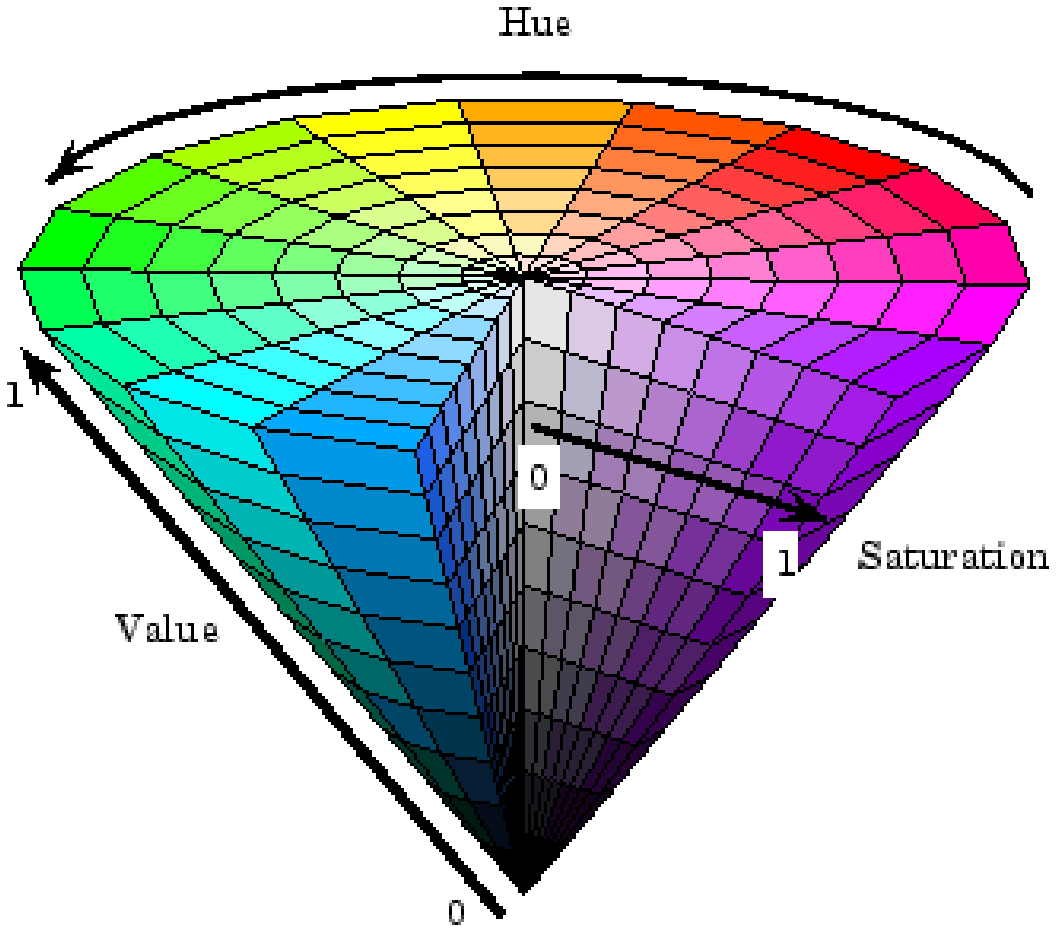
B
(R=0,G=0)

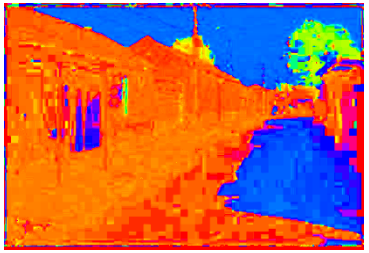
- Remarks
  - Easy for devices
  - But not perceptual
  - Where do the grays live?
  - Where is hue and saturation?

Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Example Color Space: HSV

- Intuitive Color Space



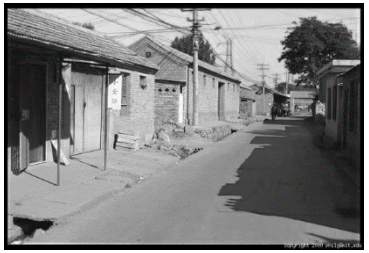


H

(S=1,V=1)

S

(H=1,V=1)

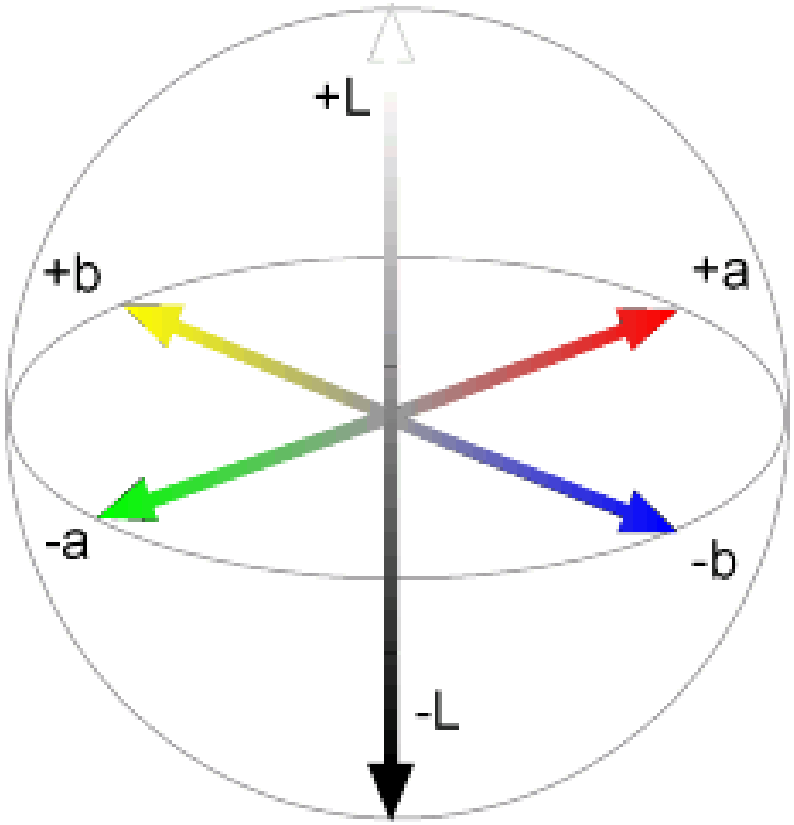


V

(H=1,S=0)

Example Color Space: L*a*b*

- "Perceptually uniform/linear" color space



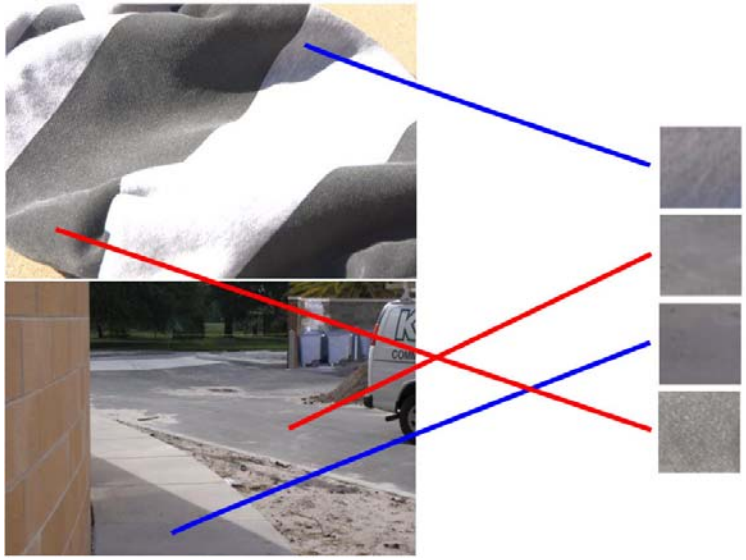
L
(a=0,b=0)

a
(L=65,b=0)

b
(L=65,a=0)

Light

- A pixel's brightness is determined by
  - Light source (strength, direction, color)
  - Surface orientation
  - Surface material and albedo
  - Reflected light and shadows from surrounding surfaces
  - Gain on the sensor
- A pixel's brightness tell us nothing but itself.



Slide credit: J.-B. Huang

Light (cont'd)

- Yet, we are able to interpret images from lightness.
  - For nearby pixels/points, most factors do not change much.
  - The information is mainly contained in *local differences* of brightness

Slide credit: J.-B. Huang

What is this?


HONDA

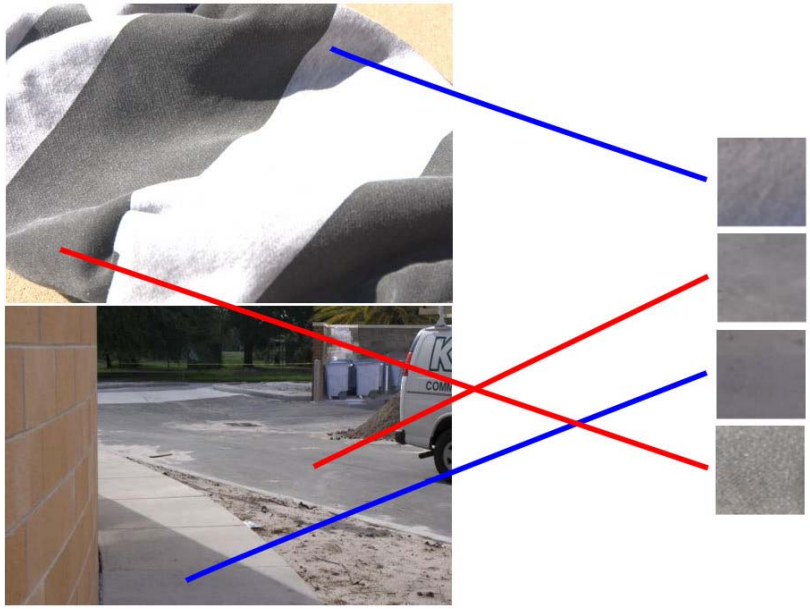
What differences in intensity tell us about shape?

- Changes in surface normal
- Texture
- Proximity
- Indents and bumps
- Grooves and creases

Photos Koenderink slides on image texture and the flow of light

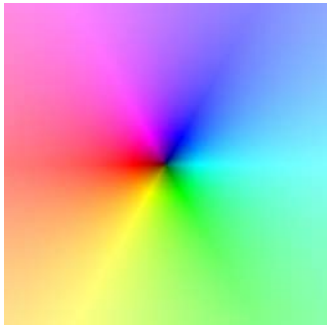
Color Constancy

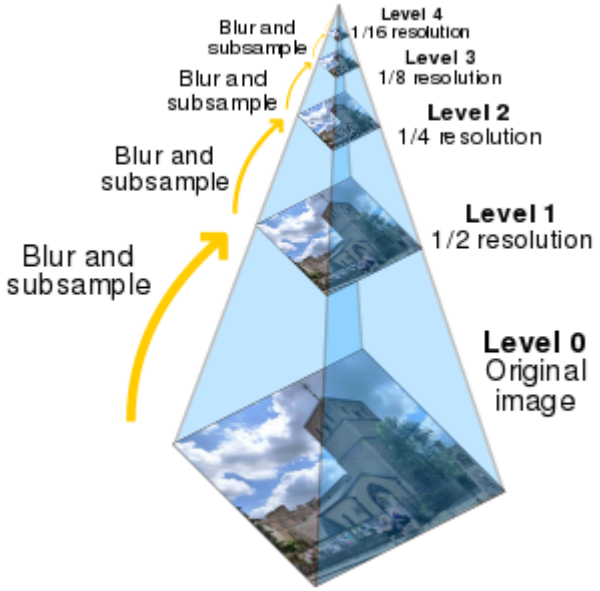
- Interpret surface in terms of albedo or “true color”, rather than observed intensity.
  - Humans are good at it.
  - Computers are not nearly as good…



What's to Be Covered Today…

- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid





Why Image Filtering?

Input Smoothing Sharpening

https://en.wikipedia.org/wiki/Albert_Einstein_in_popular_culture#/media/File:Einstein_tongue.jpg

Why Image Filtering?

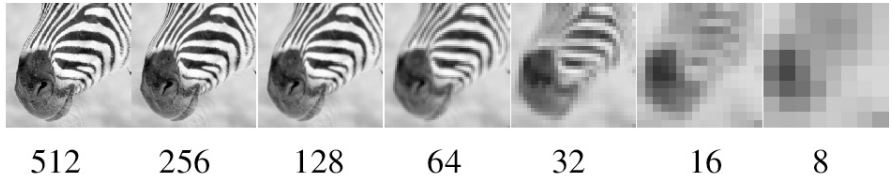


Image Pyramid

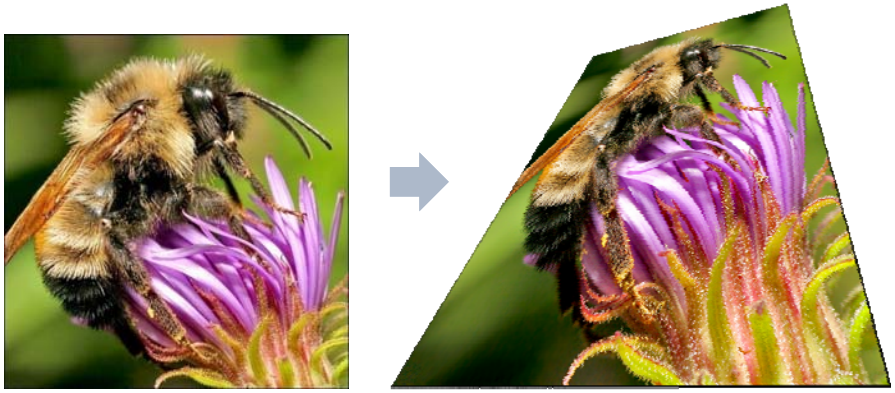
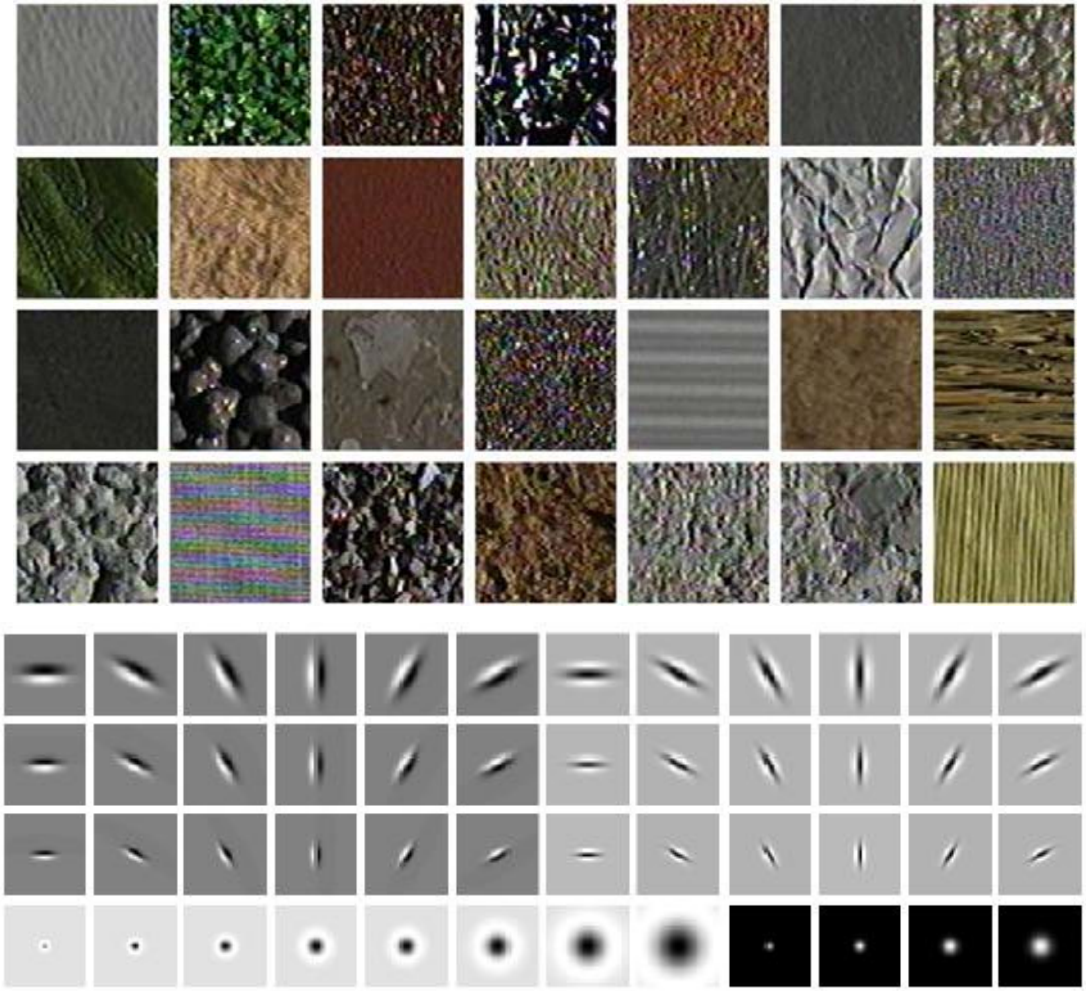


Image interpolation/resampling

Image credit: D. Forsyth & N. Snavely

Why Image Filtering?

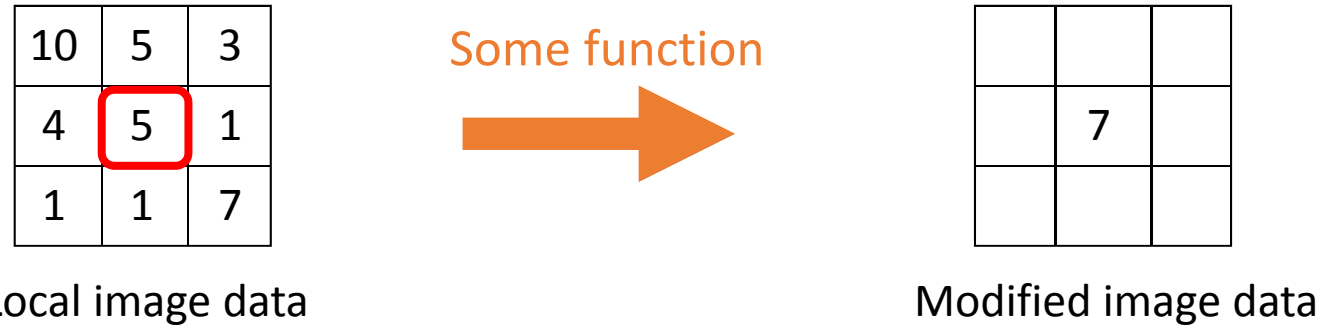
Representing textures with filter banks



LM filter bank. Code here

What's Image Filtering?

- Image Filtering
  - For each pixel of interest, compute the function of local neighborhood and output the new value.
  - Generally (while not always true), same function applied to each position
  - Typically (while not required), output and input images are of the same size.

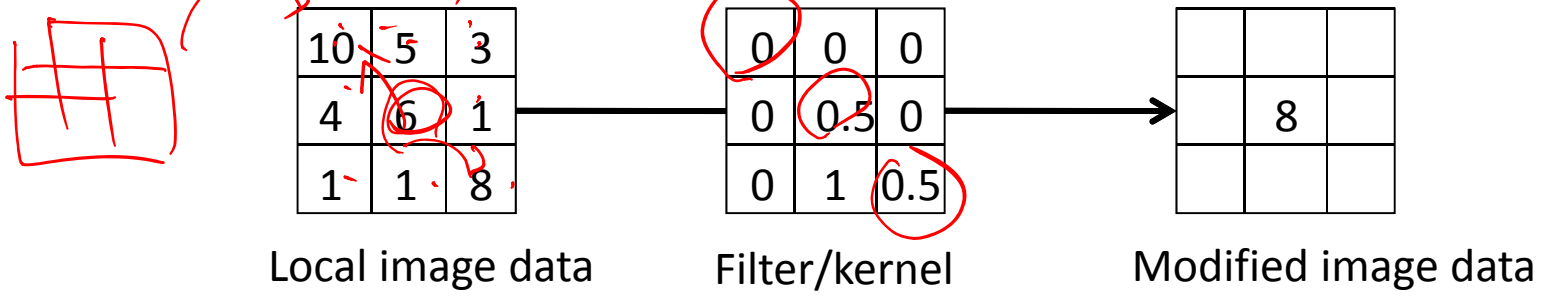


Local image data

Modified image data

What's Image Filtering?

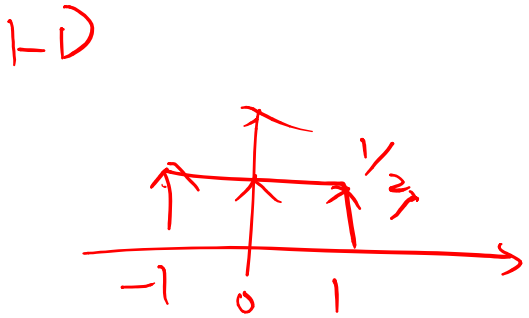
- Linear filtering
  - function is a weighted sum/difference of pixel values



- Example applications
  - Enhance images
    - Denoise, smooth, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns
    - Template matching

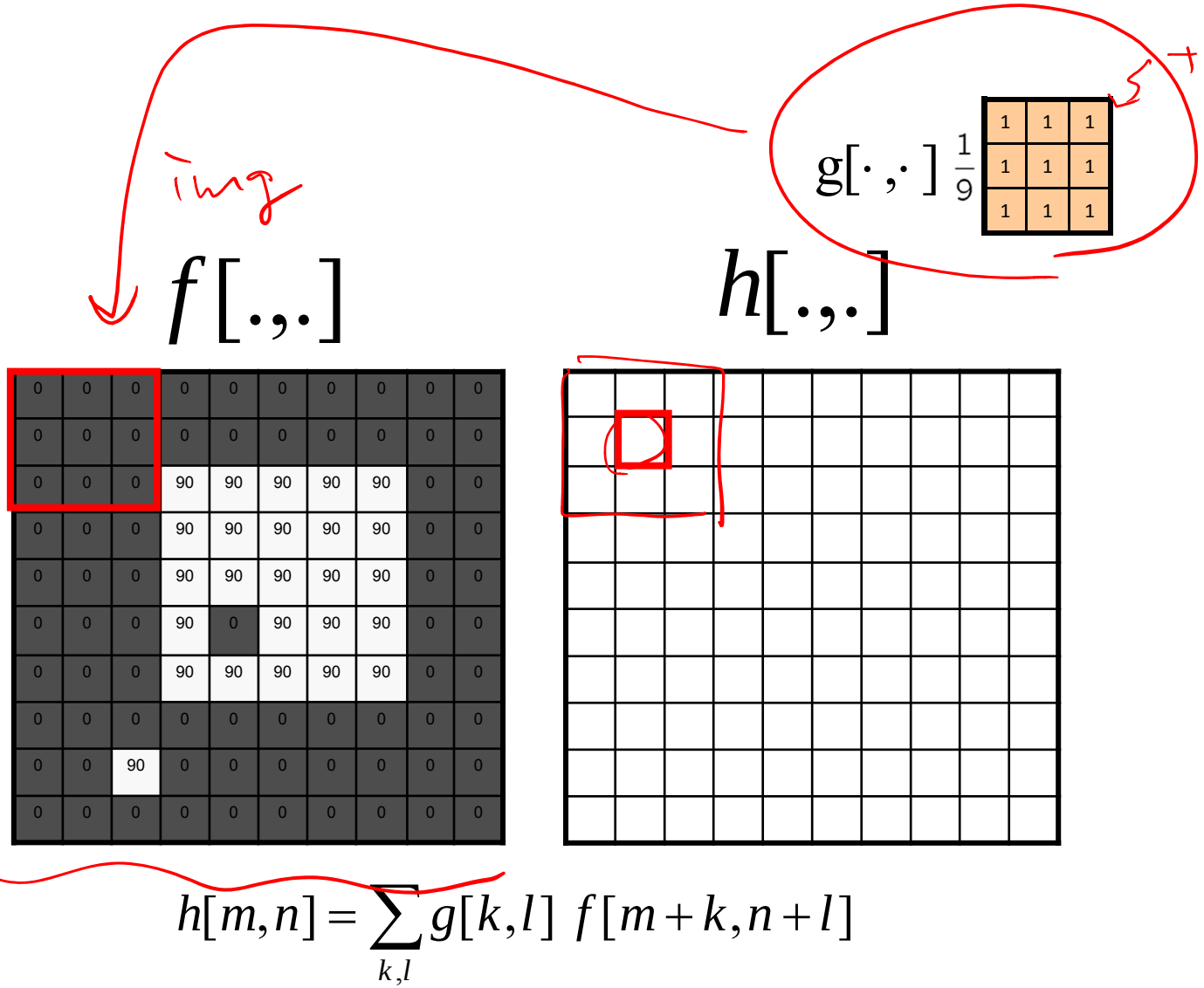
Slide credit: D. Hoiem

Example: Image Filtering with a Box Filter



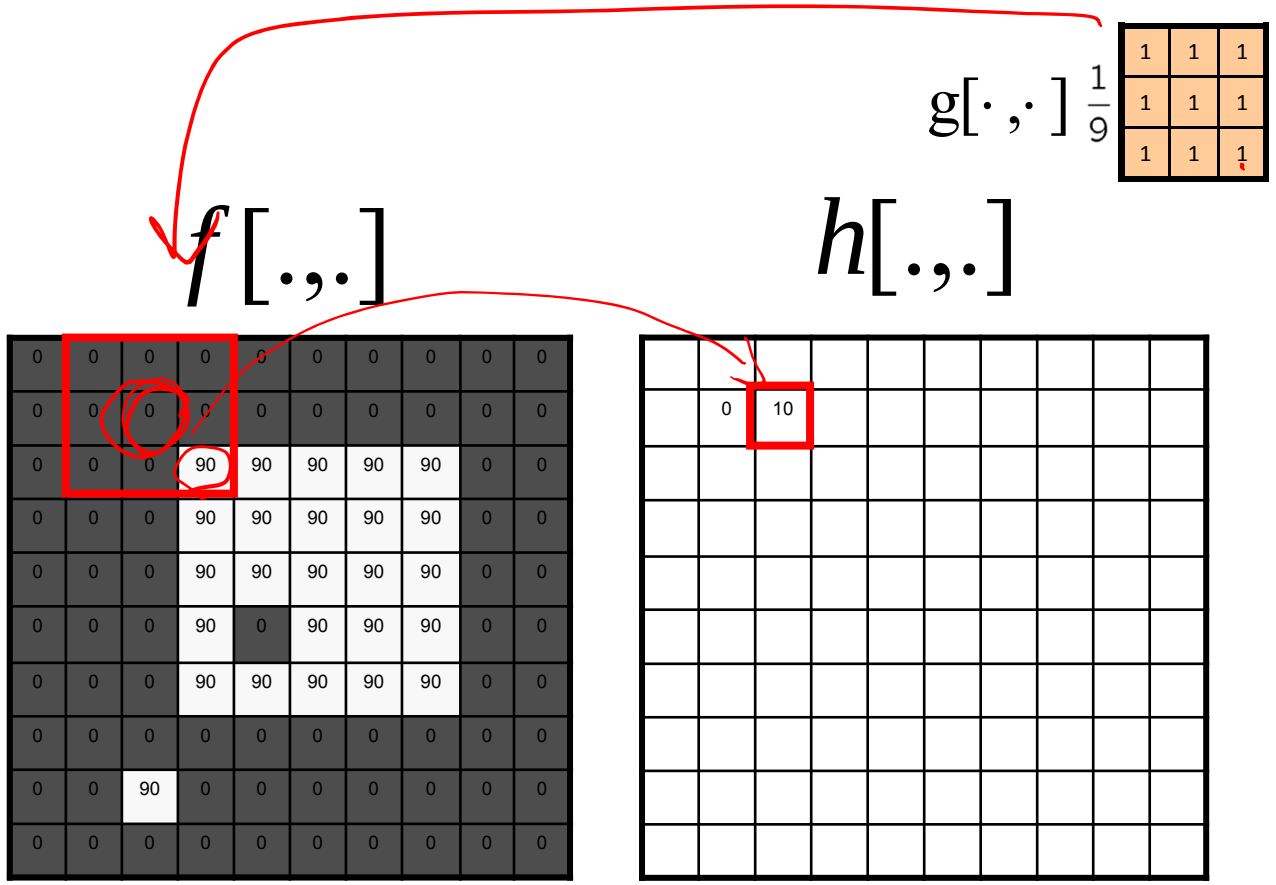
$$g[\cdot,\cdot] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: S. Seitz



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

Slide credit: S. Seitz



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

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20						

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

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30					

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

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				

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

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				?					

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

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

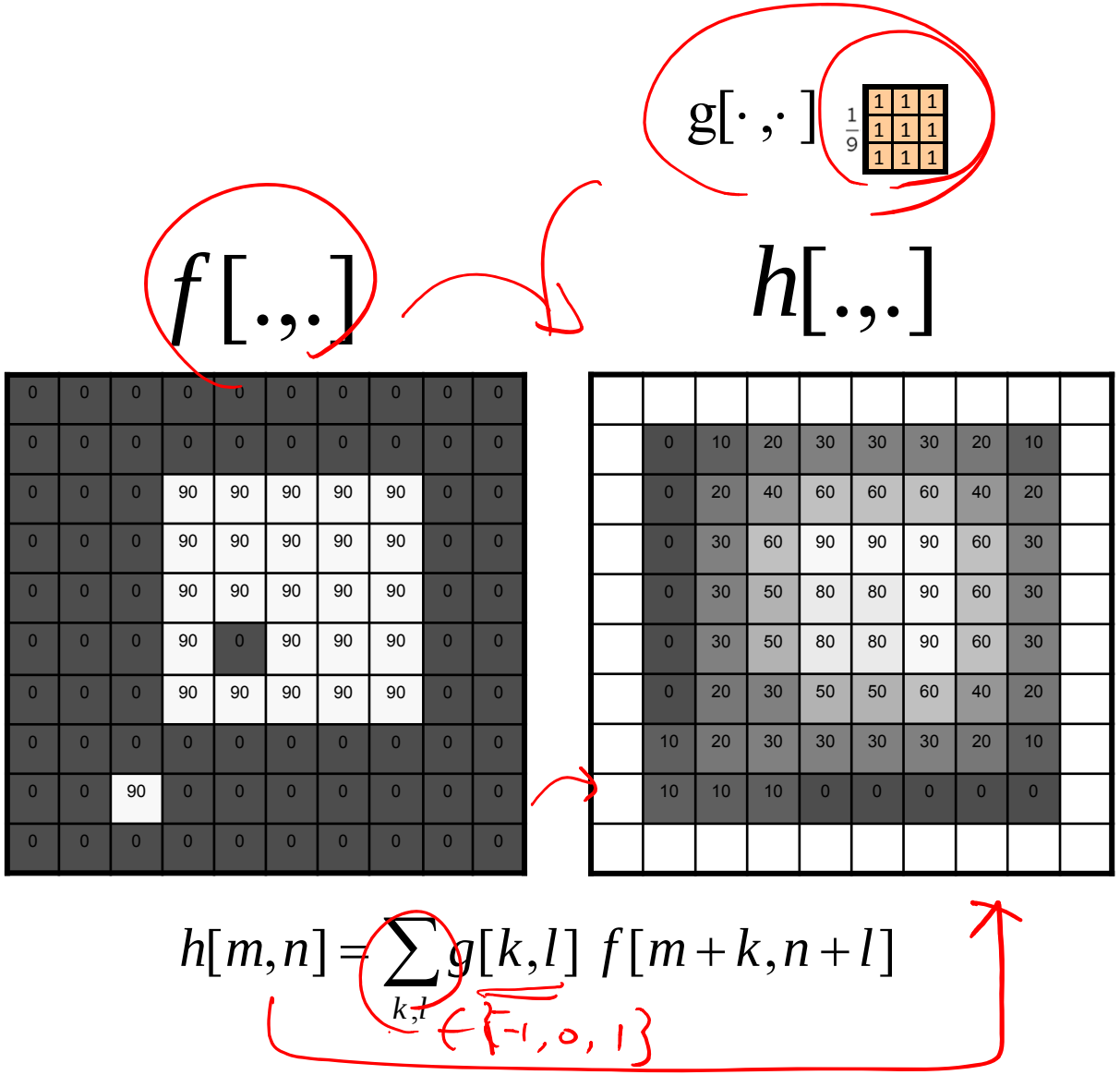
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				
						?			
				50					

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

Slide credit: S. Seitz



Slide credit: S. Seitz

Example: Image Filtering with a Box Filter

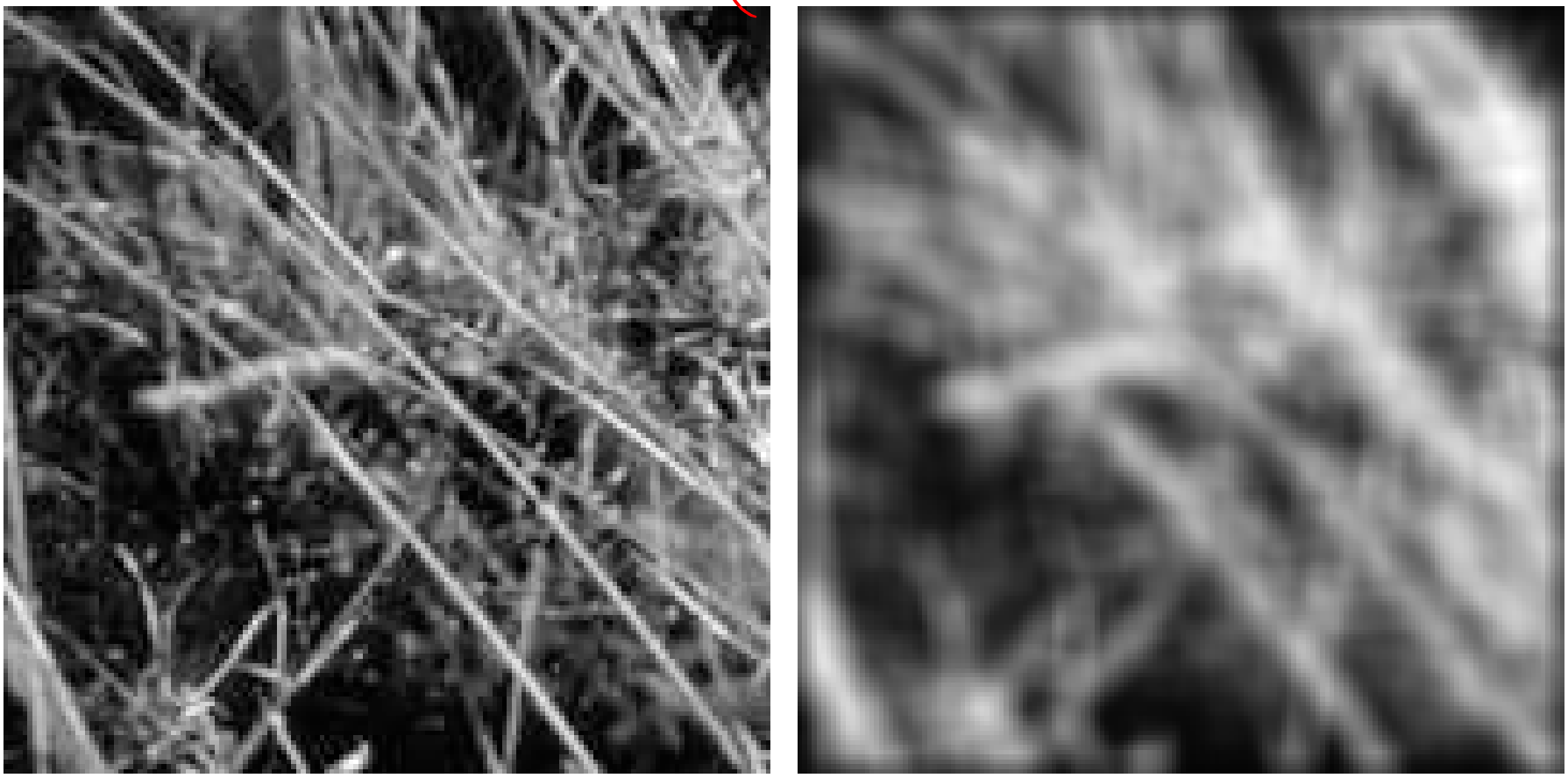
- What does Box Filter do?
  - Replaces each pixel with an average of its neighborhood
  - Performs image smoothing
  - Achieves image denoising?

$$g[\cdot,\cdot]$$

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

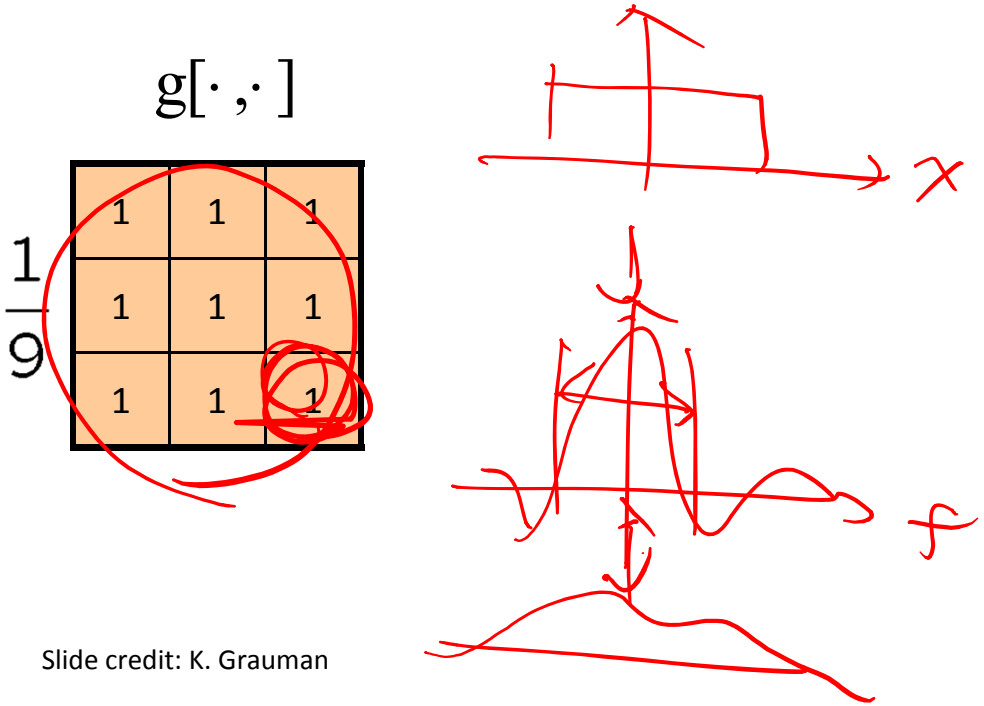
Slide credit: K. Grauman

Smoothing with a Box Filter



Properties of Smoothing Filters

- Smoothing
  - Values positive
  - Sum to 1 → constant regions same as input
  - Amount of smoothing proportional to the mask/filter size
  - Remove "high-frequency" components, i.e., low-pass filter (LPF)



Slide credit: K. Grauman

Correlation Filtering

Say the averaging window size is 2k+1 x 2k+1:

$$G[i,j] = \frac{1}{(2k+1)^2} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]}$$

Attribute uniform weight to each pixel

Now generalize to allow different weights depending on neighboring pixel's relative position:

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

Slide credit: K. Grauman

Correlation Filtering

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

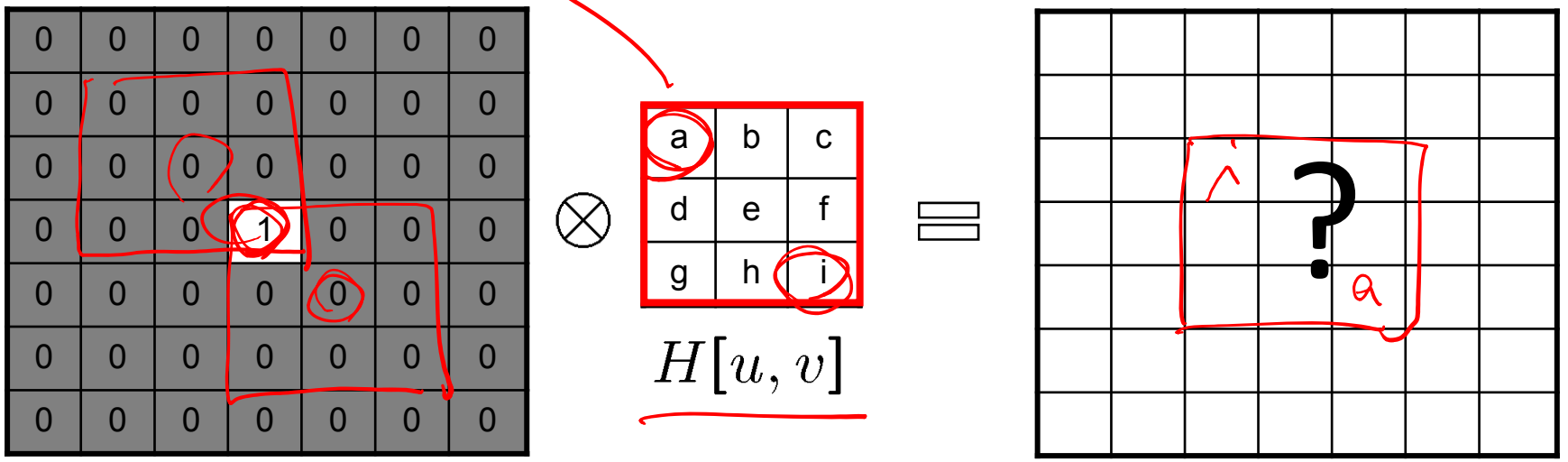
This is called cross-correlation, denoted $G = H \otimes F$

Note that the filter "kernel" or "mask" $H[u,v]$ is the prescription for the pixel weights for linear combination.

Slide credit: K. Grauman

Filtering an Impulse Signal

What is the result of filtering the impulse signal (image) *F* with an arbitrary kernel *H*?



$F[x, y]$ $H[u, v]$ $G[x, y]$

Slide credit: K. Grauman

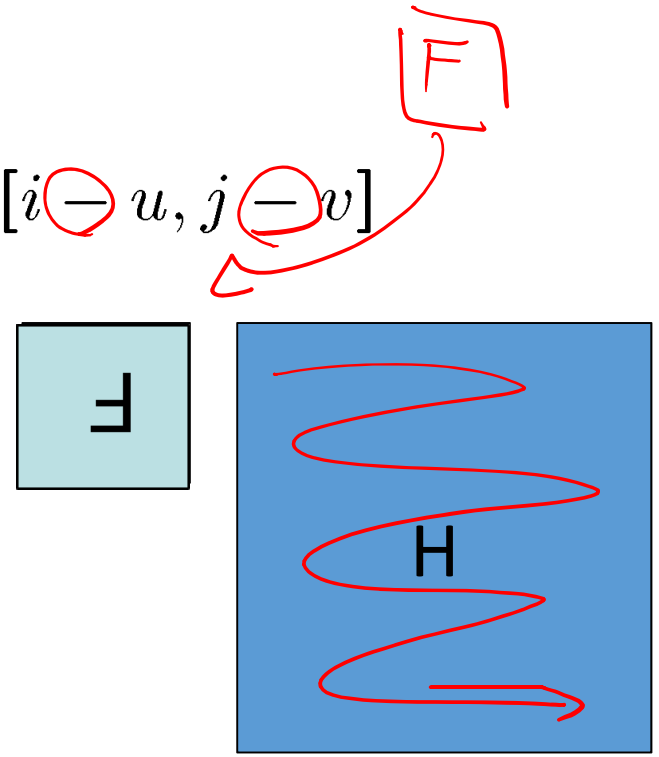
Convolution

- Convolution:
  - Flip the filter in both dimensions (bottom to top, right to left)
  - Then apply cross-correlation

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

$$G = H \star F$$

Notation for convolution operator



Slide credit: K. Grauman

Convolution vs. Correlation

Convolution

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

```
G=conv2(H,F);
```

$$G = H \star F$$

Cross-correlation

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

```
G=filter2(H,F); or
G=imfilter(F,H);
```

$$G = H \otimes F$$

For a Gaussian or box filter, how will the outputs differ?

If the input is an impulse signal, how will the outputs differ?

Slide credit: K. Grauman

Practice with linear filters

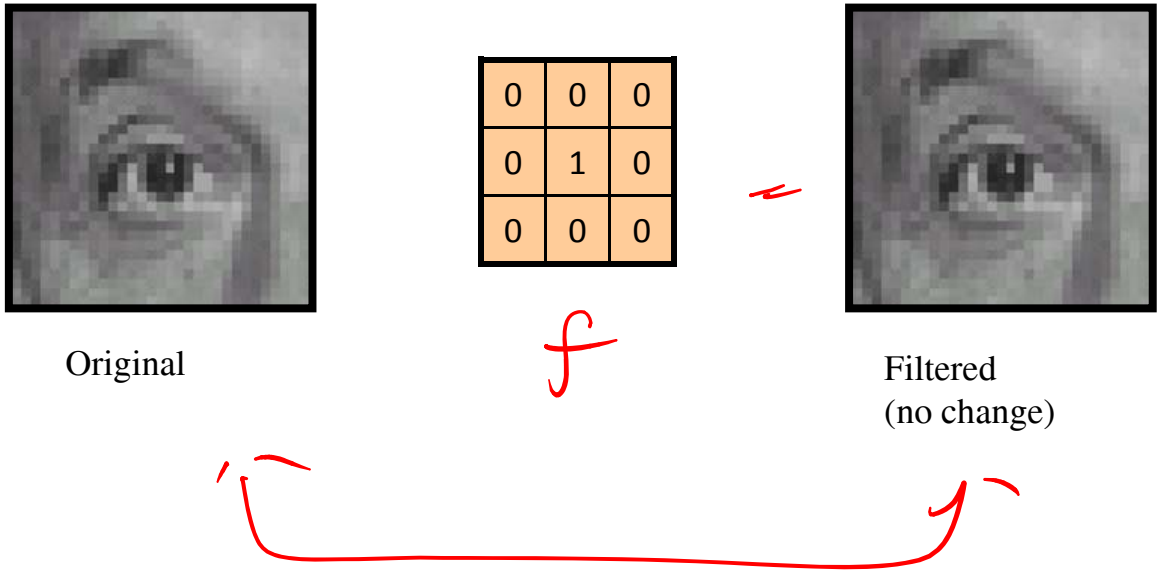
Original

0	0	0
0	1	0
0	0	0

?

Slide credit: David Lowe (UBC)

Practice with linear filters



Slide credit: David Lowe (UBC)

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0

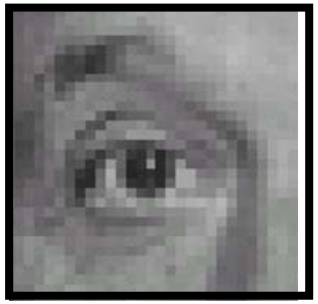
?

Slide credit: David Lowe (UBC)

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0



Shifted left or right?
By 1 pixel

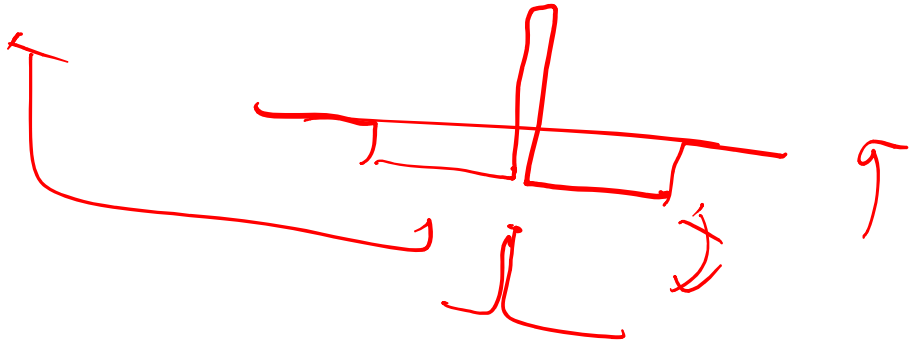
Slide credit: David Lowe (UBC)

Practice with linear filters

Original

$$\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) \quad ?$$

(Note that filter sums to 1)

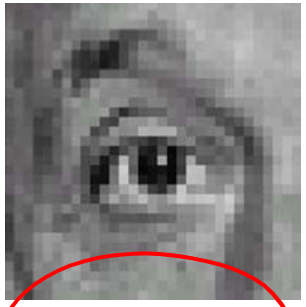


Slide credit: David Lowe (UBC)

Practice with linear filters

Original

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

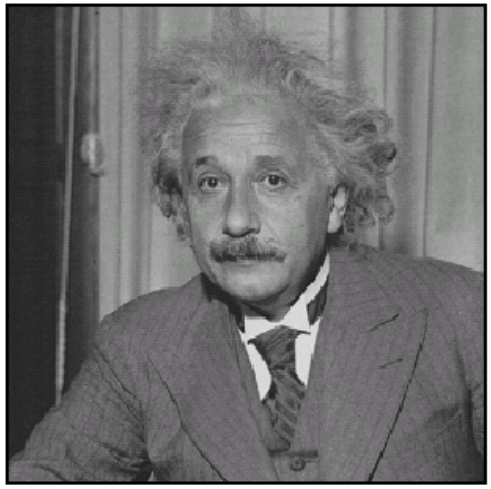


Sharpening filter

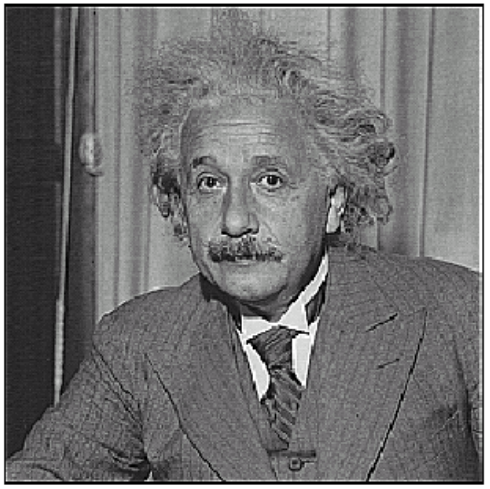
Accentuates differences with local average

Slide credit: David Lowe (UBC)

Sharpening



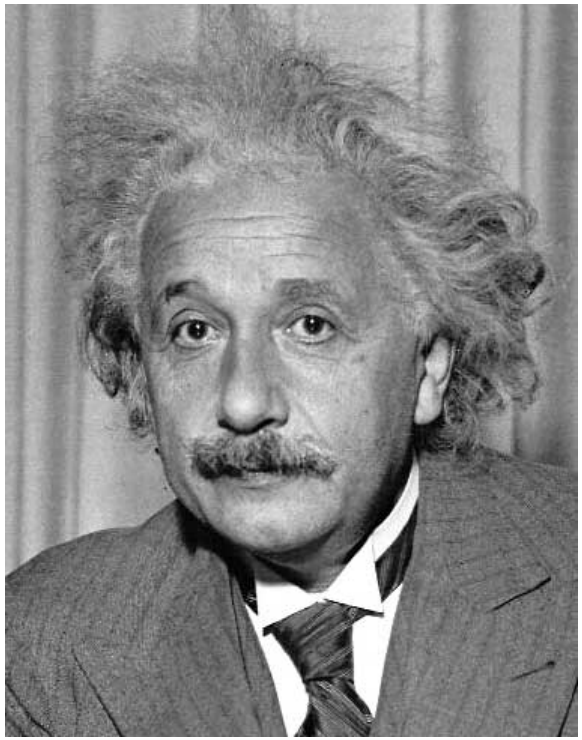
before



after

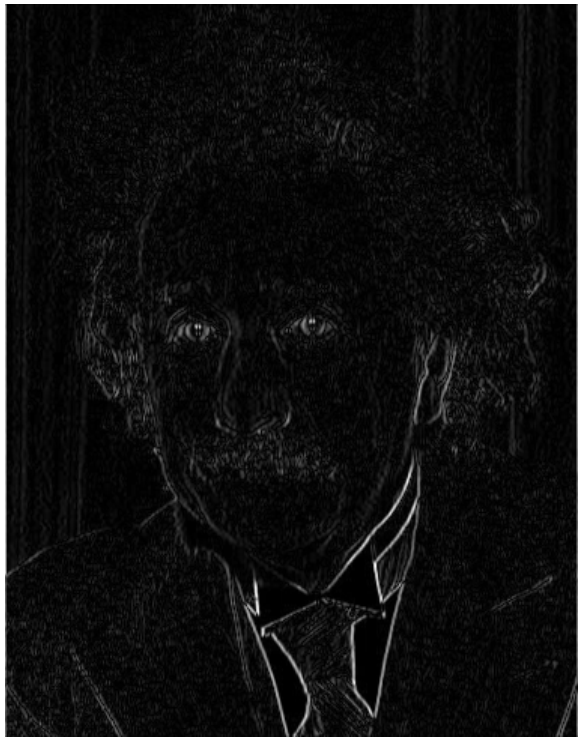
Slide credit: David Lowe (UBC)

Other filters



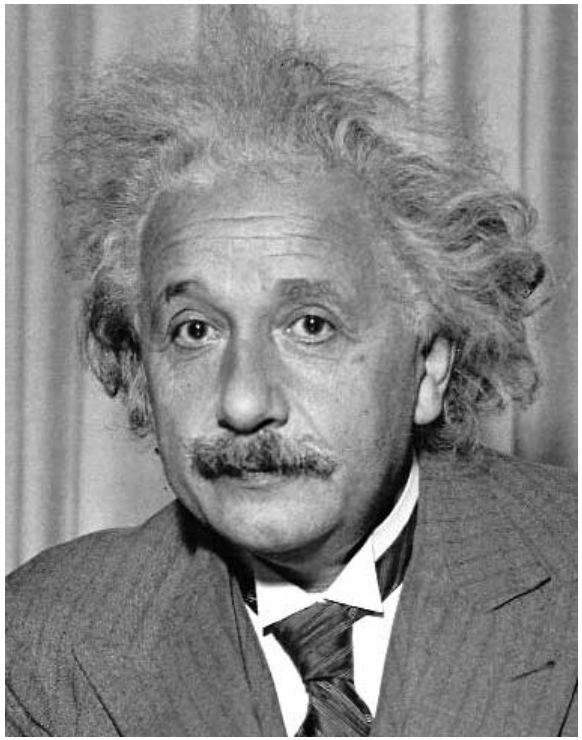
1	0	-1
2	0	-2
1	0	-1

Sobel



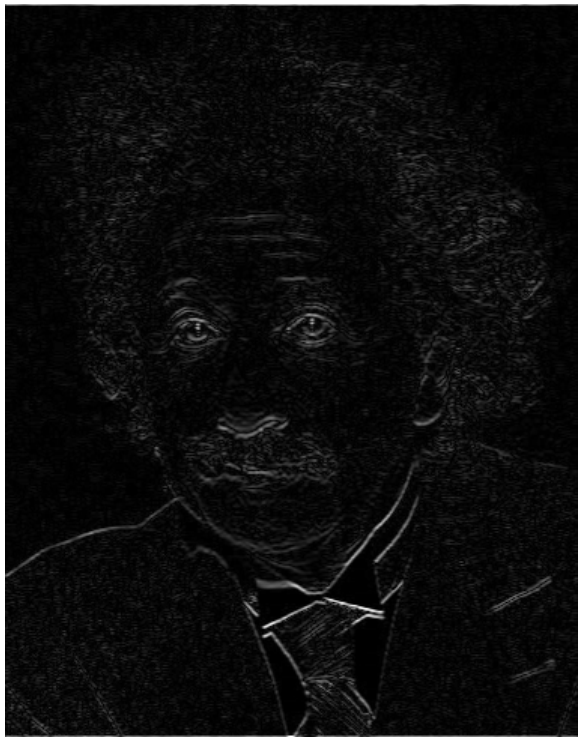
Vertical Edge
(absolute value)

Other filters



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

Sobel



Horizontal Edge
(absolute value)

Basic gradient filters

Horizontal Gradient

0	0	0
-1	0	1
0	0	0

or

-1	0	1

Vertical Gradient

0	-1	0
0	0	0
0	1	0

or

-1
0
1

Questions

Write as filtering operations, plus some pointwise operations: +, -, .*, >, etc.

1. Sum of four adjacent neighbors plus 1

2. Sum of squared values of 3x3 windows around each pixel:

3. Center pixel value is larger than the average of the pixel values to the left and right:

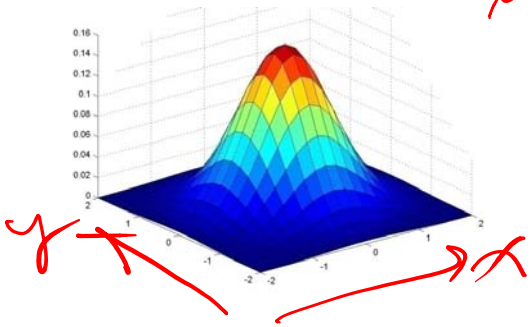
More properties

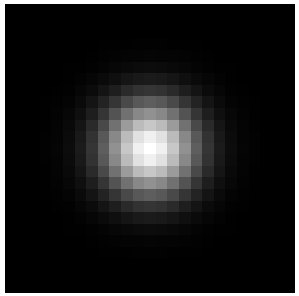
- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: $a * (b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k (a * b)$
- Identity: unit impulse $e = [0, 0, 1, 0, 0]$, $a * e = a$

Slide credit: S. Lazebnik

Gaussian Filter

- Spatially-weighted averaging





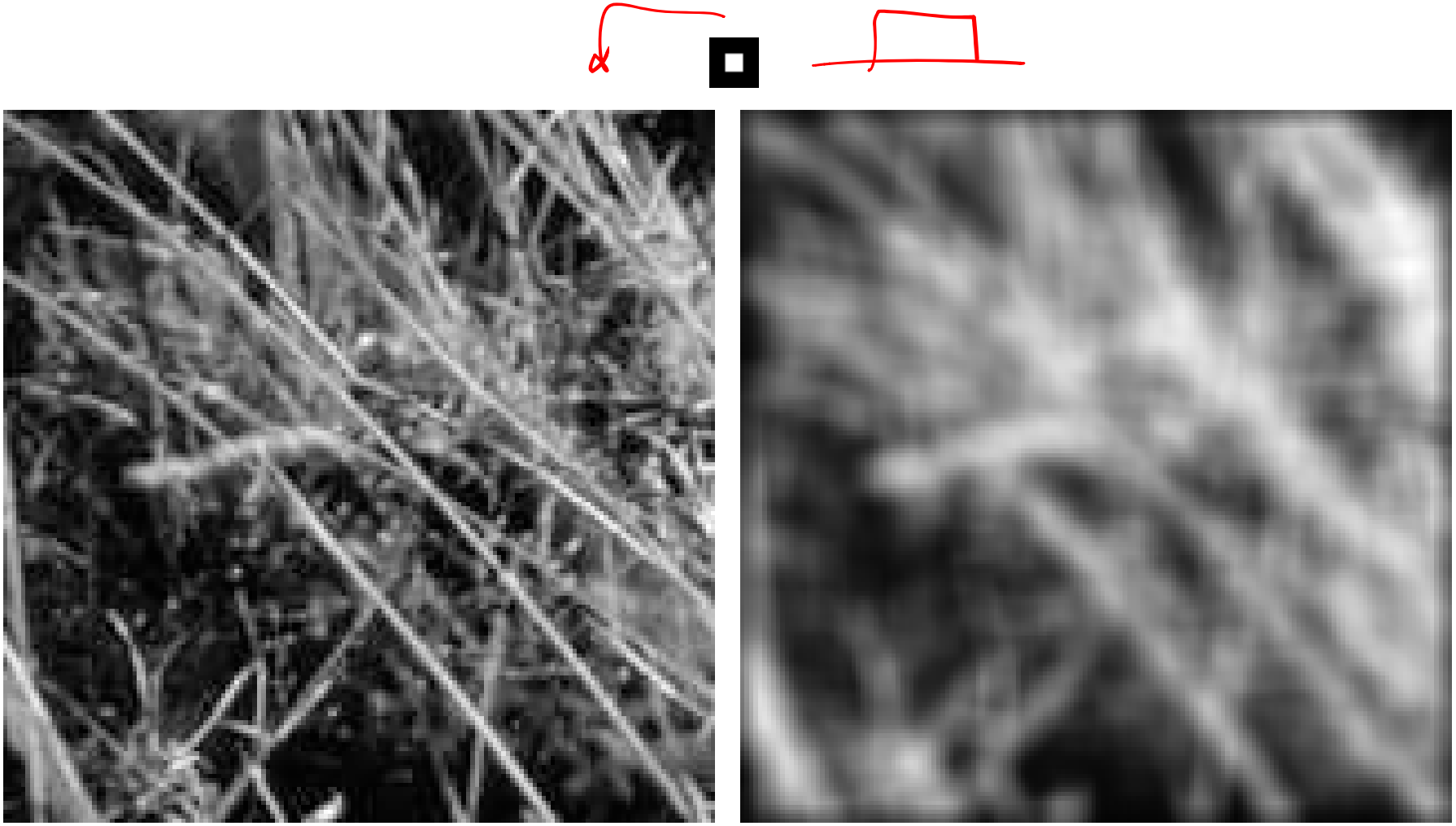
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

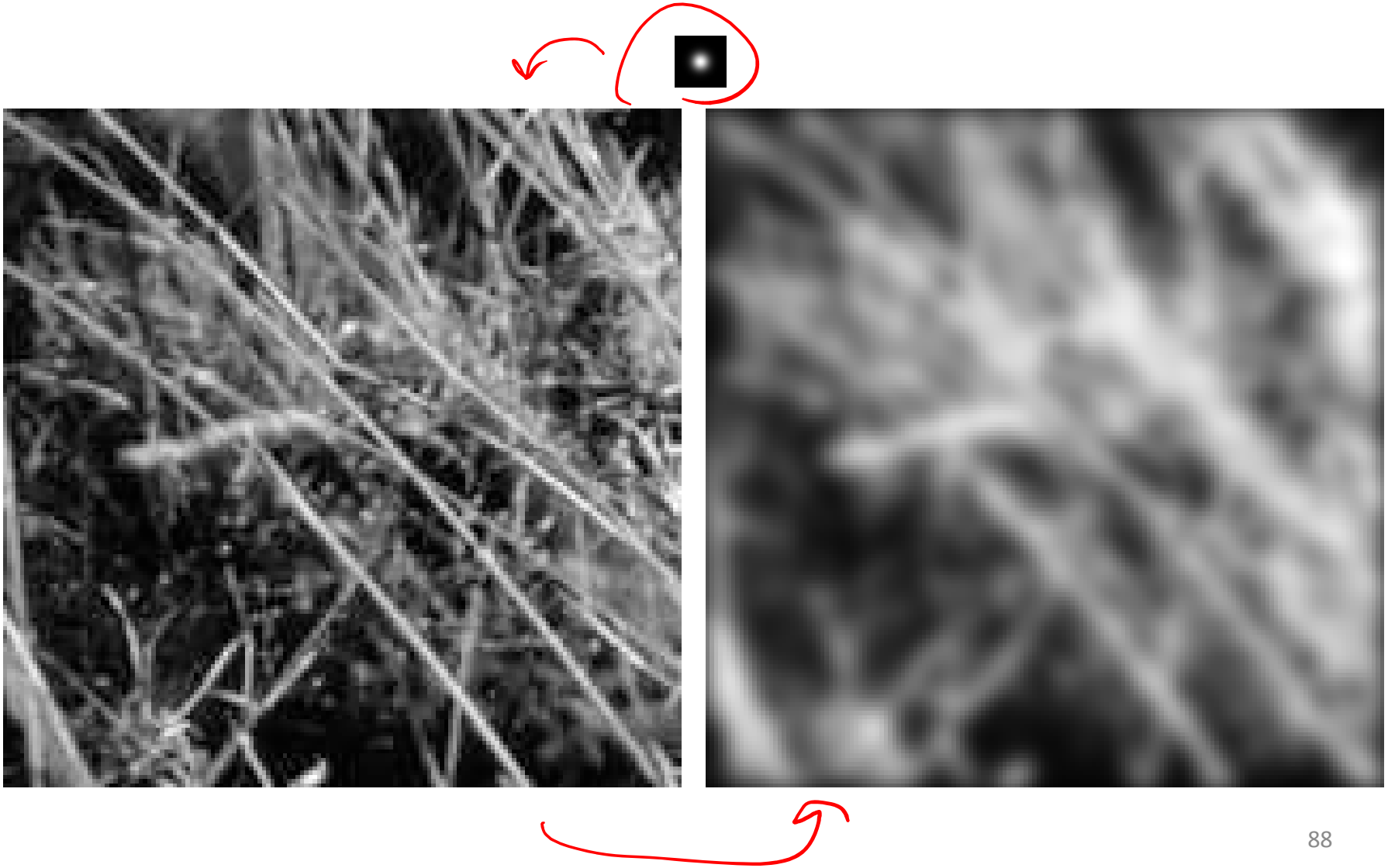
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Slide credit: Christopher Rasmussen

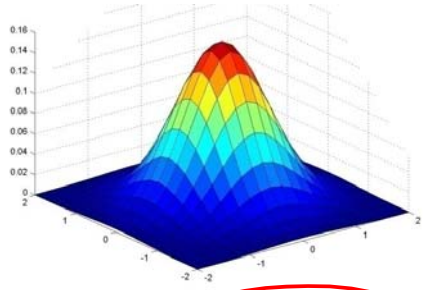
Smoothing with a Box Filter



Smoothing with a Gaussian Filter



Gaussian Filter

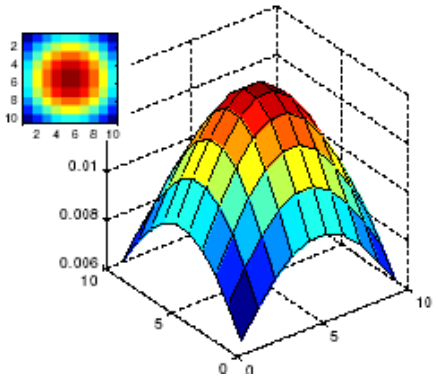


- Remove "high-frequency" components from the image (i.e., low-pass filter)
  - Images become more smooth
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width σ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernels
  - Factors into product of two 1D Gaussians

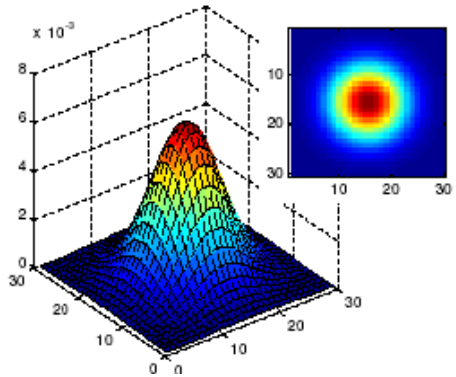
Slide credit: K. Grauman

Gaussian Filter

- What parameters matter here?
- **Size** of filter (or kernel/mask)



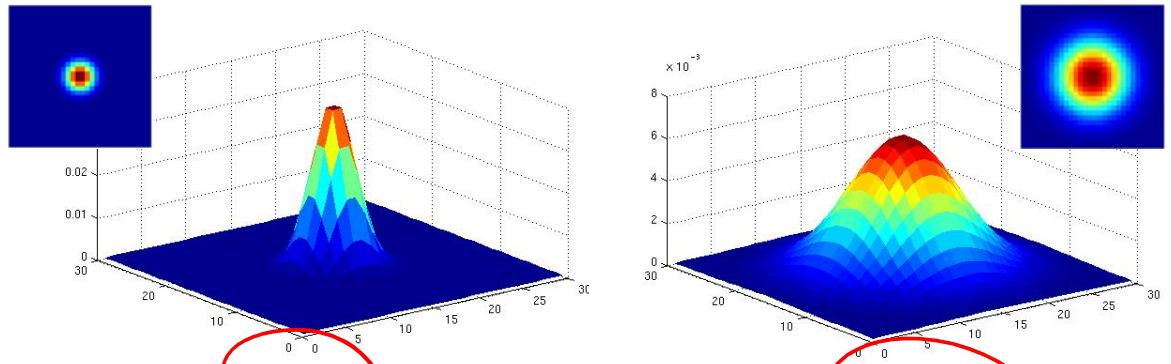
σ = 5 with 10 x 10 kernel



σ = 5 with 30 x 30 kernel

Gaussian Filter

- What parameters matter here?
- **Variance** of Gaussian: determines extent of smoothing



Separability of the Gaussian Filter

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

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

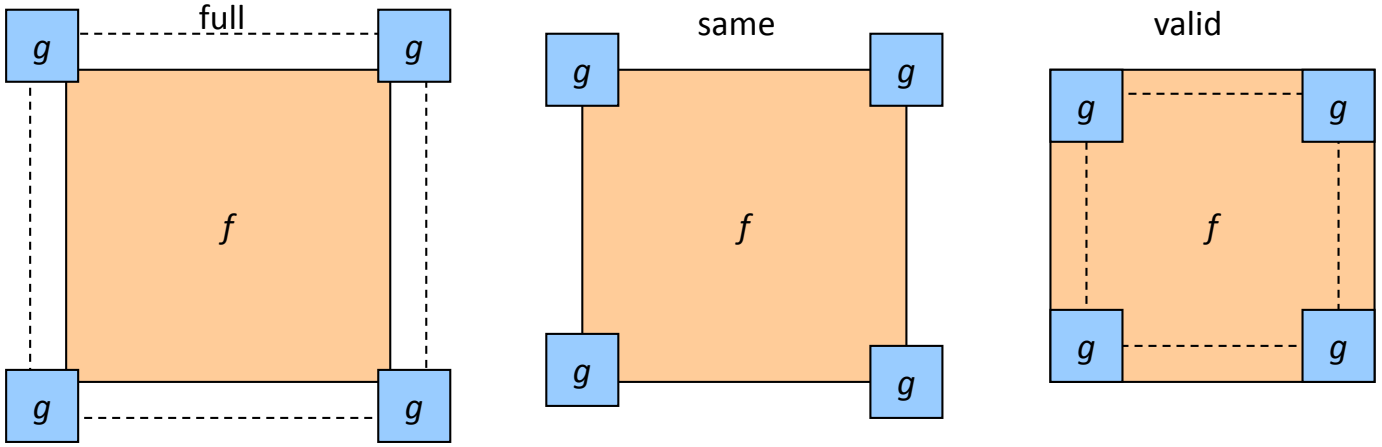
Source: D. Lowe

Separability of the Gaussian Filter

- Why is separability useful in practice?
- Separability means that a 2D convolution can be reduced to two 1D convolutions (one among rows and one among columns)
- What is the complexity of filtering an $n \times n$ image with an $m \times m$ kernel?
  - $O(n^2 m^2)$
- What if the kernel is separable?
  - $O(n^2 m)$

Practical matters

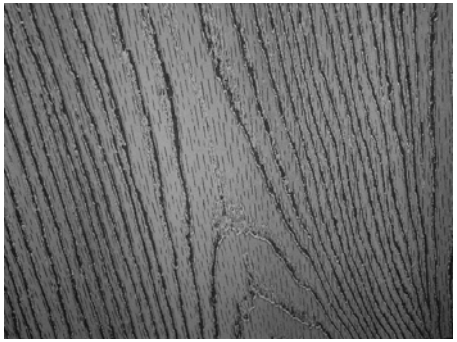
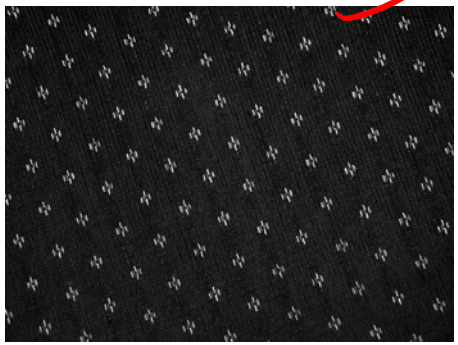
- What is the size of the output?
- MATLAB: filter2(g, f, *shape*)
  - *shape* = 'full': output size is sum of sizes of f and g
  - *shape* = 'same': output size is same as f
  - *shape* = 'valid': output size is difference of sizes of f and g



Source: S. Lazebnik

Applications: Representing Image Texture

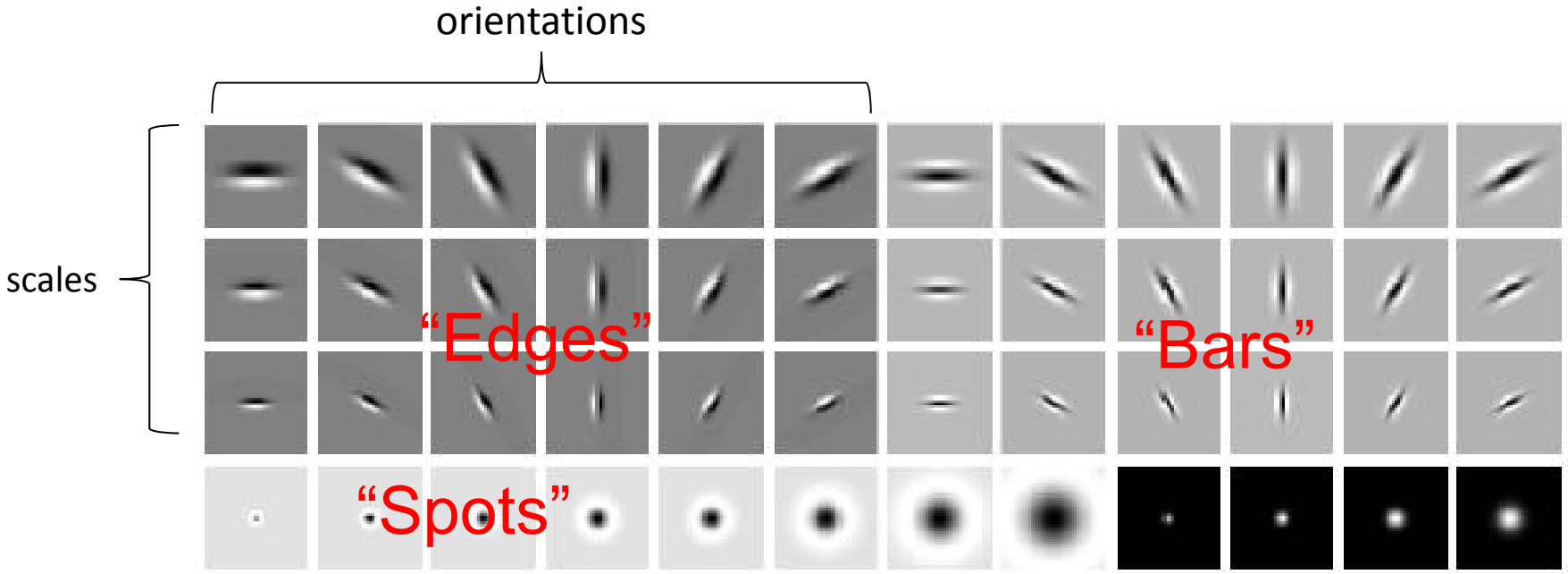
- Texture: Orientation, Material, and Scale


http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

How to Describe Textures?

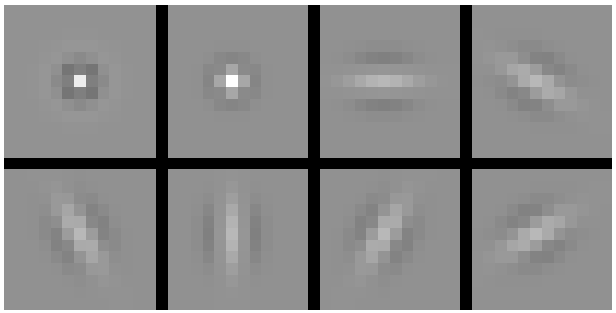
- Filter banks
  - Typically overcomplete representation
  - E.g., wavelet or more complex filters (basis functions)

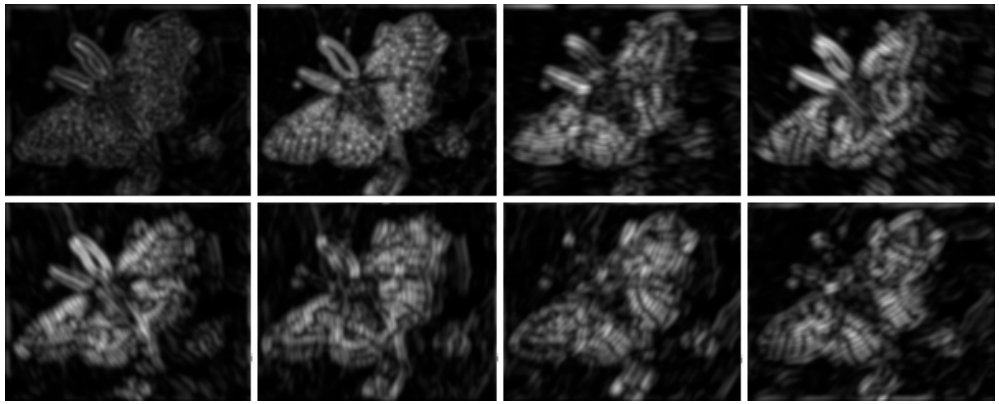


Code for filter banks: www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

How to Describe Textures?

- Filter banks
  - Typically overcomplete representation
  - E.g., wavelet or more complex filters (basis functions)
  - Example





Can you match the filtering response?

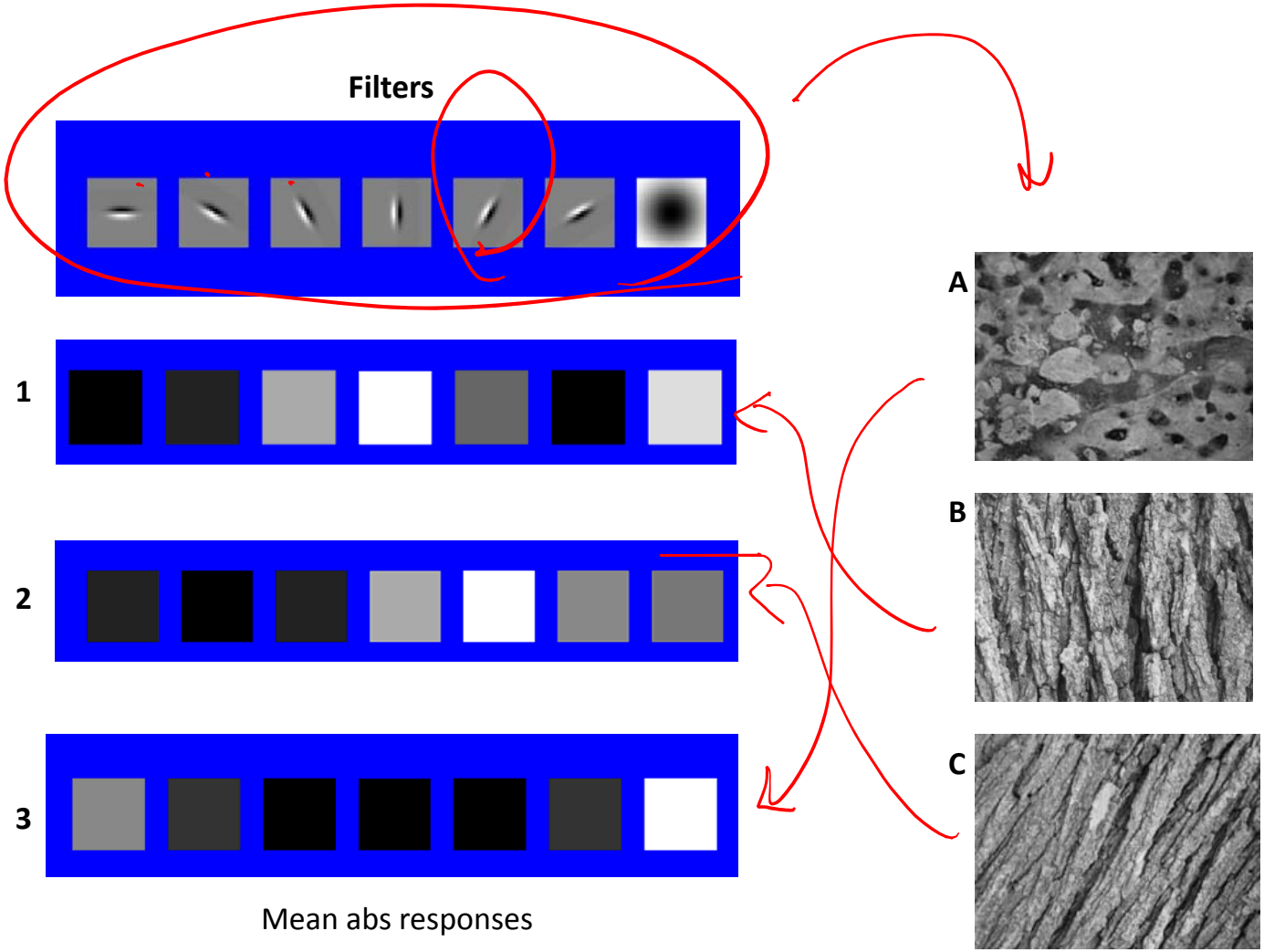


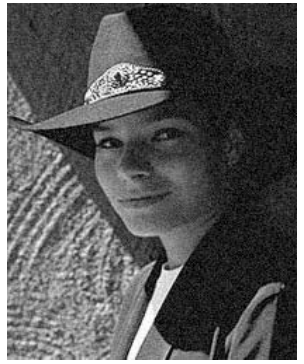
Image Denoising with Various Noise Types

- Salt and pepper
  - Contains random occurrences of black/white pixels
- Impulse noise
  - Contains random occurrences of white pixels
- Gaussian noise
  - Random pixels with variations in intensity drawn from a Gaussian normal distribution

Original

Salt and pepper noise

Impulse noise

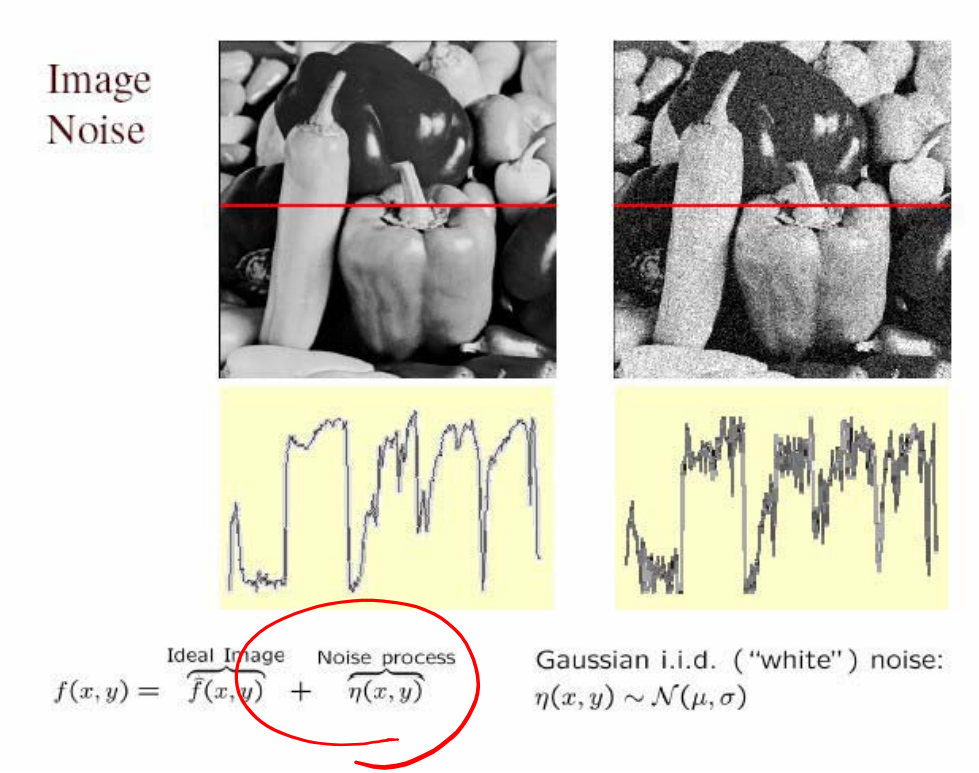


Gaussian noise

Slide credit: S. Seitz

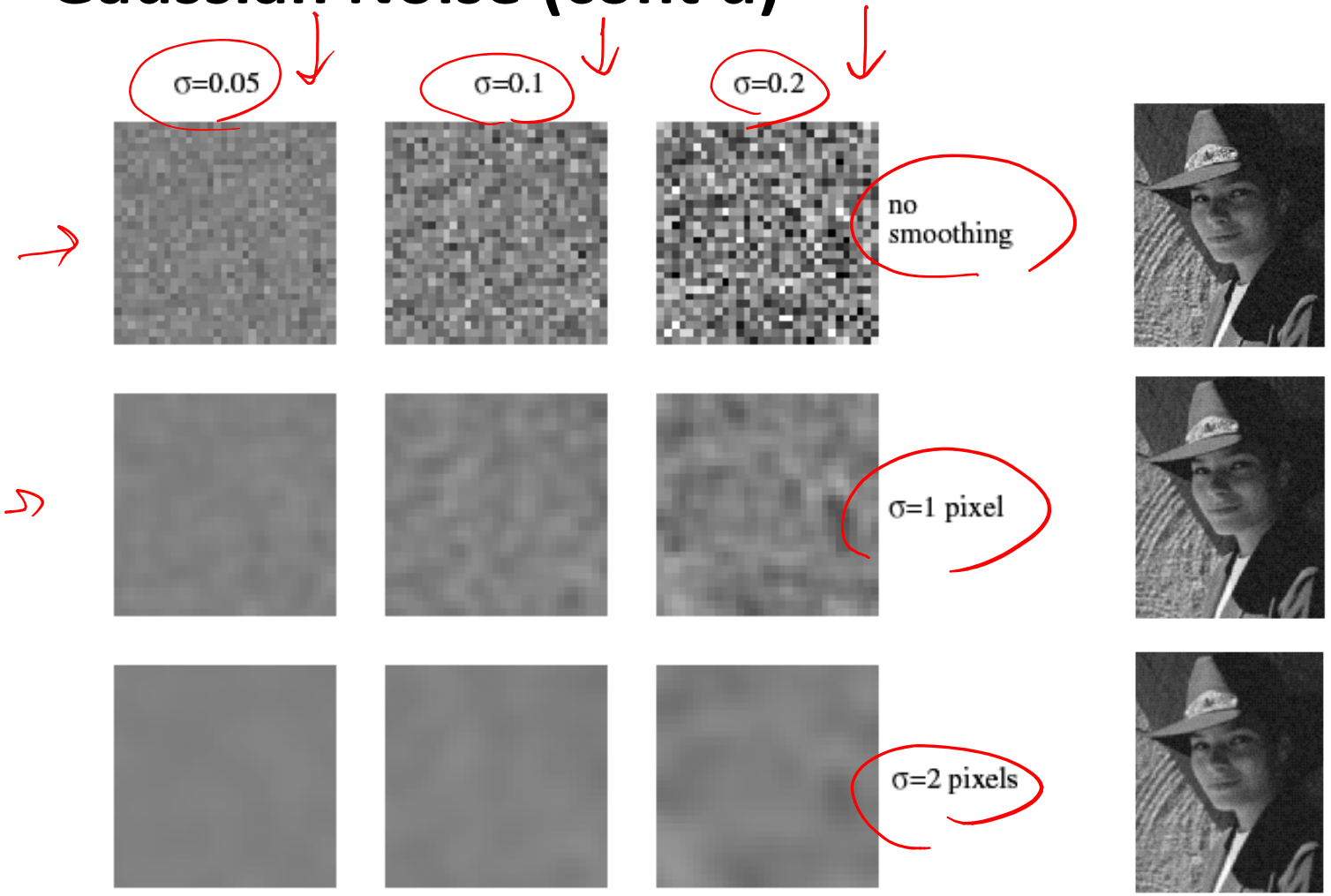
Gaussian Noise

- Assumption: independent, zero-mean noise
- Mathematical model: sum of many independent factors
- Good for small standard deviations



Slide credit: M. Hebert

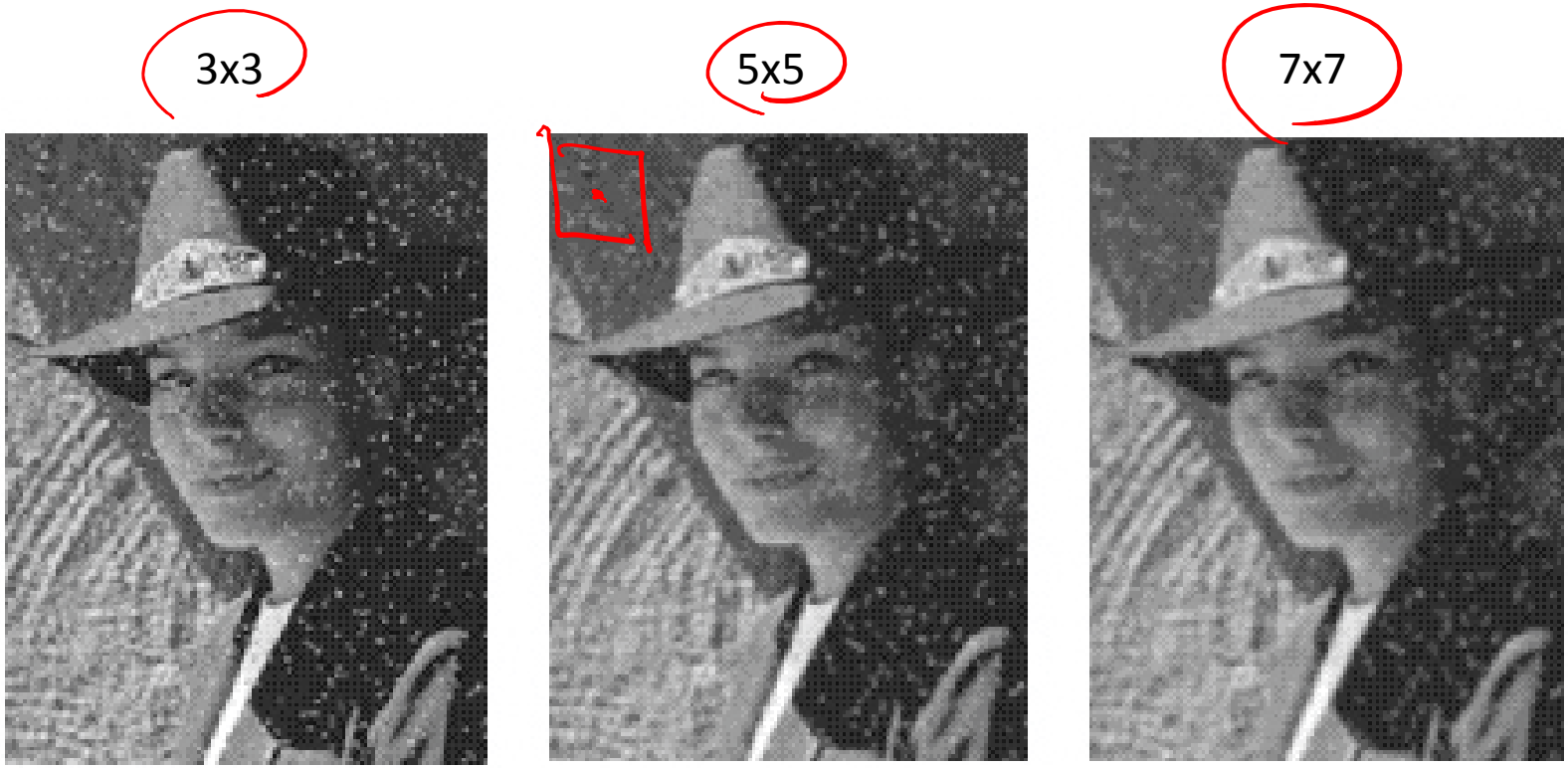
Gaussian Noise (cont'd)



- Smoothing with larger σ suppresses noise but also results in blurry images.

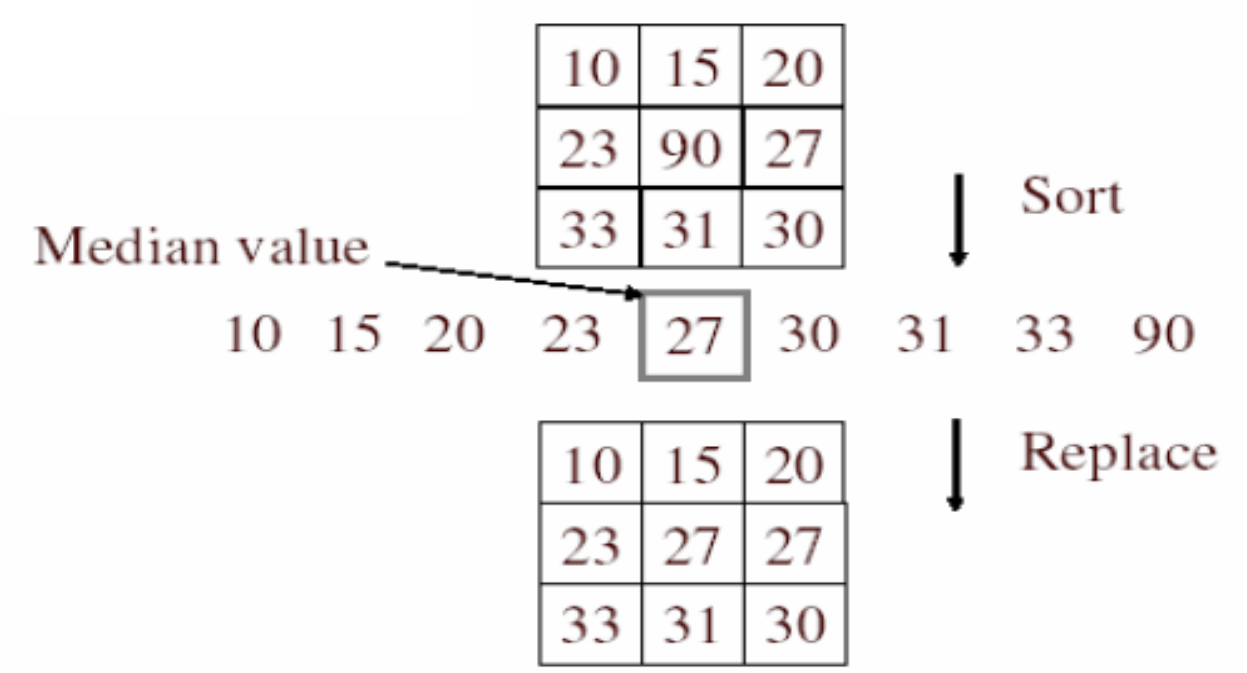
Salt-and-Pepper Noise

- What's wrong with the results?



Alternative Idea…

- Median Filtering
  - A median filter operates over a window by selecting the median intensity of the pixels in that window.
  - Is median filtering a linear operation?

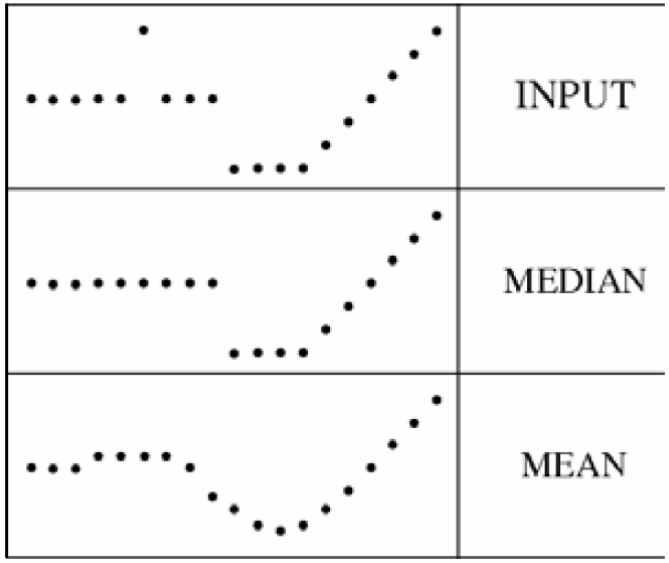


Slide credit: K. Grauman

Alternative Idea…

- Median Filtering
  - A median filter operates over a window by selecting the median intensity of the pixels in that window.
  - Is median filtering a linear operation?
  - What advantage does median filter have over the Gaussian one?
    - Robustness to outliers.

filters have width 5 :

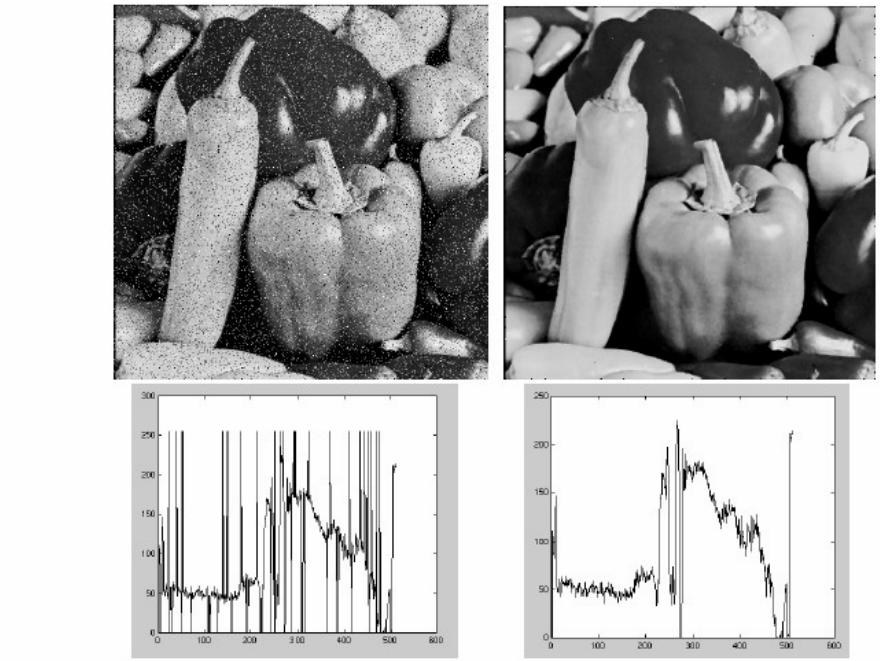


Slide credit: K. Grauman

Alternative Idea…

- Median Filtering
  - A median filter operates over a window by selecting the median intensity of the pixels in that window.
  - What advantage does median filter have over the Gaussian one?
    - Robustness to outliers.
  - Example

Salt-and-pepper noise Median filtered



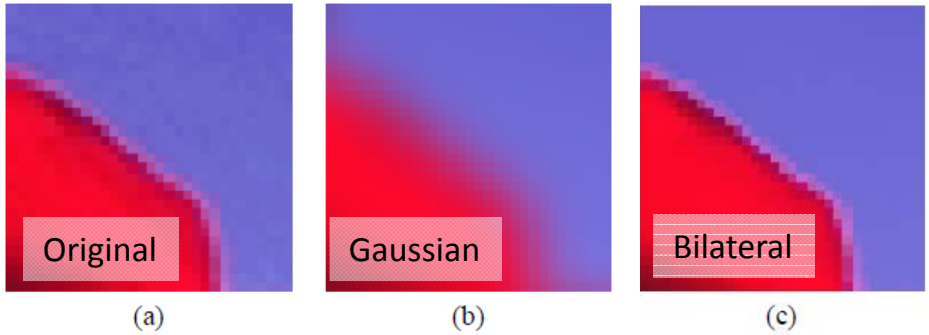
Other Non-Linear Filters

- Weighted median (pixels further from center count less)
- Clipped mean (average, ignoring few brightest and darkest pixels)
- Bilateral filtering (weight by spatial distance *and* intensity difference)

Bilateral filtering

Bilateral Filters

- Edge preserving: weights similar pixels more



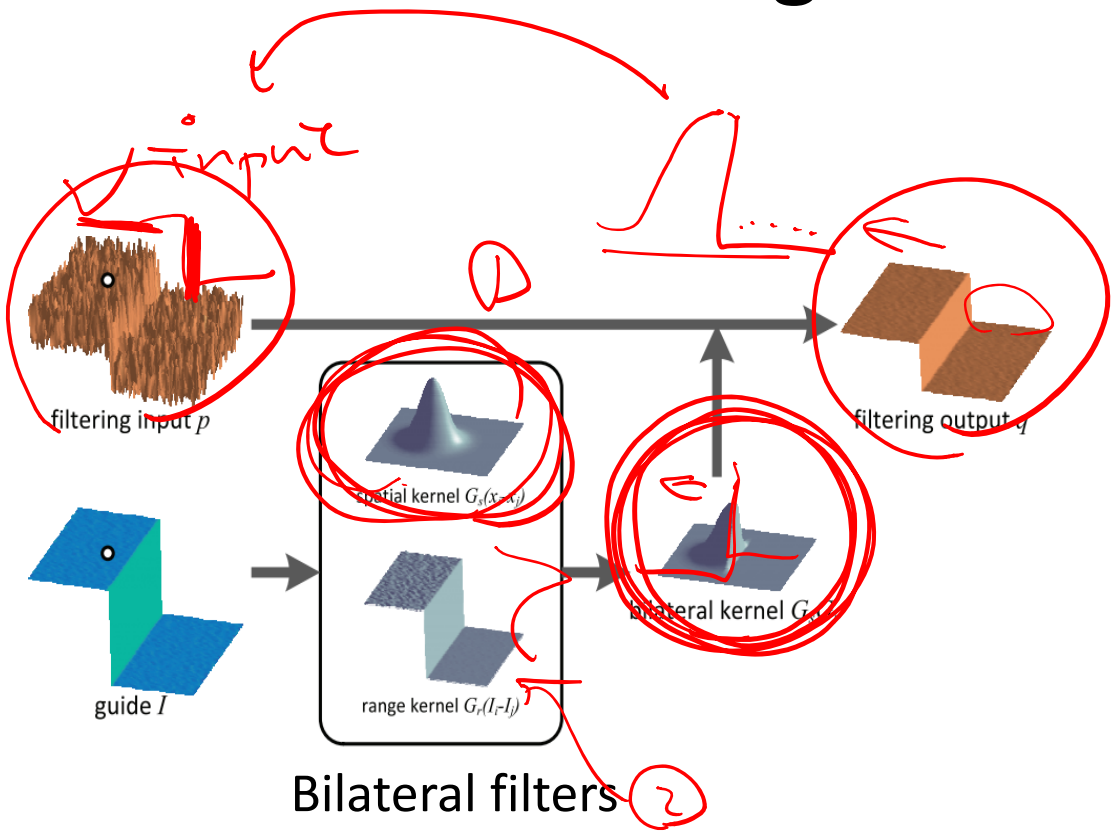
spatial similarity (e.g., intensity)

$$I^{\mathrm{b}}_{\mathbf{p}} = \frac{1}{W^{\mathrm{b}}_{\mathbf{p}}} \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$

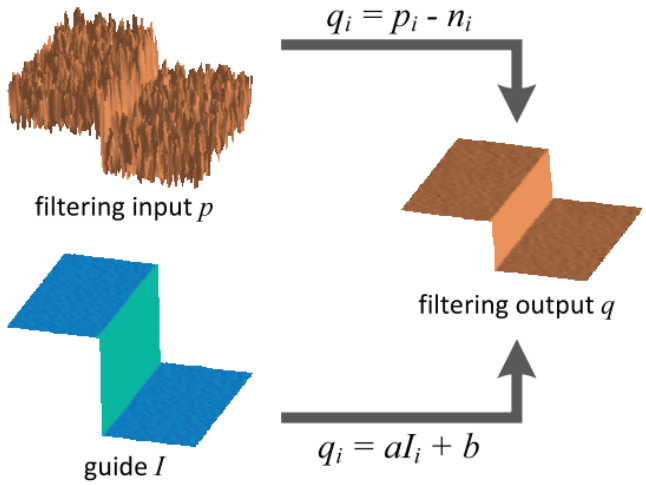
$$\text{with} \quad W^{\mathrm{b}}_{\mathbf{p}} = \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)$$

Carlo Tomasi, Roberto Manduchi, Bilateral Filtering for Gray and Color Images, ICCV, 1998.

BF as Guided Image Filters



Bilateral filters



Guided filters

```
B = imguidedfilter(A,G);
```

Kaiming He, Jian Sun, Xiaou Tang, Guided Image Filtering. PAMI 2013