Intelligent Multimedia Systems

Master AI, 2012, Lecture 3

Lecturers: Theo Gevers

Lab: Intelligent Systems Lab Amsterdam (ISLA)

Email: th.gevers@uva.nl

http://staff.science.uva.nl/~gevers

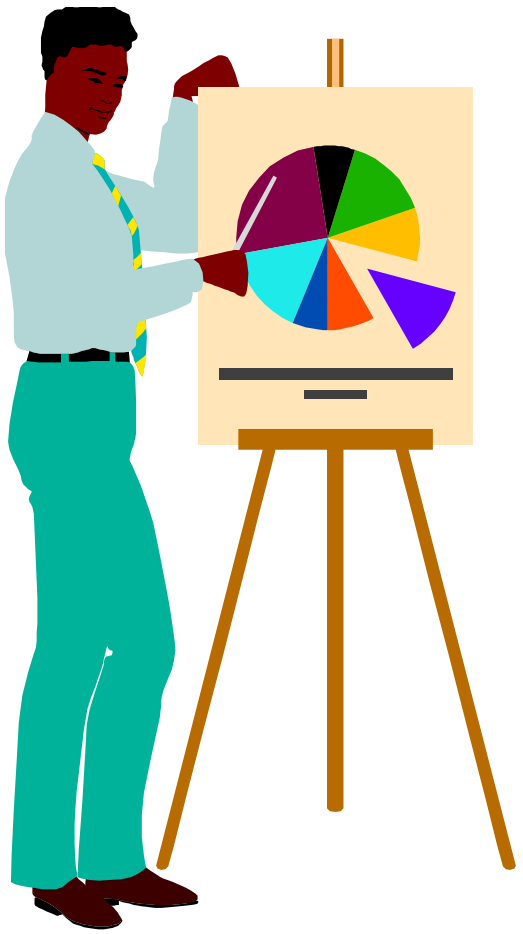
Lectures

- 29-10-2012, Monday, 15:00-17:00, Science Park A1.04 - Introduction
- 05-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Image and Video Formation
- 12-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Color Invariance and Image Processing
- 19-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Feature Extraction and Object Recognition
- 26-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Learning and Tracking
- 03-12-2011, Monday, 15:00-17:00, Science Park A1.04 - Visual Attention and Affective Computing
- 10-12-2011, Monday, 15:00-17:00, Science Park A1.04 - Human Behavior Analysis
- 18-12-2011, Tuesday, 15:00-18:00, Science Park, C1.10 - Examination

Class today

Color Invariance

Image Processing



Mobile robots

NASA's Mars Spirit Rover
http://en.wikipedia.org/wiki/Spirit_rover

http://www.robocup2013.org/

Saxena et al. 2008
STAIR at Stanford

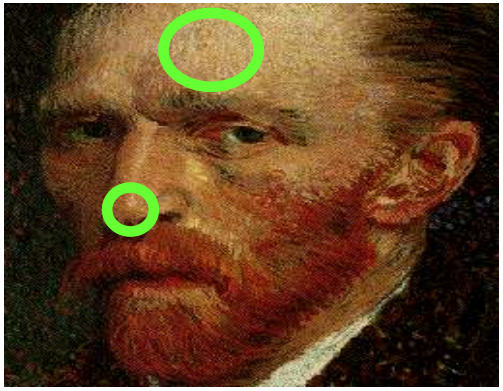
Reflection Model

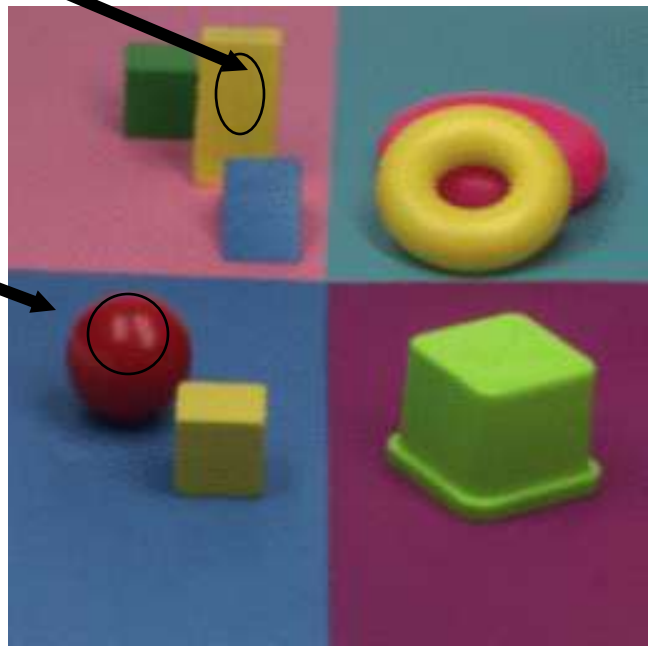
Reflectance model [Shafer]

body = $m_b(\vec{n},\vec{s})\int_\lambda f_C(\lambda)e(\lambda)c_b(\lambda)d\lambda$

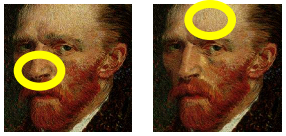
surface = $m_s(\vec{n},\vec{s},\vec{v})\int_\lambda f_C(\lambda)e(\lambda)c_s(\lambda)d\lambda$

for {R,G,B} giving an R-, B-, G-sensor response





Reflection Model



$$C = m_b(\vec{n},\vec{s})\int_{\lambda} e(\lambda)c_b(\lambda)\,f_C(\lambda)d\lambda + m_s(\vec{n},\vec{s},\vec{v})\int_{\lambda} e(\lambda)c_s(\lambda)\,f_C(\lambda)d\lambda$$

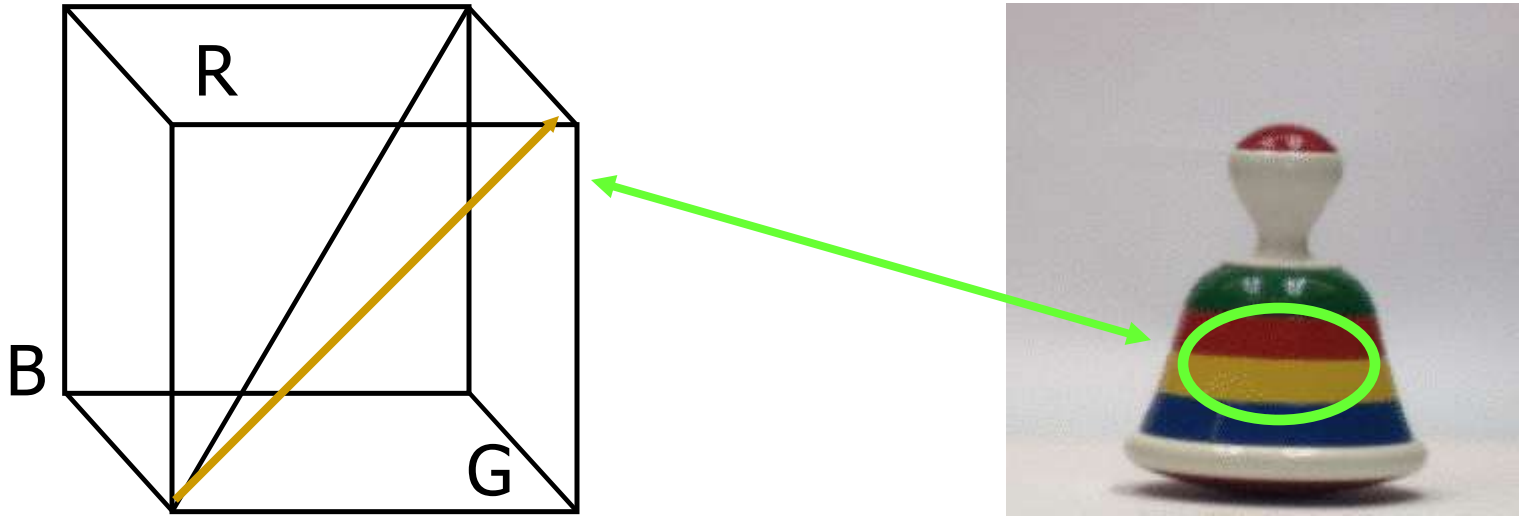
$c_b(\lambda)$	surface albedo	scene & viewpoint invariant
$e(\lambda)$	illumination	scene dependent
$\vec{n}$	object surface normal	object shape variant
$\vec{s}$	illumination direction	scene dependent
$\vec{v}$	viewer's direction	viewpoint variant
$f_C(\lambda)$	sensor sensitivity	scene dependent

body reflectance in RGB - space

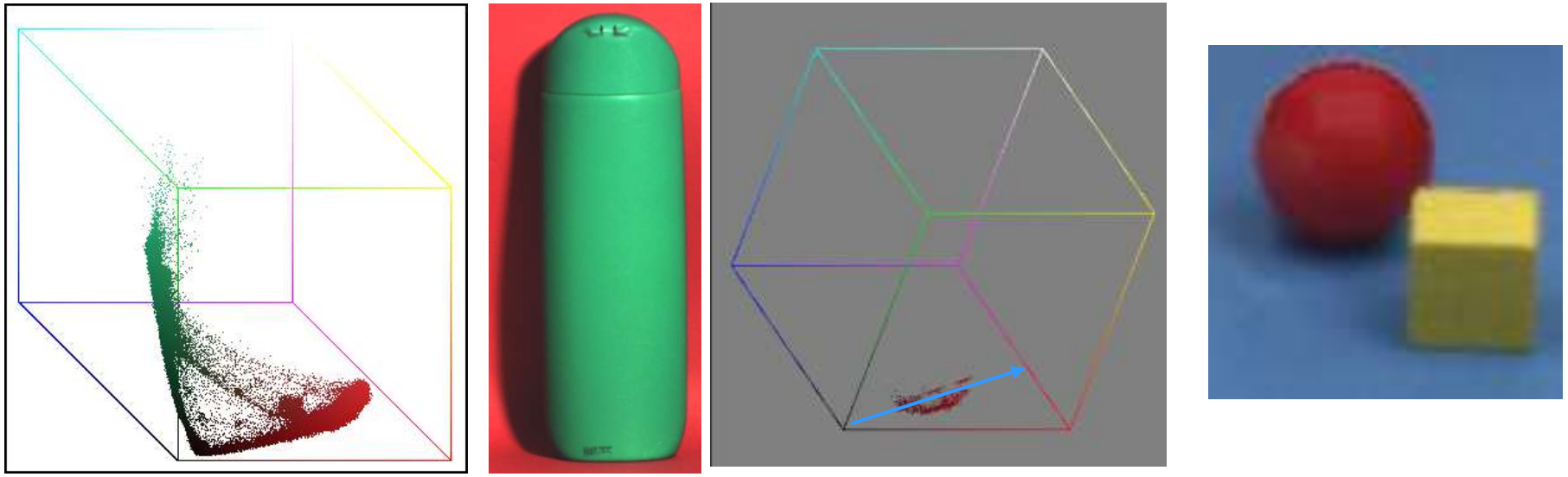
Consider the body reflection term:

$$C = m_b(\vec{n}, \vec{s}) \int_{\lambda} f_C(\lambda) e(\lambda) c_b(\lambda) d\lambda$$

for example the geometric term is Lambertian i.e. $m_b(\vec{n}, \vec{s}) = \vec{n} \cdot \vec{s}$



Body Reflectance in RGB – Matte Surfaces



An object with its representation in RGB-colour space.

Reflectance under White Light

Dichromatic reflection:

$$C = m_b(\vec{n},\vec{s})\int_{\lambda} f_C(\lambda)e(\lambda)c_b(\lambda)d\lambda + m_s(\vec{n},\vec{s},\vec{v})\int_{\lambda} f_C(\lambda)e(\lambda)c_s(\lambda)d\lambda$$

Considering dichromatic reflectance and white illumination, then $e(\lambda) = e$ and $c_s(\lambda) = c_s$. Then, for $C_W = \{R,G,B\}$ giving the red, green and blue sensor response under white light. This gives:

$$C_W = em_b(\vec{n},\vec{s})k_c + em_s(\vec{n},\vec{s},\vec{v})c_s f$$

with $k_C = \int f_C(\lambda)c_b(\lambda)d\lambda$

rgb – Photometric Invariance: Proof

Consider the body reflection term:

$$C_b = em_b(\vec{n}, \vec{s})k_C$$

For $C_b = \{R, G, B\}$ giving the red, green and blue sensor response under white light. Further, with $k_C = \int f_C(\lambda)c_b(\lambda)d\lambda$

$$r(R_b, G_b, B_b) = \frac{em_b(\vec{n}, \vec{s})k_R}{em_b(\vec{n}, \vec{s})(k_R + k_G + k_B)} = \frac{k_R}{(k_R + k_G + k_B)}$$

$$g(R_b, G_b, B_b) = \frac{em_b(\vec{n}, \vec{s})k_G}{em_b(\vec{n}, \vec{s})(k_R + k_G + k_B)} = \frac{k_G}{(k_R + k_G + k_B)}$$

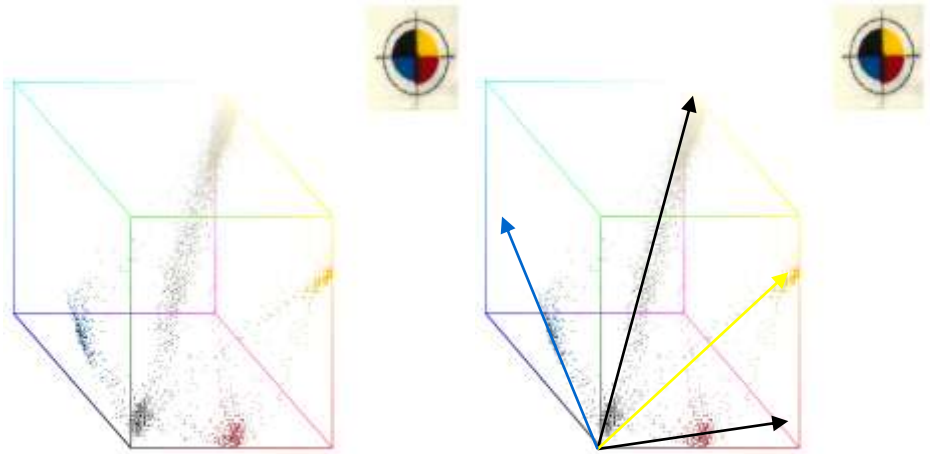
$$b(R_b, G_b, B_b) = \frac{em_b(\vec{n}, \vec{s})k_B}{em_b(\vec{n}, \vec{s})(k_R + k_G + k_B)} = \frac{k_B}{(k_R + k_G + k_B)}$$

Matte Objects

$$\frac{R}{G} = \frac{em_b(\vec{n},\vec{s})k_R}{em_b(\vec{n},\vec{s})k_G} = \frac{k_R}{k_G}$$

$$\frac{R}{B} = \frac{em_b(\vec{n},\vec{s})k_R}{em_b(\vec{n},\vec{s})k_B} = \frac{k_R}{k_B},$$

$$\frac{G}{B} = \frac{em_b(\vec{n},\vec{s})k_G}{em_b(\vec{n},\vec{s})k_B} = \frac{k_G}{k_B}$$



Then

$$C = \frac{\sum R^p G^q B^r}{\sum R^s G^t B^u}, \text{ where } p+q+r = s+t+u$$

$$e.g.\ \{\ \frac{R+G}{B}, \frac{3G+B}{R+B}, \frac{R+G+B}{G+B}, \frac{4R+G}{R+G+B}, ...,\}$$

$$e.g.\ \{\ \frac{R^2+BG}{B^2}, \frac{3GR+GB}{R^2+B^2}, \frac{R^2+G^2+B^2}{G^2+B^2}, \frac{4RG+GB}{GR+BG+B^2}, ...,\}$$

etc.

Colour Invariance of Matte Objects

First order color invariants :

$$\{\frac{R+G}{B}, \frac{3G+B}{R+B}, \frac{R+G+B}{G+B}, \frac{4R+G}{R+G+B}, \frac{R}{R+G+B}, ..., \}$$

Second order color invariants :

$$\{\frac{R^2+BG}{B^2}, \frac{3GR+GB}{R^2+B^2}, \frac{R^2+G^2+B^2}{G^2+B^2}, \frac{4RG+GB}{GR+BG+B^2}, \frac{R^2}{R^2+G^2+B^2}, ..., \}$$

Third order color invariants :

$$\{\frac{R^3+RBG}{RB^2}, \frac{3GR^2+GB^2}{R^3+B^3}, \frac{R^3+G^3+B^3}{G^3+B^3}, \frac{4R^2G+GB^2}{GR^2+BG^2+B^3}, \frac{R^3}{R^3+G^3+B^3}, ..., \}$$

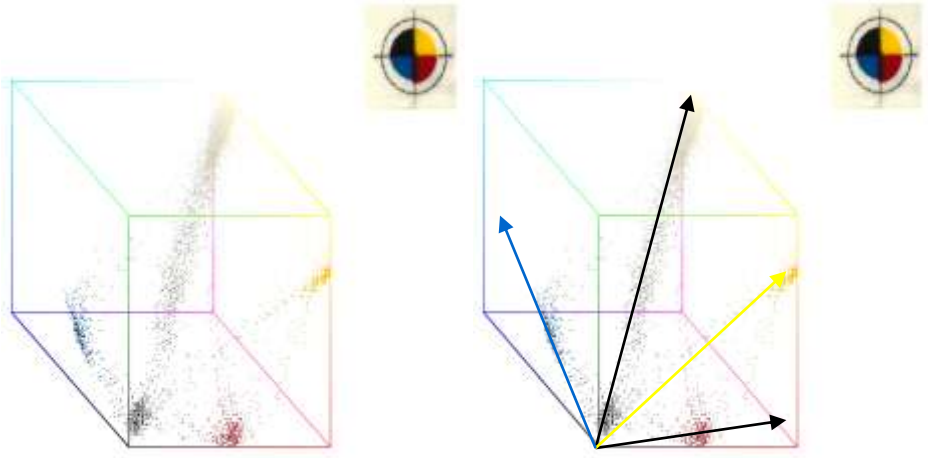
etc, etc,..., etc

Colour invariance / c1c2c3 space

$$c_1(R,G,B) = \arctan \frac{R}{\max\{G,B\}}$$

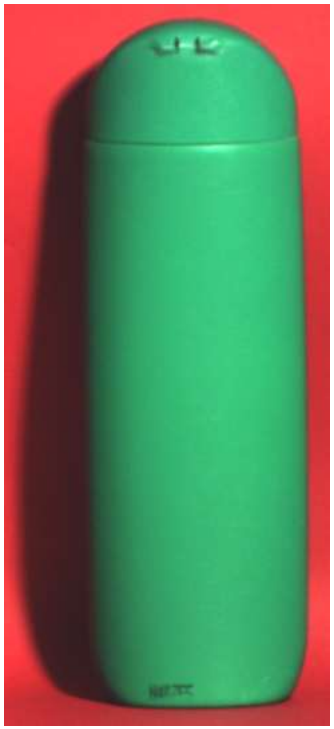
$$c_2(R,G,B) = \arctan \frac{G}{\max\{R,B\}}$$

$$c_3(R,G,B) = \arctan \frac{B}{\max\{R,G\}}$$

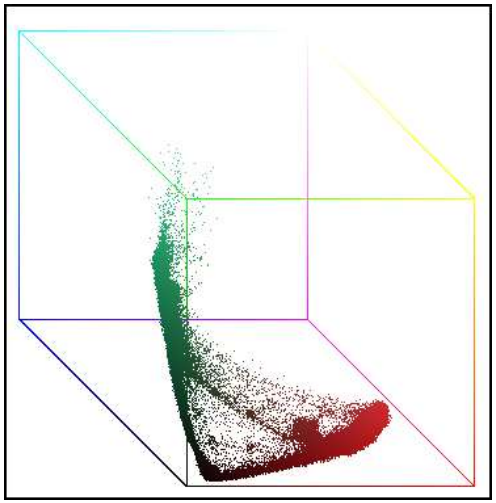


$$c_1(R_b,G_b,B_b) = \arctan\left(\frac{em_b(\vec{n},\vec{s})k_R}{\max\{em_b(\vec{n},\vec{s})k_G, em_b(\vec{n},\vec{s})k_B\}}\right) = \arctan\left(\frac{k_R}{\max\{k_G,k_B\}}\right)$$

Colour invariance / c1c2c3 space

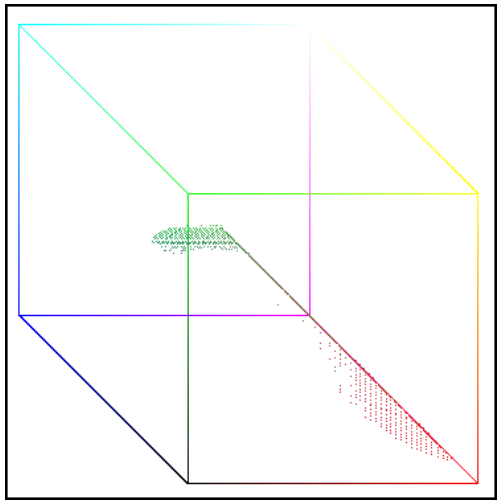


Original RGB image



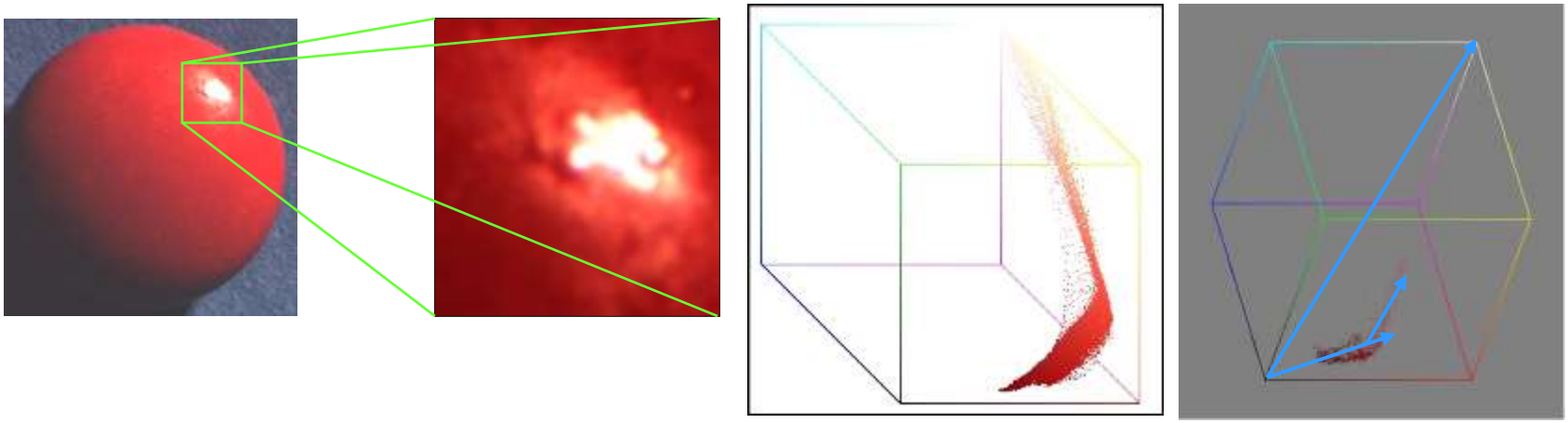
3D plot of RGB image

c1c2c3 image



3D plot of c1c2c3 image

Body Reflectance in RGB - space



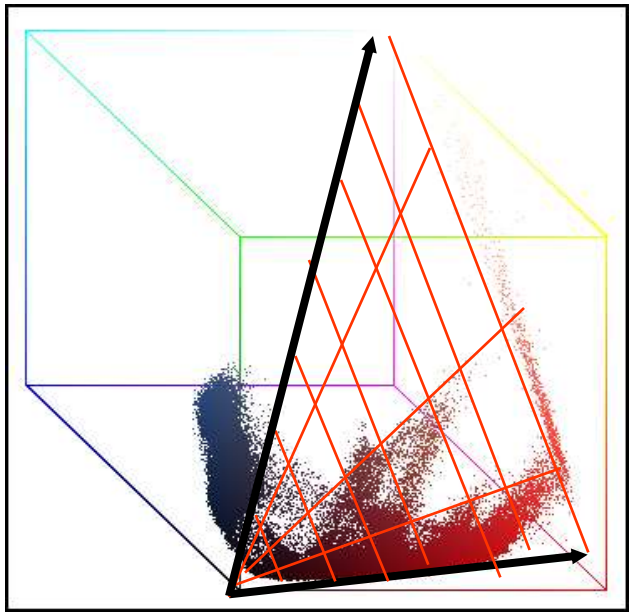
Consider the surface reflection term:

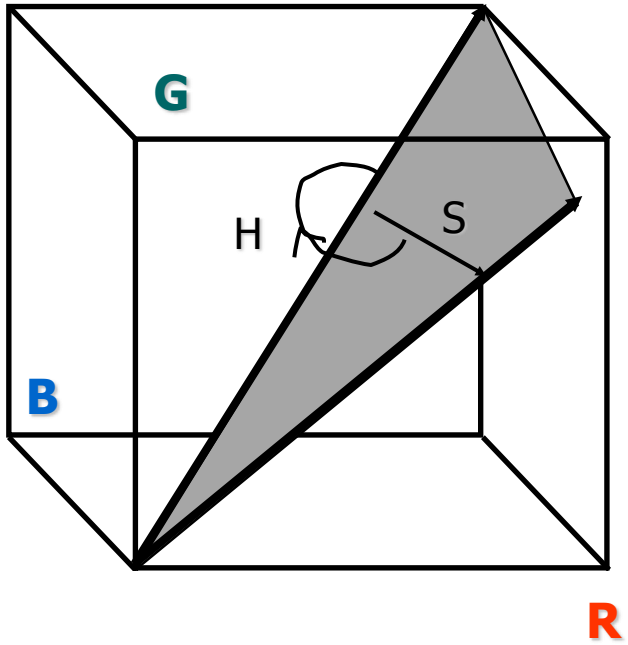
$$m_s(\vec{n},\vec{s},\vec{v})\int_{\lambda} f_C(\lambda)e(\lambda)c_s(\lambda)d\lambda$$

where the geometric term is for example the phong model $\cos^n(\alpha)$ where α only depends on $\vec{n},\vec{s}$ and $\vec{v}$.

Colour invariance

Hue is viewpoint invariant





Hue: $H(R,G,B) = \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right)$

Colour invariance

Shiny objects: l1l2l3l - space

$$l1(R,G,B) = \frac{(R-G)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

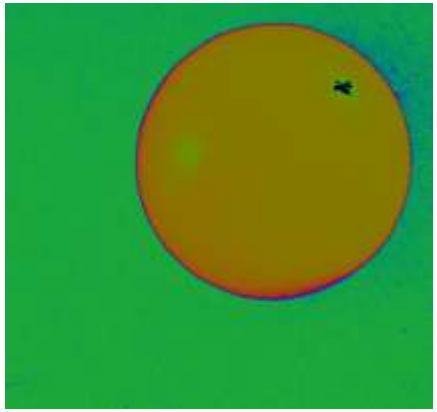
$$l2(R,G,B) = \frac{(R-B)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

$$l3(R,G,B) = \frac{(G-B)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

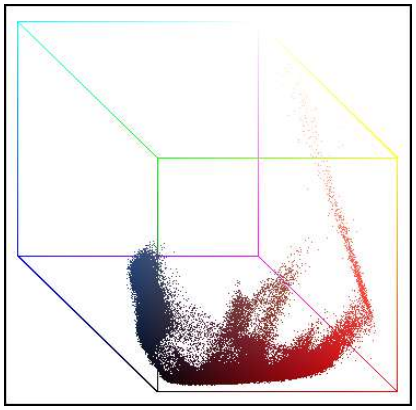
Colour invariance
Shiny objects: l1l2l3l - space

Original colour image

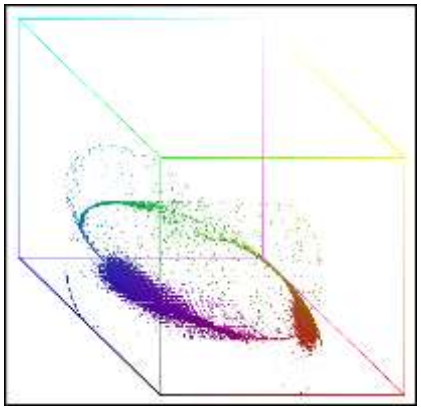
L1l2l3 transformation



L1l2l3 image



3D plot of colour image



3D plot of l1l2l3 image

Colour invariance - Summary

	shadows	shading	highlights	ill. intensity	ill. Colour
I	-	-	-	-	-
R,G,B	-	-	-	-	-
r,g,b	+	+	-	+	-
c1,c2,c3	+	+	-	+	-
Hue	+	+	+	+	-
l1,l2,l3	+	+	+	+	-
m1m2m3	+	+	-	+	+

- no invariance

+ invariance

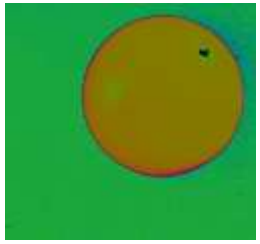
Summary

- Reflection models help to understand the image formation process.

$$m_b(\vec{n},\vec{s})\int_{\lambda} f_C(\lambda)e(\lambda)c_b(\lambda)d\lambda$$

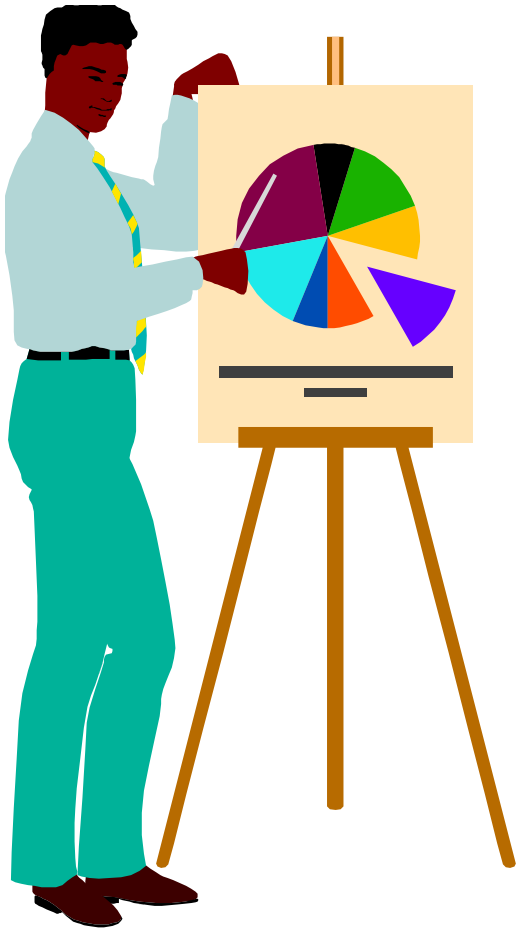
- Color invariance at the pixel. White light source!

- Be aware of instabilities



Color Invariance

Color constancy



Color Constancy

- Model images assuming Lambertian reflectance:

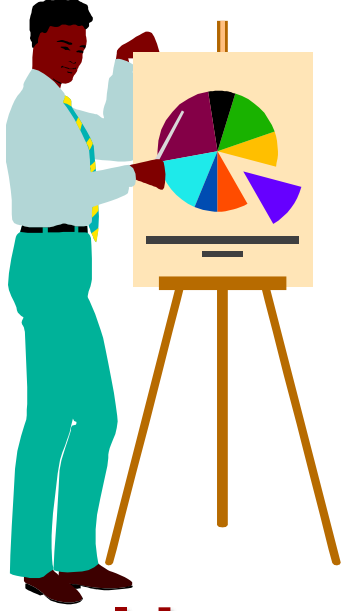
$$\mathbf{f}(\mathbf{x}) = \int_{\omega} e(\lambda)\rho_k(\lambda)s(\mathbf{x},\lambda)d\lambda$$

- Image transformation using von Kries model[1]:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \gamma \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

(1) - J. von Kries. “Influence of adaptation on the effects produced by luminous stimuli.” In D. MacAdam, editor, *Sources of Color Vision*, pages 109–119. MIT Press, 1970.

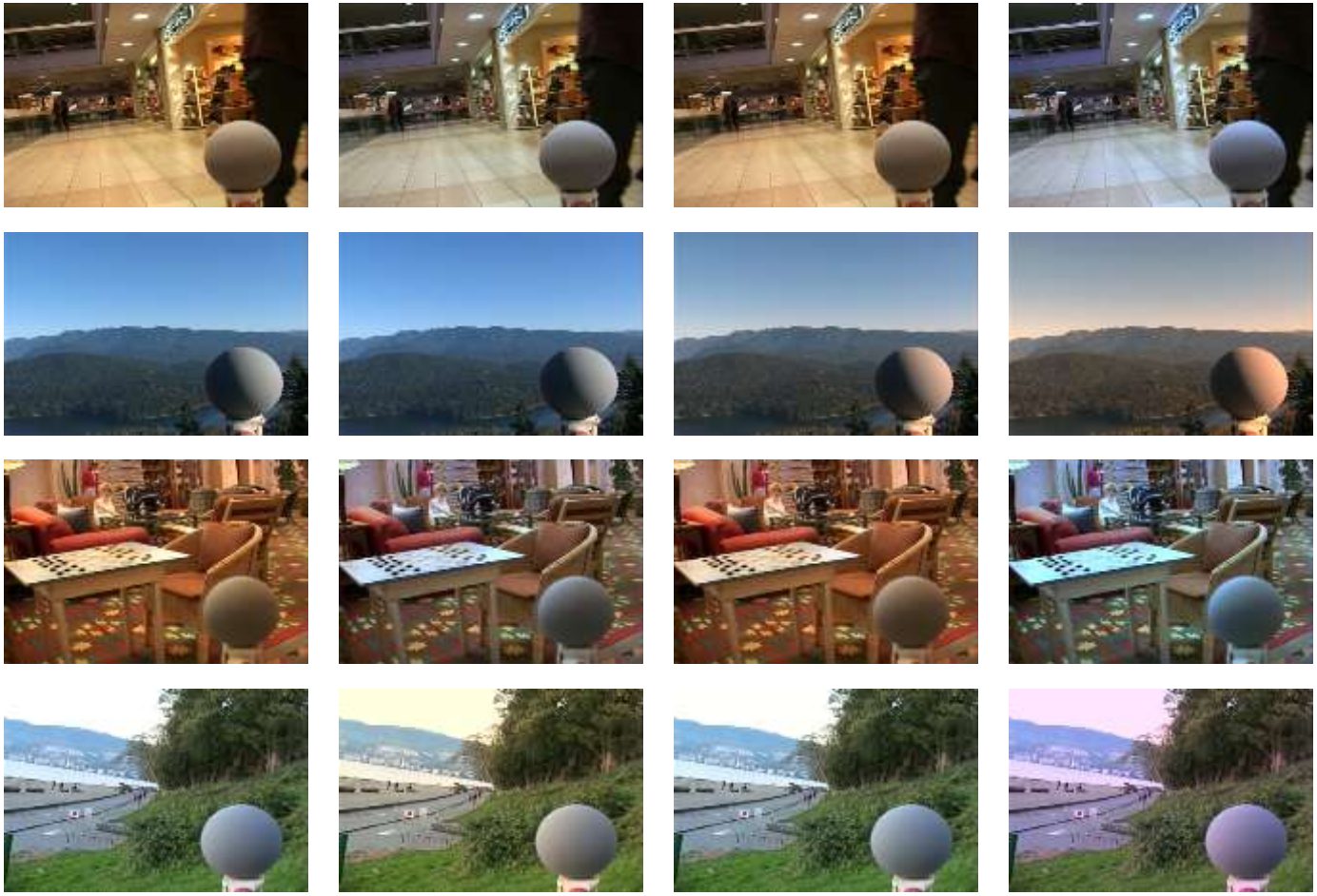
Overview



- **Color constancy: white patch and grey-world**
- Color constancy: grey-edge
- Color constancy: natural image statistics
- Color constancy: Derivative-based Gamut Mapping

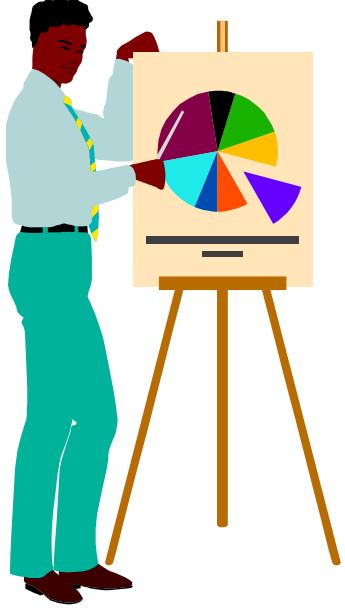
White Patch and Grey-world Examples

Original image **Ideal correction** **White-Patch** **Grey-world**



Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

Overview



- Color constancy: white patch and grey-world
- **Color constancy: grey-edge**
- Color constancy: natural image statistics
- Color constancy: Derivative-based Gamut Mapping

Assumptions

color constancy : the ability to recognize colors of objects invariant of the color of the light source.

Grey world hypothesis : the average reflectance in a scene is grey.

White patch hypothesis : the highest value in the image is white.

Shades of Grey hypothesis : the n-Minkowsky norm based average of a scene is achromatic.

- unifies Grey-World and White Patch : $e^p \approx \sqrt[p]{\int \left|\mathbf{f}(\mathbf{x})\right|^p d\mathbf{x}}$

Color Constancy: Experiment

- real-world data set (F. Ciurea and B. Funt : Vision Lab - Simon Fraser)

Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

Color Constancy: Experiment

- Angular error: $\cos^{-1}(\hat{\mathbf{e}}_l \cdot \hat{\mathbf{e}}_e)$

- Three data sets:
  - Laboratory setting(4)
  - Real-world images(5)

(4) – Images from “A data set for color research”, by K. Barnard et al. in CRA 27(3), 2002.

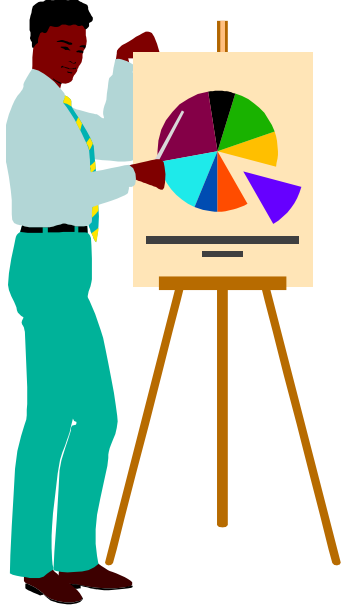
Color Constancy: Experiment

• real-world data set (F. Ciurea and B. Funt : Vision Lab - Simon Fraser)

	error
Grey-World	8.2
White-Patch	7.1
General Grey-World	6.2
Grey-Edge	5.7
2nd order Grey-Edge	5.8

Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

Overview



- Color constancy: white patch and grey-world
- Color constancy: grey-edge
- **Color constancy: natural image statistics**
- Color constancy: Derivative-based Gamut Mapping

Natural Image Statistics: Weibull

- Why use Natural Image Statistics to solve Color Constancy?
- White-Patch

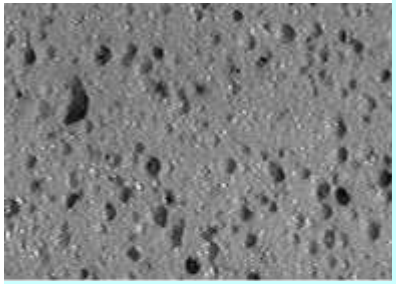


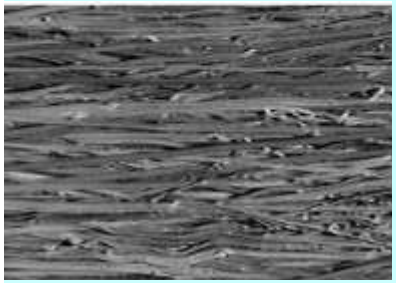

- Grey-world




Natural Image Statistics: Weibull

Geusebroek 2005 IJCV





When you go to an arbitrary place and take a picture: fair chance on Weibull statistics

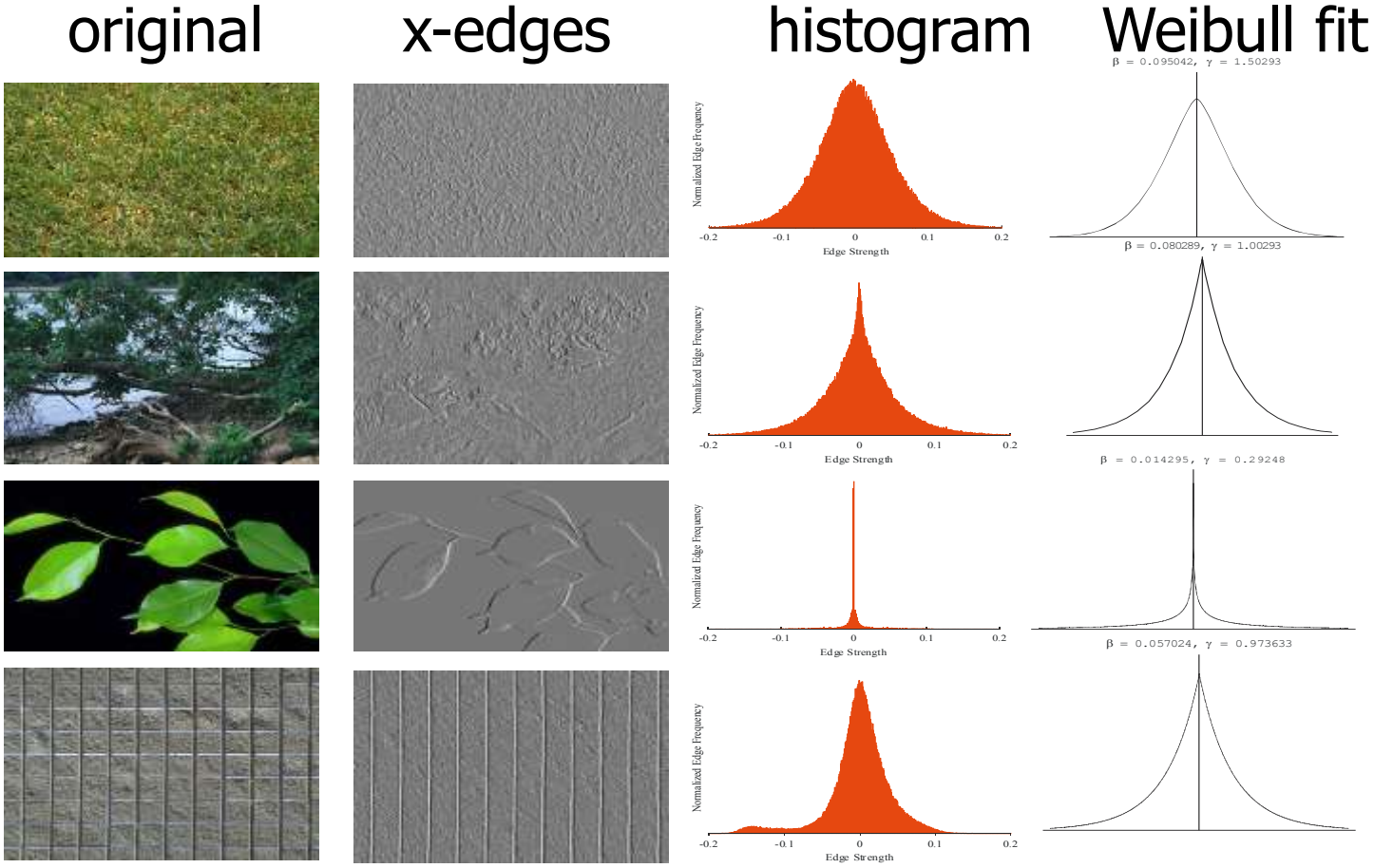
A three parameter family:

μ: average	normalize out
β: variation	contrast
γ: power	grains

$$p(f) = c\, e^{\left(\frac{f-\mu}{\beta}\right)^{\gamma}}$$

Natural Image Statistics: Weibull

Geusebroek 2005 IJCV



J van Gemert 2006

Weibull

Gamma app. resolution

1.5

b=0.012192, g=1.81543

b=0.095042, g=1.50293

b=0.251007, g=1.354492

1

b=0.01023, g=0.989258

b=0.100916, g=1.02832

b=0.178397, g=1.043945

0.5

b=0.010076, g=0.510742

b=0.101547, g=0.641602

b=0.182459, g=0.647461

0.01 0.1 0.2

Beta app. contrast

Natural Image Statistics

2nd order Grey-edge

White-Patch

Grey-world

Grey-edge

2nd order Grey-edge

Gamma app. resolution

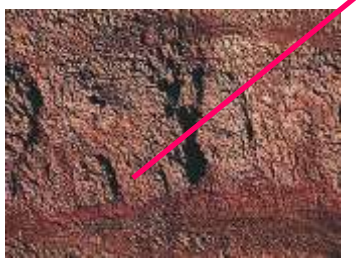
1.5

1

0.5


b=0.012192, g=1.81543


b=0.095042, g=1.50293


b=0.251007, g=1.354492


b=0.01023, g=0.989258


b=0.100916, g=1.02832


b=0.178397, g=1.043945


b=0.010076, g=0.510742


b=0.101547, g=0.641602


b=0.182459, g=0.647461

0.01

0.1

0.2

Beta app. Contrast

Natural Image Statistics

- Geusebroek and Smeulders (2005) – Weibulls

- Examples:

Natural Image Statistics

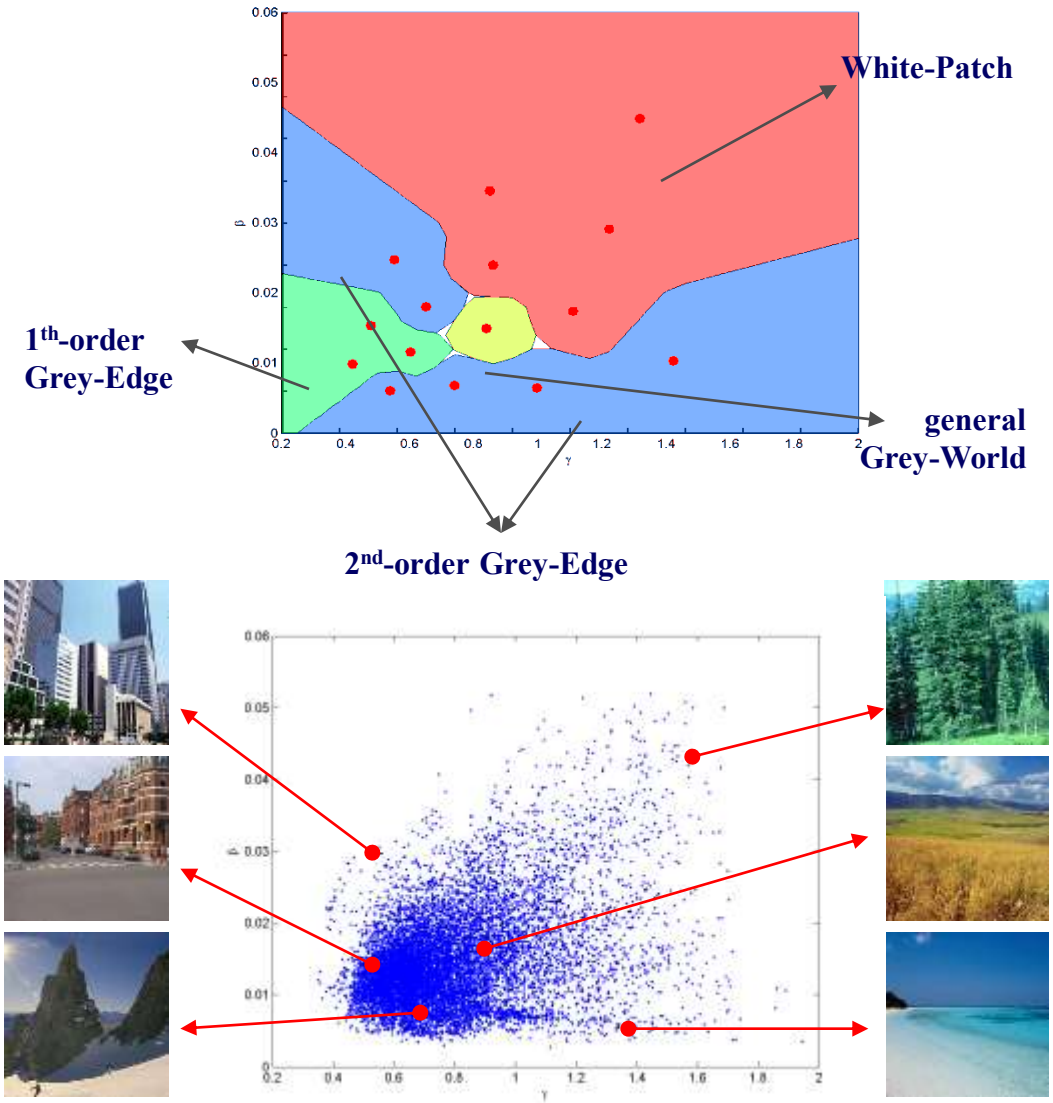
- Distribution of edge responses follows Weibull distribution
- Parameters of Weibull distribution are indicative for statistics of a (natural) scene:
  - β – Contrast of the image. A higher value indicates more contrast
  - γ – Grain size. A Higher value indicates more fine textures

Color Constancy – Selection

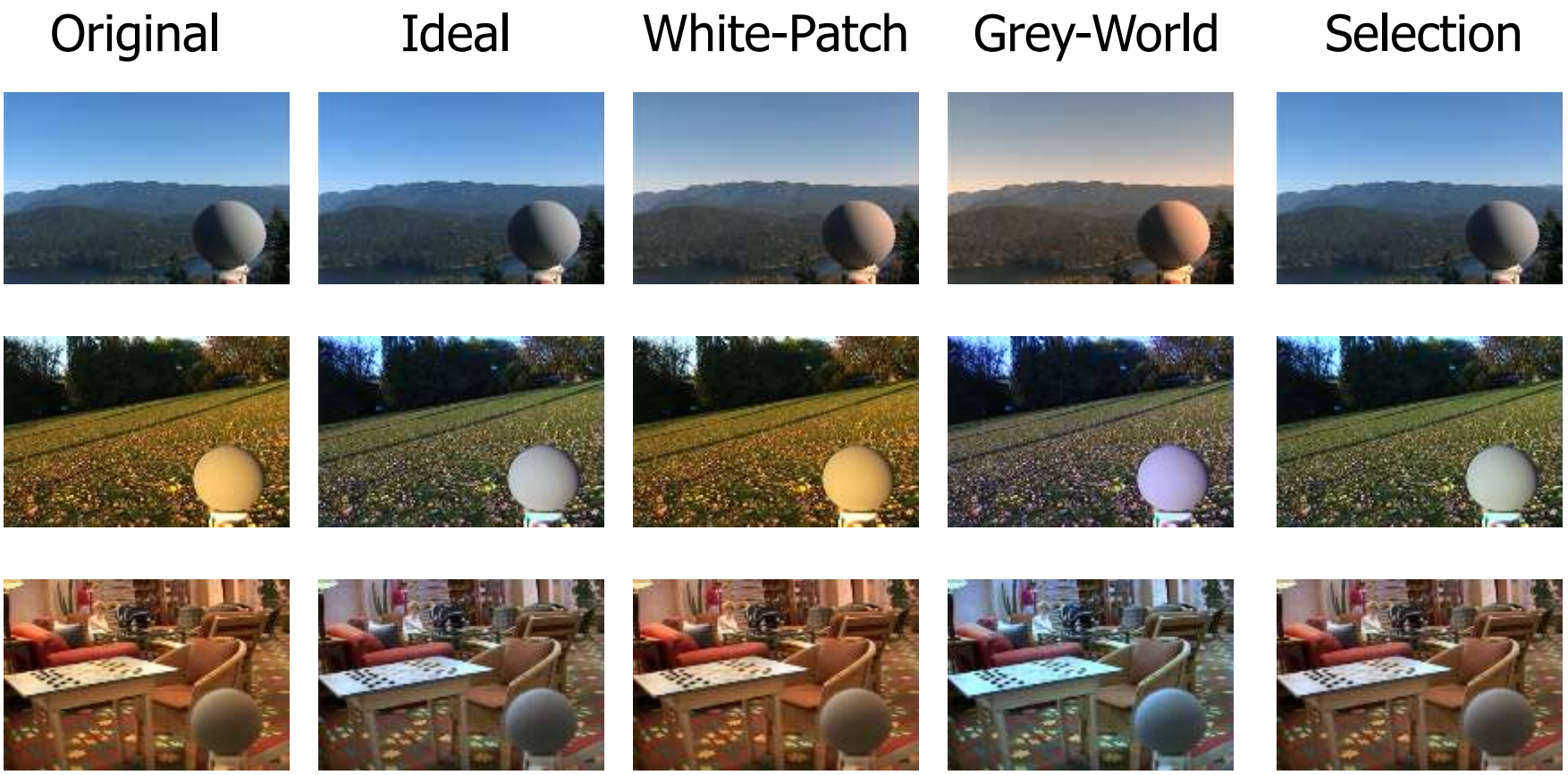
Postsupervised Prototype

Classifier:

- Compute Weibull-parameters for all images
- Cluster weibull-parameters using *k*-means
- Label cluster centers according to the minimum mean angular error
- Build 1-NN Classifier on these cluster centers



Experiments – Results



Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

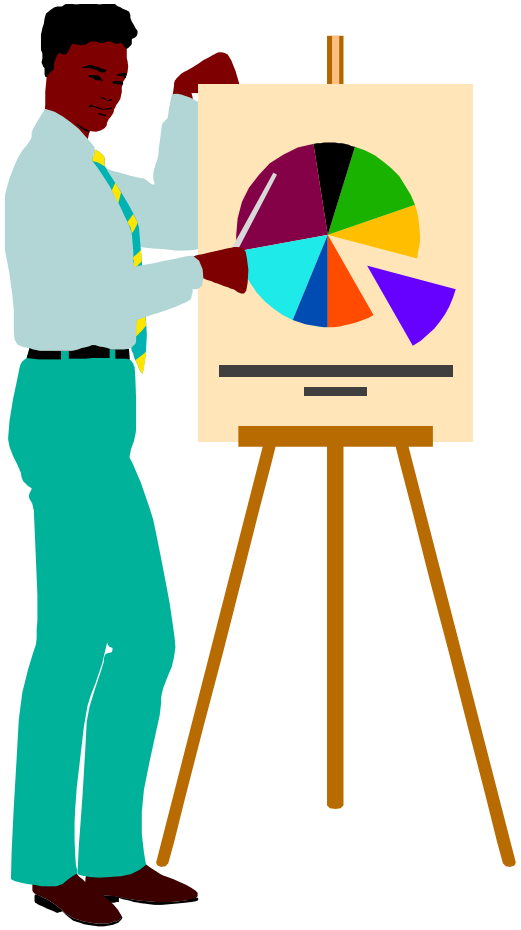
Experiments – Results

Original	Ideal	White-Patch	Grey-World	Selection
				
				
				

Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

Color constancy

- cross ratios



Color Ratio's

Assuming body reflection

$$C = m_b(\vec{n}, \vec{s}) \int_{\lambda} f_C(\lambda) e(\lambda) c_b(\lambda) d\lambda \text{ for } C = \{R, G, B\}$$

And narrow-band filters: $f_C(\lambda) = \delta(\lambda - \lambda_C)$

Then: $C = m_b(\vec{n}, \vec{s}) e(\lambda_C) c_b(\lambda_C) \text{ for } C = \{R, G, B\}$

Therefore, we have:

$$R = m_b(\vec{n}, \vec{s}) e(\lambda_R) c_b(\lambda_R)$$

$$G = m_b(\vec{n}, \vec{s}) e(\lambda_G) c_b(\lambda_G)$$

$$B = m_b(\vec{n}, \vec{s}) e(\lambda_B) c_b(\lambda_B)$$

Color Ratio's

- Colour ratio of two neighboring positions:

$$\frac{R^{\mathbf{x}_1}}{R^{\mathbf{x}_2}} = \frac{m_b(\vec{n},\vec{s})^{\mathbf{x}_1} e(\lambda_R)^{\mathbf{x}_1} c_b(\lambda_R)^{\mathbf{x}_1}}{m_b(\vec{n},\vec{s})^{\mathbf{x}_2} e(\lambda_R)^{\mathbf{x}_2} c_b(\lambda_R)^{\mathbf{x}_2}} = \frac{m_b(\vec{n},\vec{s})^{\mathbf{x}_1} c_b(\lambda_R)^{\mathbf{x}_1}}{m_b(\vec{n},\vec{s})^{\mathbf{x}_2} c_b(\lambda_R)^{\mathbf{x}_2}} = \frac{c_b(\lambda_R)^{\mathbf{x}_1}}{c_b(\lambda_R)^{\mathbf{x}_2}}$$

- Colour ratio with multiplication:

$$\frac{R^{\mathbf{x}_1} G^{\mathbf{x}_2}}{R^{\mathbf{x}_2} G^{\mathbf{x}_1}} = \frac{m_b(\vec{n},\vec{s})^{\mathbf{x}_1} e(\lambda_R)^{\mathbf{x}_1} c_b(\lambda_R)^{\mathbf{x}_1} m_b(\vec{n},\vec{s})^{\mathbf{x}_2} e(\lambda_G)^{\mathbf{x}_2} c_b(\lambda_G)^{\mathbf{x}_2}}{m_b(\vec{n},\vec{s})^{\mathbf{x}_2} e(\lambda_R)^{\mathbf{x}_2} c_b(\lambda_R)^{\mathbf{x}_2} m_b(\vec{n},\vec{s})^{\mathbf{x}_1} e(\lambda_G)^{\mathbf{x}_1} c_b(\lambda_G)^{\mathbf{x}_1}} =$$

$$\frac{m_b(\vec{n},\vec{s})^{\mathbf{x}_1} c_b(\lambda_R)^{\mathbf{x}_1} m_b(\vec{n},\vec{s})^{\mathbf{x}_2} c_b(\lambda_G)^{\mathbf{x}_2}}{m_b(\vec{n},\vec{s})^{\mathbf{x}_2} c_b(\lambda_R)^{\mathbf{x}_2} m_b(\vec{n},\vec{s})^{\mathbf{x}_1} c_b(\lambda_G)^{\mathbf{x}_1}} = \frac{c_b(\lambda_R)^{\mathbf{x}_1} c_b(\lambda_G)^{\mathbf{x}_2}}{c_b(\lambda_R)^{\mathbf{x}_2} c_b(\lambda_G)^{\mathbf{x}_1}}$$

Color Ratio's

$$m_1 = \frac{R^{x_1} G^{x_2}}{R^{x_2} G^{x_1}}, m_2 = \frac{R^{x_1} B^{x_2}}{R^{x_2} B^{x_1}}, m_3 = \frac{G^{x_1} B^{x_2}}{G^{x_2} B^{x_1}}$$

Taking the natural logarithm of both sides results for m_1 in :

$$\ln m_1 = \ln\left(\frac{R^{x_1} G^{x_2}}{R^{x_2} G^{x_1}}\right) = \ln R^{x_1} + \ln G^{x_2} - \ln R^{x_2} - \ln G^{x_2} =$$

$$\ln R^{x_1} + \ln G^{x_2} - (\ln R^{x_2} + \ln G^{x_2}) =$$

$$\ln\left(\frac{R^{x_1}}{G^{x_1}}\right) - \ln\left(\frac{R^{x_2}}{G^{x_2}}\right) = \ln\left(\frac{R}{G}\right)^{x_1} - \ln\left(\frac{R}{G}\right)^{x_2}$$

Overview

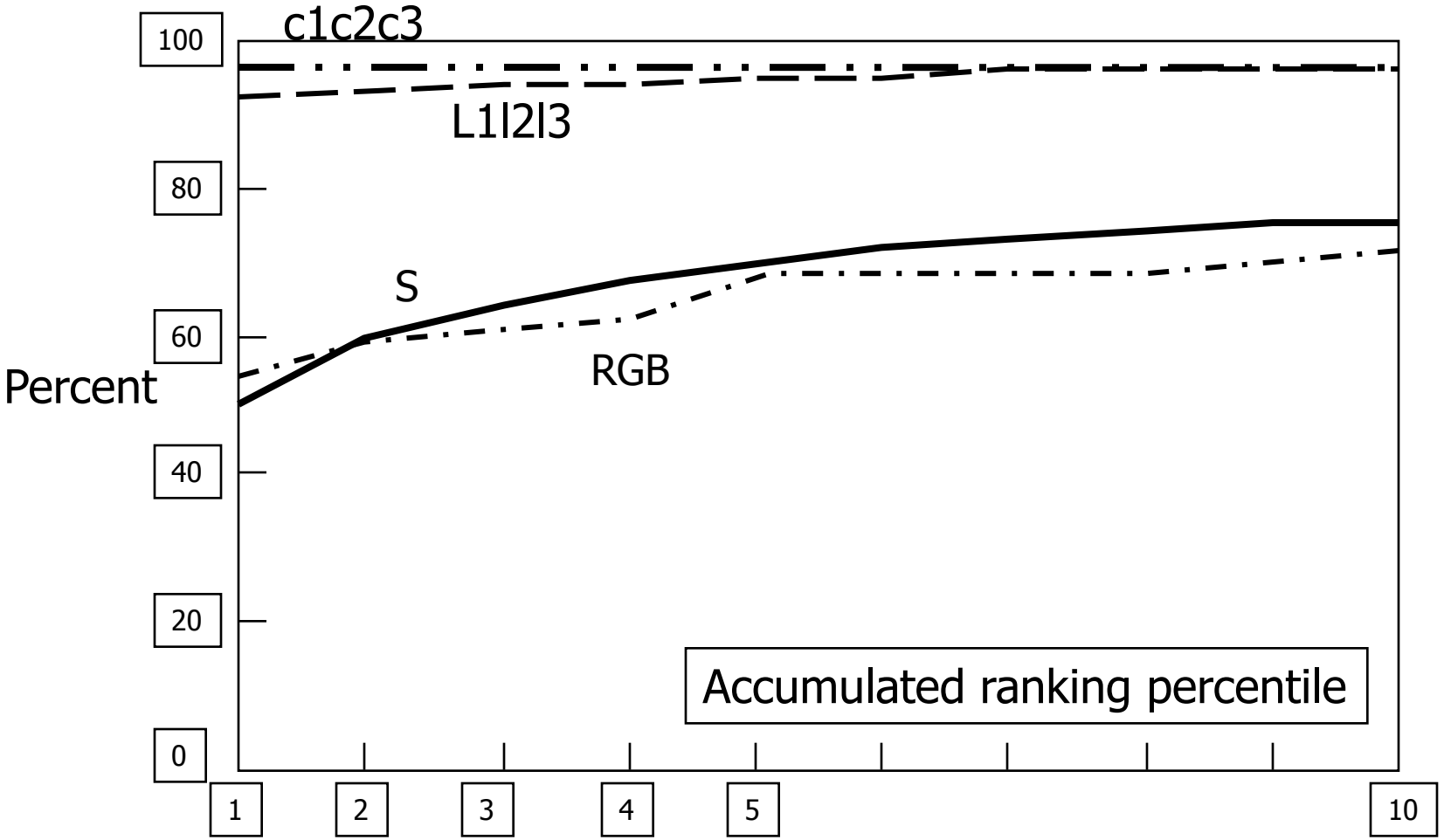
	shadows	shading	highlights	ill. intensity	ill. Colour
I	-	-	-	-	-
R,G,B	-	-	-	-	-
r,g,b	+	+	-	+	-
c1,c2,c3	+	+	-	+	-
Hue	+	+	+	+	-
l1,l2,l3	+	+	+	+	-
m1m2m3	+	+	-	+	+

- no invariance

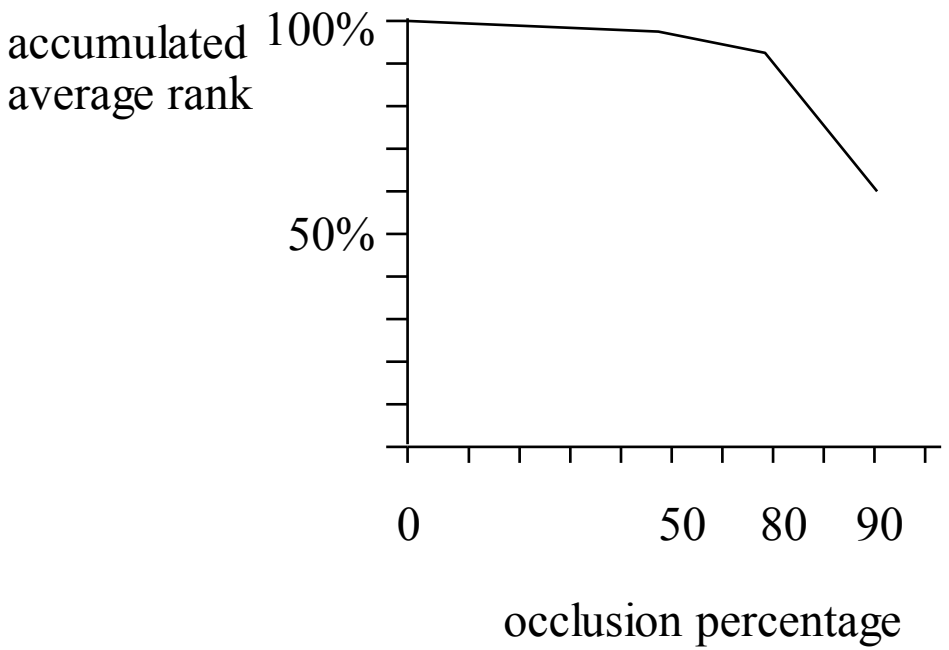
+ invariance

Experiments: Object Recognition

Experiments: Object Recognition

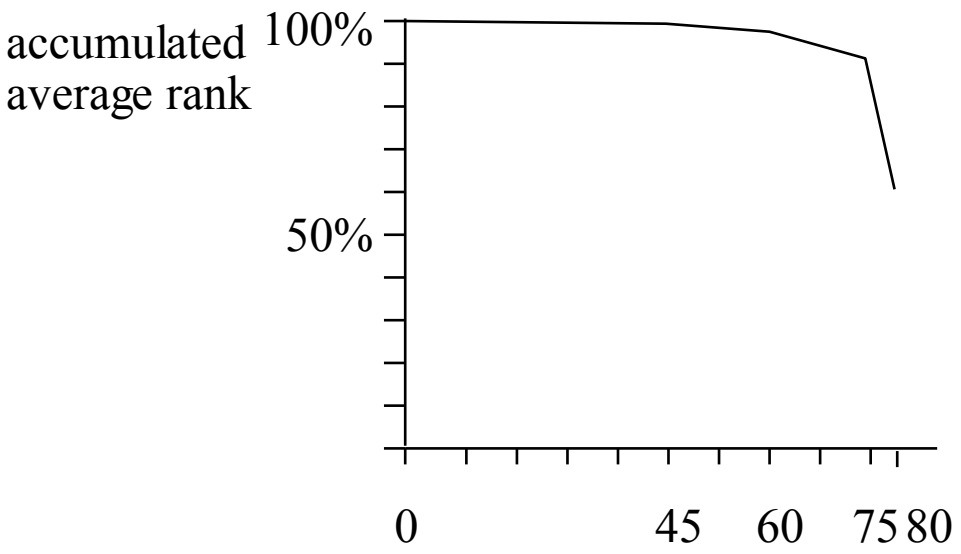


Experiments: Occlusion



Yes! Even 80% occluded.

Experiments: Viewpoint



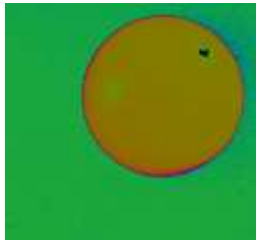
Yes! Even 75° out of sight.

Summary

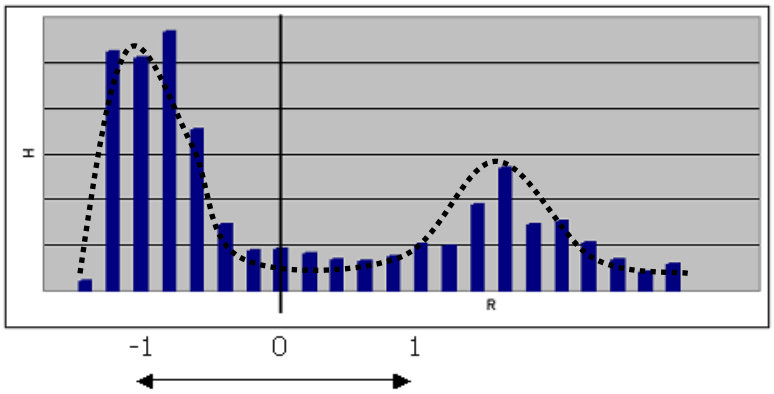
- Estimation of the light source

- Invariant color features

- Be aware of instabilities

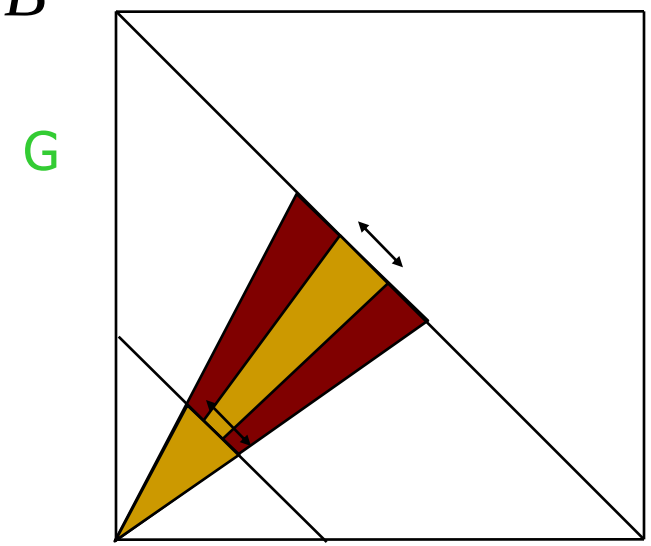


Kernel Density Estimation for Object Recognition



Color Invariants: Instabilities

$$r = \frac{R}{R+G+B}$$



$$I = (R+G+B)/3$$

Color Invariants: Instabilities

Measuring two quantities x and y to calculate their sum $x + y$

Then the highest probable value of $x + y = x_{\text{best}} + y_{\text{best}} + (\delta x + \delta y)$

Then the lowest probable value of $x + y = x_{\text{best}} + y_{\text{best}} - (\delta x + \delta y)$

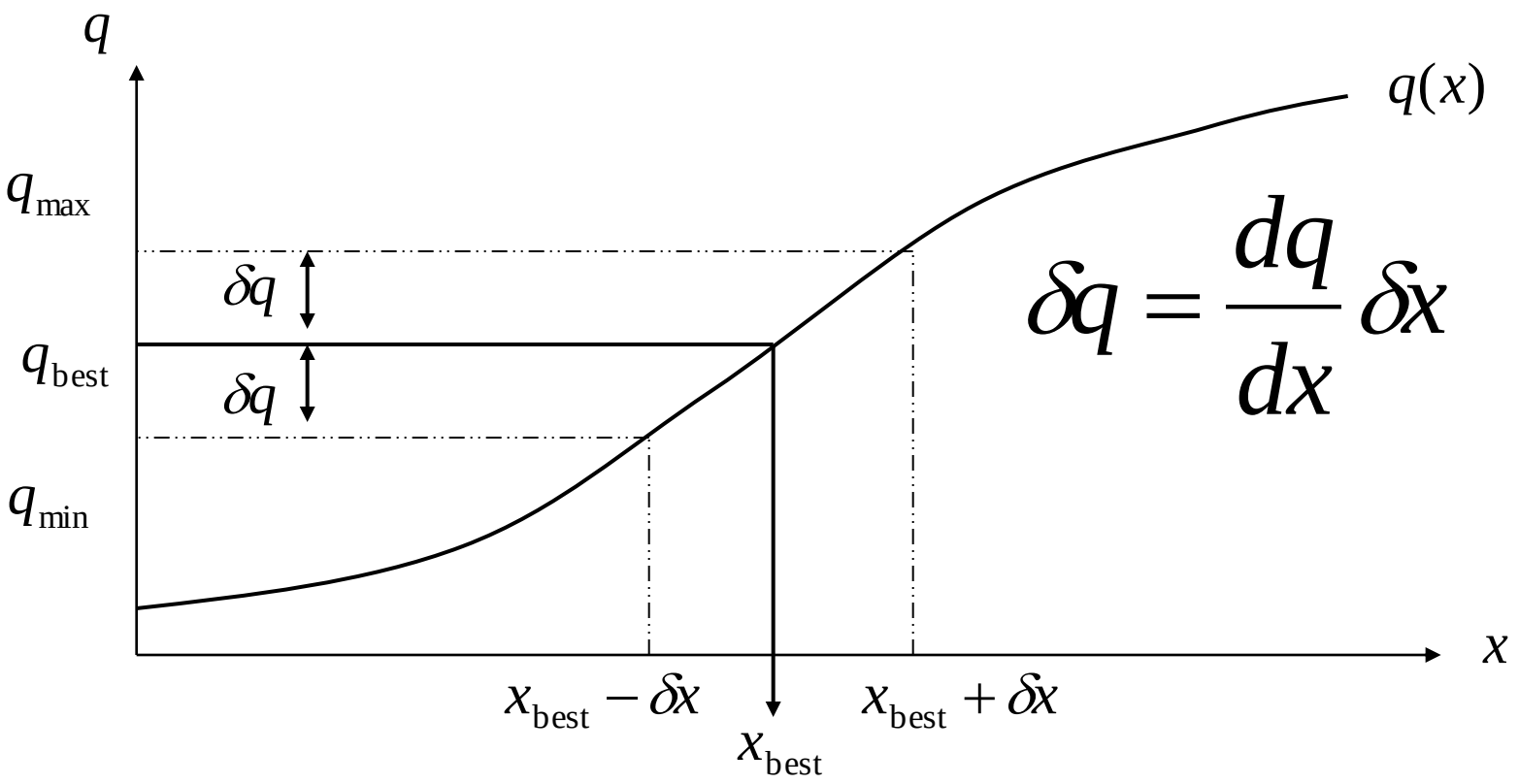
Where x_{best} is the best estimate of x and δx the uncertainty of x

$$q = x + \ldots + z - (u + \ldots + w)$$

$$\delta q \approx \delta x + \ldots + \delta z + \delta u + \ldots + \delta w$$

Color Invariants: Noise Propagation

$$\delta q = q(x_{\text{best}} + \delta x) - q(x_{\text{best}})$$



$$\delta q = \frac{dq}{dx}\delta x$$

Color Invariants: Noise Propagation

Suppose that $u,\ldots,w$ are measured with corresponding uncertainties $\sigma_u,\ldots,\sigma_w$ to compute function $q(u,\ldots,w)$.

The predicted uncertainty is defined by :

$$\sigma_q = \sqrt{\left(\frac{\partial q}{\partial u}\sigma_u\right)^2 + \ldots + \left(\frac{\partial q}{\partial w}\sigma_w\right)^2}$$

The uncertainty in q is never larger than the ordinary sum

$$\sigma_q \leq \left|\frac{\partial q}{\partial u}\right|\sigma_u + \ldots \left|\frac{\partial q}{\partial w}\right|\sigma_w$$

if and only if the uncertainties $\sigma_u,\ldots,\sigma_w$ are relatively small.

Color Invariants: Noise Propagation

Example: rgb

$$\text{Function } q(x,\ldots,z) \text{ then } \delta_q = \sqrt{\left(\frac{\partial q}{\partial x}\delta x\right)^2 + \ldots + \left(\frac{\partial q}{\partial z}\delta z\right)^2}$$

$$r(R,G,B) = \frac{R}{R+G+B}$$

Normalized color value

$$\delta_r = \frac{\sqrt{R^2(\delta_B^2+\delta_G^2)+(B+G)\delta_R^2}}{(B+G+R)^4}$$

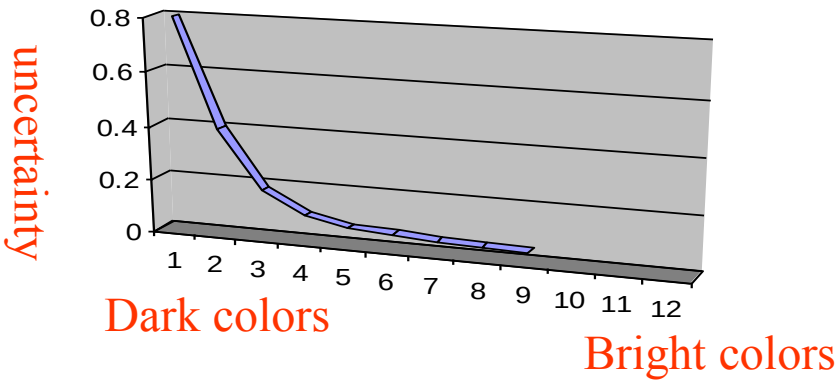
Predicted uncertainty of normalized color value

Measured values and predicted uncertainty

Color Invariants: Noise Propagation Example: Hue

$$H = \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right)$$

$$\sigma_H = \frac{1}{2}\sqrt{3}\left(\frac{-2BR\sigma_G^2 + R^2\sigma_B^2 + \sigma_G^2 + G^2(\sigma_B^2+\sigma_R^2) + B^2(\sigma_G^2+\sigma_R^2) - 2G(R\sigma_B^2 + B\sigma_R^2)}{B^2 + G^2 - GR + R^2 - B(G+R)^2}\right)$$



Histogram Construction

$$\tilde{f}(x) = \frac{1}{nh}(\text{number of } X_i \text{ in the same bin as } x)$$

n is the number of pixels X_i in the image, h is the bin width and x the range of the data

Kernel density estimator:

$$\tilde{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X}{h}\right)$$

where K is a function satisfying $\int K(x)dx = 1$

Variable Kernel Density Estimation

Variable kernel density is defined by:

$$\tilde{f}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\alpha(X_i)} K\left(\frac{x - X_i}{\alpha(X_i)}\right)$$

where X_i has associated scale (smoothing) parameter $\alpha(X_i)$

Variable Kernel Density Estimation

The most frequently occuring noise is additive Gaussian noise.

It is widely used to model thermal noise and is the limiting behaviour of photon counting noise and film grain noise

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp^{\frac{-x^2}{2}}$$

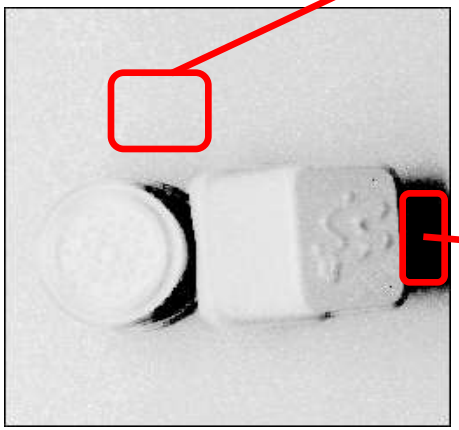
Variable kernel density for *rg* is defined by :

$$\tilde{f}(x) = \frac{1}{n} \sum_{i=1}^{n} \sigma_r^{-1} K(\frac{x-r}{\sigma_r}) \sigma_g^{-1} K(\frac{x-g}{\sigma_g})$$

where σ_r , σ_g denote the uncertainties in normalized color

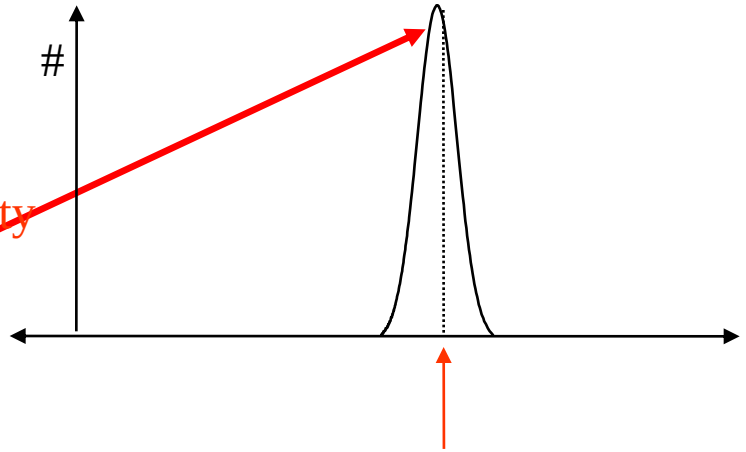
Variable Kernel Density Estimation Example

image



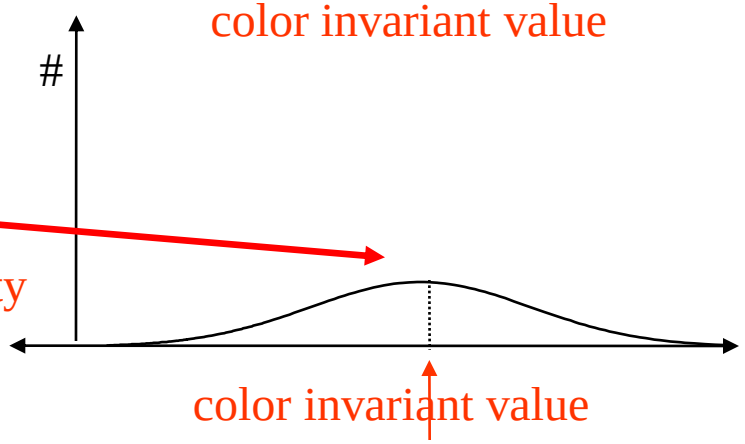
uncertainty

High certainty



color invariant value

Low certainty



color invariant value

Object Recognition

Dataset

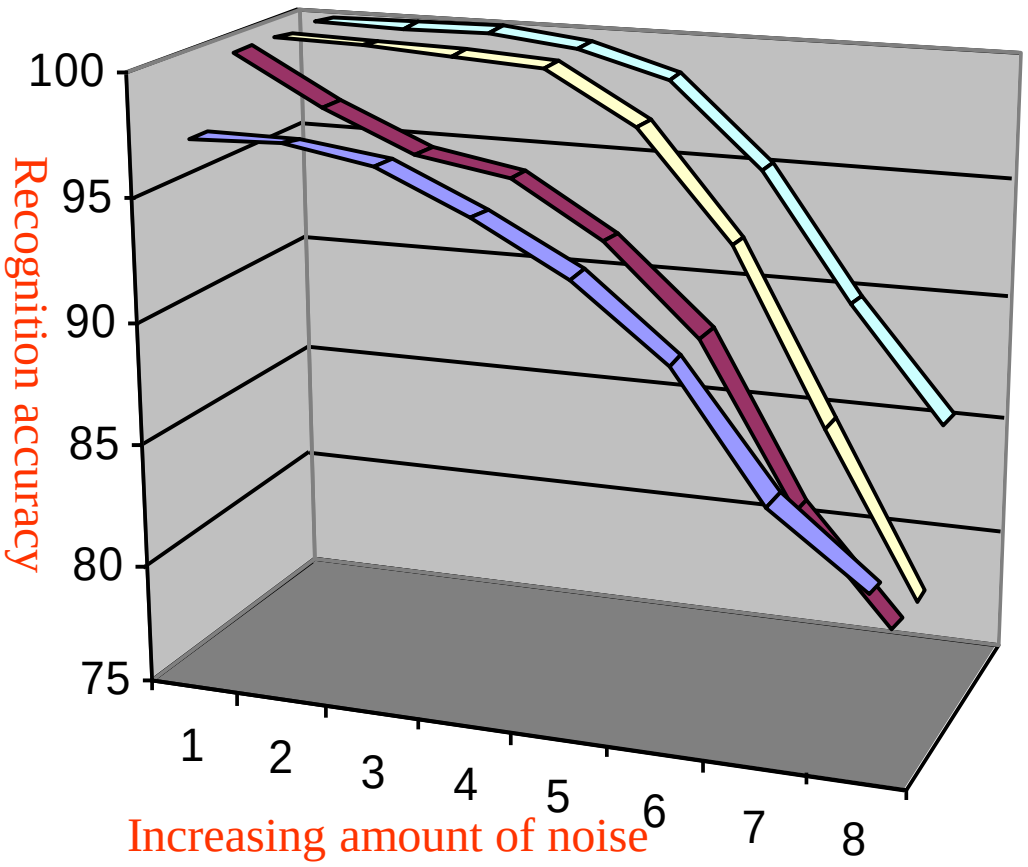
Let rank r^{Q_i} denote the position of the correct match for test image $Q_i, i = 1, \cdots, N_2$, in the ordered list of N_1 match values.

The average ranking percentile is defined by :

$$\bar{r} = \left(\frac{1}{N}\sum_{i=1}^{N_2}\frac{N_1 - r^{Q_i}}{N_1 - 1}\right)100\%$$

$\sigma_n = 8$ $\sigma_n = 16$ $\sigma_n = 32$ $\sigma_n = 64$

Experiment



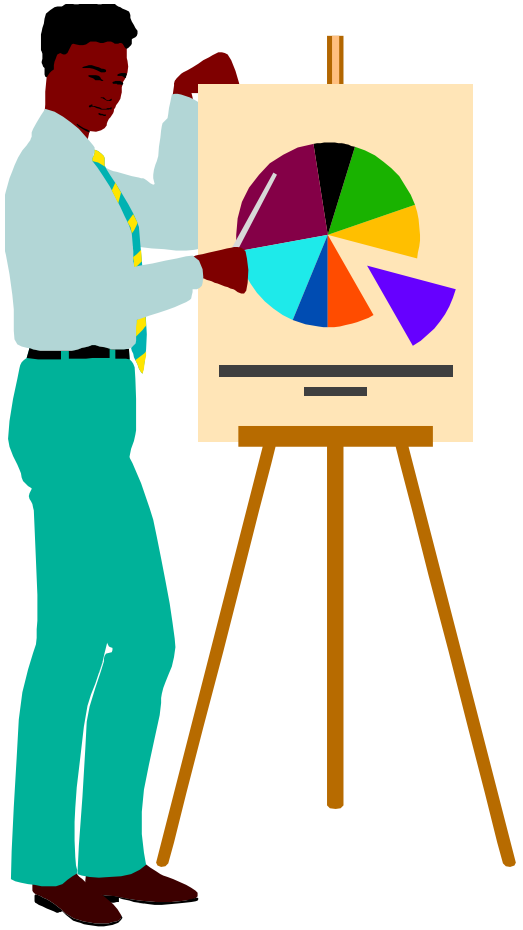
$$I = (R + G + B)/3$$

$$S = 1 - \frac{\min\{R, G, B\}}{R + G + B}$$

- without threshold
- threshold on S
- threshold on I and S
- variable kernel density

Image Processing

Image Filtering



Example: box filter

$g[\cdot,\cdot]$

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

Slide credit: David Lowe (UBC)

Image filtering

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

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

$h[.,.]$

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

Credit: S. Seitz

Image filtering

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

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							

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

Credit: S. Seitz

Image filtering

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

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						

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

Credit: S. Seitz

Image filtering

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

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					

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

Credit: S. Seitz

Image filtering

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

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				

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

Credit: S. Seitz

Image filtering

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

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				
				?					

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

Credit: S. Seitz

Image filtering

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

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					

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

Credit: S. Seitz

Image filtering

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

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	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	

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

Credit: S. Seitz

Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood

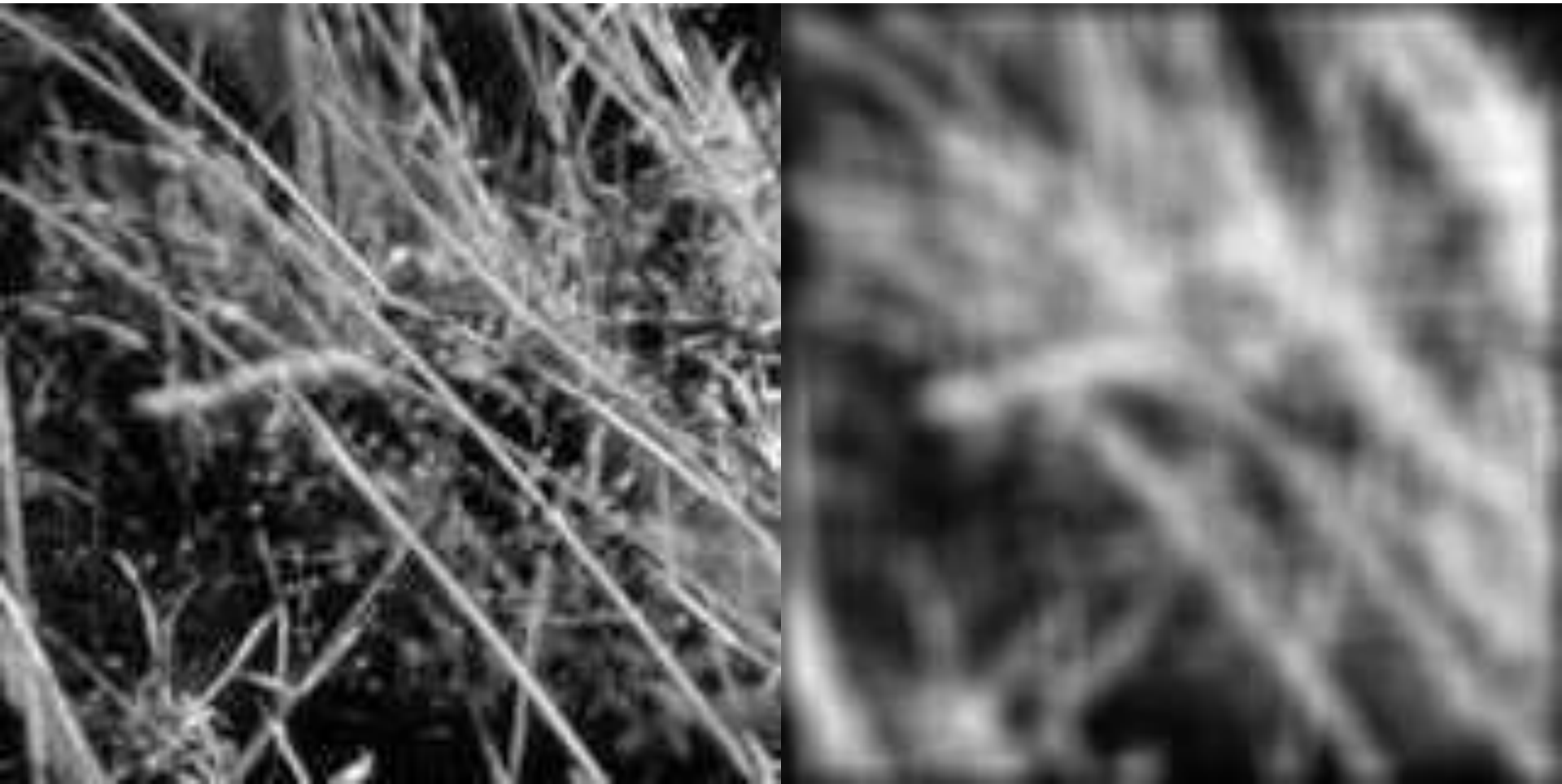
- Achieve smoothing effect (remove sharp features)

$g[\cdot,\cdot]$

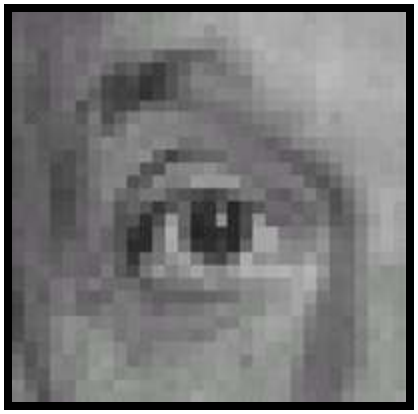
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

Smoothing with box filter



Practice with linear filters



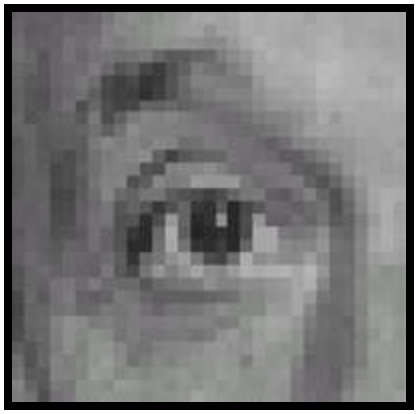
Original

0	0	0
0	1	0
0	0	0

?

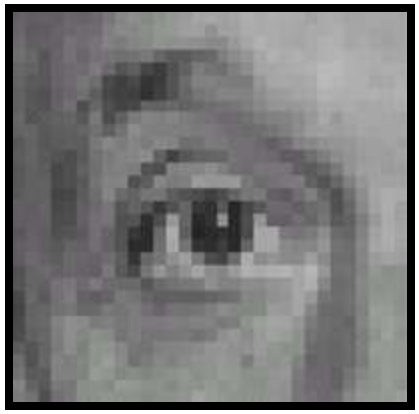
Source: D. Lowe

Practice with linear filters



Original

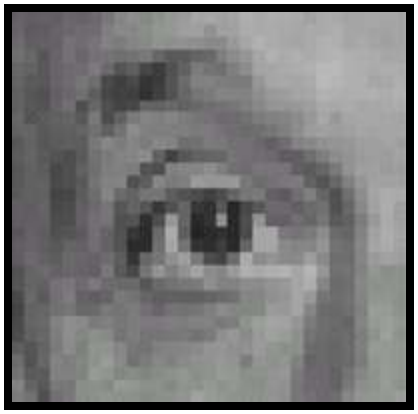
0	0	0
0	1	0
0	0	0



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



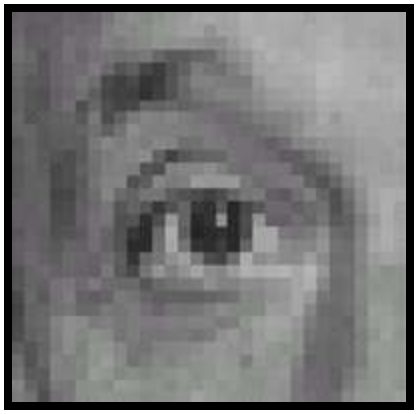
Original

0	0	0
0	0	1
0	0	0

?

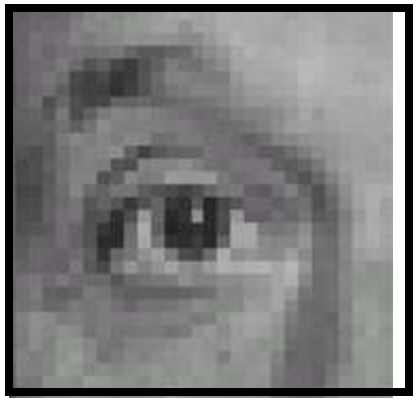
Source: D. Lowe

Practice with linear filters



Original

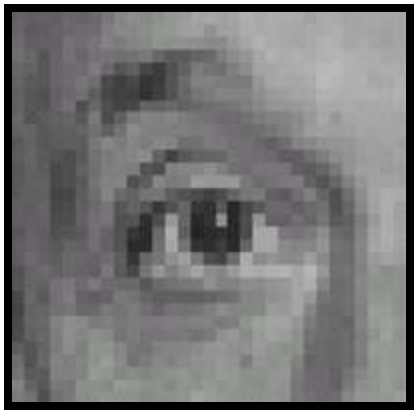
0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with linear filters

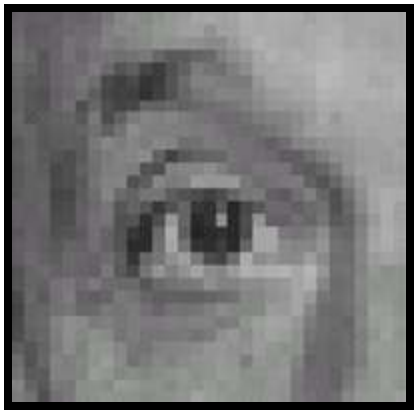


Original

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

Source: D. Lowe

Practice with linear filters



Original

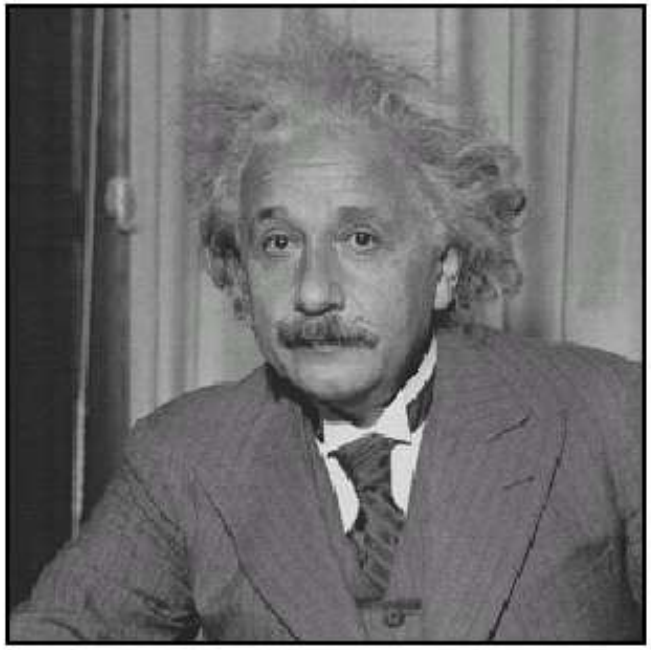
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening filter

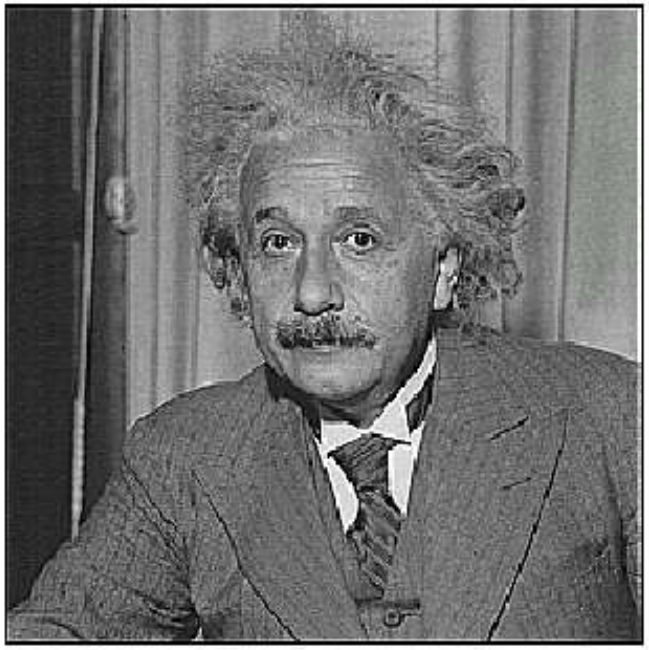
- Accentuates differences with local average

Source: D. Lowe

Sharpening



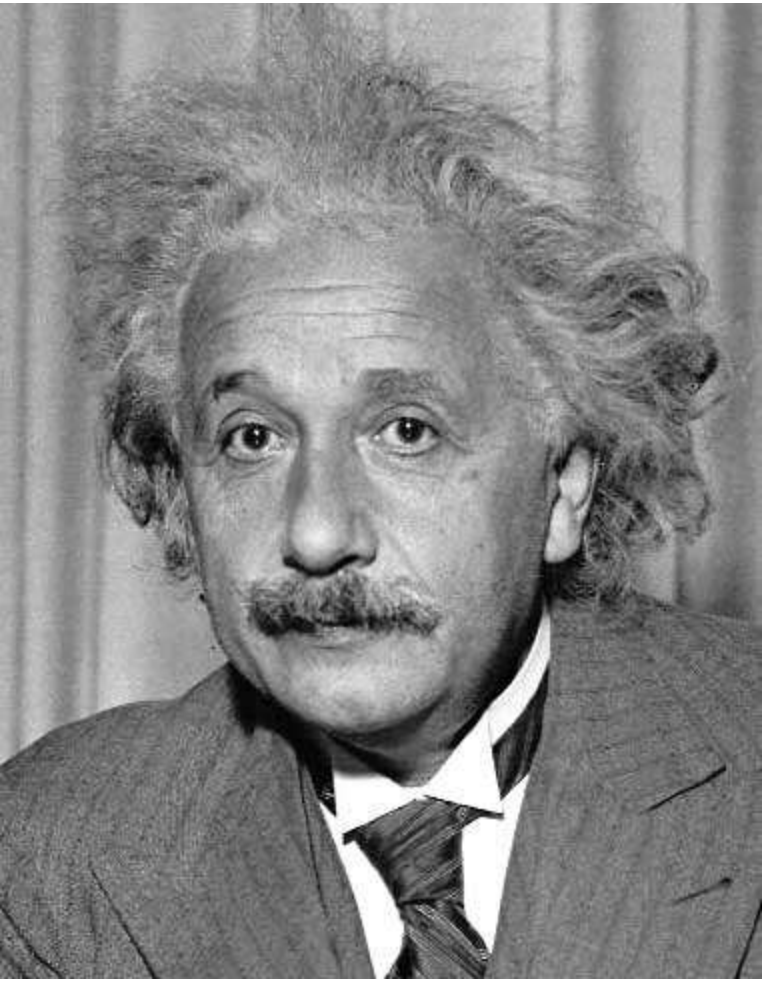
before



after

Source: D. Lowe

Other filters

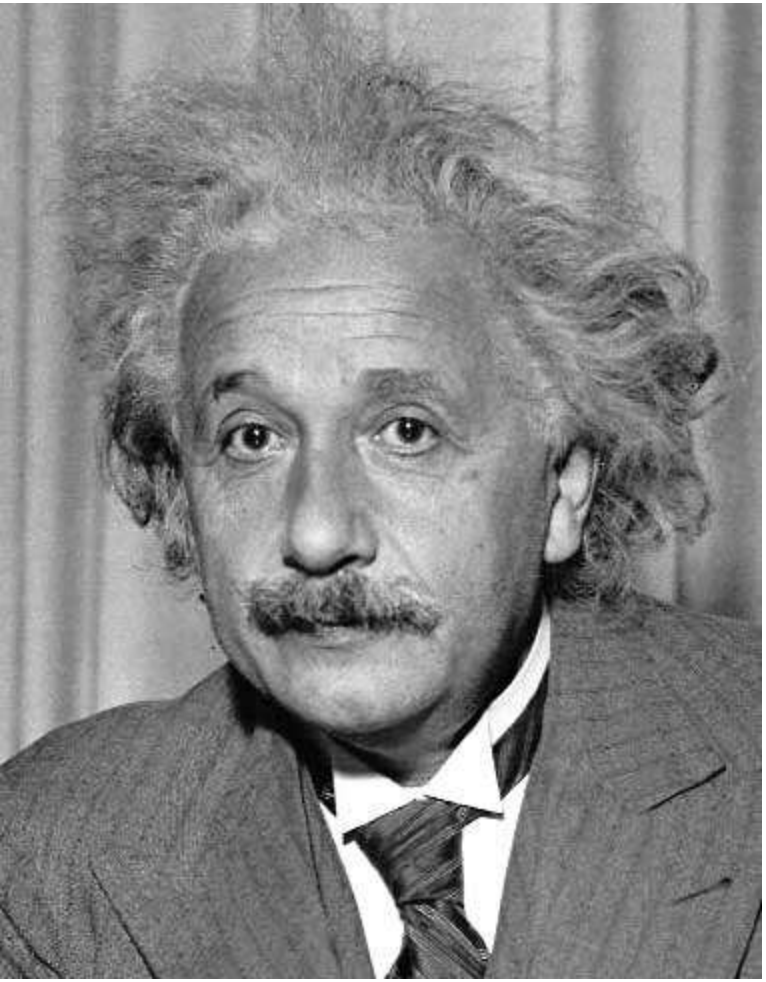


1	0	-1
2	0	-2
1	0	-1

Sobel

Vertical Edge
(absolute value)

Other filters



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

Sobel

Horizontal Edge
(absolute value)

Filtering vs. Convolution

- 2d filtering

g=filter f=image

– h=filter2(g,f); or h=imfilter(f,g);

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

- 2d convolution

– h=conv2(g,f);

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

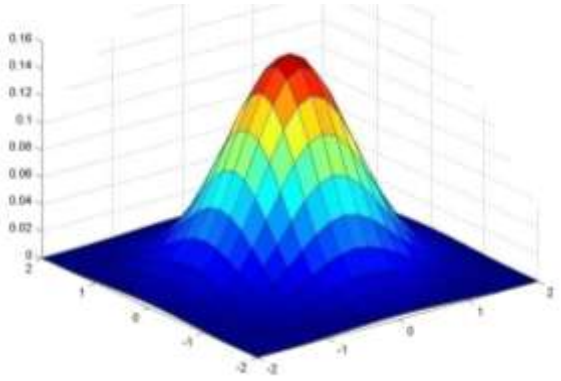
Properties

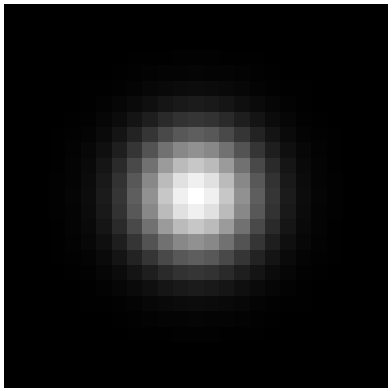
- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: $a * (b_1 * b_2 * b_3)$
- Distributes over addition: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

Source: S. Lazebnik

Important filter: Gaussian

Weight contributions of neighboring pixels by nearness





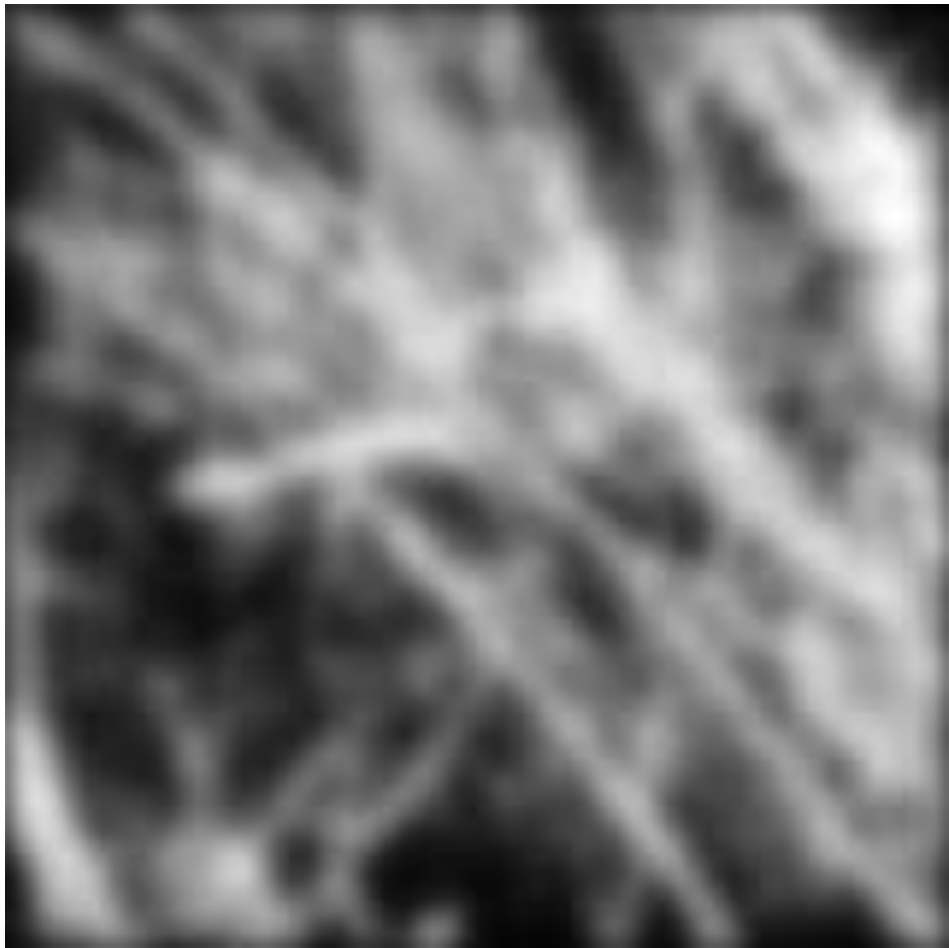
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

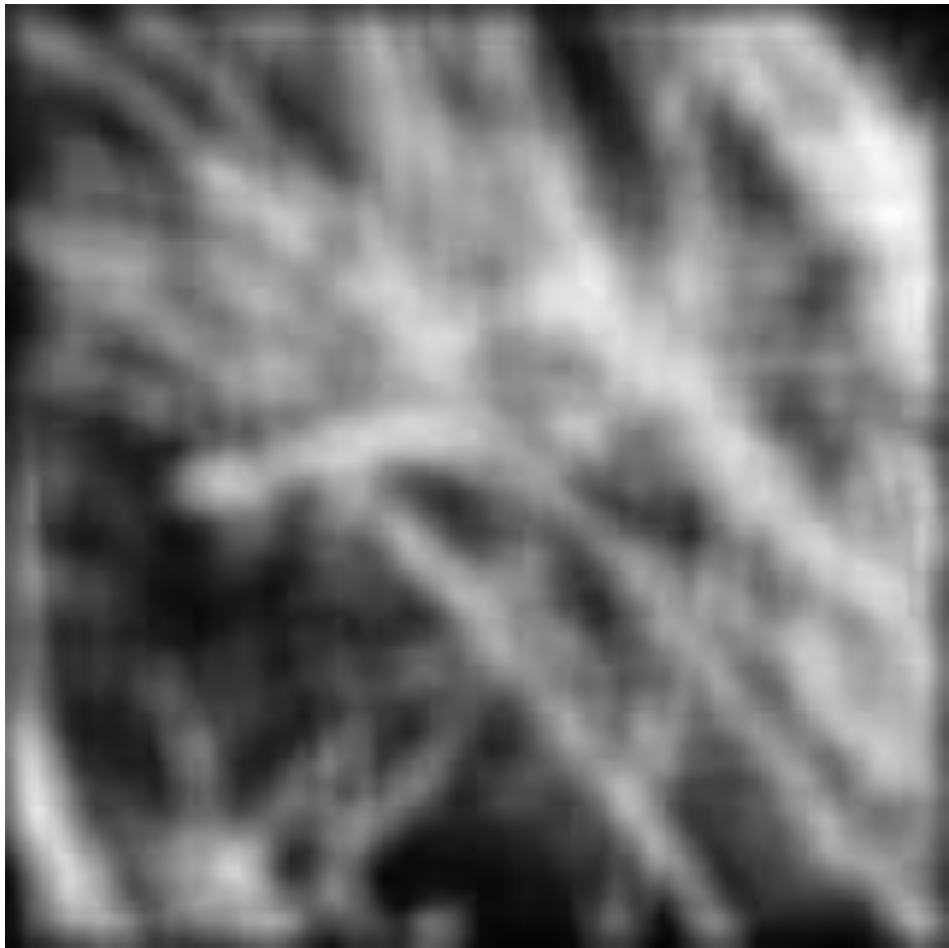
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Slide credit: Christopher Rasmussen

Smoothing with Gaussian filter



Smoothing with box filter



Gaussian filters

- Remove “high-frequency” components from the image (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* kernel
  - Factors into product of two 1D Gaussians

Source: K. Grauman

Separability of the Gaussian filter

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

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

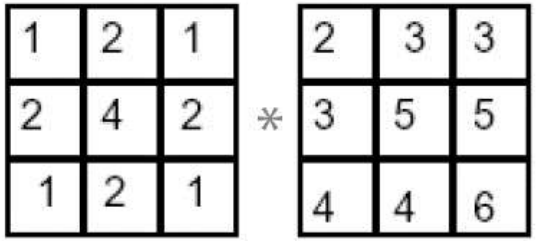
The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

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

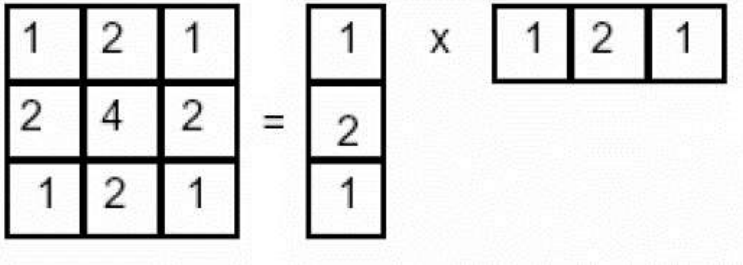
Source: D. Lowe

Separability example

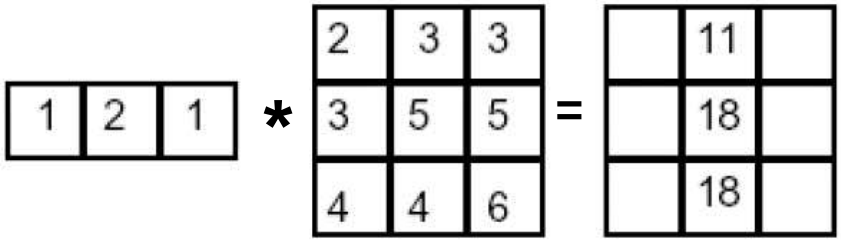
2D convolution
(center location only)



The filter factors
into a product of 1D
filters:



Perform convolution
along rows:



Followed by convolution
along the remaining column:

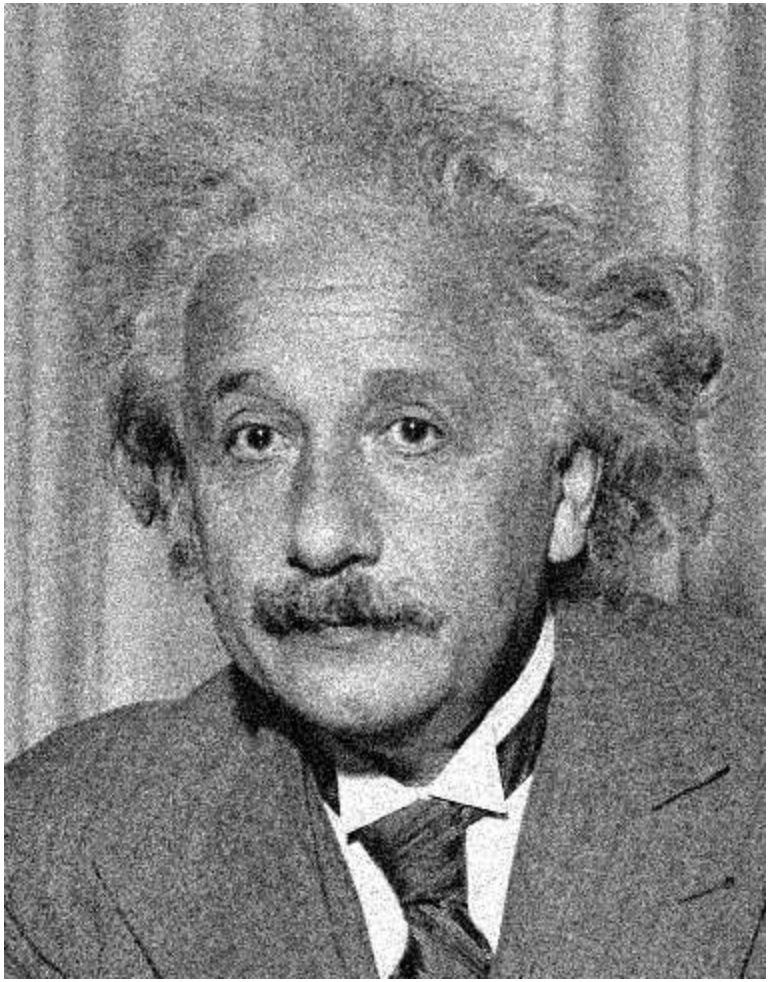
Source: K. Grauman

Practical matters

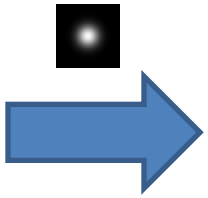
- What about near the edge?
  - the filter window falls off the edge of the image
  - need to extrapolate
  - methods:
    - clip filter (black)
    - wrap around
    - copy edge
    - reflect across edge

Source: S. Marschner

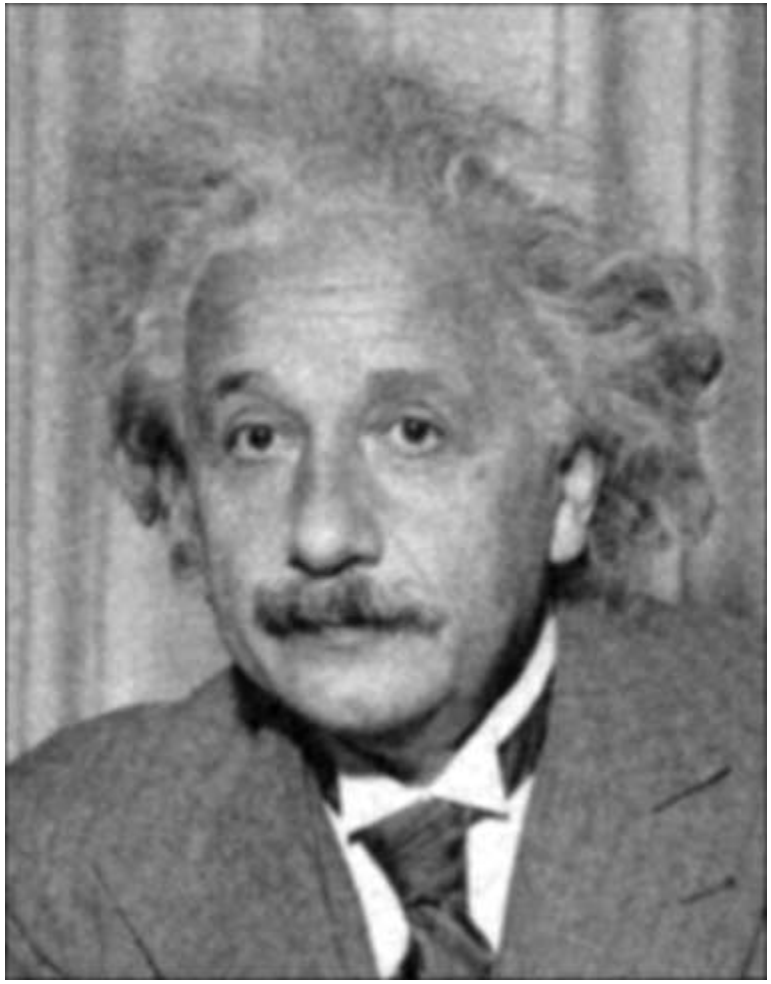
Denoising



Additive Gaussian Noise



Gaussian
Filter



Slide: Hoiem

Reducing salt-and-pepper noise by Gaussian smoothing

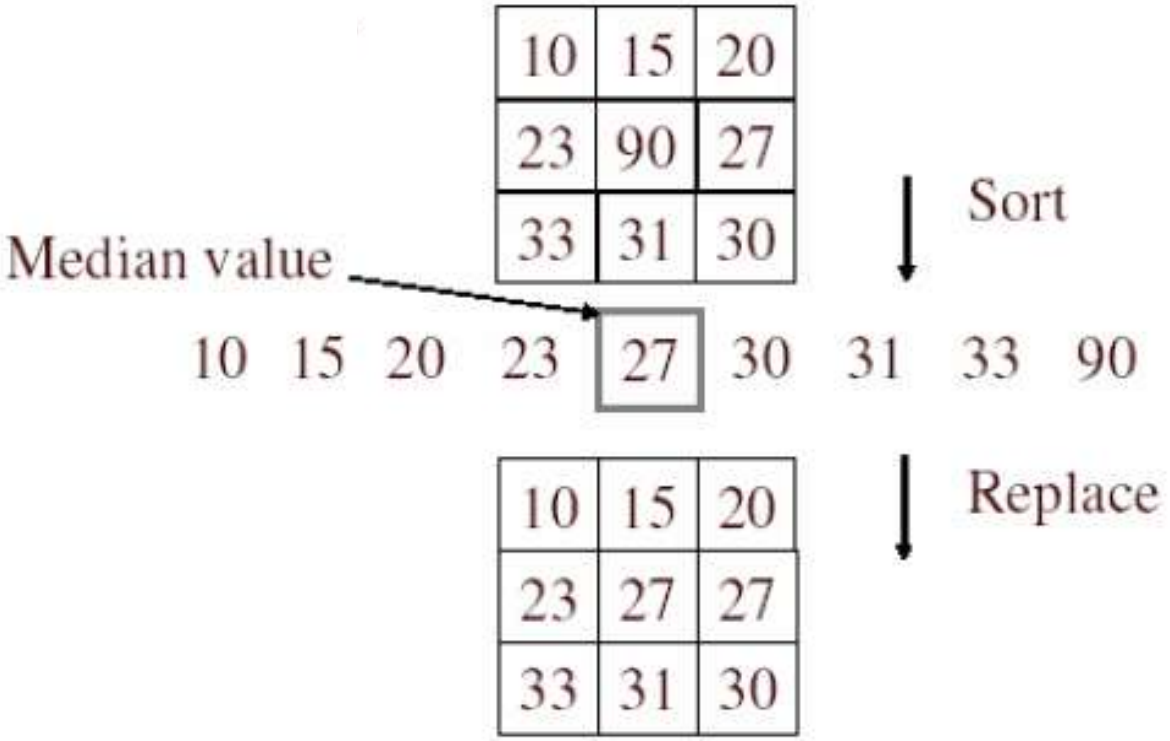
3x3

5x5

7x7

Alternative idea: Median filtering

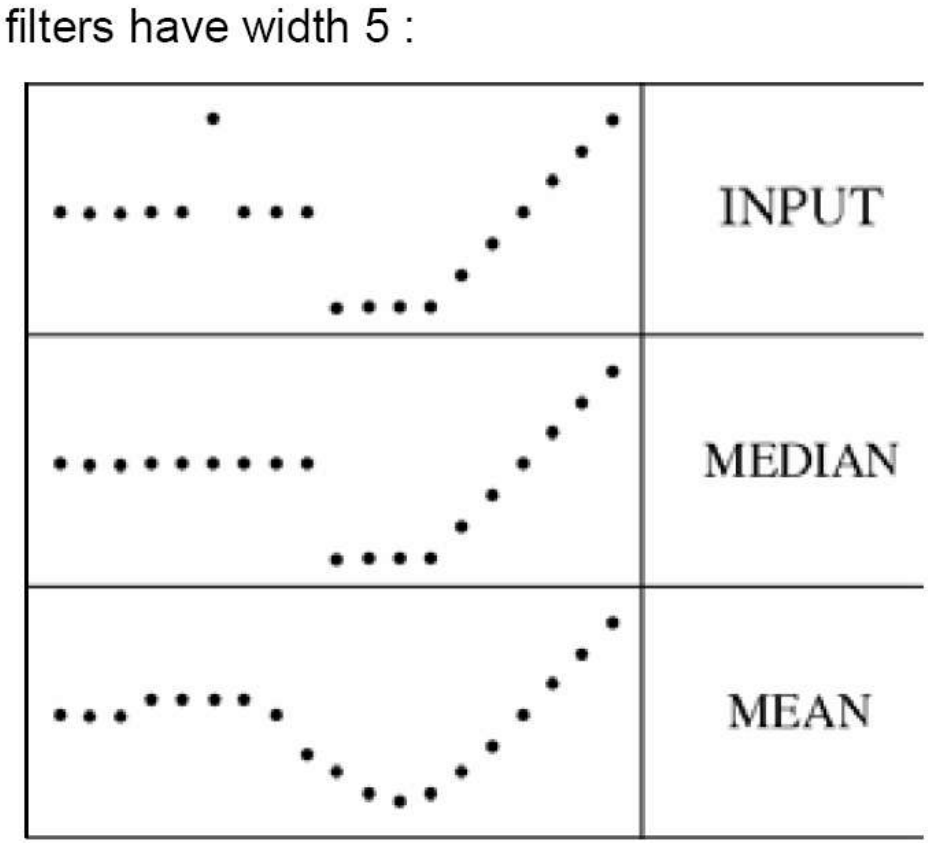
- A **median filter** operates over a window by selecting the median intensity in the window



Source: K. Grauman

Median filter

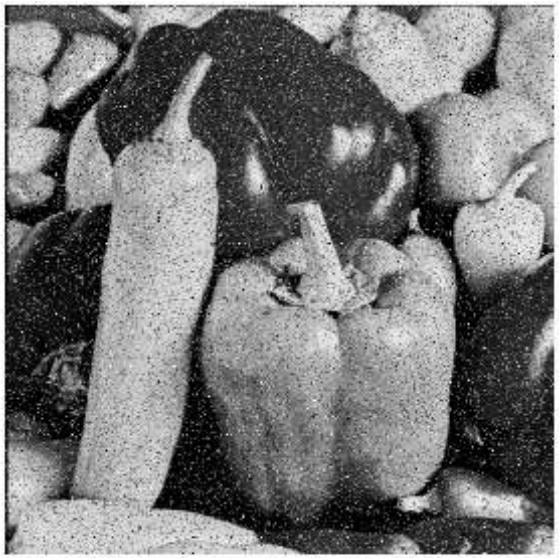
- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers

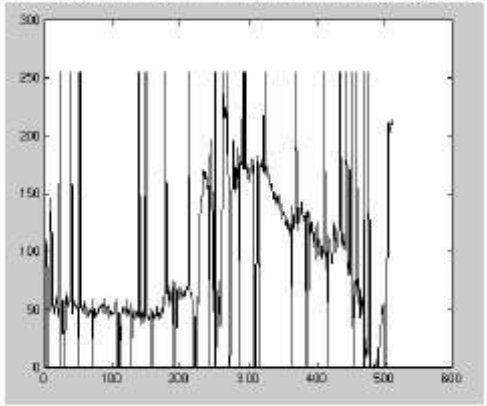


Source: K. Grauman

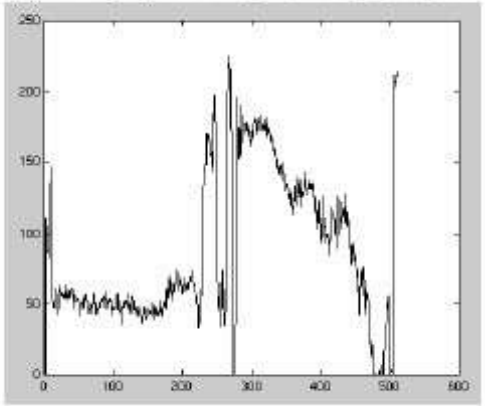
Median filter

Salt-and-pepper noise



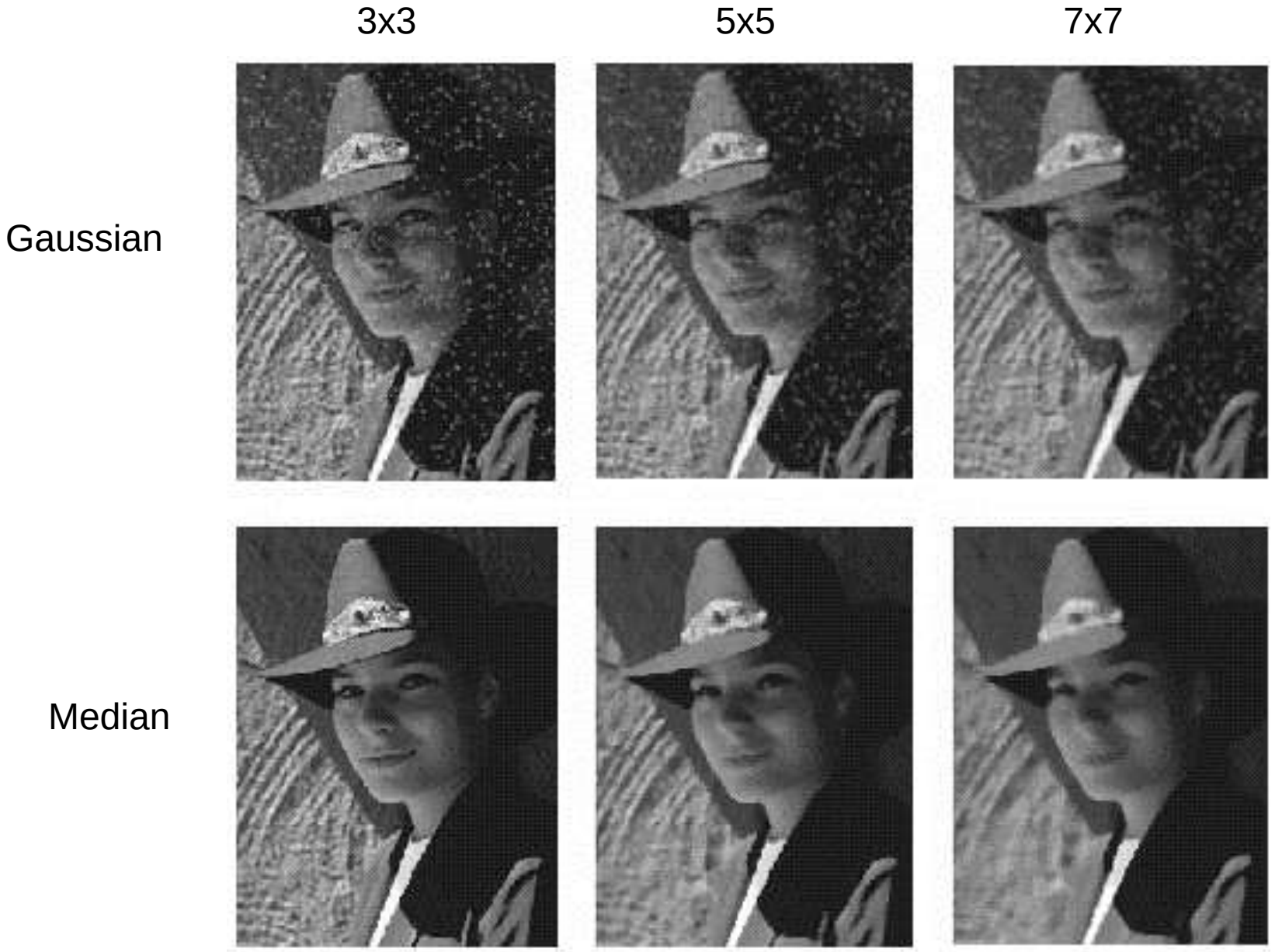


Median filtered



Source: M. Hebert

Median vs. Gaussian filtering



Other non-linear filters

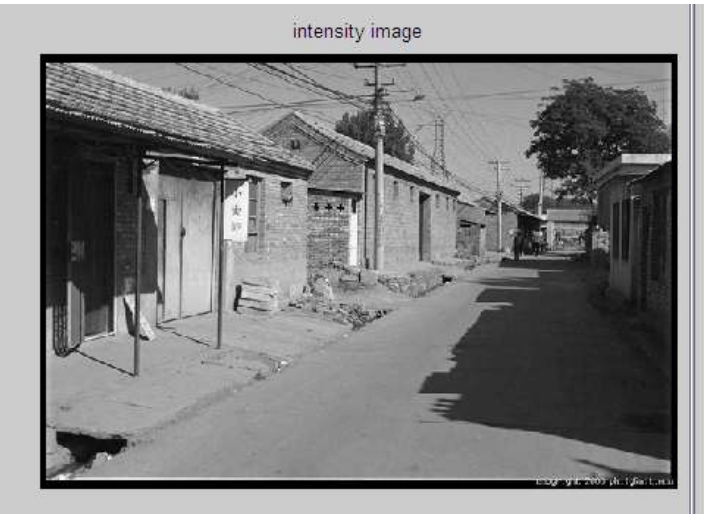
- Weighted median (pixels further from center count less)
- Clipped mean (average, ignoring few brightest and darkest pixels)
- Bilateral filtering (weight by spatial distance *and* intensity difference)

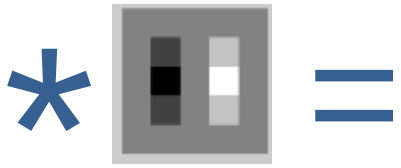
Bilateral filtering

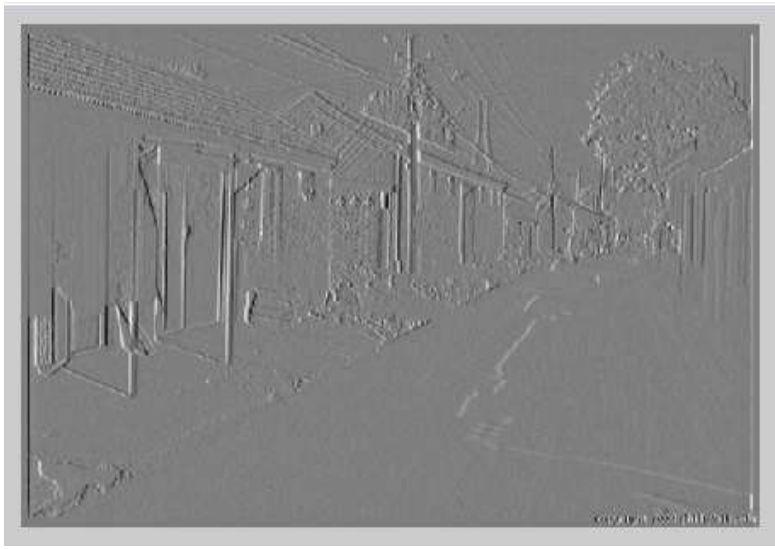
Image: http://vision.ai.uiuc.edu/?p=1455

Review: Edge Detection

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

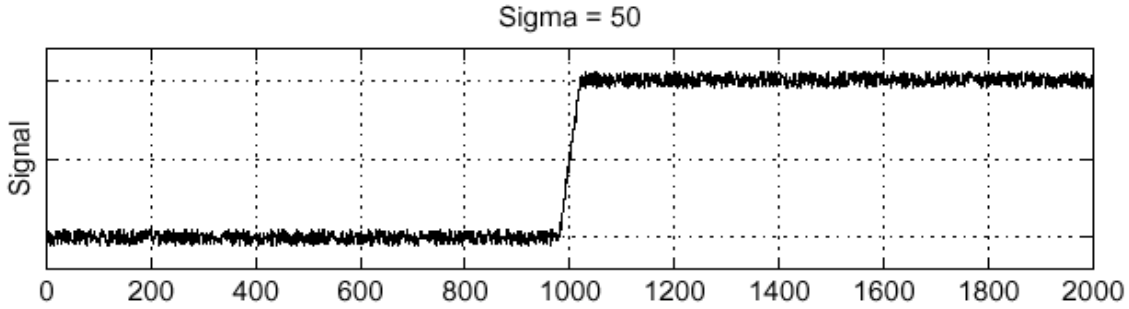






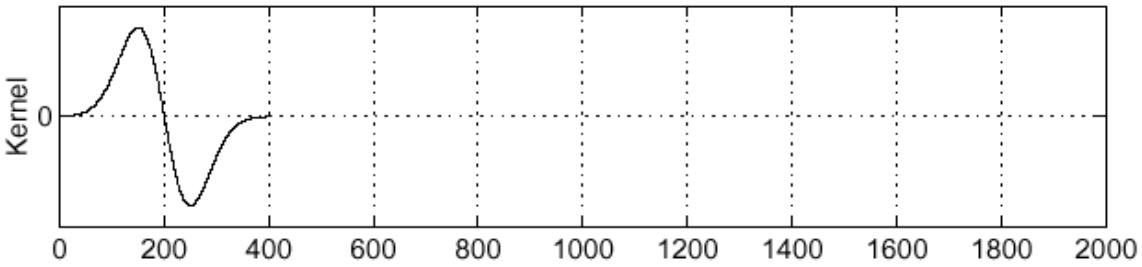
Review: Edge Detection

f



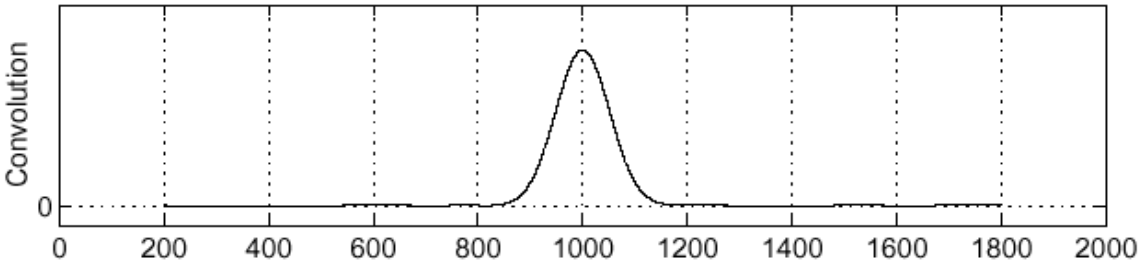
Edge

$\frac{d}{dx} g$



Derivative of Gaussian

$f * \frac{d}{dx} g$

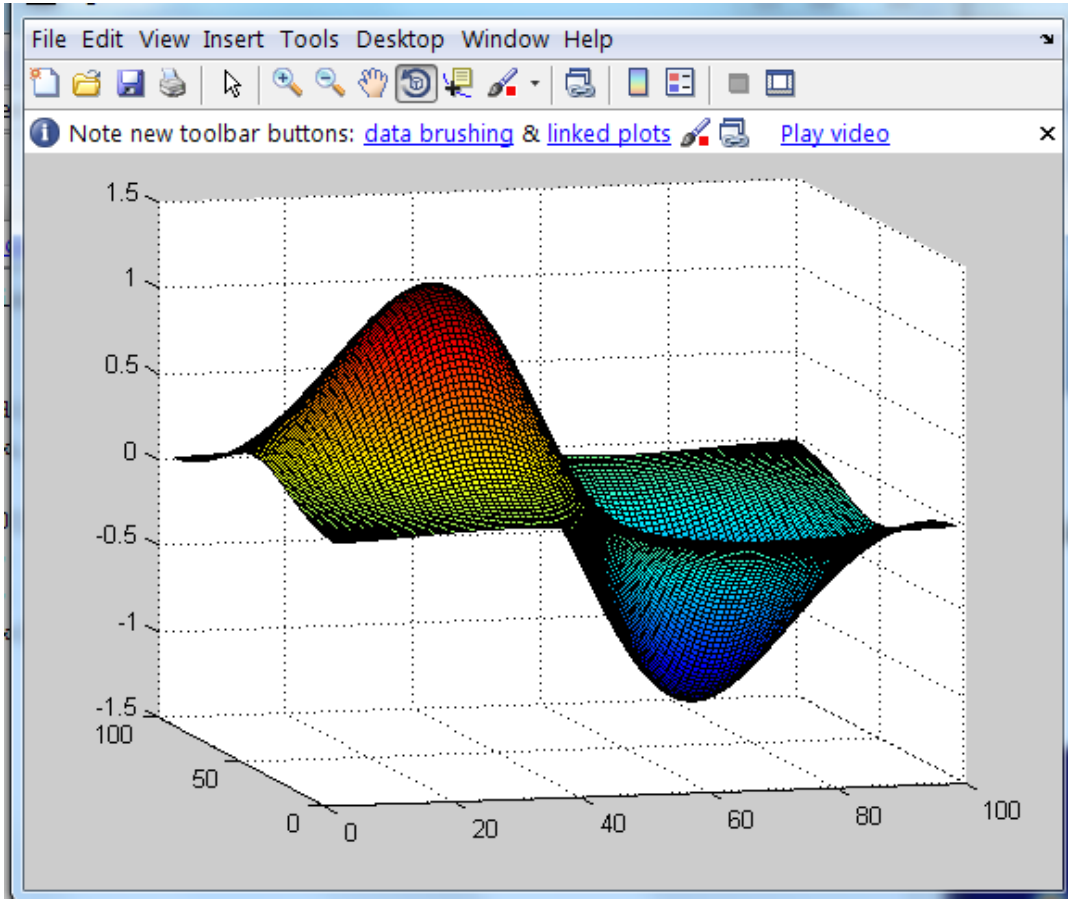


Edge = maximum of derivative

Source: S. Seitz

Review

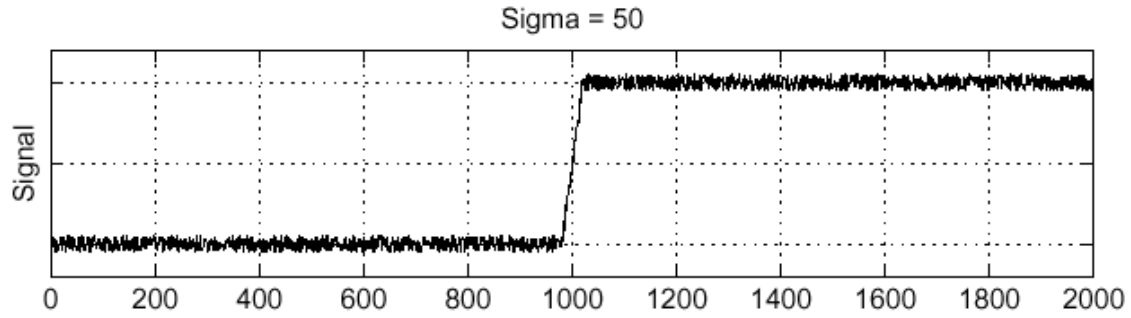
Derivative of Gaussian



Slide: Hoiem

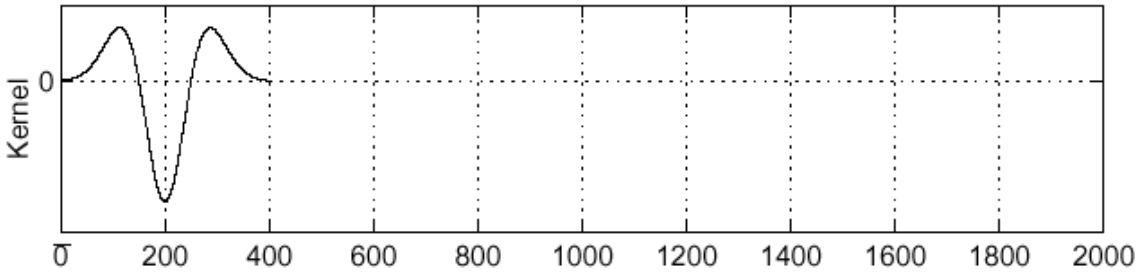
Review: Edge Detection

f



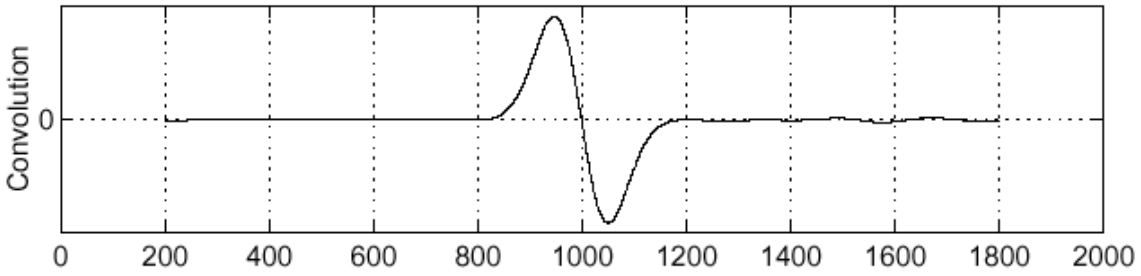
Edge

$\frac{d^2}{dx^2} g$



Second derivative of Gaussian (Laplacian)

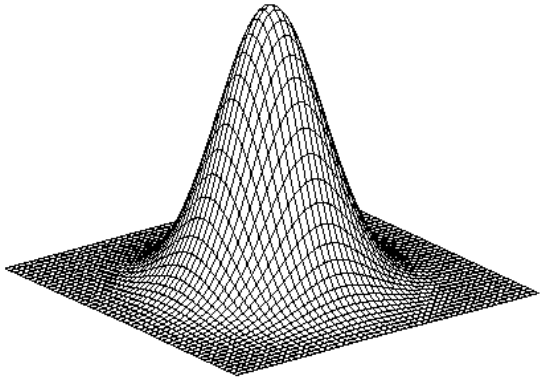
$f * \frac{d^2}{dx^2} g$



Edge = zero crossing of second derivative

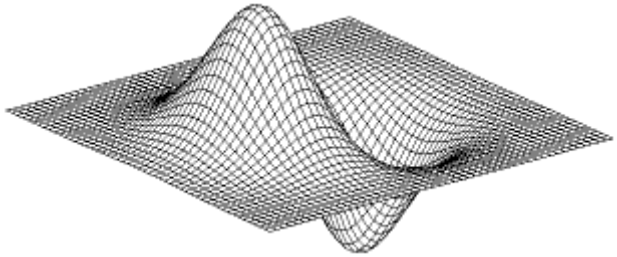
Source: S. Seitz

2D Edge Detection Filters



Gaussian

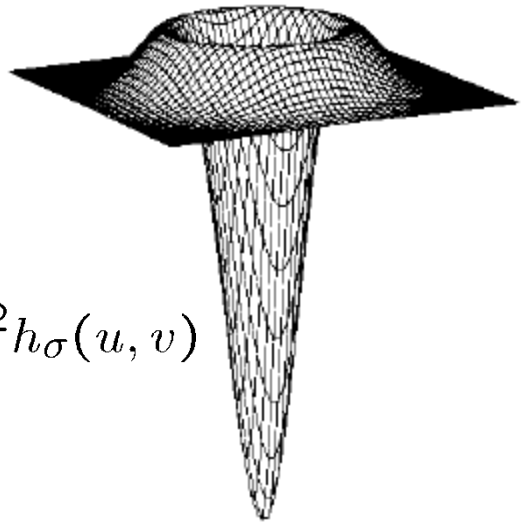
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



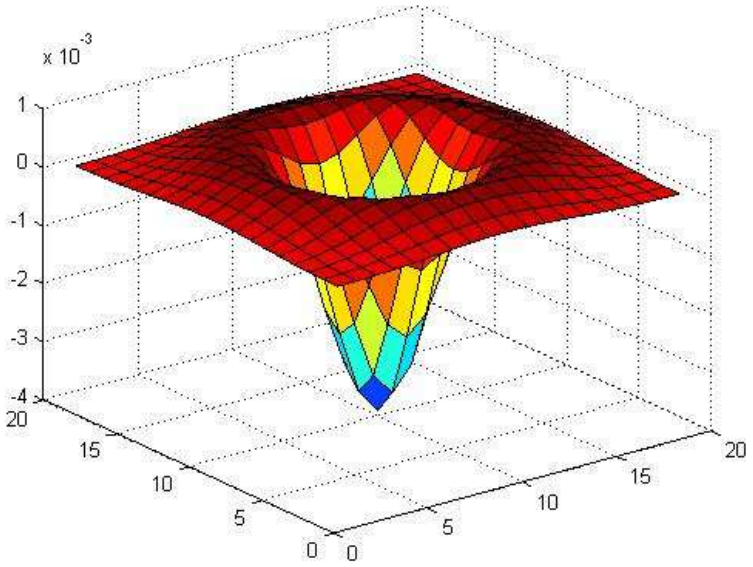
$$\nabla^2 h_\sigma(u,v)$$

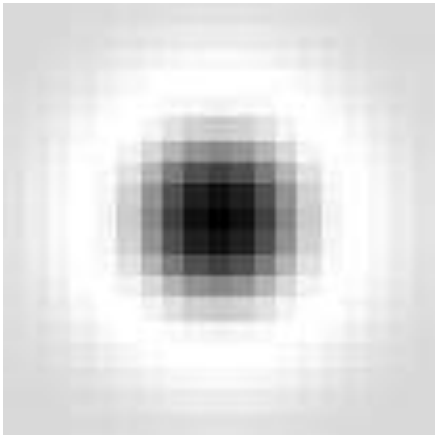
∇^2 is the **Laplacian** operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Laplacian of Gaussian

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

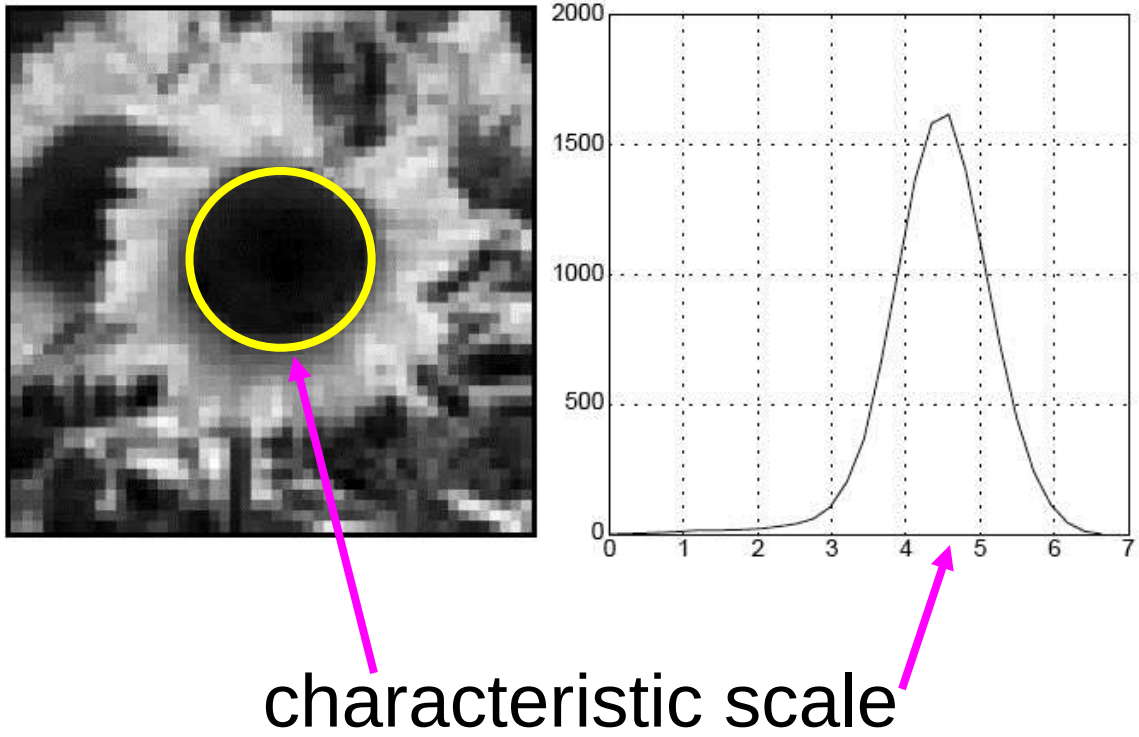




$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Blob Detection in 2D

- We define the *characteristic scale* as the scale that produces peak of Laplacian response

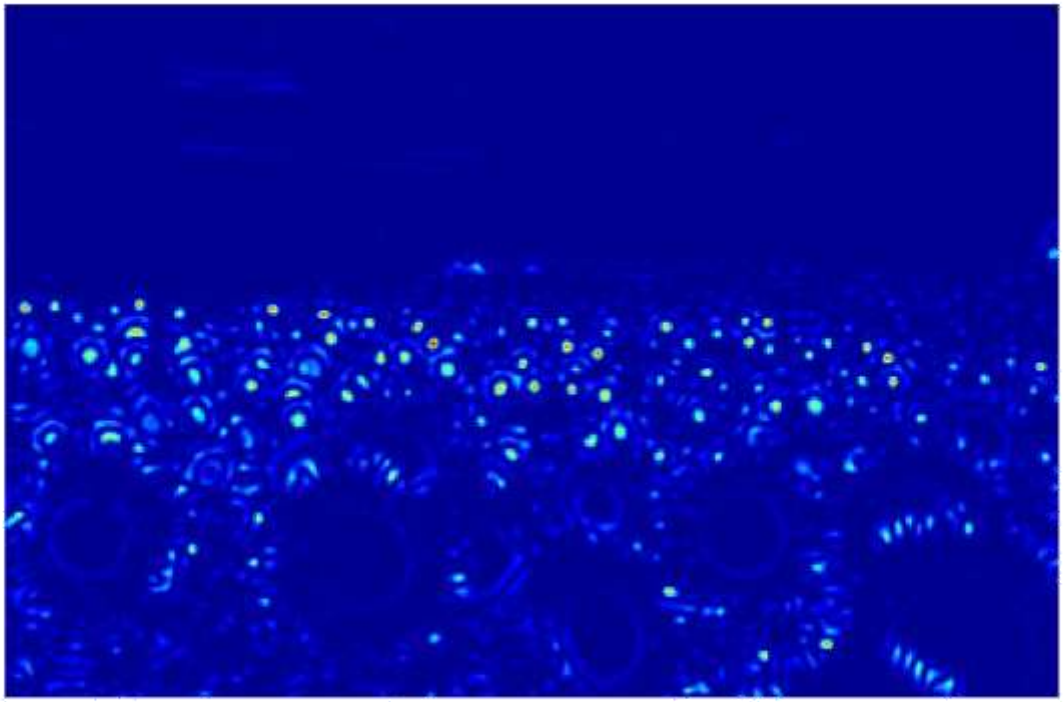


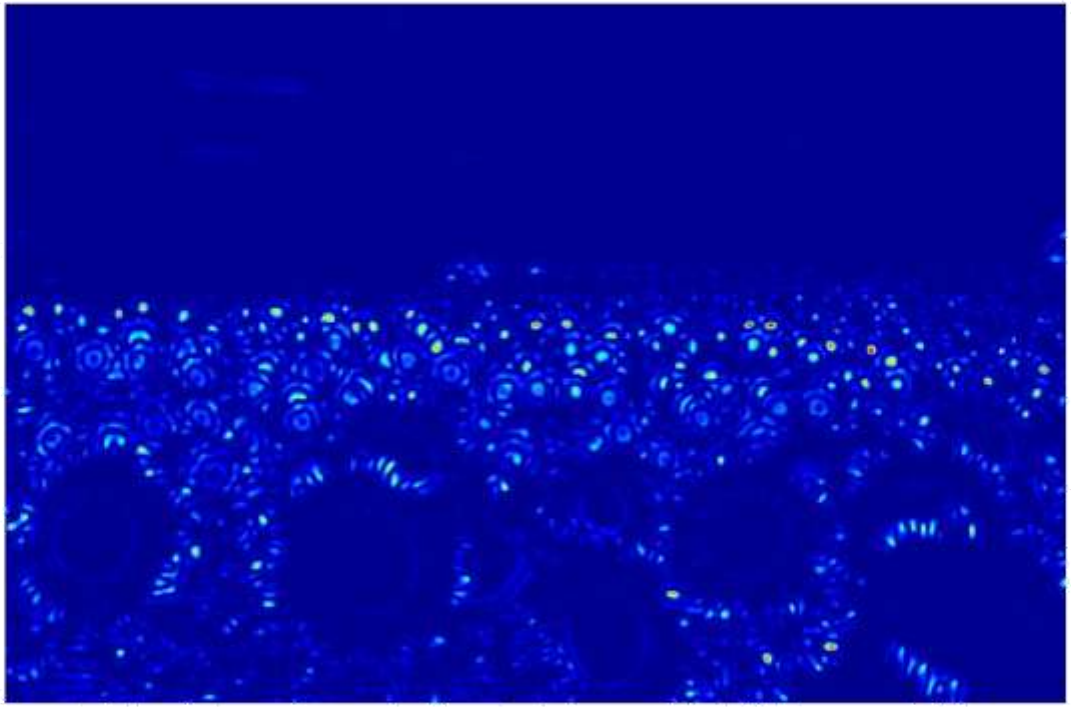
Slide credit: Lana Lazebnik

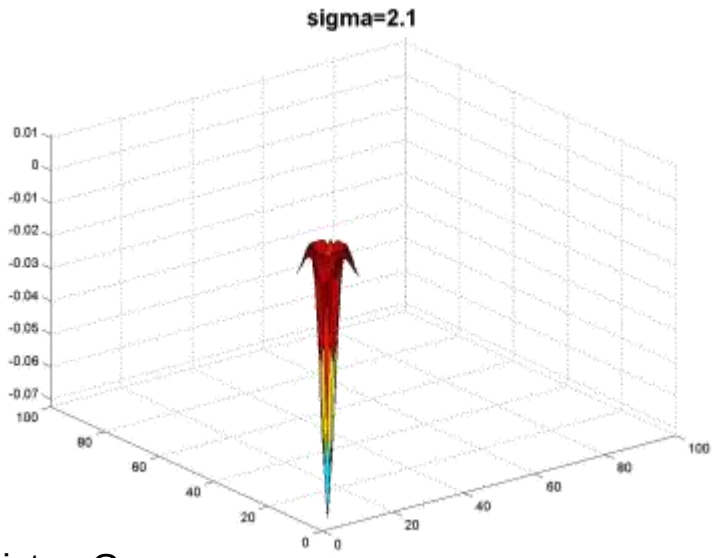
Example

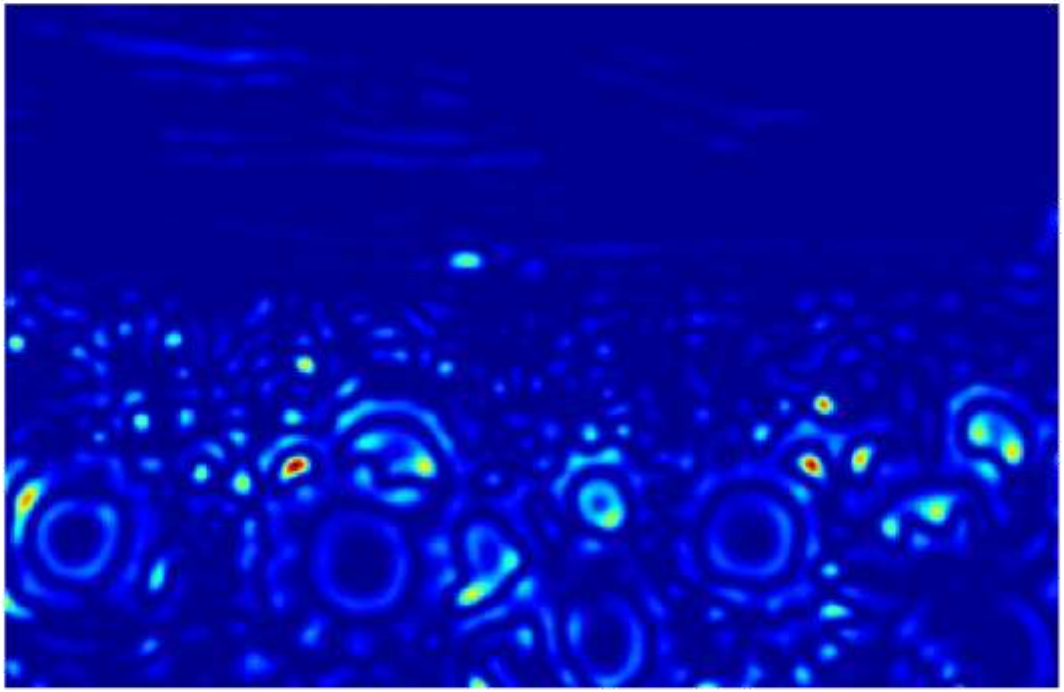
Original image
at ¾ the size

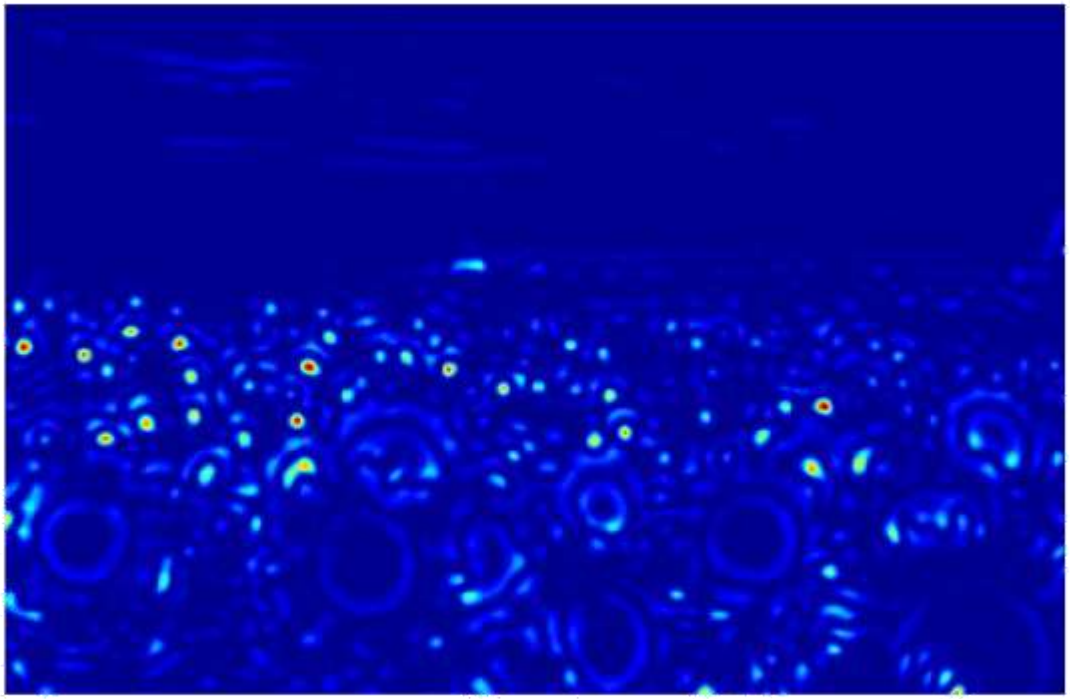
Original image
at ¾ the size

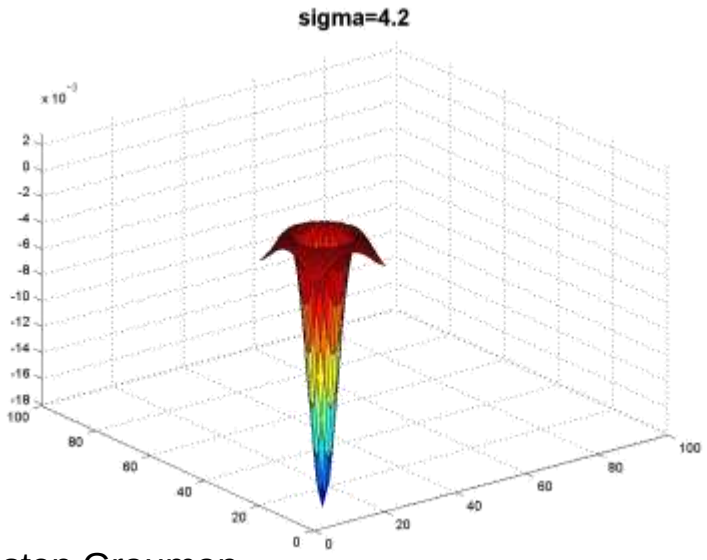




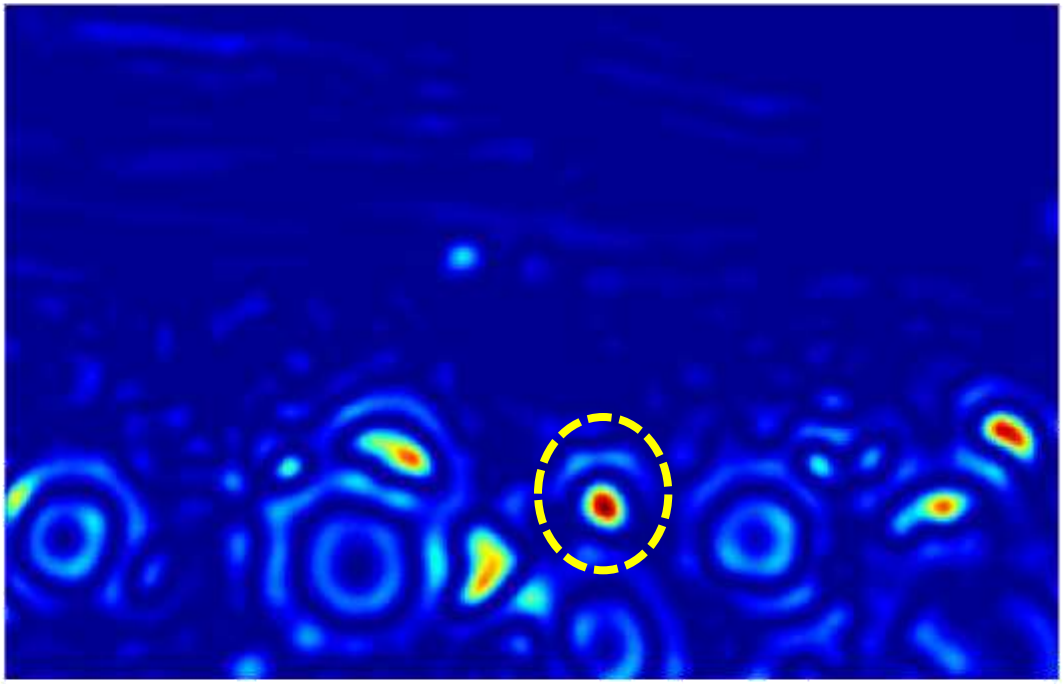


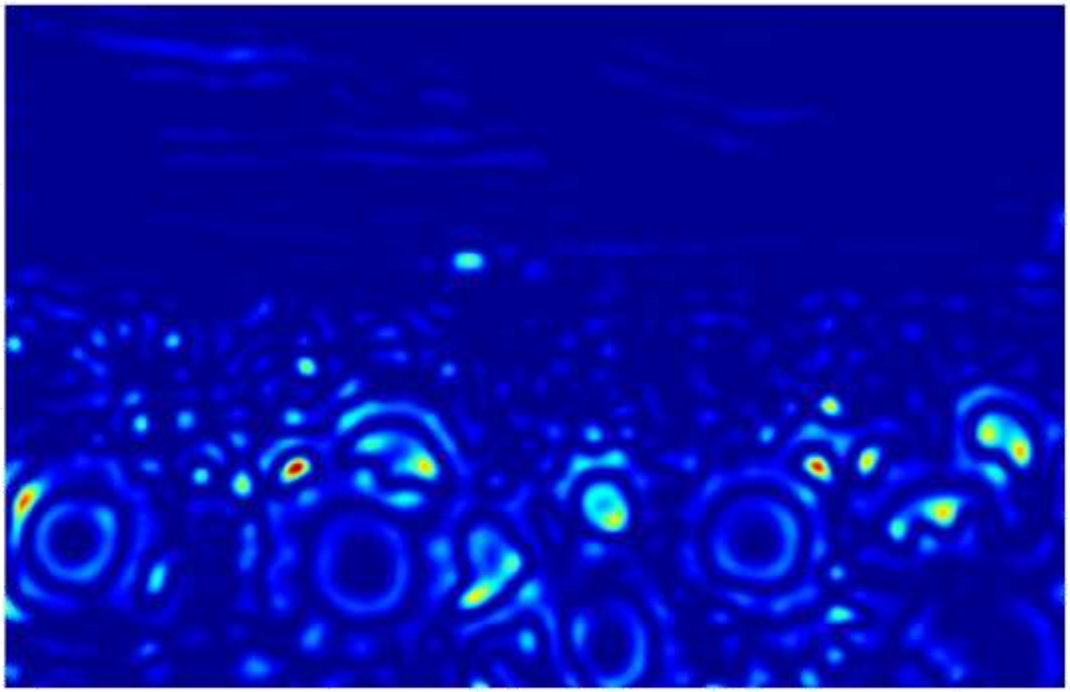


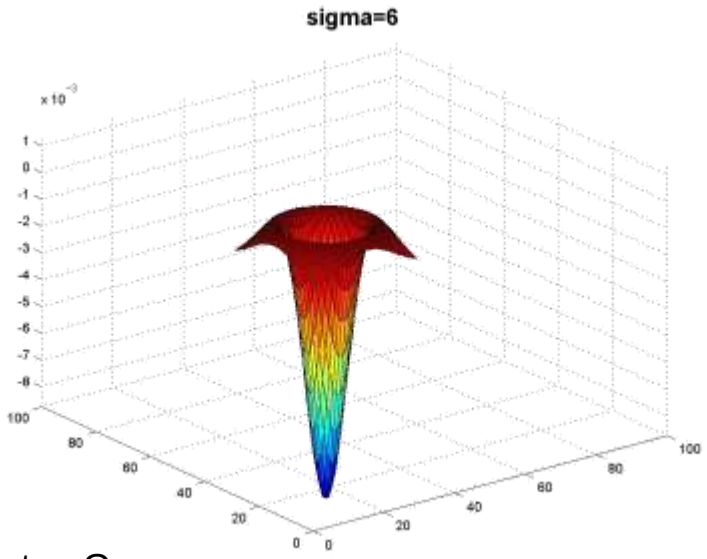




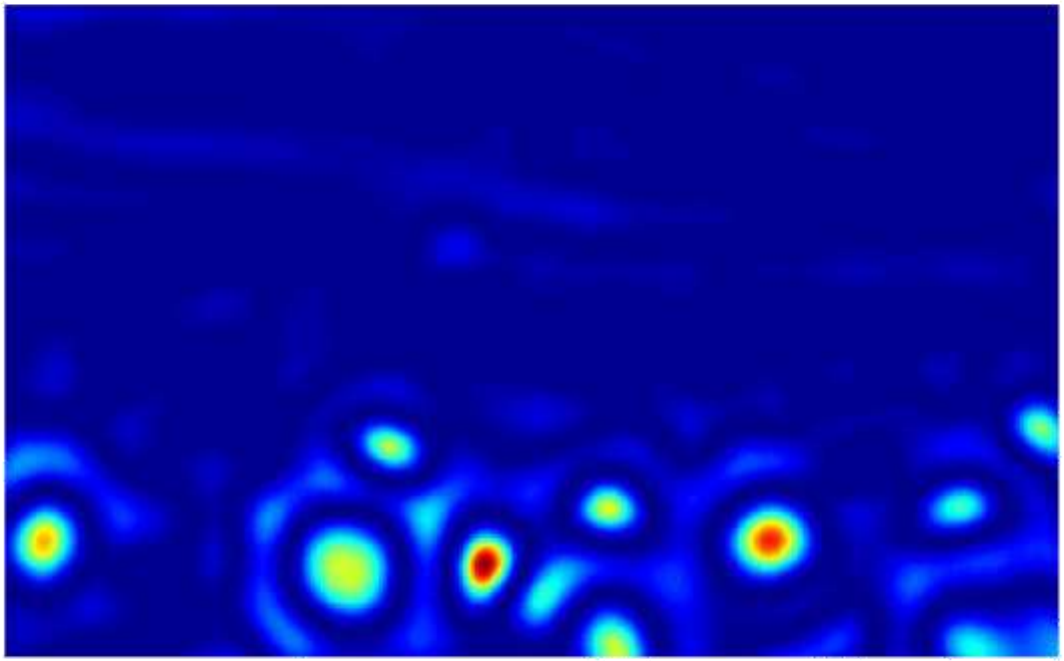
Kristen Grauman

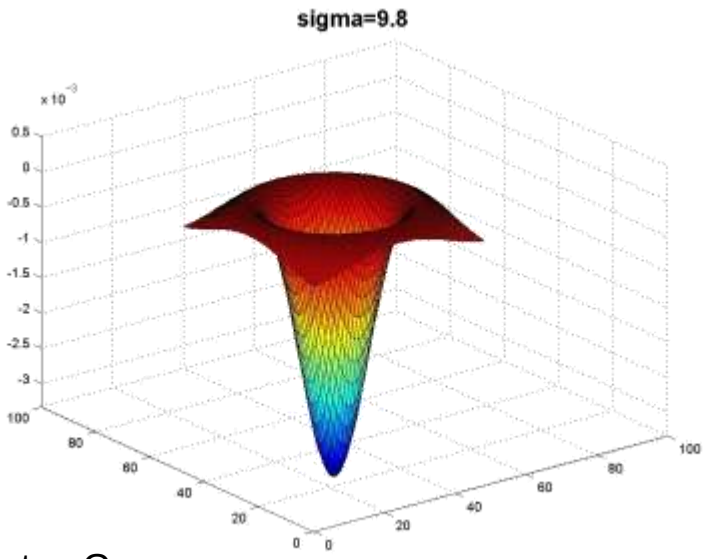


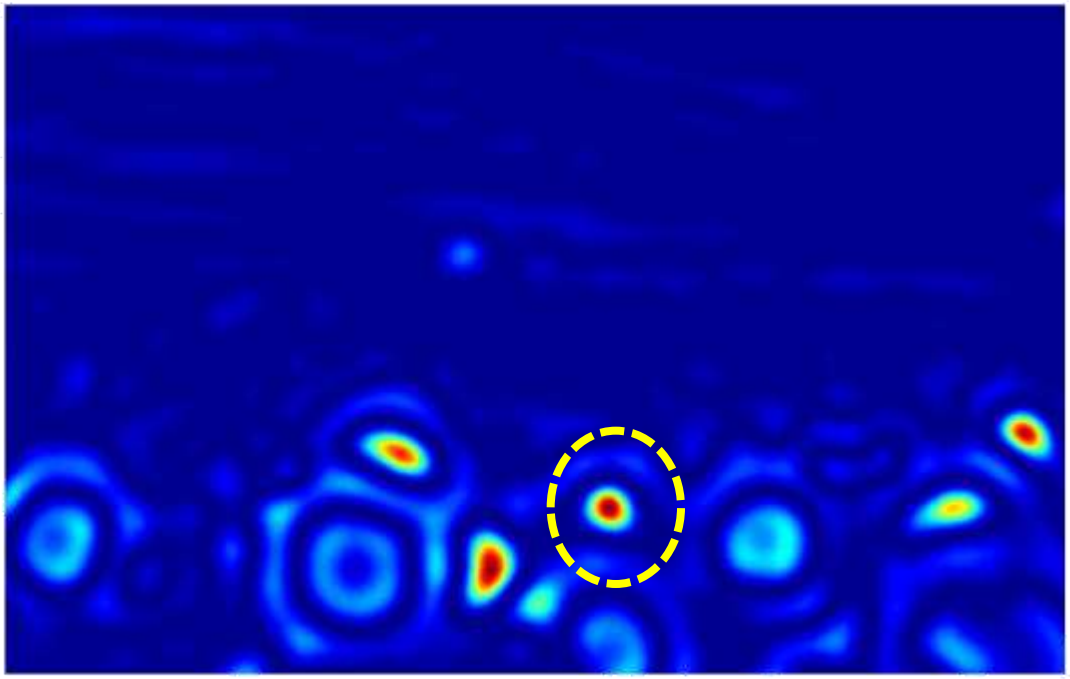


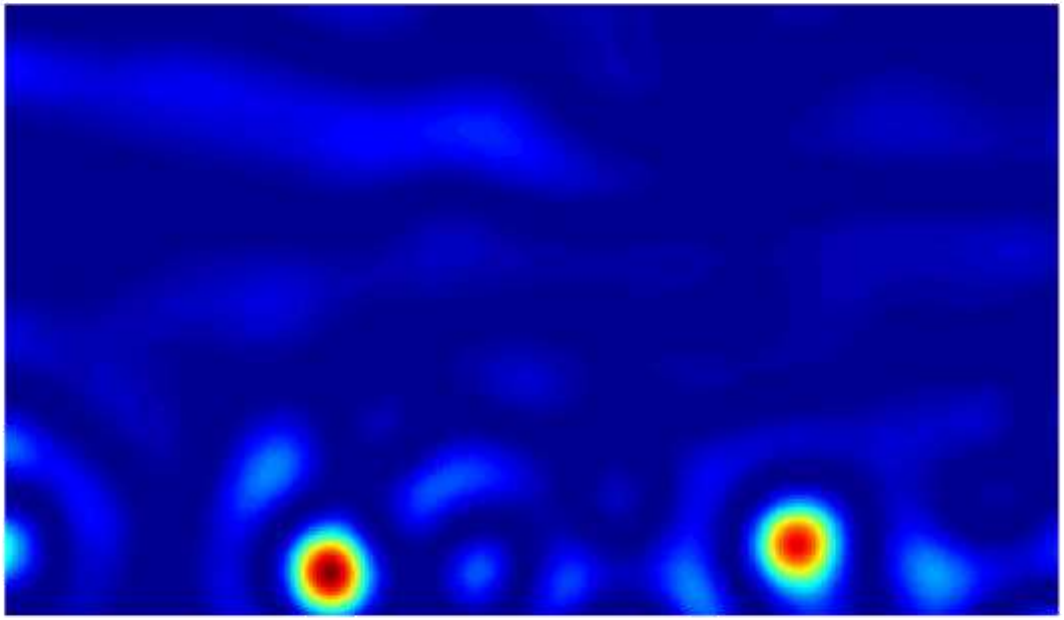


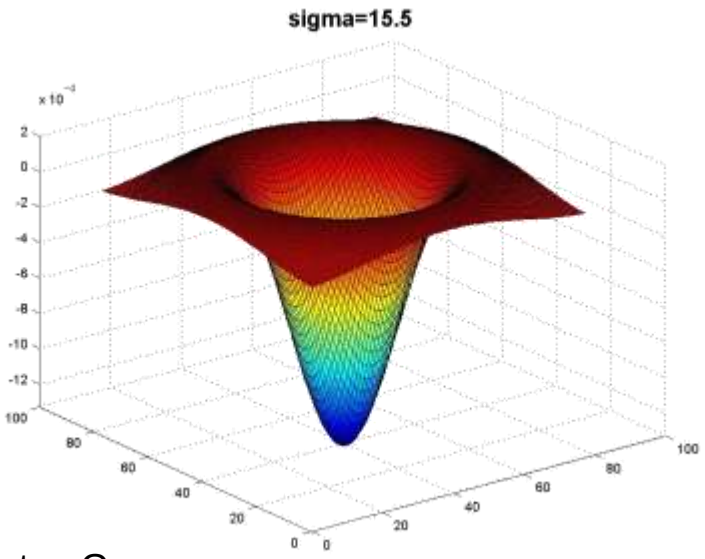
Kristen Grauman

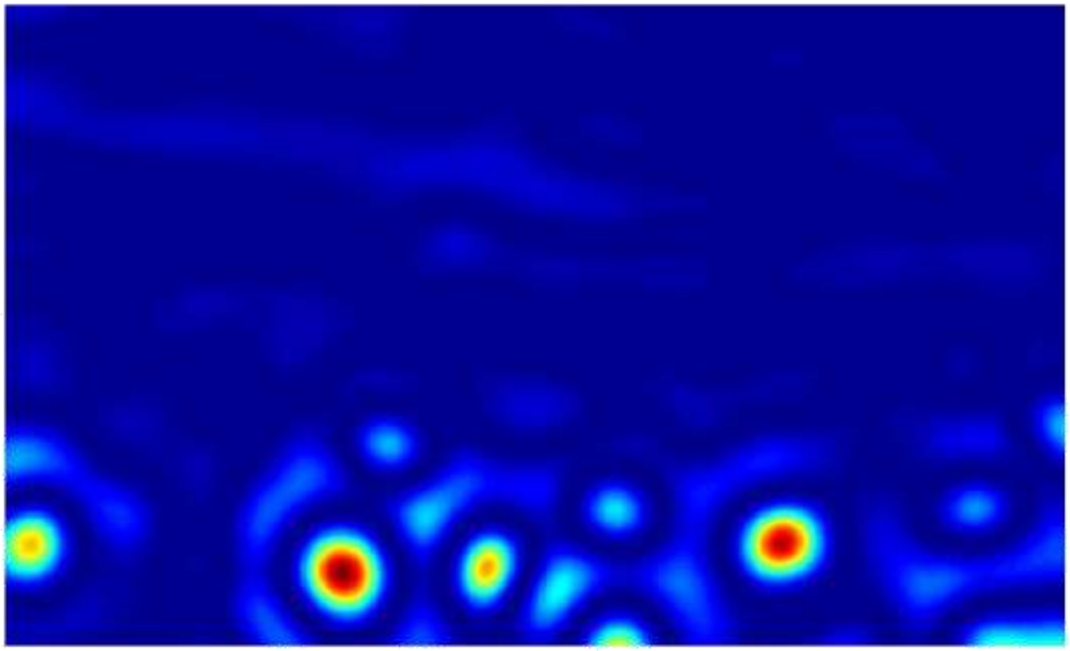



sigma=9.8
x 10^-3
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
100
80
60
40
20
0
0
20
40
60
80
100





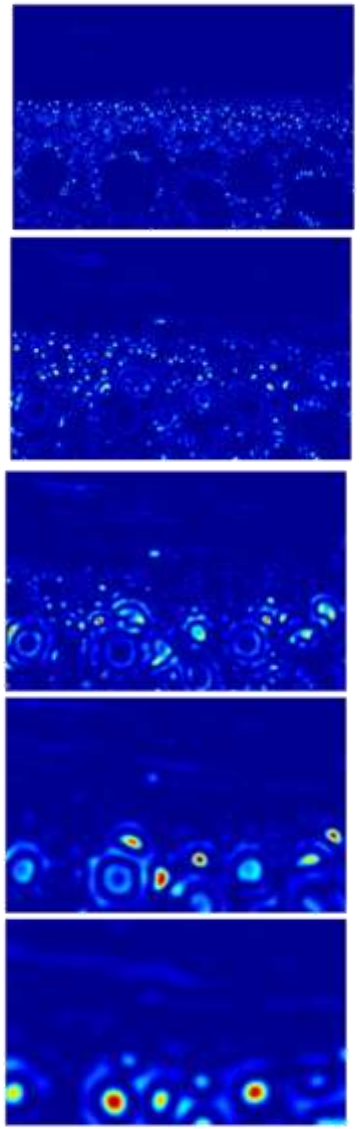

sigma=15.5
x 10^-4
2
0
-2
-4
-6
-8
-10
-12
100
80
60
40
20
0
0
20
40
60
80
100

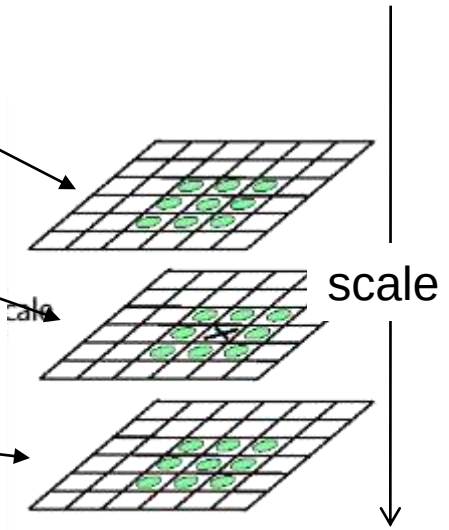


Scale invariant interest points

Interest points are local maxima in both position and scale.

$$L_{xx}(\sigma)+L_{yy}(\sigma)$$





Squared filter response maps

⇒ **List of** ***(x, y, σ)***

Template Matching with Image Pyramids

Input: Image, Template

1. Match template at current scale

2. Downsample image

3. Repeat 1-2 until image is very small

4. Take responses above some threshold, perhaps with non-maxima suppression

Scale-space blob detector: Example

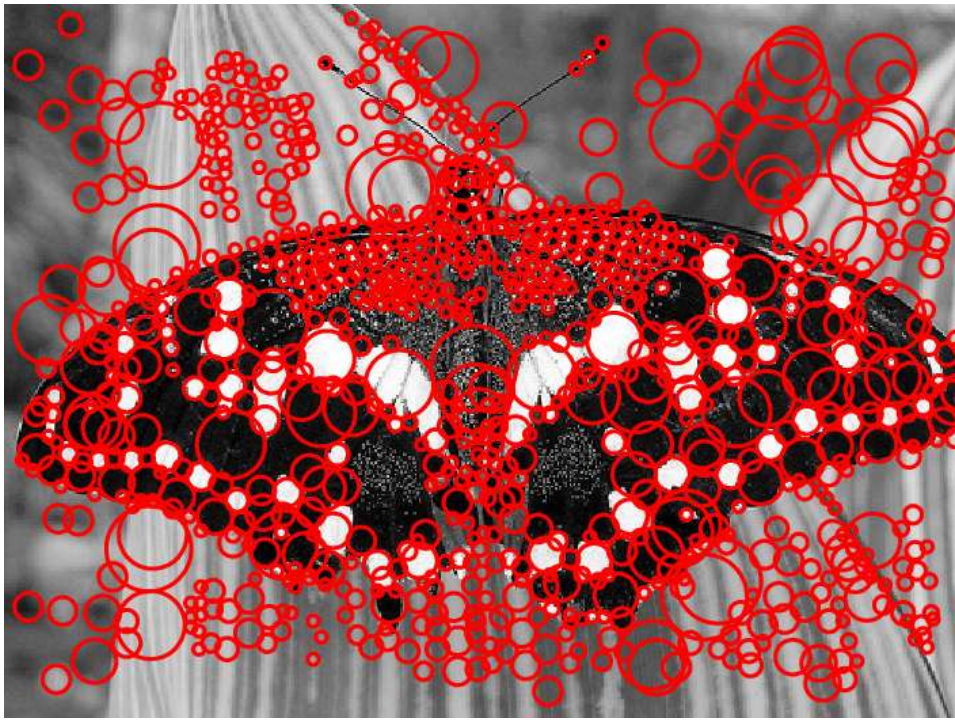


Image credit: Lana Lazebnik

Summary

- Edge Detection
  - Non-maximum suppression on gradient.
  - Zero-crossings for Laplacian.
  - Scale invariance
    - Gaussian pyramids. Coarse-to-fine search, multi-scale detection.
    - Laplacian pyramid. More compact image representation

Image Structures

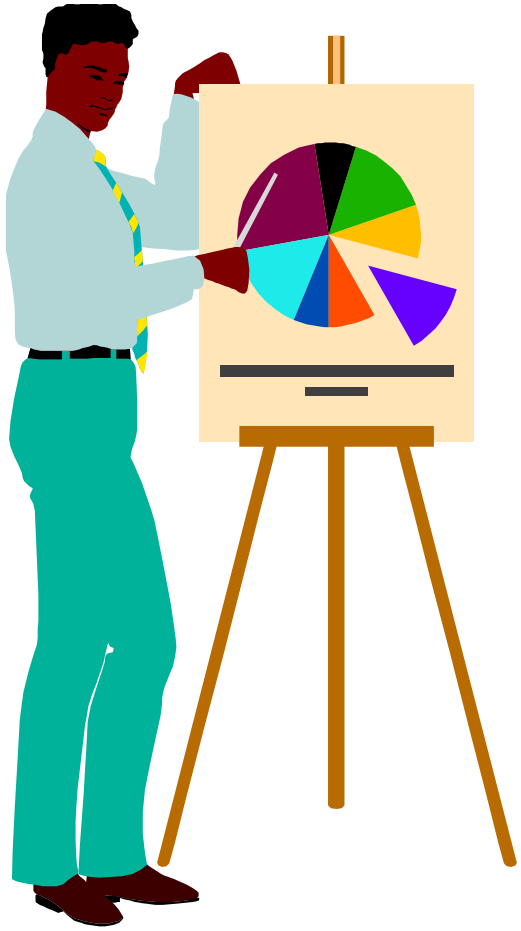
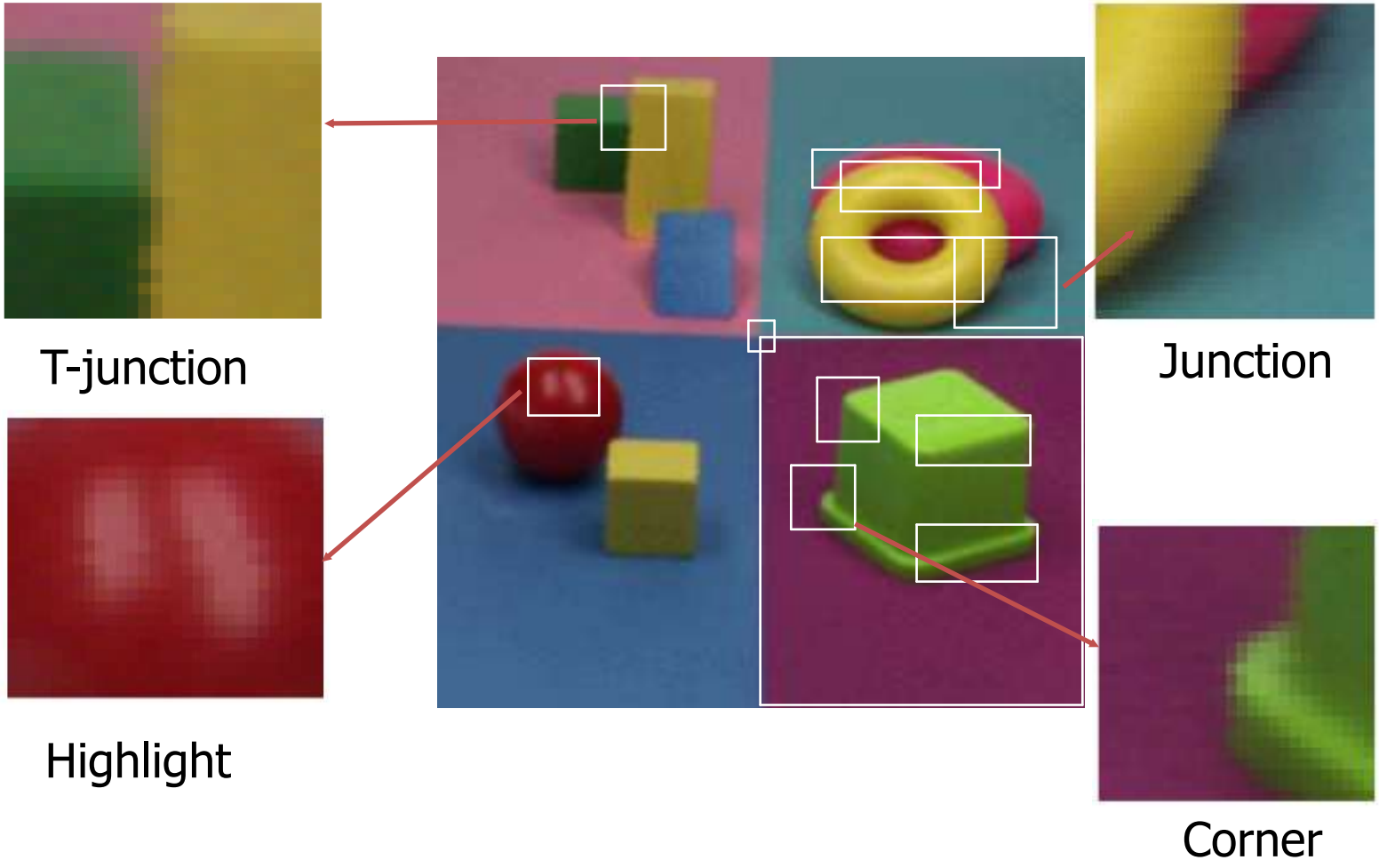


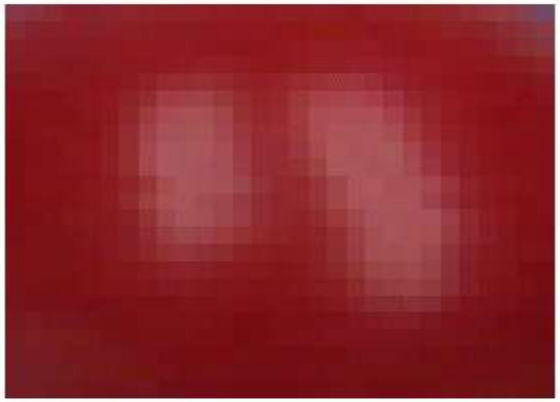
Image Structures

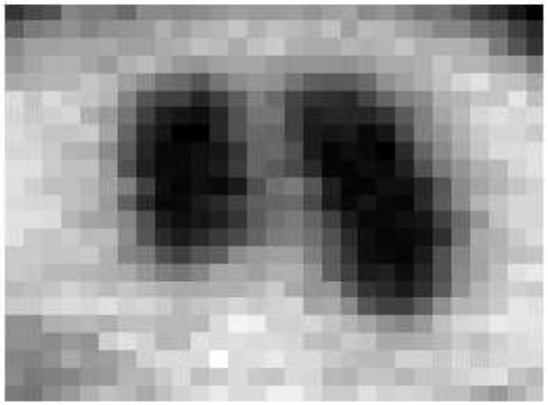


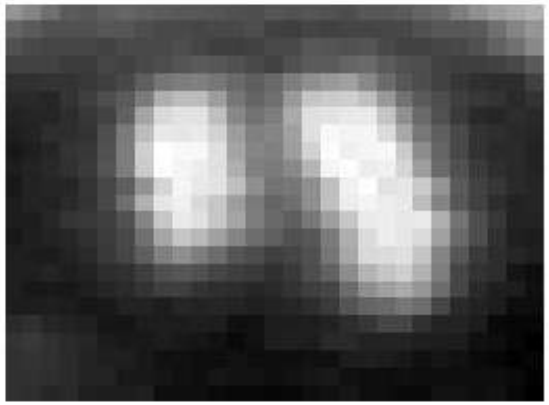
Classification of Highlights

Properties

- Hue remains the same
- Saturation will decrease
- Brightness will increase







Classification of Highlights

$f_x(\mathbf{p}_X, \mathbf{p}_Y) = 0$ and $f_y(\mathbf{p}_X, \mathbf{p}_Y) = 0$

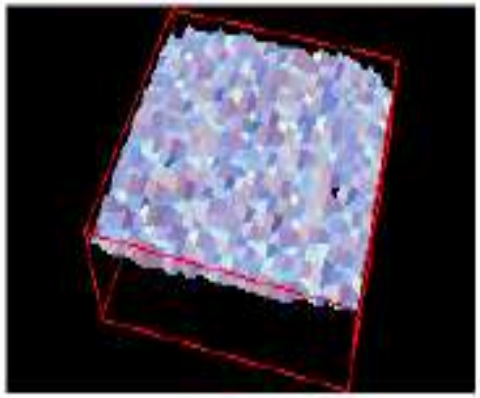
where $d = f_{xx}(\mathbf{p}_X, \mathbf{p}_Y) f_{yy}(\mathbf{p}_X, \mathbf{p}_Y) - [f_{xy}(\mathbf{p}_X, \mathbf{p}_Y)]^2$

then

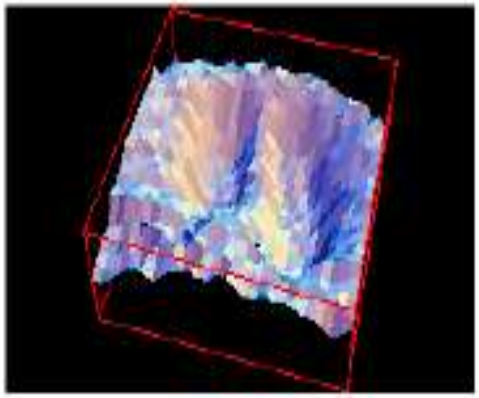
i. $f(\mathbf{p}_X, \mathbf{p}_Y)$ is a relative minimum if $d > 0$ and $f_{xx}(\mathbf{p}_X, \mathbf{p}_Y) > 0$

ii. $f(\mathbf{p}_X, \mathbf{p}_Y)$ is a relative maximum if $d > 0$ and $f_{xx}(\mathbf{p}_X, \mathbf{p}_Y) < 0$

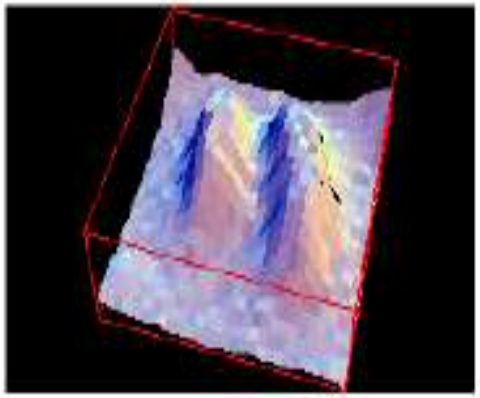
iii. $f(\mathbf{p}_X, \mathbf{p}_Y)$ is a saddle point if $d < 0$



H



S



I

N-jet: description of local image patches

Taylor series (second order):

$$f(\mathbf{p}_X + x, \mathbf{p}_Y + y) \approx f(\mathbf{p}_X, \mathbf{p}_Y) + x f_x(\mathbf{p}_X, \mathbf{p}_Y) + y f_y(\mathbf{p}_X, \mathbf{p}_Y) + \frac{1}{2} x^2 f_{xx}(\mathbf{p}_X, \mathbf{p}_Y) + xy f_{xy}(\mathbf{p}_X, \mathbf{p}_Y) + \frac{1}{2} y^2 f_{yy}(\mathbf{p}_X, \mathbf{p}_Y)$$

Taylor series (second order) gauge coordinates:

$$\begin{pmatrix} f_{ww} & f_{vw} \\ f_{vw} & f_{ww} \end{pmatrix} = \frac{1}{f_x^2 + f_y^2} \begin{pmatrix} f_x^2 f_{yy} - 2 f_x f_y f_{xy} + f_x^2 f_{xx} & (f_y^2 - f_x^2) f_{xy} + f_x f_y (f_{xx} + f_{yy}) \\ (f_y^2 - f_x^2) f_{xy} + f_x f_y (f_{xx} - f_{yy}) & f_x^2 f_{xx} - 2 f_x f_y f_{xy} + f_y^2 f_{xx} \end{pmatrix}$$

Hessian and curvature gauge

Hessian :

$$H = \begin{pmatrix} f_{xx} & f_{yy} \\ f_{xy} & f_{yx} \end{pmatrix}$$

eigenvalues are $f_{xx} - f_{yy} \pm \sqrt{(f_{xx} + f_{yy})^2 + 4 f_{xy}^2}$

eigenvalues in gauge coordinates are

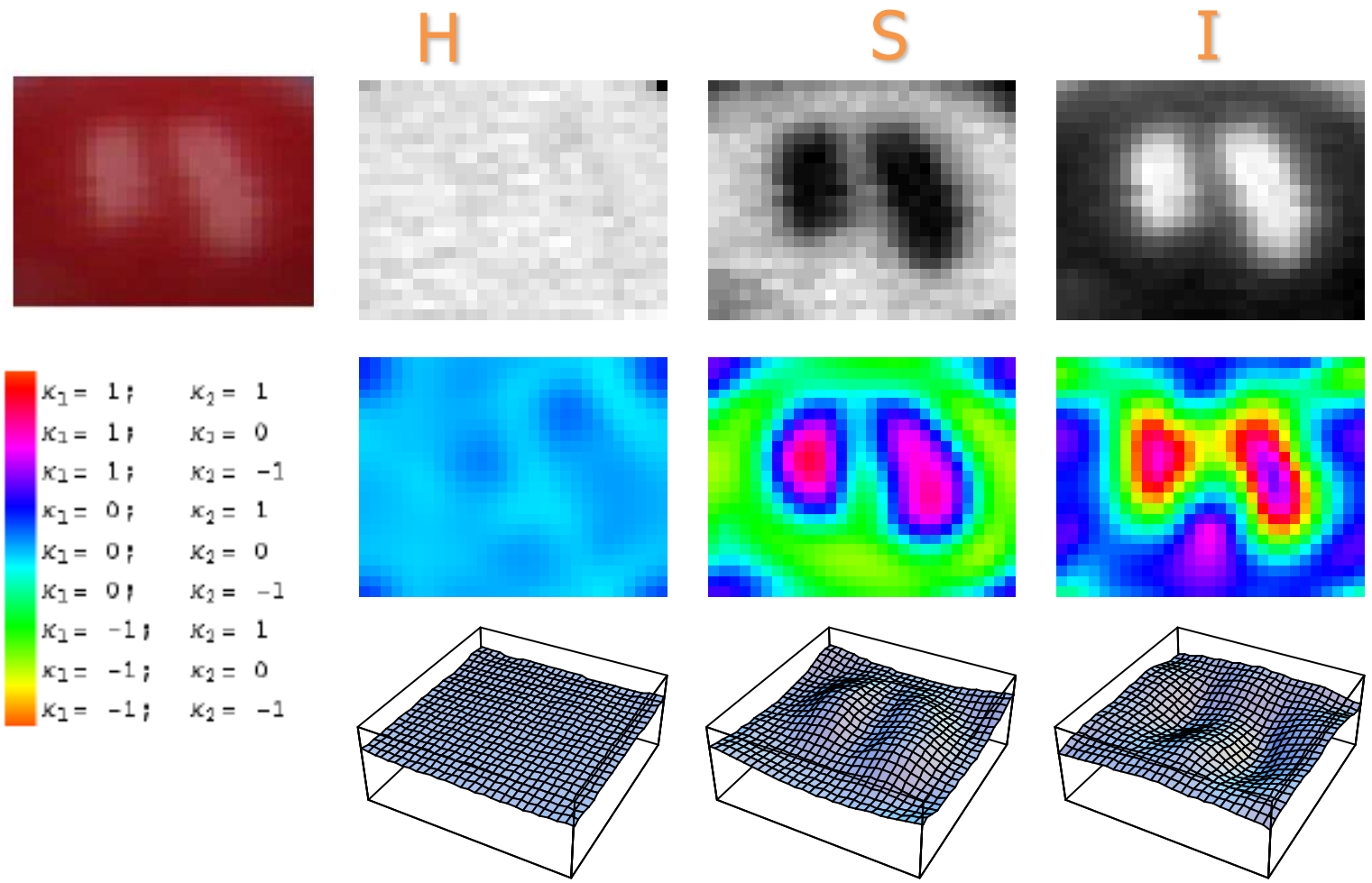
$$\kappa_1 = f_{xx} + f_{yy} - \sqrt{(f_{xx} + f_{yy})^2 + 4 f_{xy}^2}$$

$$\kappa_2 = f_{xx} + f_{yy} - \sqrt{(f_{xx} + f_{yy})^2 + 4 f_{xy}^2}$$

Higher-order N-jet: blobs, bars, ridges, saddle points

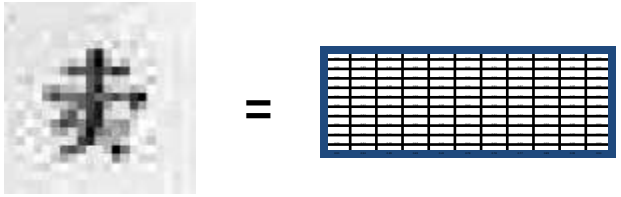
- Dark blob on bright background: $f_w \approx 0, \kappa_1 \approx \kappa_2 > 0$
- Dark bar on bright background: $f_w \approx 0, \kappa_1 \approx 0, \kappa_2 > 0$
- Bright blob on dark background: $f_w \approx 0, \kappa_1 \approx \kappa_2 < 0$
- Bright bar on dark background: $f_w \approx 0, \kappa_1 < 0, \kappa_2 \approx 0$
- Constant patch: $f_w \approx 0, \kappa_1 \approx 0, \kappa_2 \approx 0$
- Saddle point: $f_w \approx 0, \kappa_1 < 0, \kappa_2 > 0$

Classification of Highlights

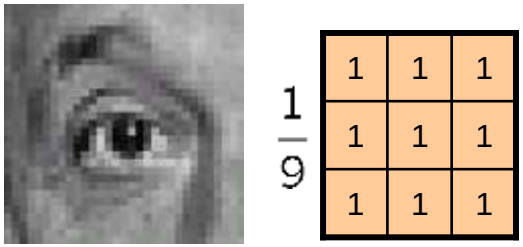


Summary

- Image is a matrix of numbers



- Linear filtering is sum of dot product at each position
  - Can smooth, sharpen, translate (among many other uses)



- Be aware of details for filter size, extrapolation, cropping