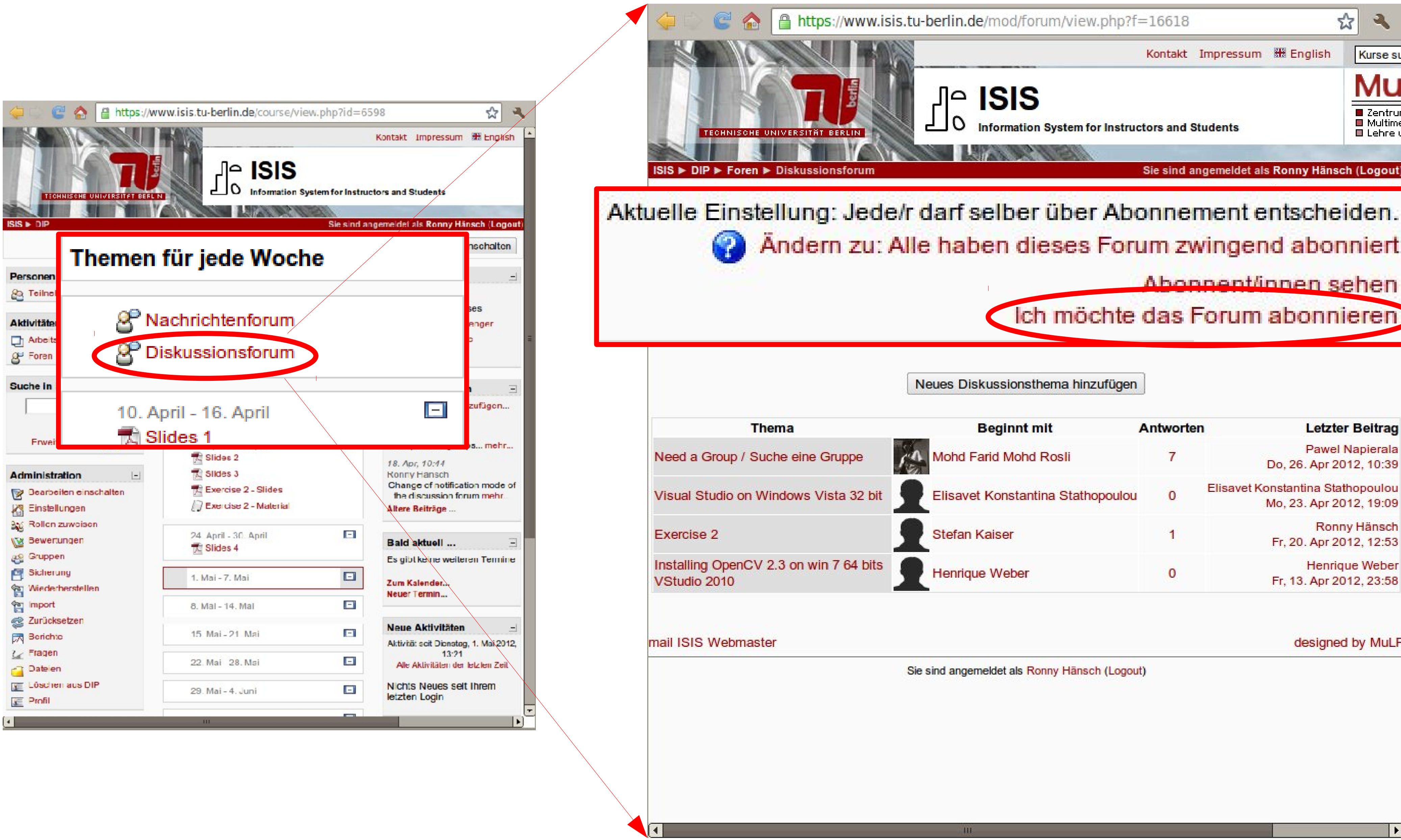
Digital Image Processing

Berlin University of Technology (TUB),
Computer Vision and Remote Sensing Group
Berlin, Germany

Orga

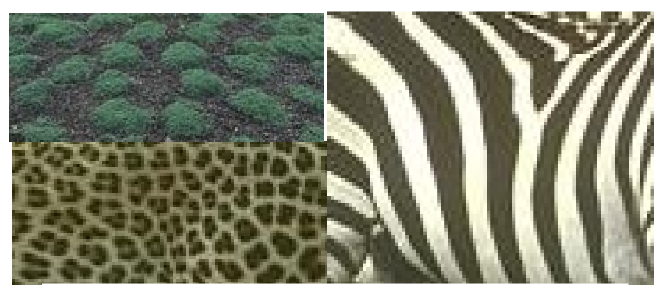

https://www.isis.tu-berlin.de/course/view.php?id=6598
Kontakt Impressum English
ISIS
Information System for Instructors and Students
TECHNISCHE UNIVERSITÄT BERLIN
ISIS ▶ DIP
Sie sind angemeldet als Ronny Hänsch (Logout)
Themen für jede Woche
Nachrichtenforum
Diskussionsforum
10. April - 16. April
Slides 1
Slides 2
Slides 3
Exercise 2 - Slides
Exercise 2 - Material
24. April - 30. April
Slides 4
1. Mai - 7. Mai
8. Mai - 14. Mai
15. Mai - 21. Mai
22. Mai - 28. Mai
29. Mai - 4. Juni
Personen
Aktivitäten
Foren
Suche in
Administration
Bearbeiten einschalten
Einstellungen
Rollen zuweisen
Bewertungen
Gruppen
Sicherung
Wiederherstellen
Import
Zurücksetzen
Berichte
Fragen
Dateien
Löschen aus DIP
Profil
18. Apr, 10:44
Ronny Hänsch
Change of notification mode of the discussion forum mehr...
Ältere Beiträge ...
Bald aktuell ...
Es gibt keine weiteren Termine
Zum Kalender...
Neuer Termin...
Neue Aktivitäten
Aktivität seit Dienstag, 1. Mai 2012, 13:21
Alle Aktivitäten der letzten Zeit
Nichts Neues seit Ihrem letzten Login
https://www.isis.tu-berlin.de/mod/forum/view.php?f=16618
Kontakt Impressum English
ISIS
Information System for Instructors and Students
TECHNISCHE UNIVERSITÄT BERLIN
ISIS ▶ DIP ▶ Foren ▶ Diskussionsforum
Sie sind angemeldet als Ronny Hänsch (Logout)
Aktuelle Einstellung: Jede/r darf selber über Abonnement entscheiden.
Ändern zu: Alle haben dieses Forum zwingend abonniert
Abonnent/innen sehen
Ich möchte das Forum abonnieren
Neues Diskussionsthema hinzufügen
Thema | Beginnt mit | Antworten | Letzter Beitrag
Need a Group / Suche eine Gruppe | Mohd Farid Mohd Rosli | 7 | Pawel Napierala Do, 26. Apr 2012, 10:39
Visual Studio on Windows Vista 32 bit | Elisavet Konstantina Stathopoulou | 0 | Elisavet Konstantina Stathopoulou Mo, 23. Apr 2012, 19:09
Exercise 2 | Stefan Kaiser | 1 | Ronny Hänsch Fr, 20. Apr 2012, 12:53
Installing OpenCV 2.3 on win 7 64 bits VStudio 2010 | Henrique Weber | 0 | Henrique Weber Fr, 13. Apr 2012, 23:58
mail ISIS Webmaster
designed by MuLF
Sie sind angemeldet als Ronny Hänsch (Logout)

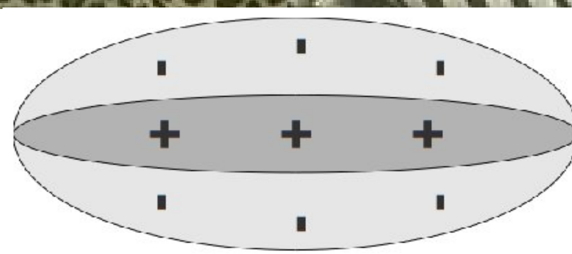
Outlook

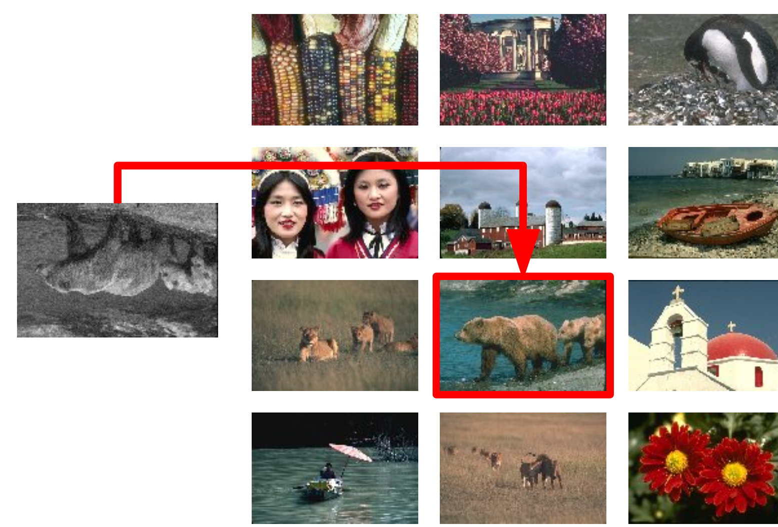
1. How to detect edges and similar image structures?
   → Directional gradients (etc.)

2. How to use this information to describe an image?
   → Texture
   → Textons

3. How to use image descriptor in an application?
   → Image retrieval

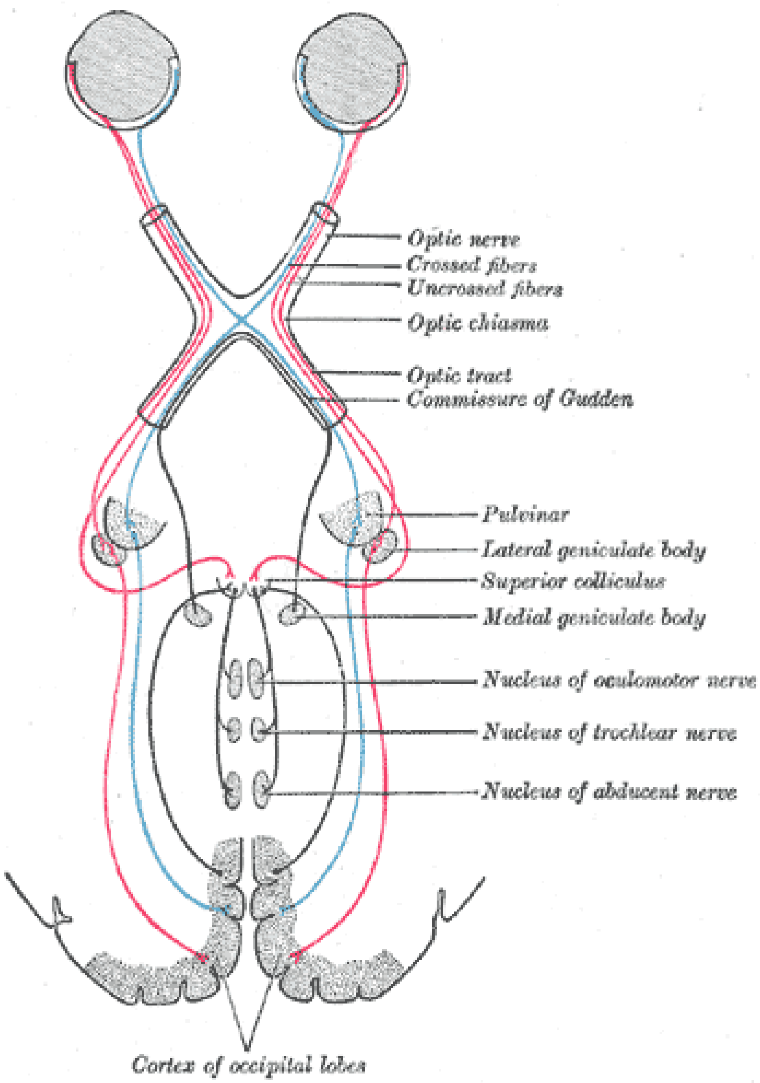


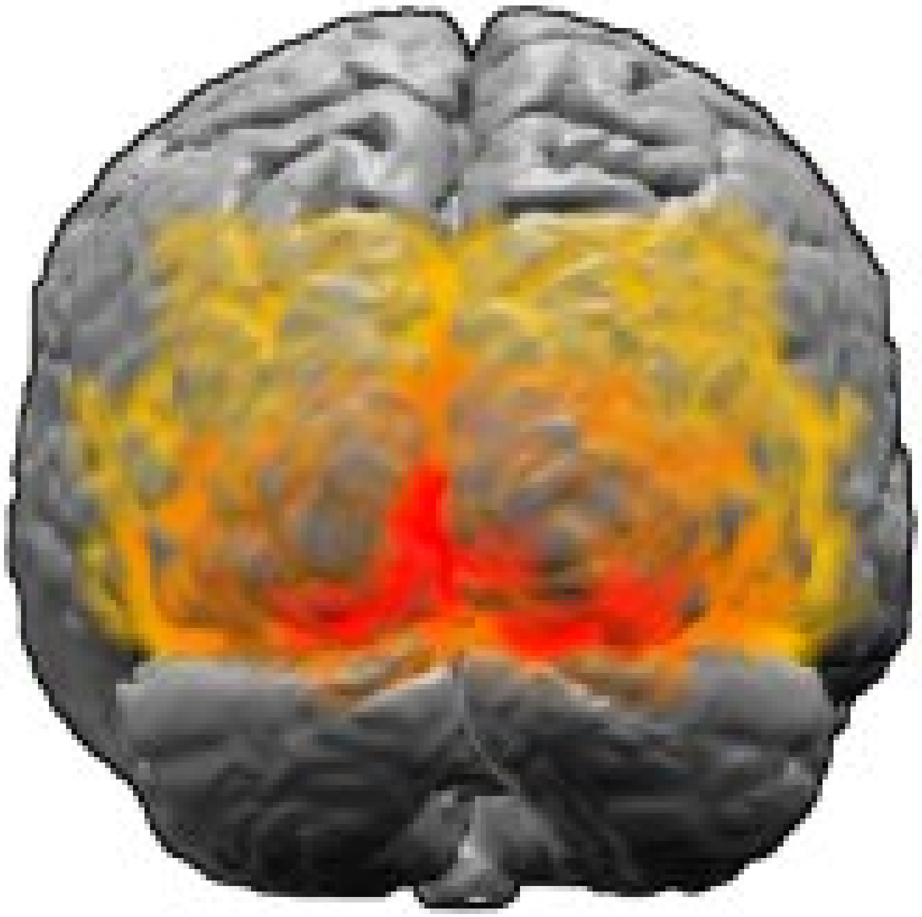




(one possible) Motivation

Currently best performing visual system humans have access to



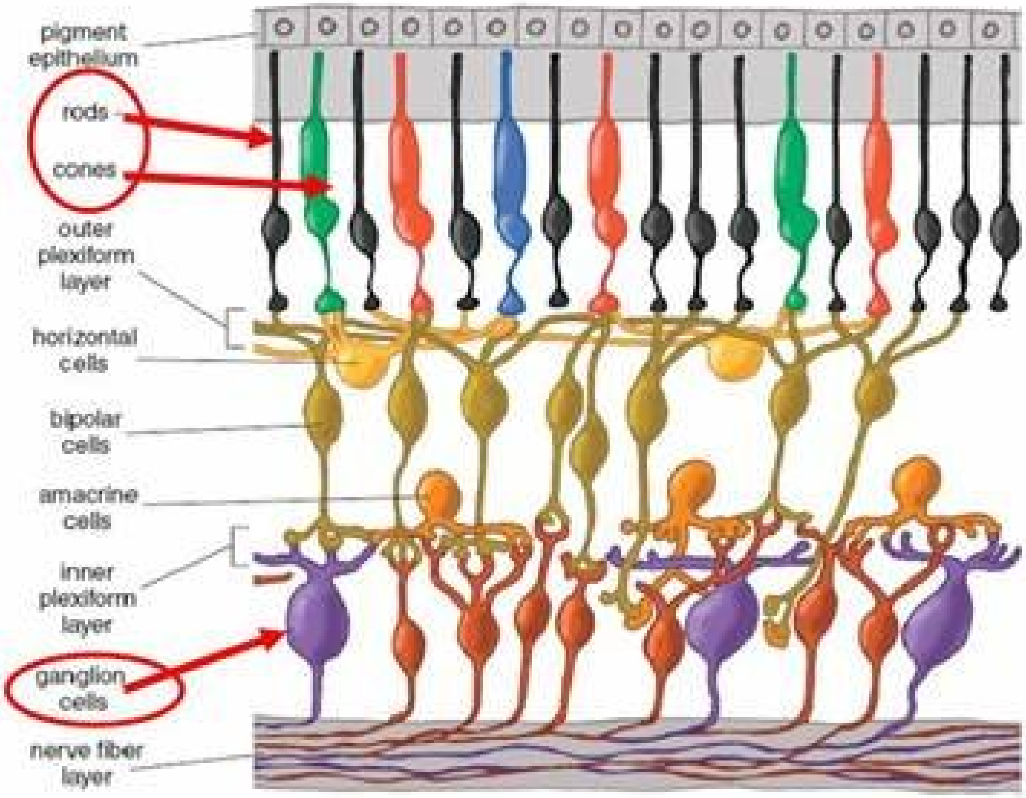


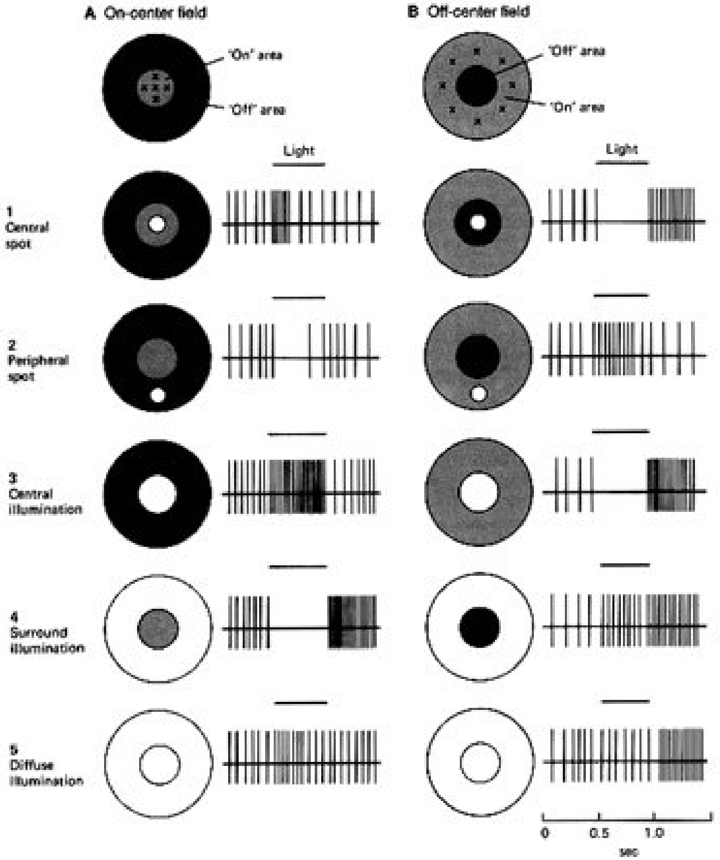
Brodmann area 17 (red)
Brodmann area 18 (orange)
Brodmann area 19 (yellow)

(one possible) Motivation

Receptive field of neurons

- Auditory, somatosensory, and visual system
- Presence of stimuli change state of neuron
- e.g. retinal ganglion cells



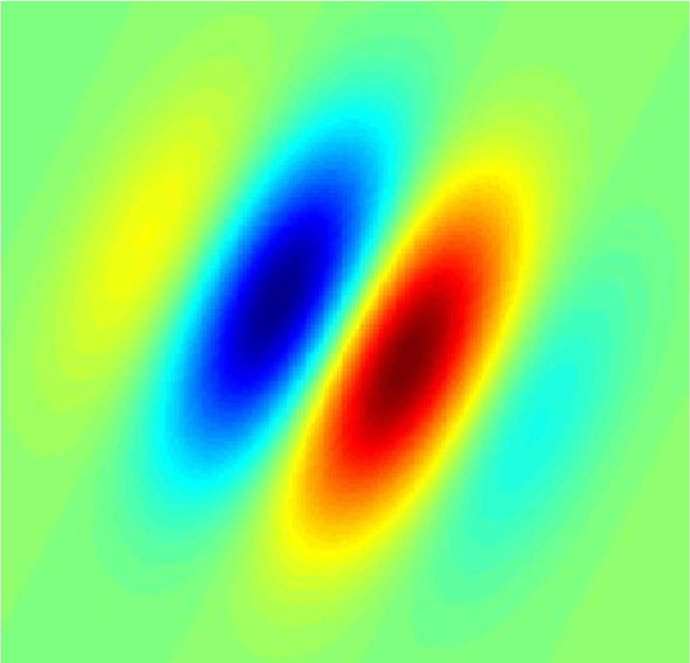


Kolb, Helga. “How the retina works.” American Scientist. 91: 28-35. (2003)

(one possible) Motivation

Receptive field of neurons

- e.g. simple cells in primary visual cortex of mammals



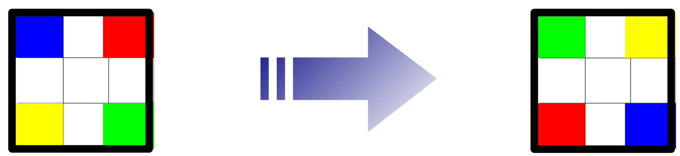
- Inhibitory and excitatory areas
- Zero activation under diffuse lighting
- Optimal stimulus: oriented edges
- Integration over spatial support

Can be modelled as **convolution** with corresponding filter kernel!

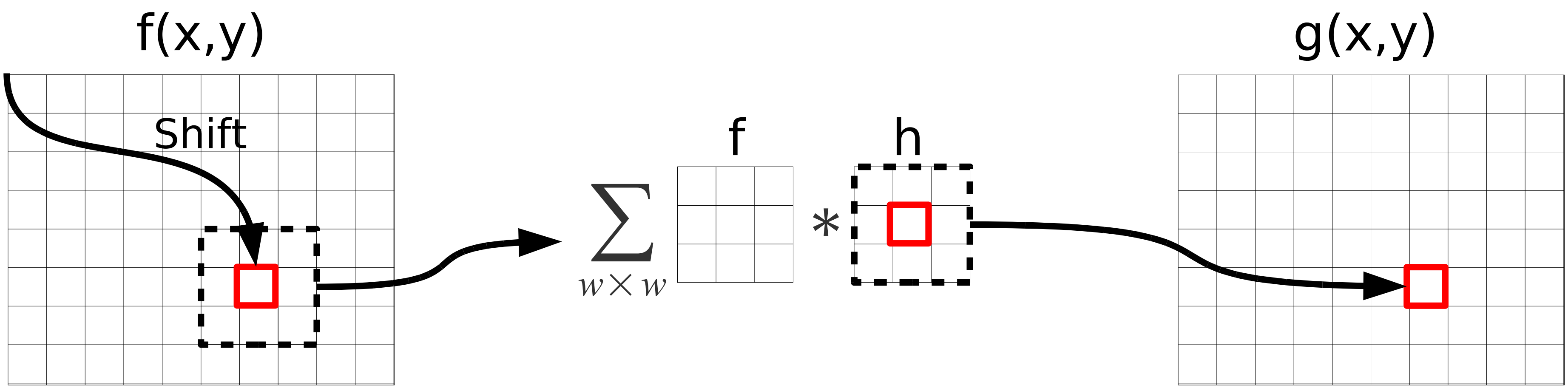
Convolution

$$g(\alpha,\beta)=\sum_{x=1}^{N}\sum_{y=1}^{M} f(x,y)\cdot h(x-\alpha,y-\beta)$$

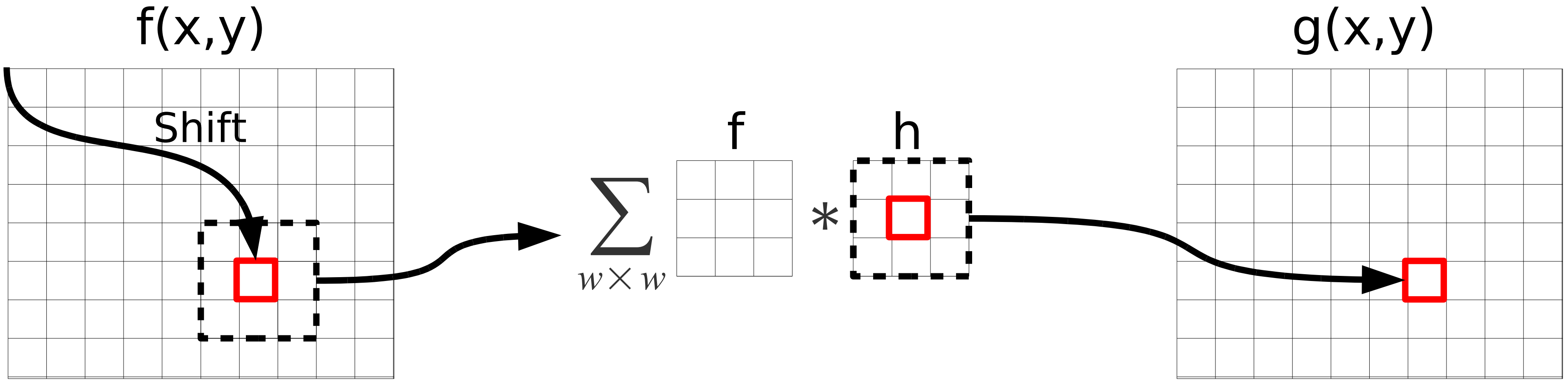
1. Flip filter kernel (about the filter centre)



2. Shift (re-centre), **Multiply** and **Integrate**



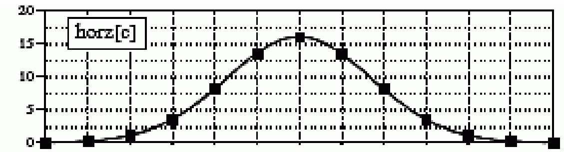
Convolution

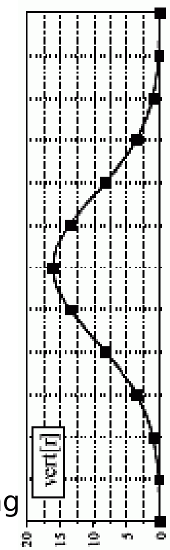


Time complexity quadratic
in terms of kernel size!

Solution: **Linear separability**

$$\begin{bmatrix} a\cdot A & b\cdot A & c\cdot A \\ a\cdot B & b\cdot B & c\cdot B \\ a\cdot C & b\cdot C & c\cdot C \end{bmatrix} = \begin{bmatrix} A \\ B \\ C \end{bmatrix} \cdot \begin{bmatrix} a & b & c \end{bmatrix}$$

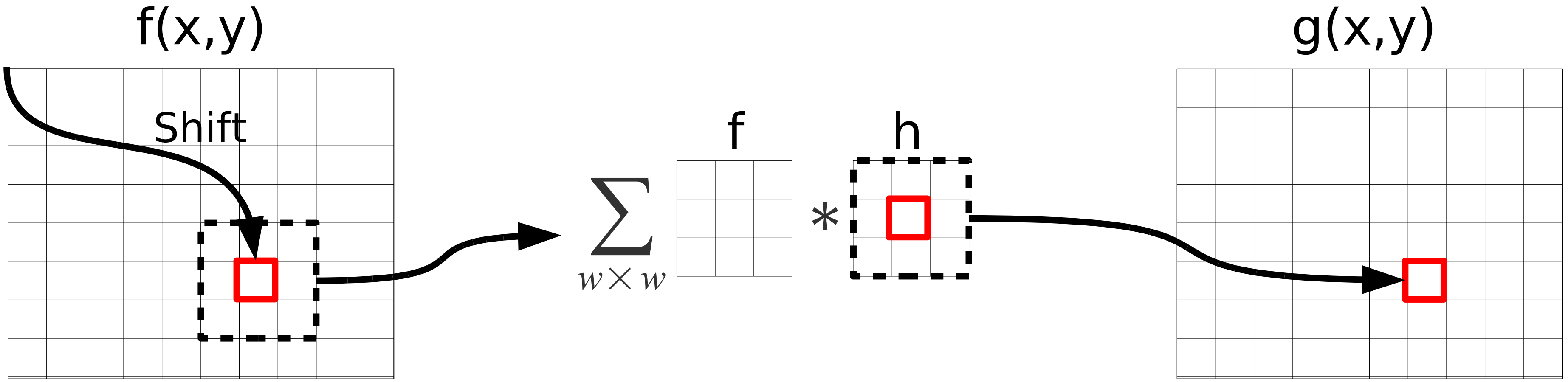




	0.04	0.25	1.11	3.56	8.20	13.5	16.0	13.5	8.20	3.56	1.11	0.25	0.04
0.04	0	0	0	0	0	1	1	1	0	0	0	0	0
0.25	0	0	0	1	2	3	4	3	2	1	0	0	0
1.11	0	0	1	4	9	15	18	15	9	4	1	0	0
3.56	0	1	4	13	29	48	57	48	29	13	4	1	0
8.20	0	2	9	29	67	111	131	111	67	29	9	2	0
13.5	1	3	15	48	111	183	216	183	111	48	15	3	1
16.0	1	4	18	57	131	216	255	216	131	57	18	4	1
13.5	1	3	15	48	111	183	216	183	111	48	15	3	1
8.20	0	2	9	29	67	111	131	111	67	29	9	2	0
3.56	0	1	4	13	29	48	57	48	29	13	4	1	0
1.11	0	0	1	4	9	15	18	15	9	4	1	0	0
0.25	0	0	0	1	2	3	4	3	2	1	0	0	0
0.04	0	0	0	0	0	1	1	1	0	0	0	0	0

The Scientist and Engineer's
Guide to Digital Signal Processing
By Steven W. Smith

Convolution



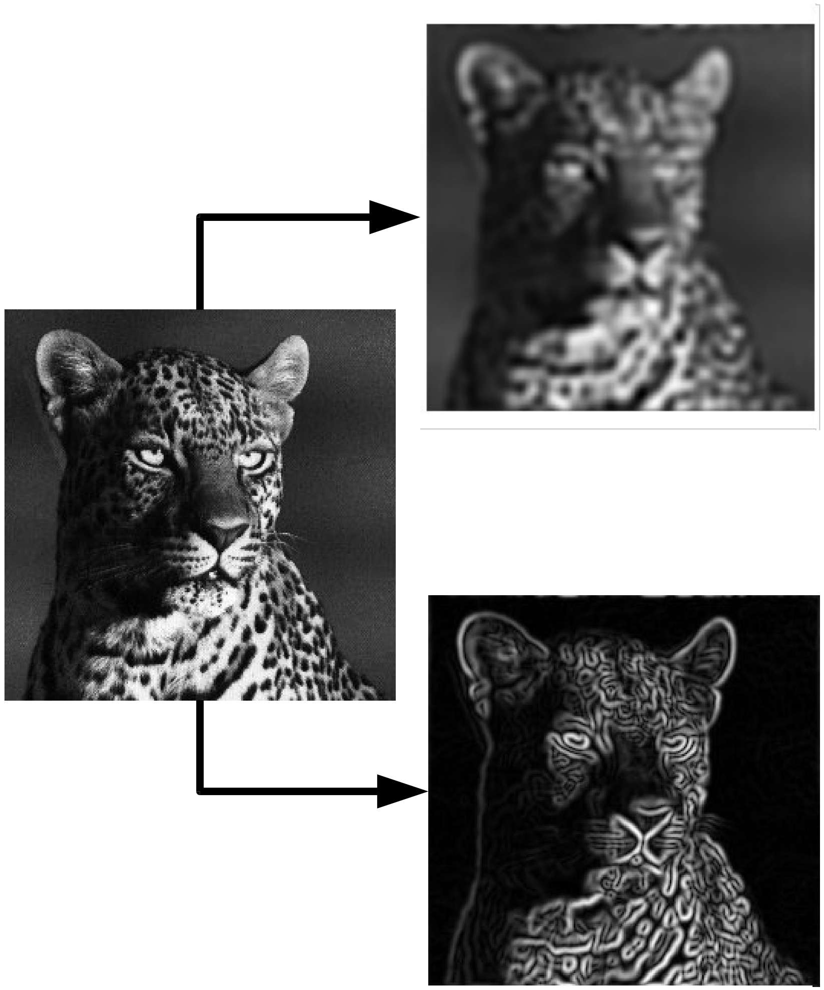
Time complexity is <u>quadratic</u> in terms of kernel size!

Solution: **<u>Linear separability</u>**

1. Convolve image rows with horizontal filter
2. Convolve result columns with vertical filter

Time complexity is <u>linear</u> in terms of kernel size!

Edge Detection



- Smoothing leads to blurring
  - edge suppression
  - Local 'Averaging' or 'Integration'

- Edges: Changes in image intensity
- Edge enhancement: Differentiation
  - The opposite of integration
- Definition in terms of 1D derivatives:

$$\nabla f(x,y) = \begin{pmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{pmatrix}$$

Edge Detection

$$\nabla f(x,y) = \begin{pmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{pmatrix}$$

- 2-Element vector for each pixel in the original image
- Gradient magnitude: Rate of change of intensity
  - Strong edges associated with large magnitude

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

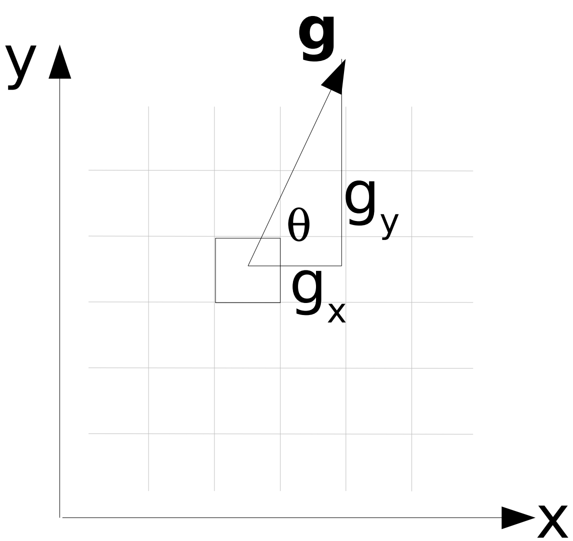
- Gradient direction: Direction of fastest intensity increase
  - At right angles to edge in image

$$\phi(f(x,y)) = \tan^{-1}\left(\frac{\partial f}{\partial y}, \frac{\partial f}{\partial x}\right)$$

Edge Detection

- **Directional gradients:** Convolution with suitable filters, e.g. G_x and G_y
  - **Image ⊗ G_x** → Gradient in x direction
  - **Image ⊗ G_y** → Gradient in y direction

- Each pixel is associated with a gradient vector **g** = $(g_x, g_y)^T$



- Gradient magnitude: $|\mathrm{g}| = \sqrt{g_x^2 + g_y^2}$
- Gradient direction: $\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$
  - Direction in which intensity increases quickest

Edge Detection

Direct Computation of Derivatives: Central Differencing

- Simple definition of derivatives from Taylor series:

$$\frac{\partial}{\partial x} f(x, y) = \frac{1}{2}\left(f(x+1, y) - f(x-1, y)\right)$$

$$\frac{\partial}{\partial y} f(x, y) = \frac{1}{2}\left(f(x, y+1) - f(x, y-1)\right)$$

  - → Average of direct left/right differences!

- Problem: Very noise-sensitive
  - → Small spatial support
  - → In practice: Consider larger local neighbourhood

Edge Detection

The Sobel Operator: First Order Derivatives

- A simple 'recipe' for calculating image gradients

$$\frac{\partial}{\partial x} f(x,y) = f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1) \\ - f(x-1,y-1) - 2f(x-1,y) - f(x-1,y+1)$$

$$\frac{\partial}{\partial y} f(x,y) = \ \ldots$$

- Simpler formulation in terms of convolution

$(\partial / \partial x) f = f * s_x$

s_x:

1	0	-1
2	$\boxed{0}$	-2
1	0	-1

$(\partial / \partial y) f = f * s_y$

s_y:

1	2	1
0	$\boxed{0}$	0
-1	-2	-1

Edge Detection

Example: Edge Detection by Sobel Operator

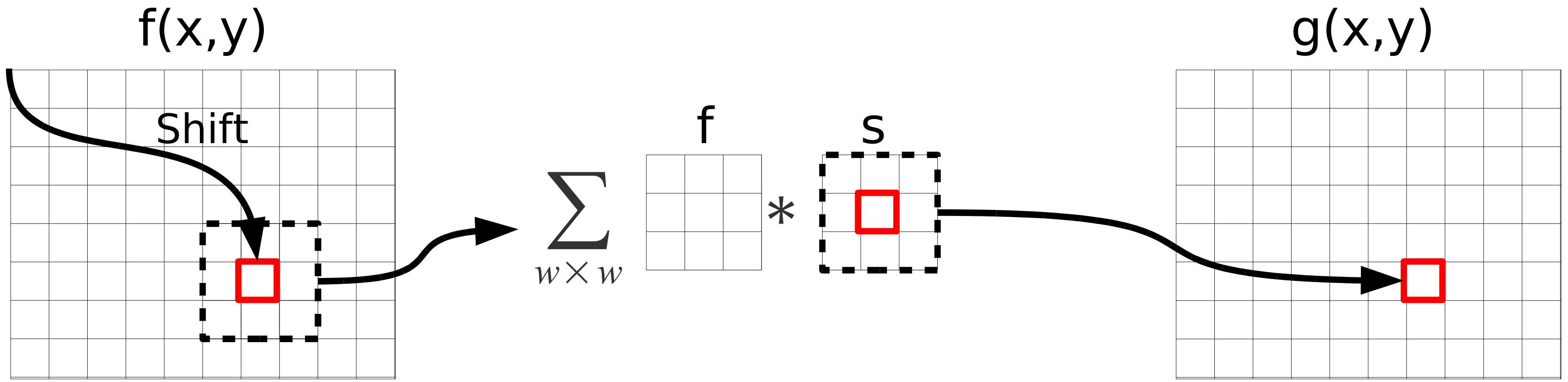
$$(\partial/\partial x) f = f * s_x$$

$$s_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & \boxed{0} & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$(\partial/\partial y) f = f * s_y$$

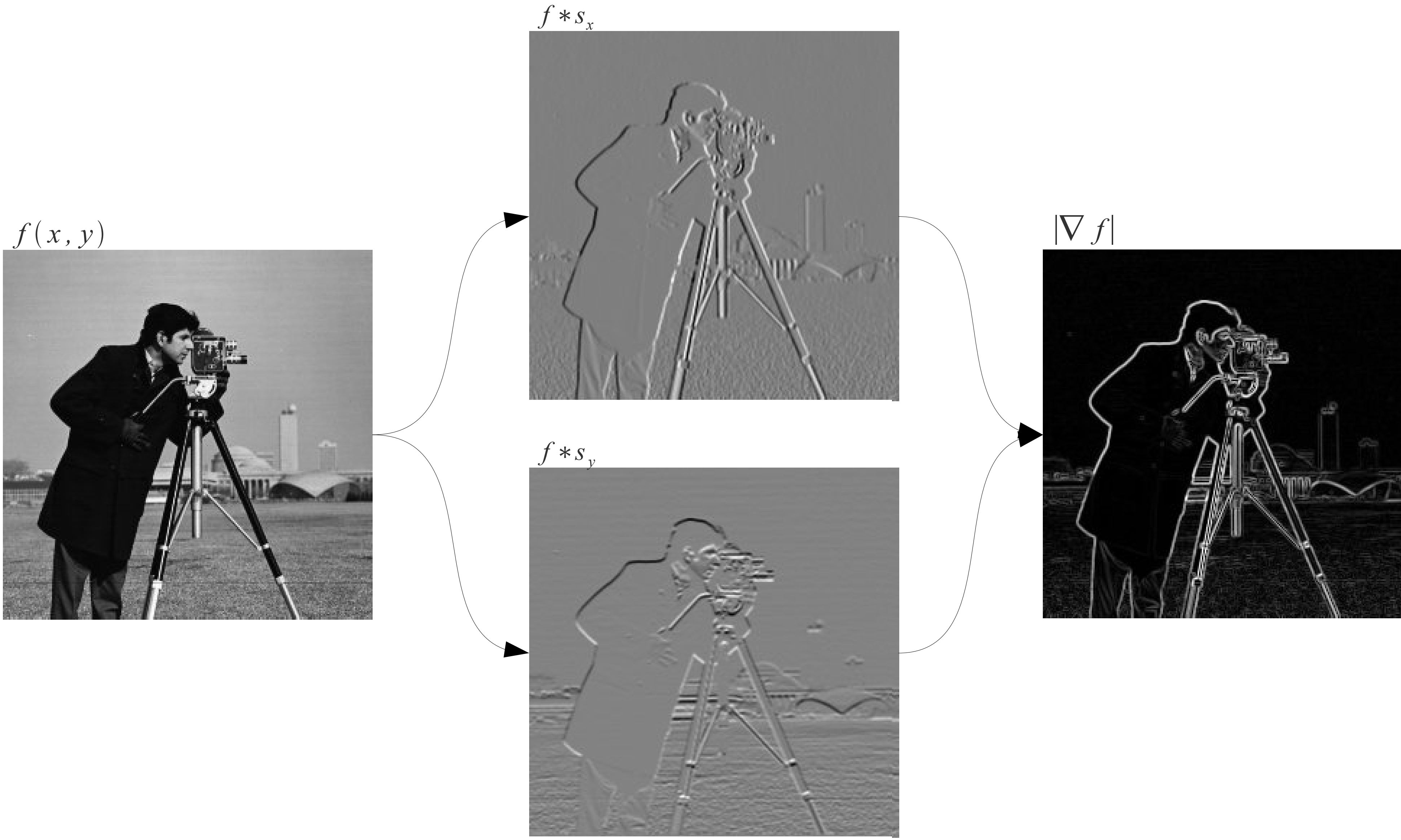
$$s_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & \boxed{0} & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Each pixel intensity is replaced by the local weighted sum...



Edge Detection

Example: Edge Detection by Sobel Operator



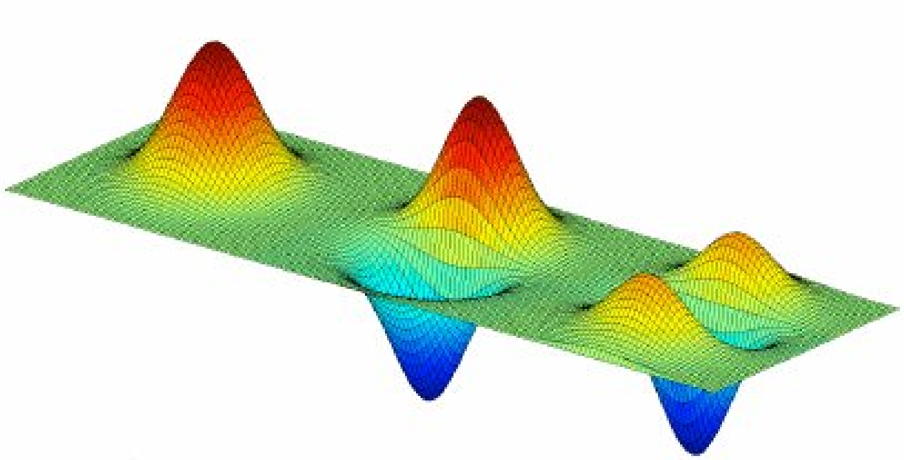
Edge Detection

- Commonly Used:
  Composition of differential operator and low-pass
- E.g. derivatives of the normal distribution:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

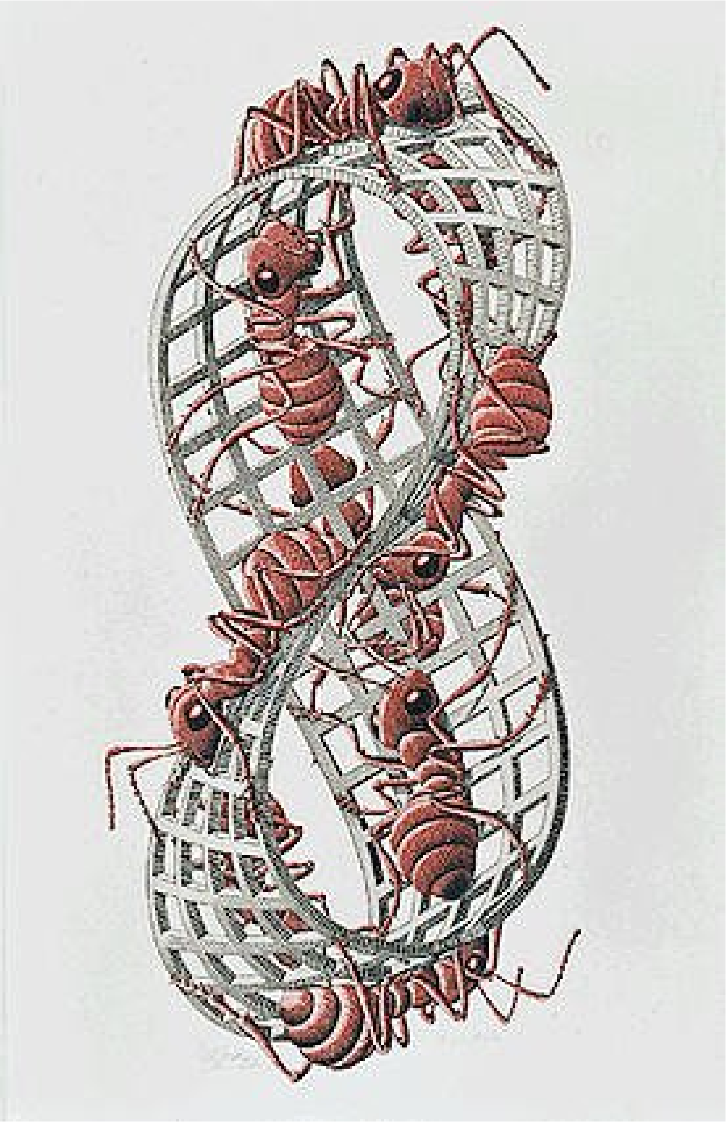
$$G_x(x, y) = \frac{\partial}{\partial x} G(x, y; \sigma) = \frac{-x}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

$$G_y(x, y) = \frac{\partial}{\partial y} G(x, y; \sigma) = \frac{-y}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

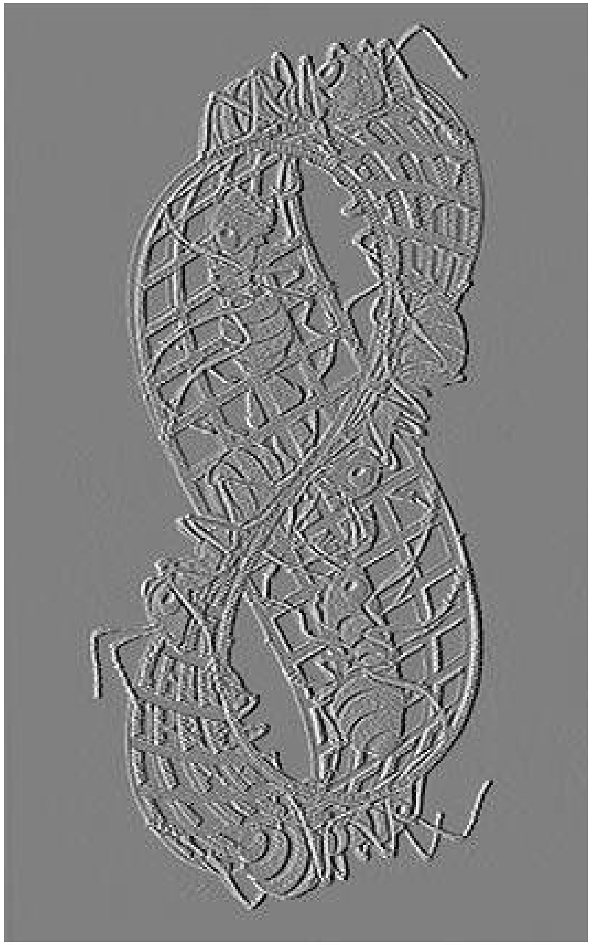


- σ: Scale and noise sensitivity
  - σ small: Small structures discernable, noise/texture preserved
  - σ large: Large structures emphasized, noise suppressed

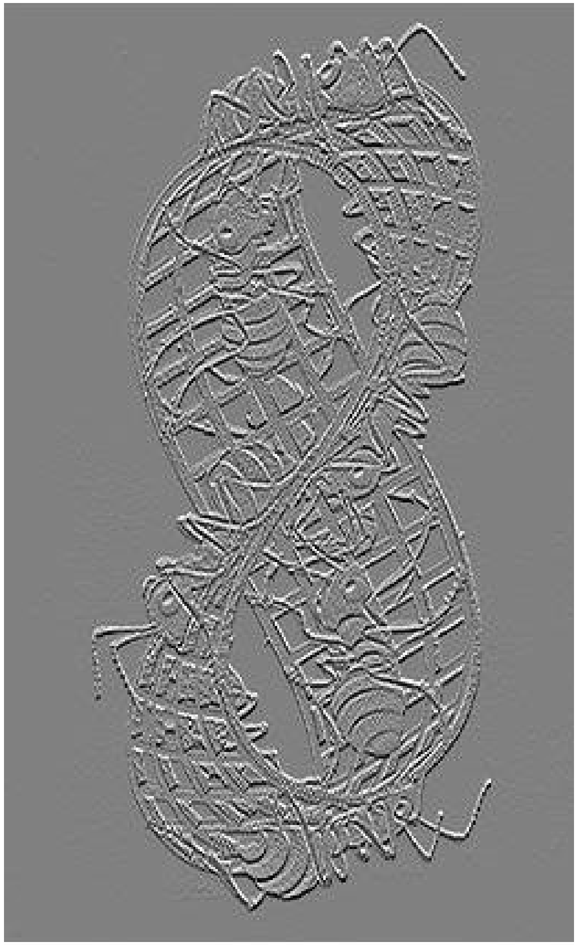
Edge Detection



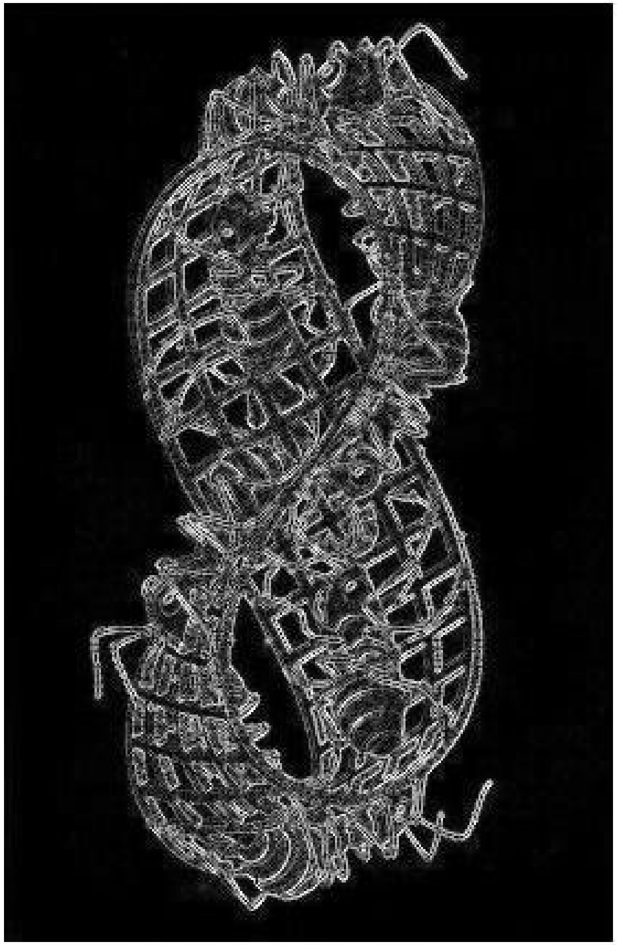
Edge Detection



g_x



g_y



$|\mathbf{g}|$

Edge Detection

g_x

g_y

$|\mathbf{g}|$

Texture

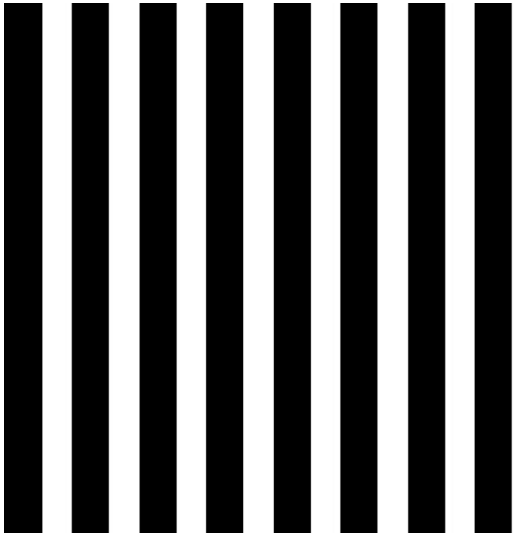
Texture in images:

Systematic spatial

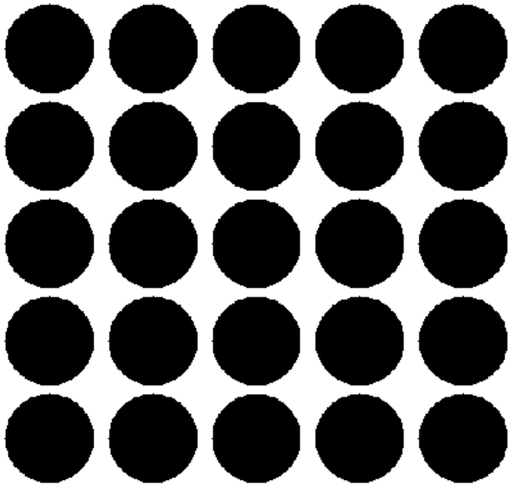
intensity variation

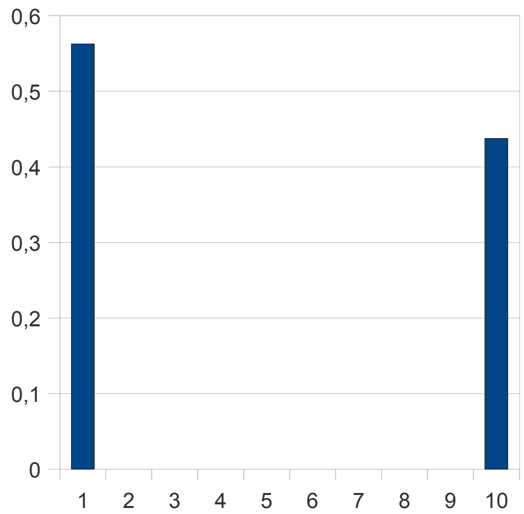
(Image parts taken from Berkeley Segmentation Dataset)

Texture

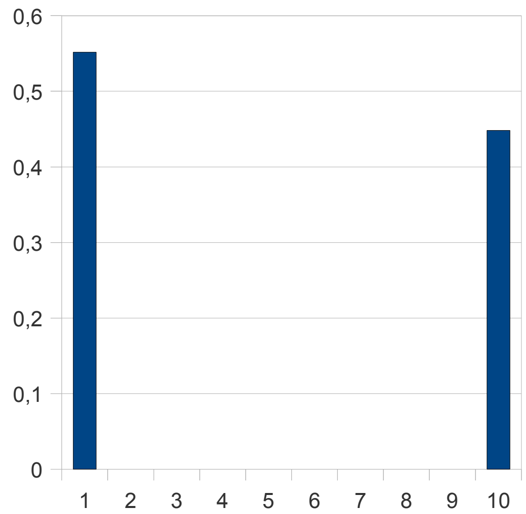


Spatial distribution is important,





but is lost if only radiometric information is taken into account.



Texture

Texture measures

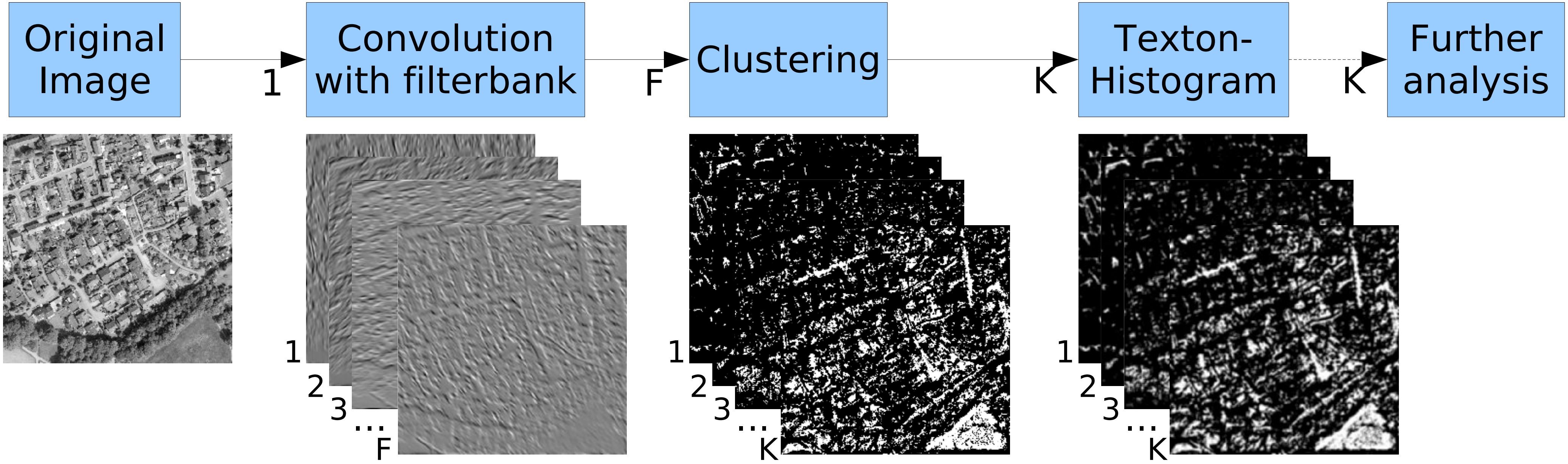
- Local statistics
  → Properties of local intensity distribution

- Histogram of Oriented Gradients (HoG)
  → Properties of local gradient distribution

- Gray-Level Co-Occurrence Matrices (GLCM)
  → Probability of intensity-pairs in a given spatial relation

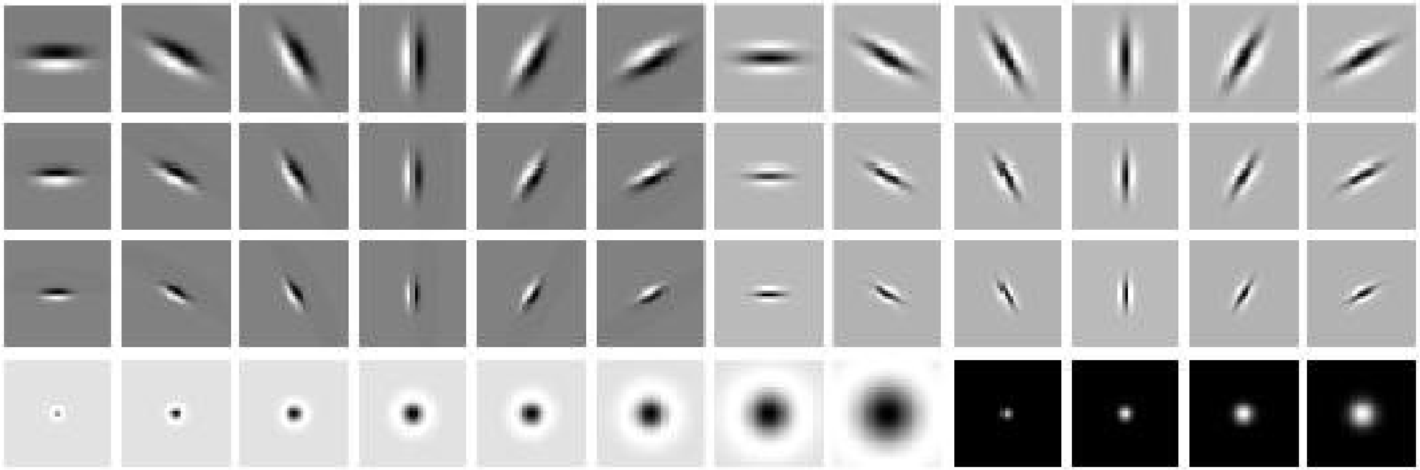
- **Textons**

Textons


Original Image
1
Convolution with filterbank
F
Clustering
K
Texton-Histogram
K
Further analysis
1
2
3
...
F
1
2
3
...
K
1
2
3
...
K

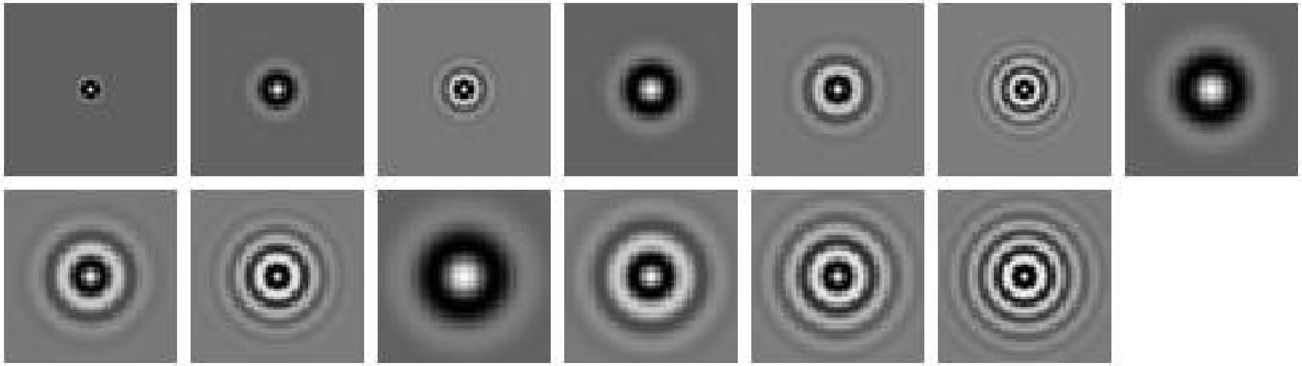
Textons

Filterbank

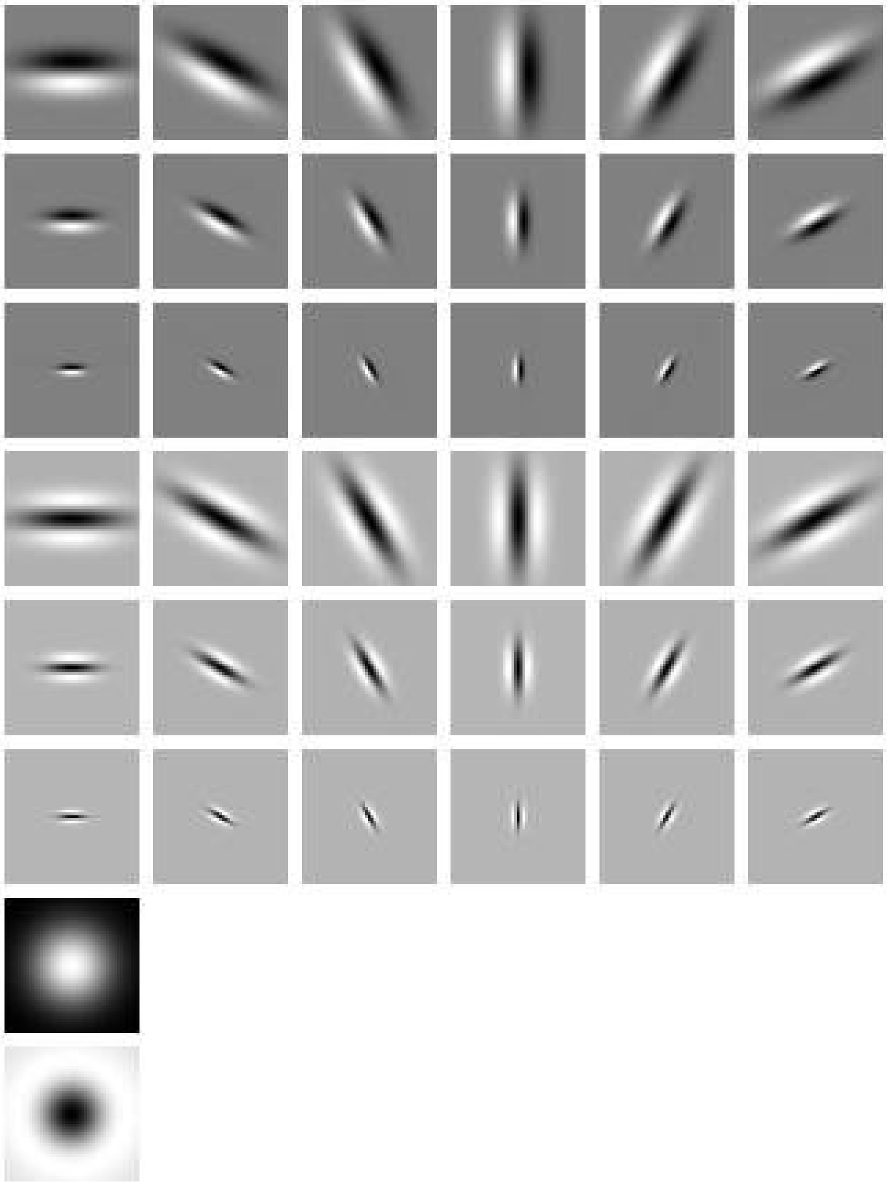


The Leung-Malik Filter Bank
T. Leung and J. Malik.
Representing and recognizing
the visual appearance of materials
using three-dimensional textons.
International Journal of
Computer Vision,
43(1):29-44, June 2001.

The Schmid (S) Filter Bank
C. Schmid. Constructing models for
content-based image retrieval.
In Proceedings of the IEEE Conference
on Computer Vision and
Pattern Recognition, volume 2,
pages 39-45, 2001.



Textons



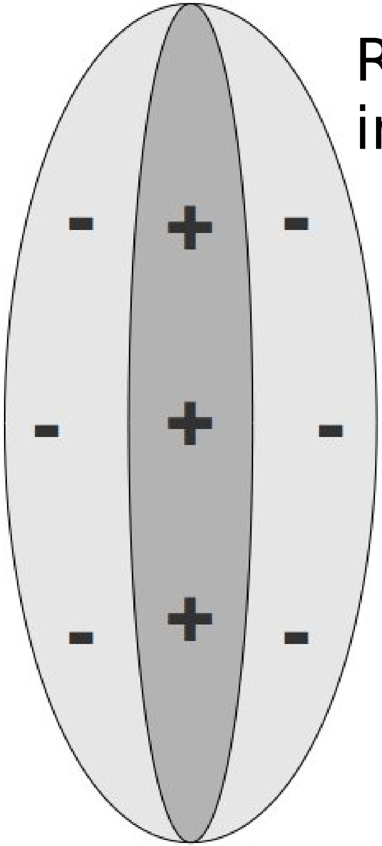
MR8 – Filterbank

- Gaussian
  - One scale
- Laplacian of Gaussian
  - One scale
- 1st Derivative of Gaussian
  - Three scales
  - Six orientations
- 2nd Derivative of Gaussian
  - Three scales
  - Six orienations

→ Maximum response over orientation!!

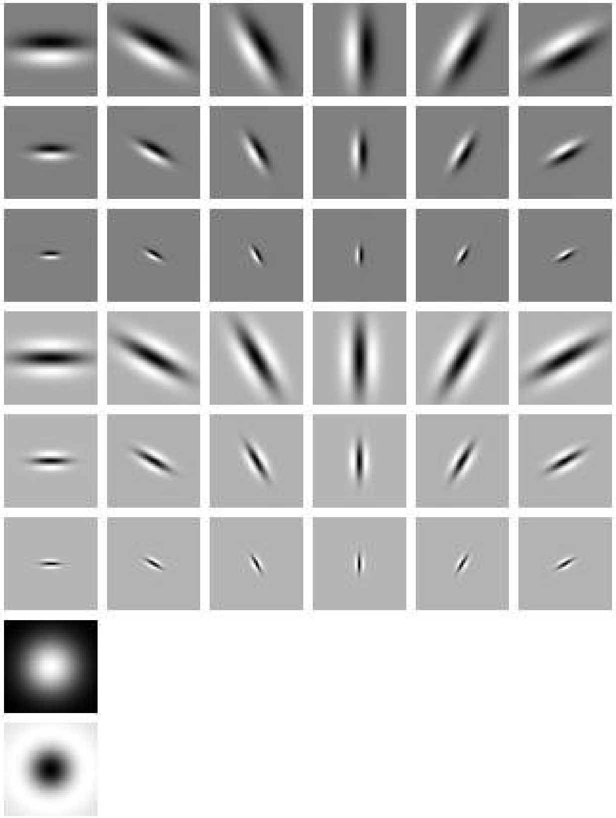
Textons

Analogy with visual cortex



Receptive fiel of on-off-cell
in visual cortex of mammals

- Excitatory/inhibitory center,
  inhibitory/excitatory sourrounding
- Zero activation
  under diffuse lighting
- Optimal impulse:
  small oriented light bar
- Other configurations possible



Textons

Gaussian function
$$G(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(\frac{-x^2+y^2}{2\sigma^2}\right)$$

Fst derivative
$$G_x(x,y)=\frac{\partial}{\partial x}G(x,y)=-\frac{x}{\sigma^2}G(x,y)$$

Snd derivative
$$G_{xx}(x,y)=\frac{\partial}{\partial x^2}G(x,y)=-\frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2}-1\right)G(x,y)$$

Mexican hat (DoG)
$$M(x,y)=G(x,y;\sigma_1)-G(x,y;\sigma_2),\qquad \sigma_1<\sigma_2$$

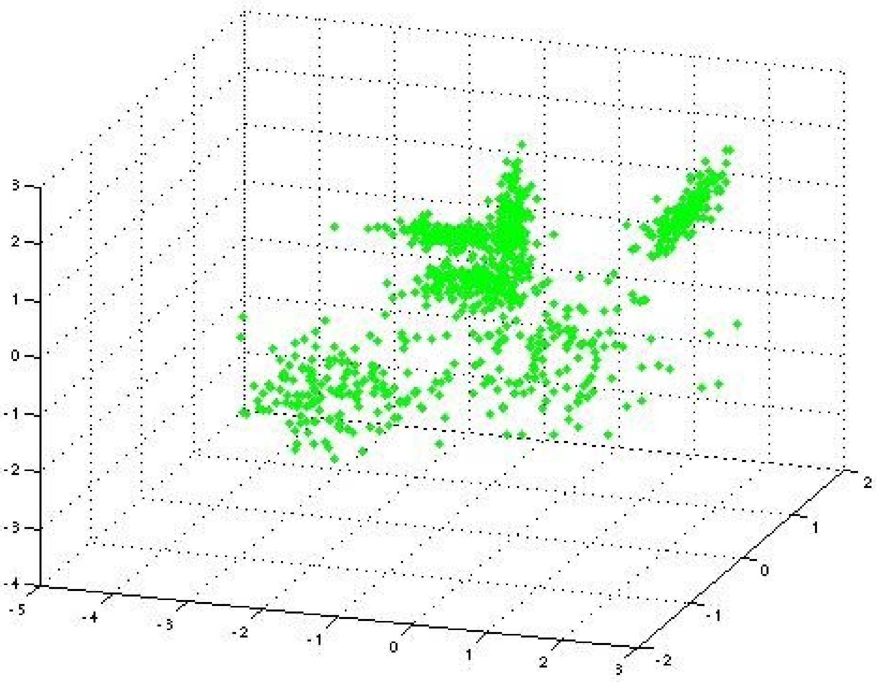
Linear separable when parallel to image axis !
Rotated versions?
→ Adjust pixel access during spatial convolution:

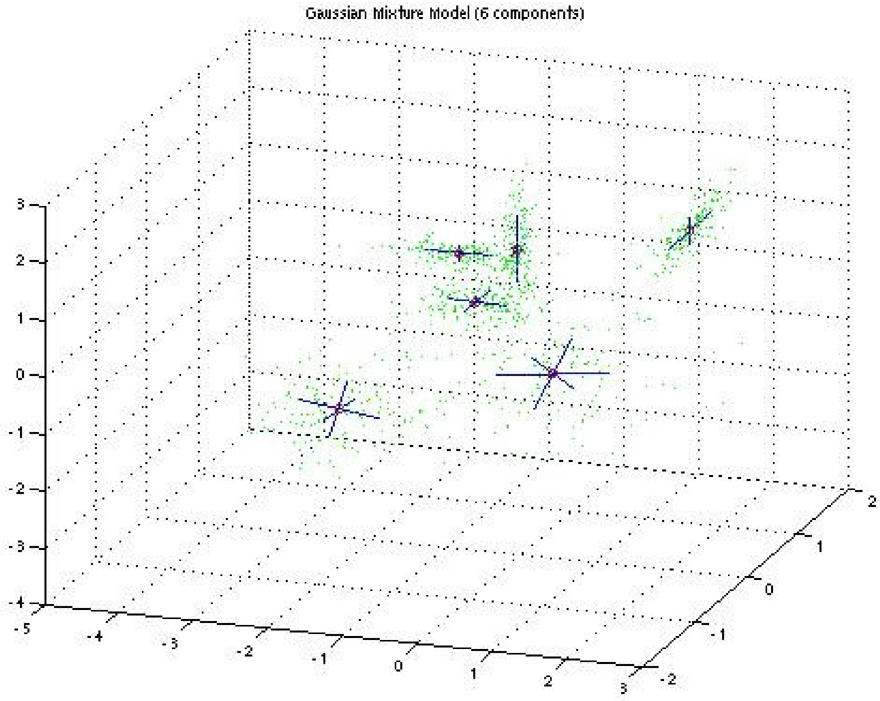
$$x=round(i+r*\cos(\phi)-s*\sin(\phi))$$
$$y=round(j+r*\sin(\phi)+s*\cos(\phi))$$

Textons

Clustering

- MR8-Filterbank: Eight filter-responses for each pixel
- Clustering in eight-dimensional space





Textons

Clustering

- MR8-Filterbank: Eight filter-responses for each pixel
- Clustering in eight-dimensional space
- K-Means clustering
  - Converges to K clusters (number of clusters pre-defined)

0. Given: Initial (sub-optimal) parameters
1. Compute membership of datapoints i to cluster j
2. Assign data point i to the most likeliy cluster
3. Cluster parameters are trivial to compute from assignment
   → Mean value of all features in a cluster
4. Iterate steps 1 to 3 until all assignments remain unchanged

Textons

Clustering

- MR8-Filterbank: Eight filter-responses for each pixel
- Clustering in eight-dimensional space
- K-Means clustering
  - Converges to K clusters (number of clusters pre-defined)
  - Cluster centers are prototypes of dominant filter-responses
    → **Textons**

Textons

- Probability of occurrence of specific textons in specific area
- Efficient and robust texture descriptor
- Applications:
  - Texture recognition
  - Scene categorization
  - Segmentation
  - Object detection
  - **<u>Image Retrieval</u>**

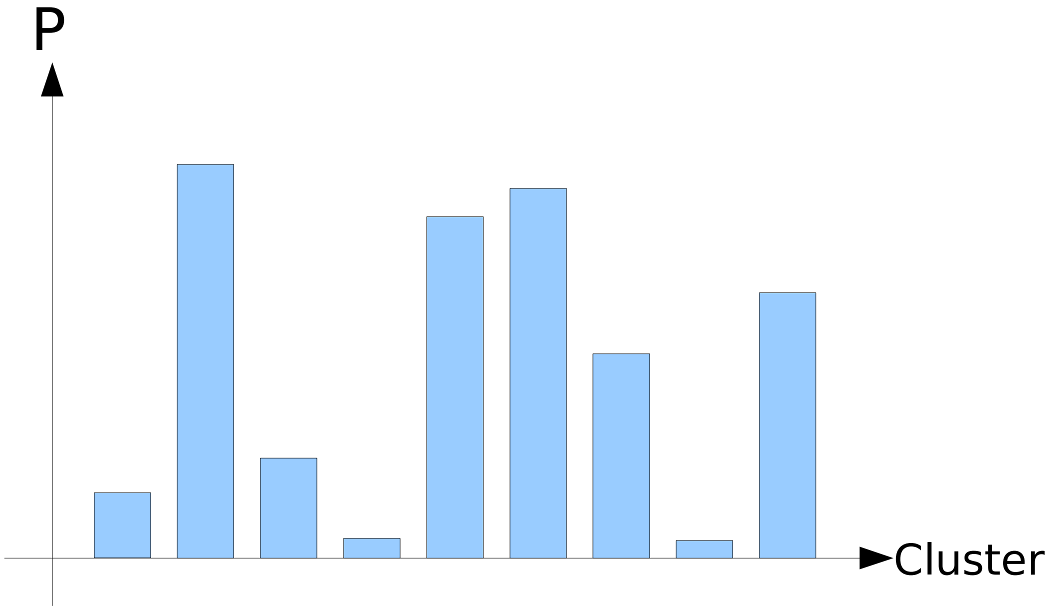
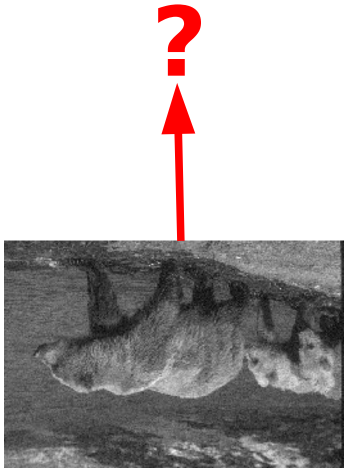


Image Retrieval



Query image

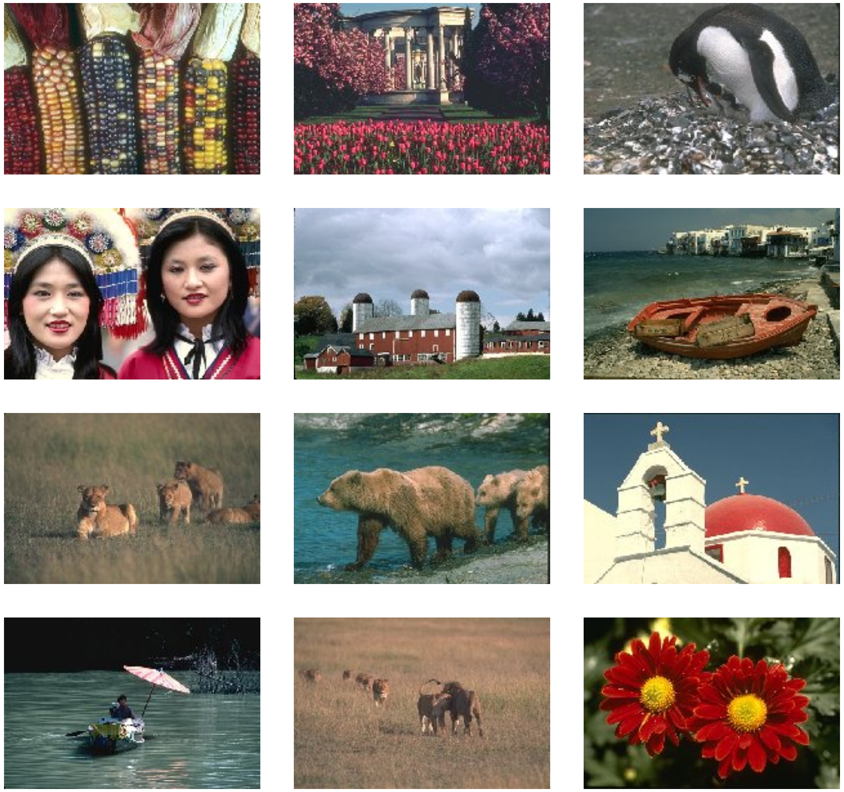


Image database

Image Retrieval

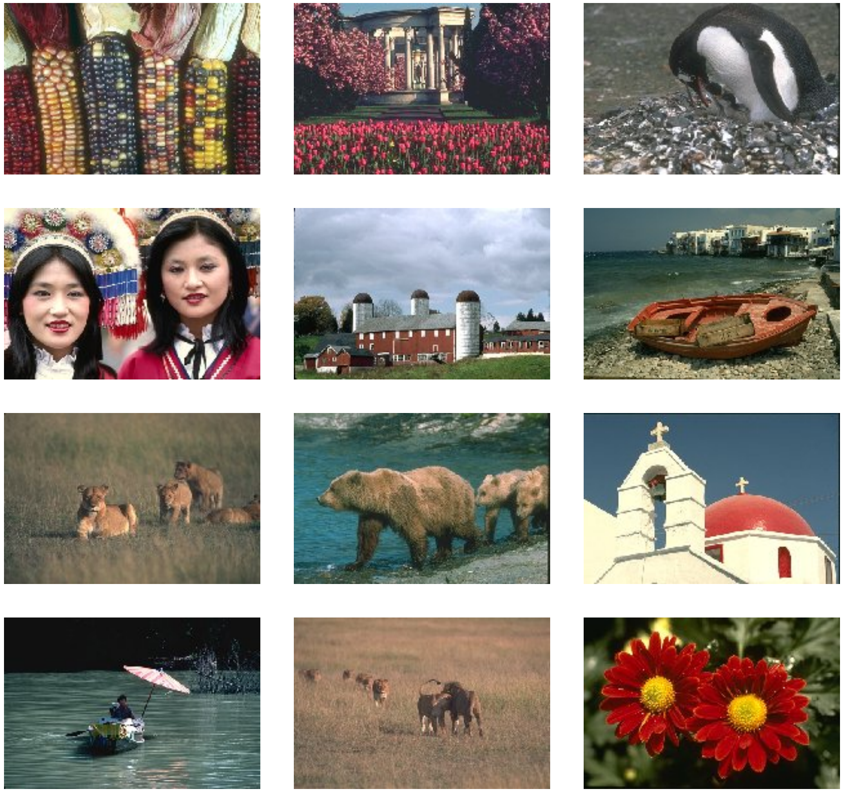


Image database

Training:

Image Retrieval

Image database

Training:

1. Convolve DB with filterbank

Image Retrieval

Image database

Training:

1. Convolve DB with filterbank

2. Clustering

Image Retrieval

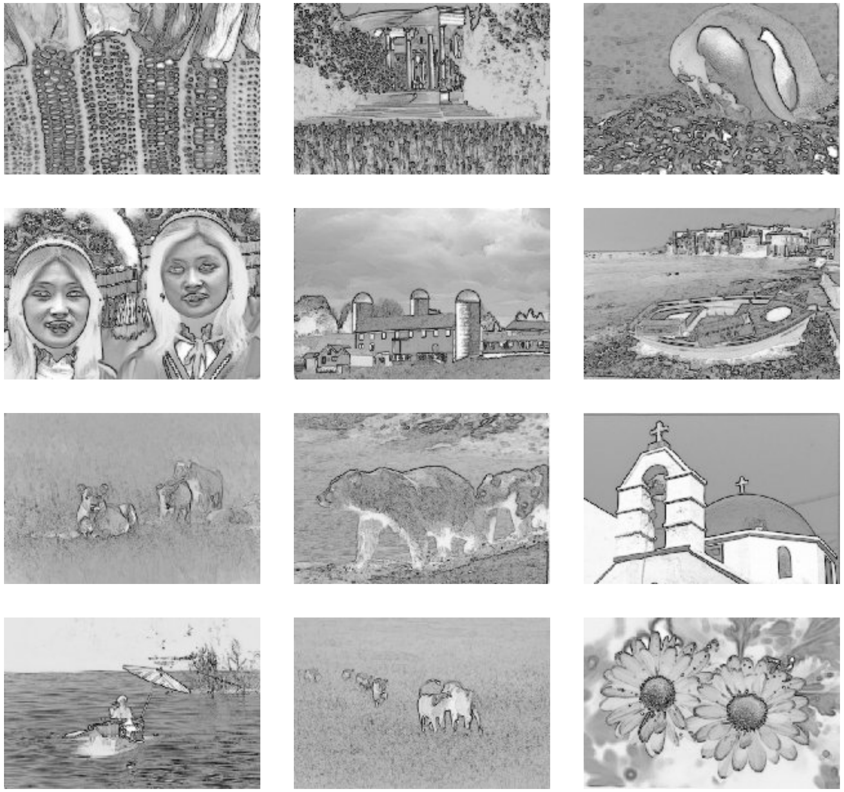


Image database

<u>Training:</u>

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images

Image Retrieval

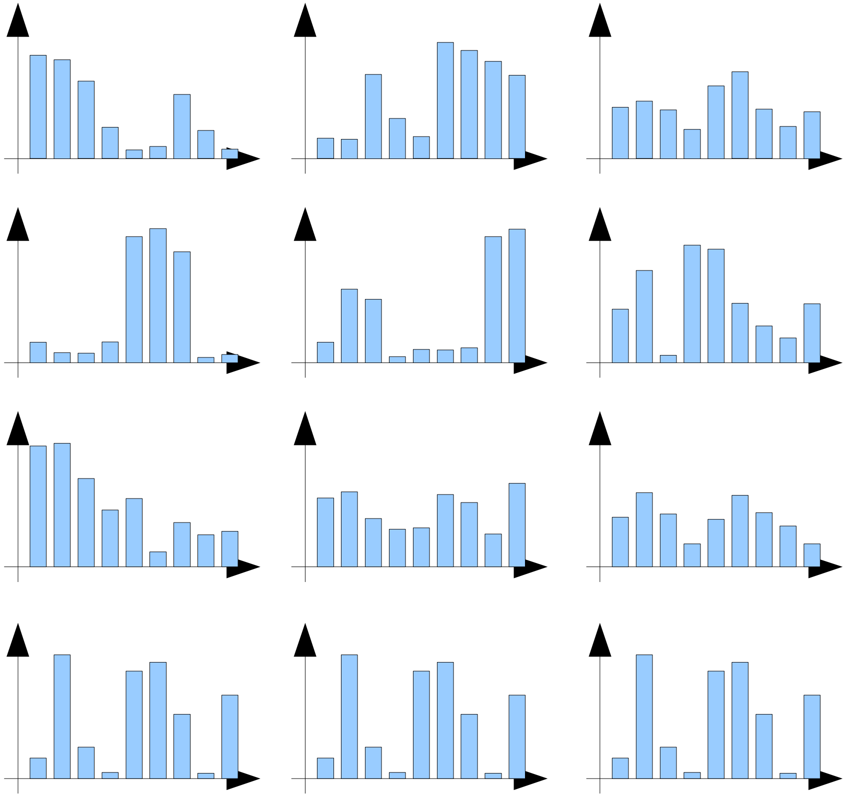
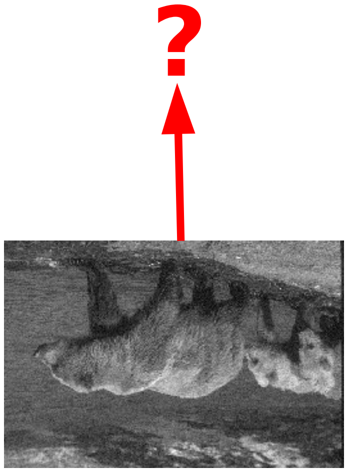


Image database

Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Image Retrieval



Query image

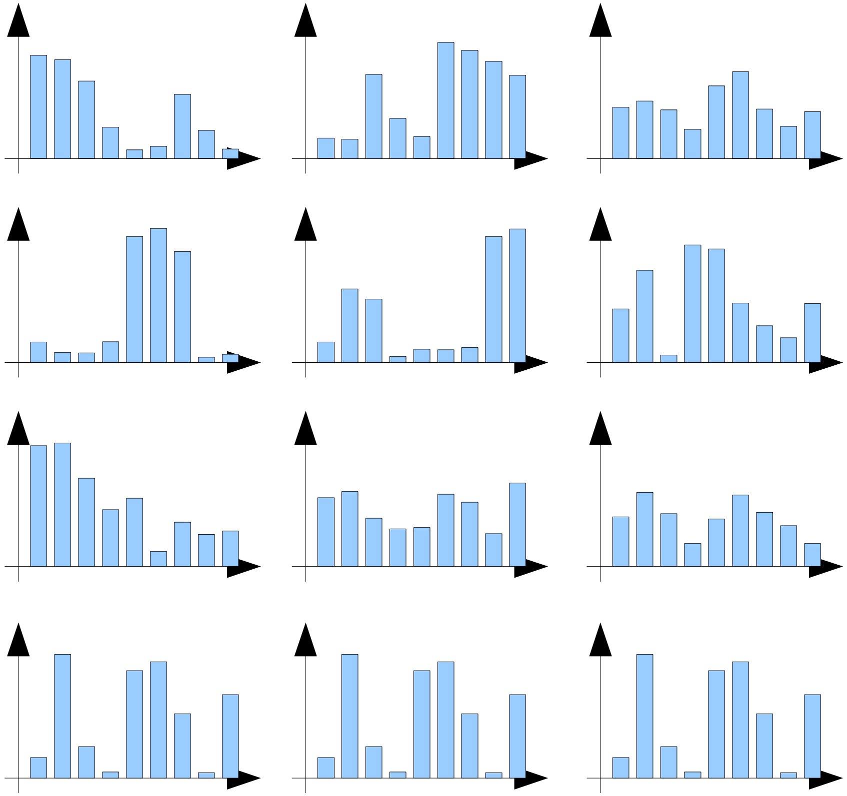


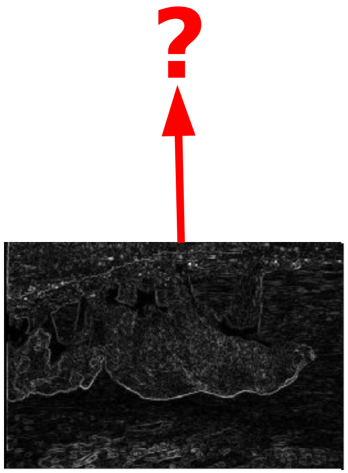
Image database

Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

Image Retrieval



Query image

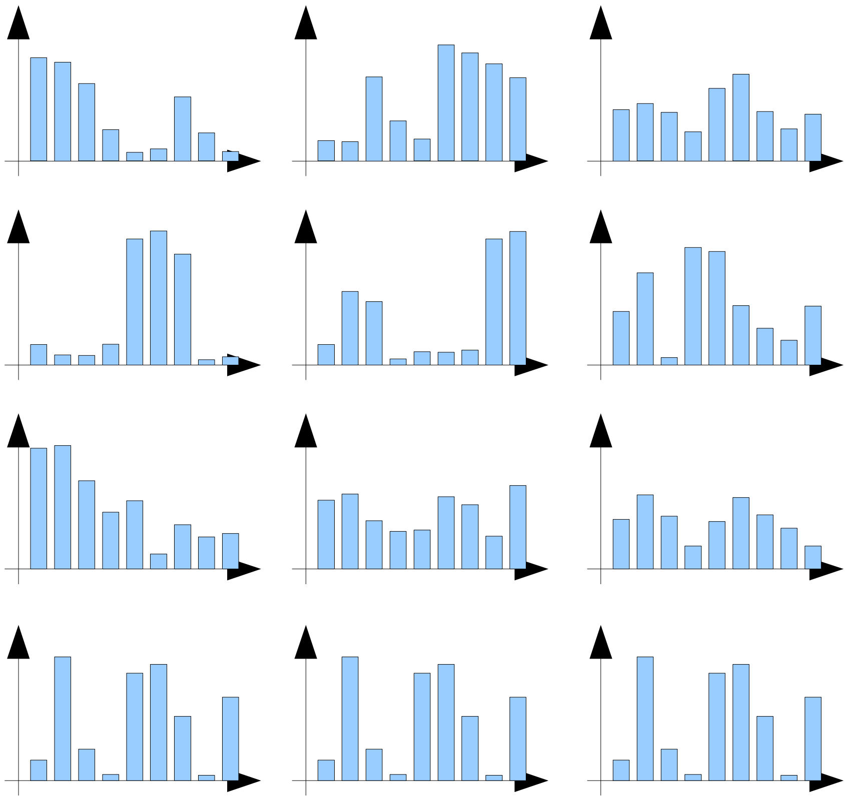


Image database

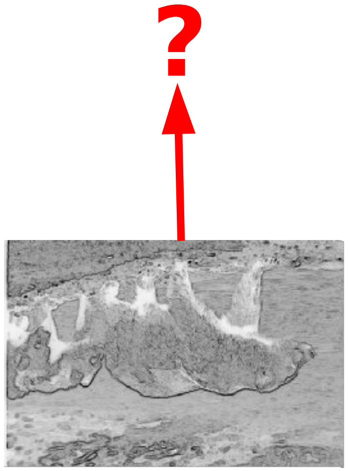
Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

1. Convolve query with filterbank

Image Retrieval



Query image

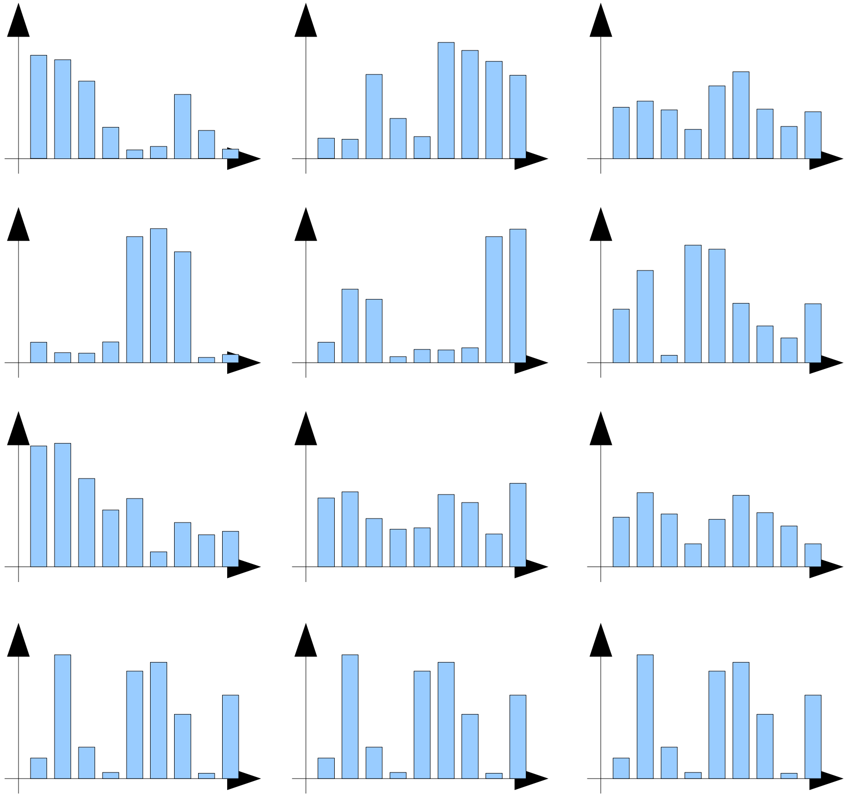


Image database

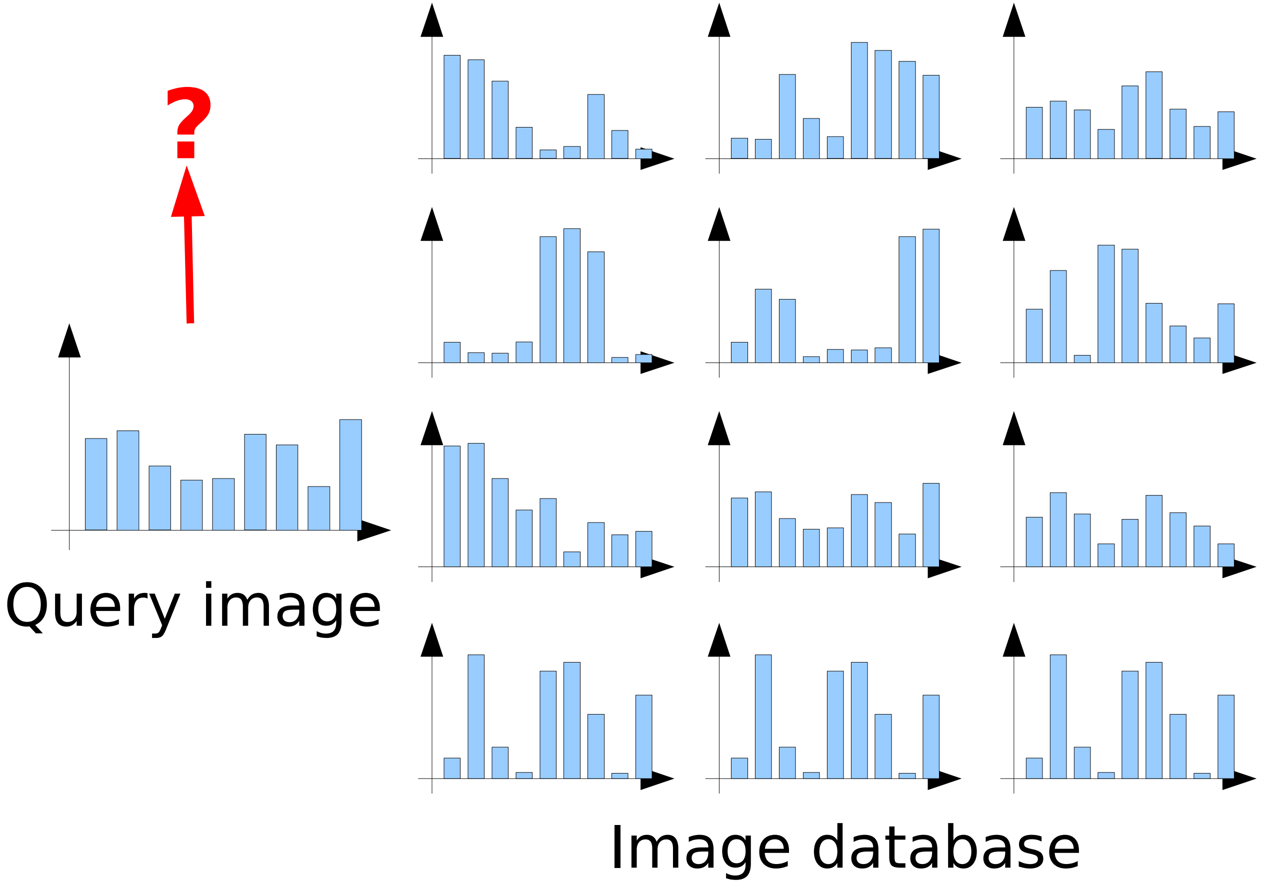
Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

1. Convolve query with filterbank
2. Calculate texton image

Image Retrieval



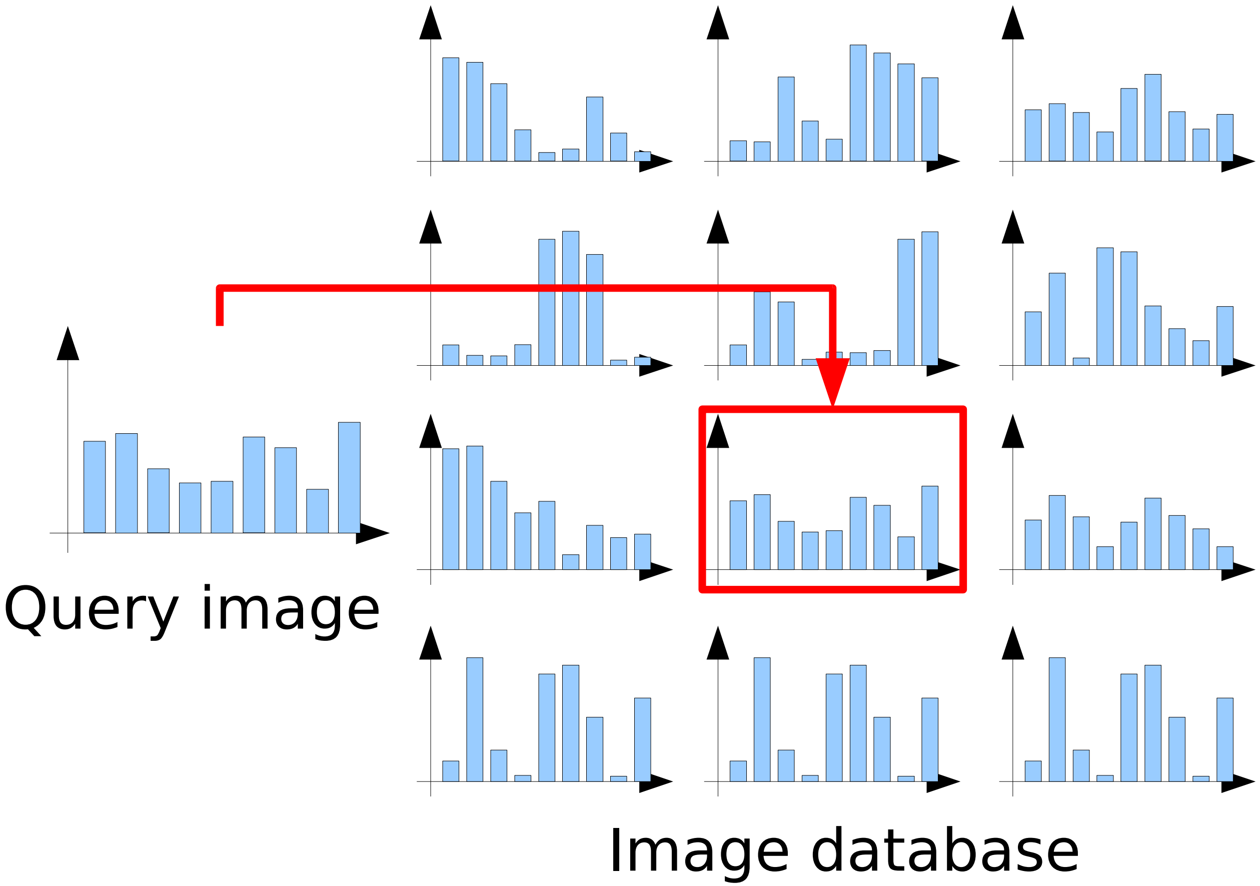
Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

1. Convolve query with filterbank
2. Calculate texton image
3. Calculate texton histogram

Image Retrieval



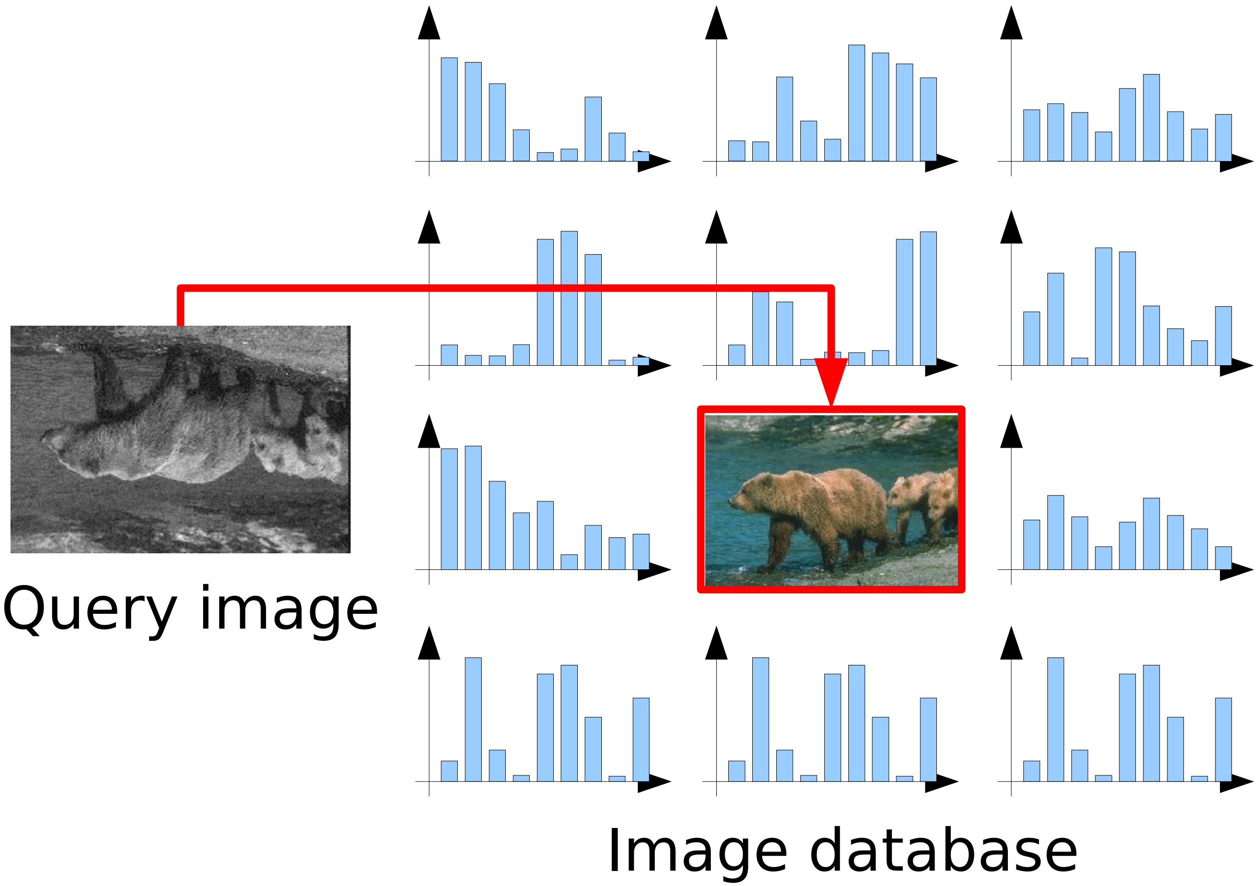
Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

1. Convolve query with filterbank
2. Calculate texton image
3. Calculate texton histogram
4. Compare histograms

Image Retrieval



Training:

1. Convolve DB with filterbank
2. Clustering
3. Calculate texton images
4. Calculate texton histogram

Application:

1. Convolve query with filterbank
2. Calculate texton image
3. Calculate texton histogram
4. Compare histograms

Given

`main(int argc, char** argv)`

- argv[1] == "generate"
  → loads image database
  → extracts textons
  → extracts texon-based image descriptors for images in database
  → saves image descriptors and textons

- argv[1] == "find"
  → loads textons and image descriptors of database
  → loads image queries (and distorts them)
  → calculates query image descriptors
  → compares query with database descriptors
  → gives image ID of the most similar one

- argv[2] == path to init file
  → used to define all necessary parameters (read function provided)

Given

- **`int loadDB(vector<Mat>& db, string fname, int numberOfImages)`**
    - `db`: contain images after loading
    - `fname`: file containing all image paths
    - `numberOfImages`: maximal number of images to be loaded
  - ➔ Loads database
  - ➔ DB-file: one path per line
  - ➔ Suitable database:
    http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

- **`void distortQuery(vector<Mat>& queries)`**
    - `queries`: query images
  - ➔ Flips the images and adds some noise

Given

- **`void clustering(vector<Mat>& filterResp, Mat& textons, int numberOfDataPoints)`**
  - `filterResp`: filterresponse images
  - `textons`: cluster centers, ie. Textons
  - `numberOfDataPoints`: number of randomly sampled data points (speed)

  ➔ Applies kMeans-clustering to filter responses in order to find textons

  ➔ `textons` is a N x 8 matrix, where N is the number of clusters, ie. textons

To Do

- **`void createKernel1D(Mat& kernel, int kSize, string name)`**
  - `kernel`: will contain computed (1D) kernel
  - `kSize`: size (length) of the kernel
  - `name`: specifies which kernel shall be computed
- ➔ Computes gaussian, fst-dev gaussian, snd-dev gaussian, or mexican hat kernel
- ➔ name = “gaussian”, “gaussianDevX”, “gaussianDevXX”, “mexHat”

Gaussian function $$G(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(\frac{-x^2+y^2}{2\sigma^2}\right)$$

Fst derivative $$G_x(x,y)=\frac{\partial}{\partial x}G(x,y)=-\frac{x}{\sigma^2}G(x,y)$$

Snd derivative $$G_{xx}(x,y)=\frac{\partial}{\partial x^2}G(x,y)=-\frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2}-1\right)G(x,y)$$

Mexican hat (DoG) $$M(x,y)=G(x,y;\sigma_1)-G(x,y;\sigma_2),\qquad \sigma_1<\sigma_2$$

To Do

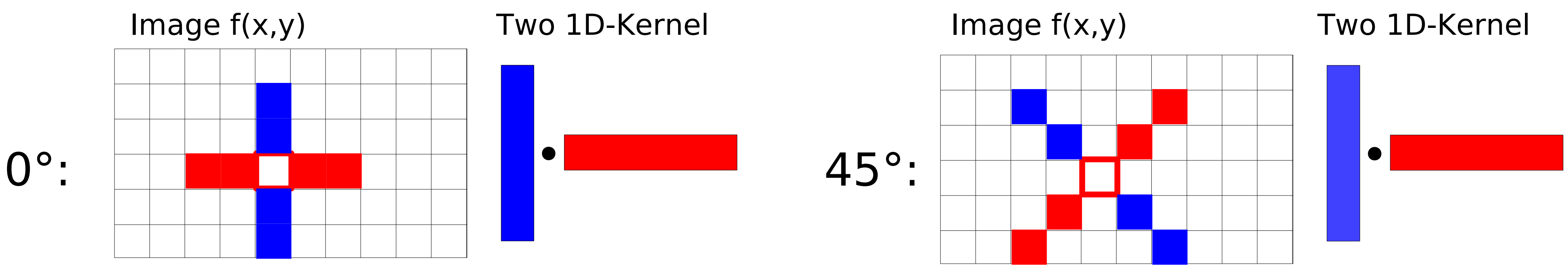
- **`void spatialConvolution(Mat& in, Mat& out, Mat& kernel, double phi)`**
  - `in`: input image
  - `out`: output image
  - `kernel`: convolution kernel
  - `Phi`: orientation of kernel

→ Computes convolution with rotated version of base kernel

→ NOTE: 1D stays 1D after rotation (just access indices are changed)

→
$$x = round(i + r * \cos(\phi) - s * \sin(\phi))$$
$$y = round(j + r * \sin(\phi) + s * \cos(\phi))$$

→ “Horizontal” and “Vertical” separated kernels differ by 90° !



To Do

- **`void applyFilterbank(vector<Mat>& db, vector<Mat>& filterResp)`**
    - `db`: image database
    - `filterResp`: filter responses of MR8-filterbank
  - → Applies MR8-filterbank to all images
    - → Takes maximum over orientation for all oriented kernel (1st and 2nd dev. of gaussian)
  - → Uses linear separable convolution
  - → `filterResp[i]-filterResp[i+7]` contains the eight filter responses of image i
  - → eg. 10 images in database result in 80 filter responses

To Do

- **`void getTextonImages(vector<Mat>& filterResp, Mat& textons, vector<Mat>& textonImages)`**
  - `filterResp`: filter responses
  - `textons`: textons
  - `textonImages`: the calculated texton images
  - ➔ Computes the distance of each filter response vector to all textons
  - ➔ `filterResp[i]-filterResp[i+7]` contain filterresponses of image i
  - ➔ `textonImages[i]-textonImages[i+N-1]` contain N texton images of image i
  - ➔ `textons` is a N x 8 matrix, where N is the number of textons
  - ➔ For each image i calculate

$$textonImage[i+t] = \sqrt{\sum_{j=1}^{8} (filterResp[i+j] - texton[t, j])^2}$$

To Do

- **`void calcTextonHistograms(vector<Mat>& textonImages, Mat& textonHistogram)`**
  - `textonImages`: calculated texton images
  - `textonHistogram`: matrix texton histograms
  - ➔ Computes the texton histogram of texton images
  - ➔ `textonImages[i]-textonImages[i+N-1]` contains N texton images of image I
  - ➔ Texton histogram h of image i:

$$h_i(t) = \frac{1}{Z} \cdot \sum_{x,y} textonImages[t](x,y)$$

$$Z = \sum_t h_i(t)$$

To Do

- **`void findQuery(Mat& textonHistogram, Mat& db)`**
  - `textonHistogram`: image descriptors of query images (one per row)
  - `db`: image descriptors of database images (one per row)

  ➔ Computes distance for each query image and each image in database as euclidian distance of image descriptors (ie. texton histograms)

  ➔ Prints index of image with minimal distance

To Do

- Mandatory:
  - Implement missing functionionality
  - State which database you used
  - Briefly discuss the performance of the implemented system
    - Easy/hard queries?
  - What problems do you expect in a real application scenario?
  - What are possible improvements?

- Optional
  - Implement improvements...

Mid-term Exam

- Friday, **01.06.2012, 10:15pm, E020**

- In place of an exercise
- Duration: ca. 30 min
- No grade, but pass is necessary to take part at the final exam

- Topics from lecture and exercise

- Questions in English, answers in English or German

- No books, no calculator, no script, no paper, ...