Advanced Topics in Computer Vision
and Robotics

Features

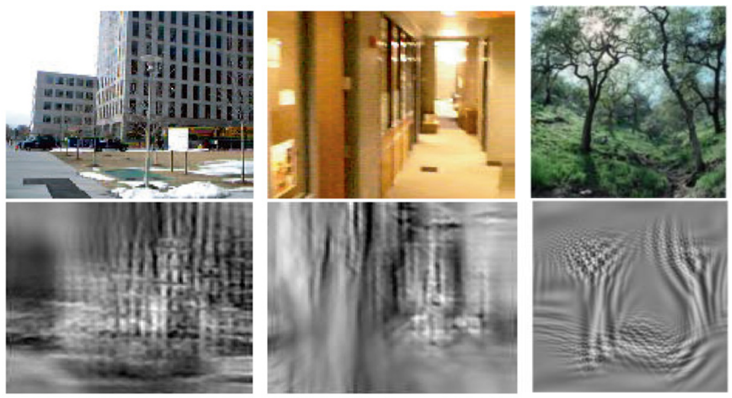
Some slides thanks to S. Lazebnik, T. Berg, Fei-Fei Li, K. Grauman and others

Outline

Image Features
- Global vs Local
- color histograms (color),
- texture histograms (texture),
- SIFT (shape)
- edges, contours

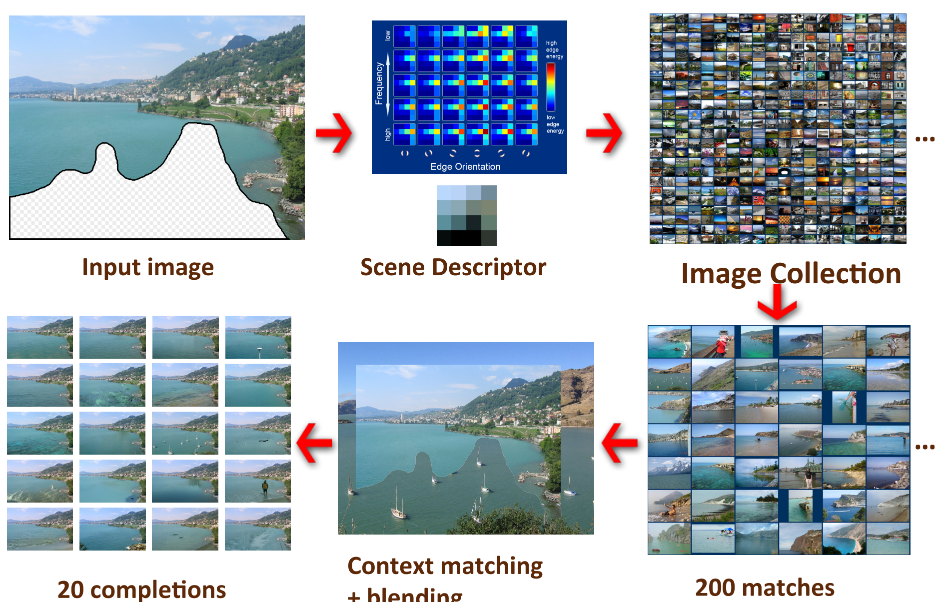
Global Features

- The “gist” of a scene: Oliva & Torralba (2001)



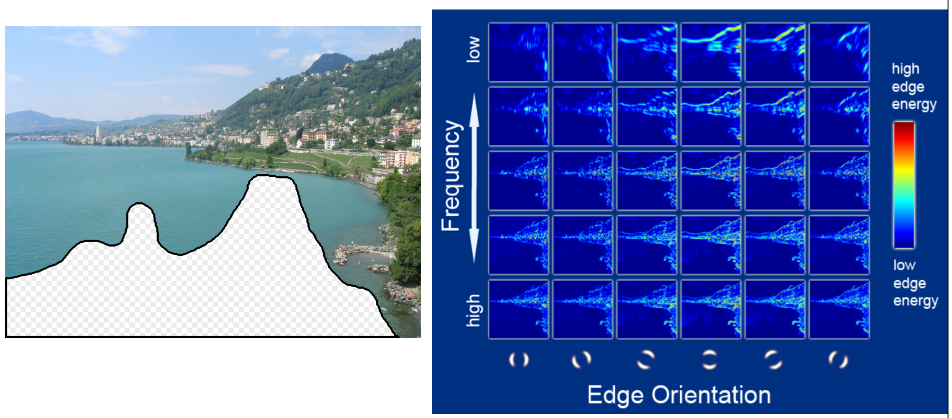
source: Svetlana Lazebnik

Example: Scene completion using millions of photographs



Hays and Efros, SIGGRAPH 2007

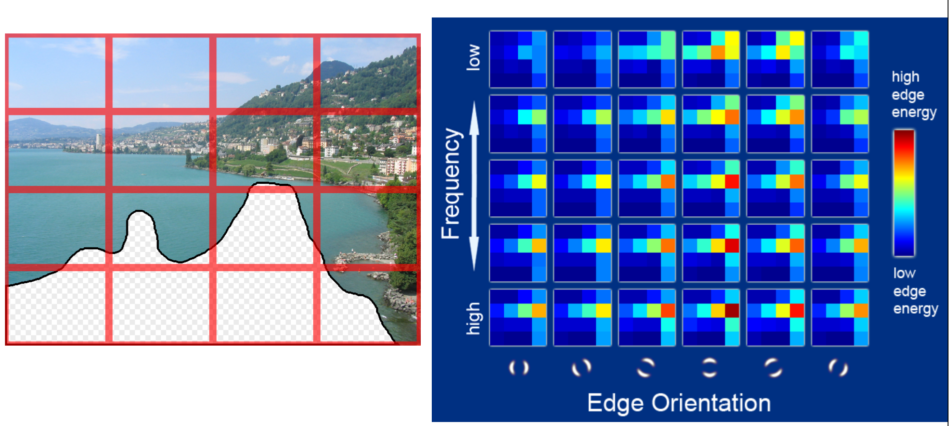
Scene Descriptor



Compute oriented edge response at multiple scales (5 spatial scales, 6 orientations)

Hays and Efros, SIGGRAPH 2007

Scene Descriptor

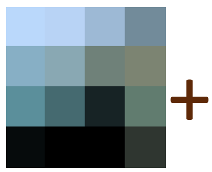


Gist scene descriptor (Oliva and Torralba 2001)

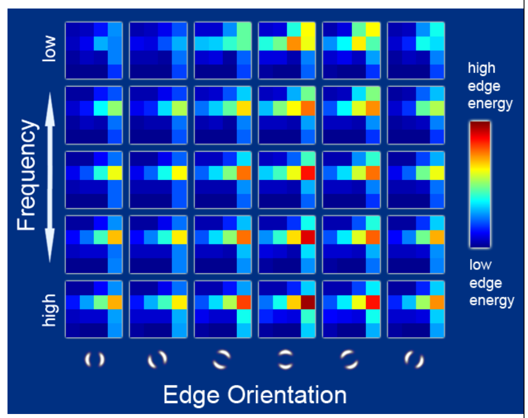
- “semantic” descriptor of image composition
- aggregated edge responses over 4x4 windows
- scenes tend to be semantically similar under this descriptor if very close

Hays and Efros, SIGGRAPH 2007

Scene Descriptor



Color descriptor – color of the query image downsampled to 4x4



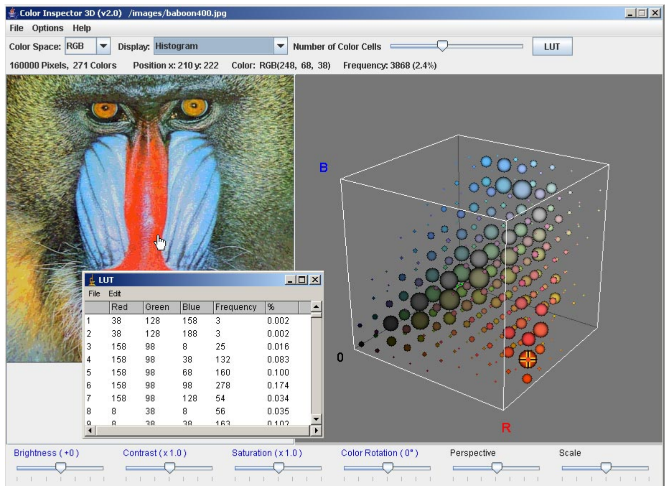
Gist scene descriptor (Oliva and Torralba 2001)

Hays and Efros, SIGGRAPH 2007

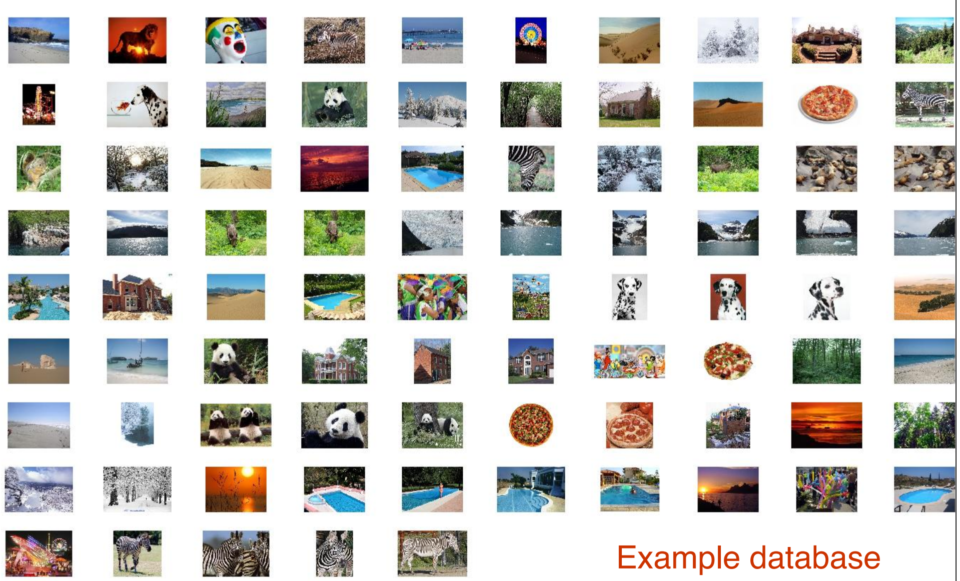
Final result – blended between the two images along the cut to merge seamlessly

Hays and Efros, SIGGRAPH 2007

Global Color Histograms



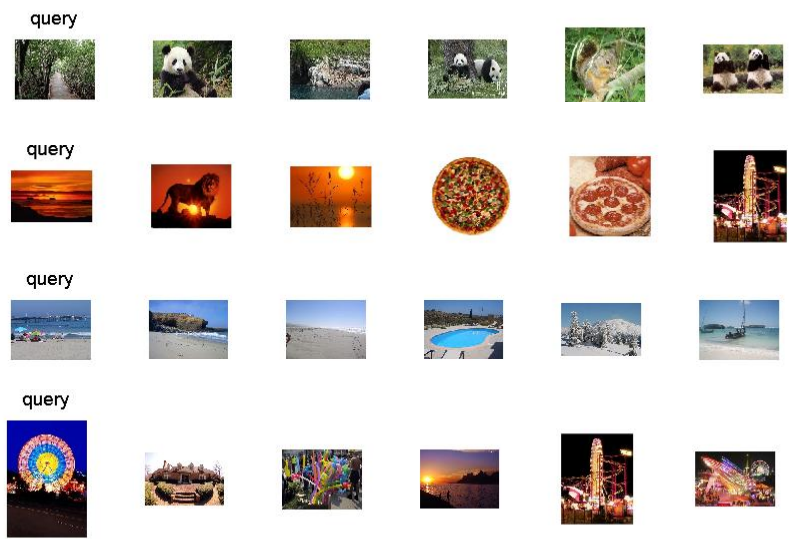
Color-based image retrieval



Example database

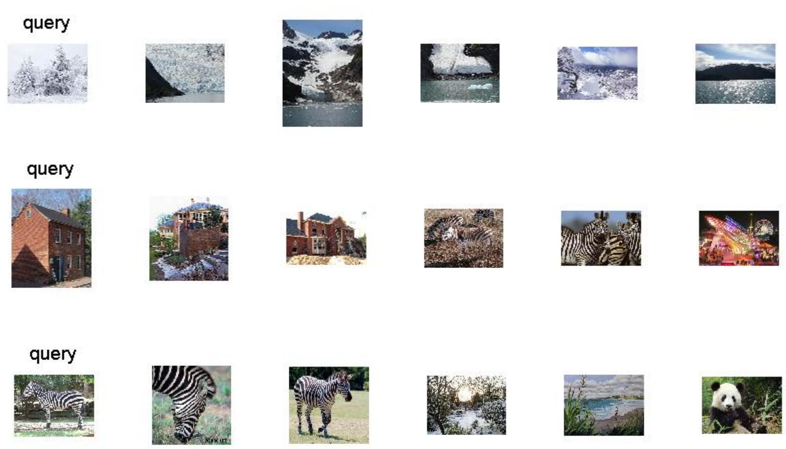
Kristen Grauman

Color-based image retrieval



Example retrievals

Color-based image retrieval



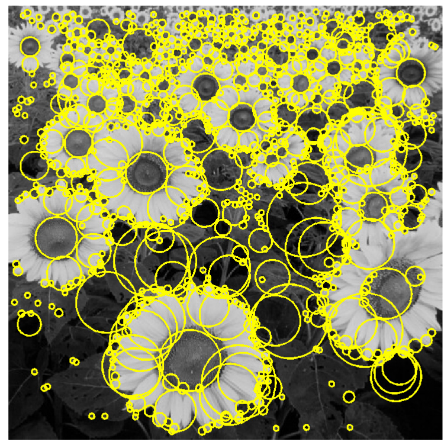
Example retrievals

See More: color and light lecture

Kristen Grauman

Local Features

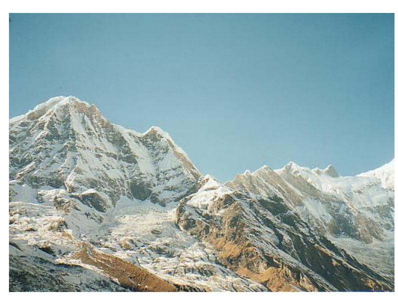
Feature points (locations) + feature descriptors



source: Svetlana Lazebnik

Why extract features?

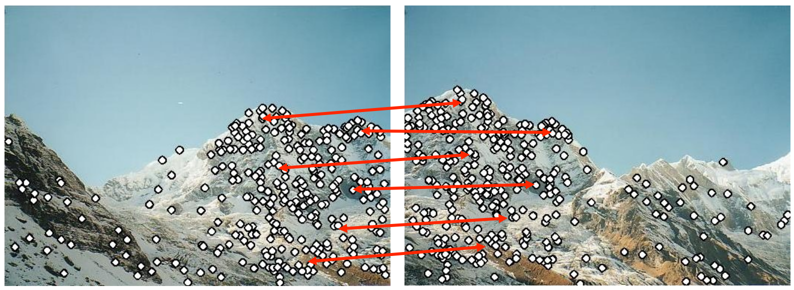
- Motivation: panorama stitching
  - We have two images – how do we combine them?



source: Svetlana Lazebnik

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



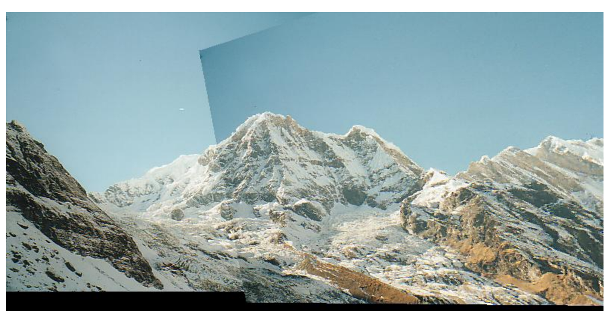
Step 1: extract features

Step 2: match features

source: Svetlana Lazebnik

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



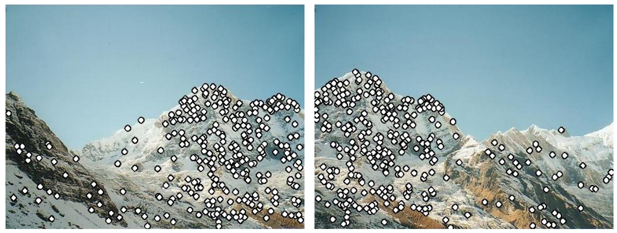
Step 1: extract features

Step 2: match features

Step 3: align images

source: Svetlana Lazeb

Characteristics of good features



- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

source: Svetlana Lazebnik

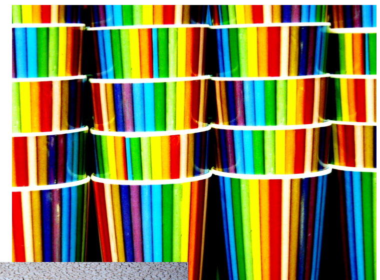
Applications

- Feature points are used for:
  - Retrieval
  - Indexing
  - Motion tracking
  - Image alignment
  - 3D reconstruction
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation

source: Svetlana Lazebnik

Feature Types

Shape!



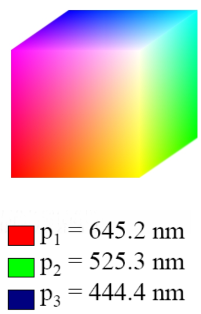
Color!

Texture!

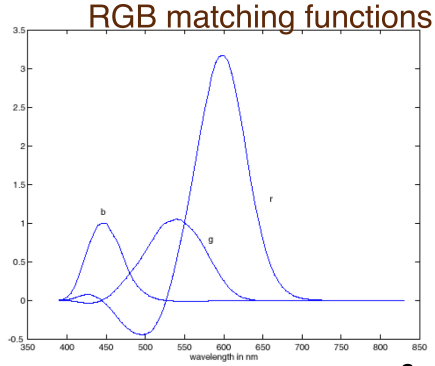
Color Features

RGB Colorspace

- Primaries are monochromatic lights (for monitors, they correspond to the three types of phosphors)

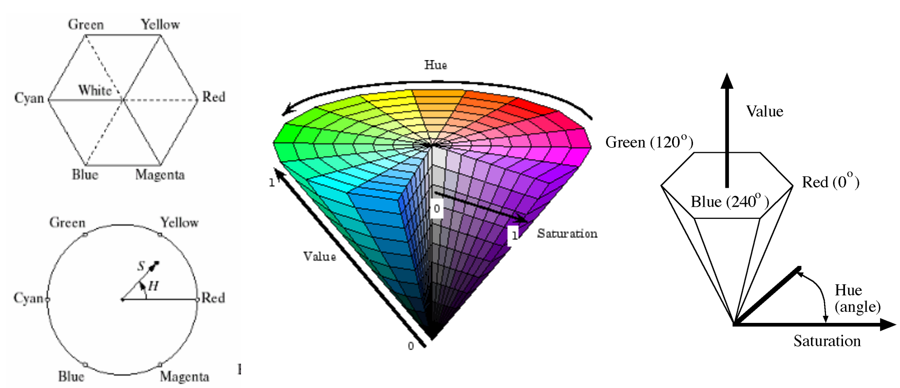


RGB matching functions



source: Svetlana La

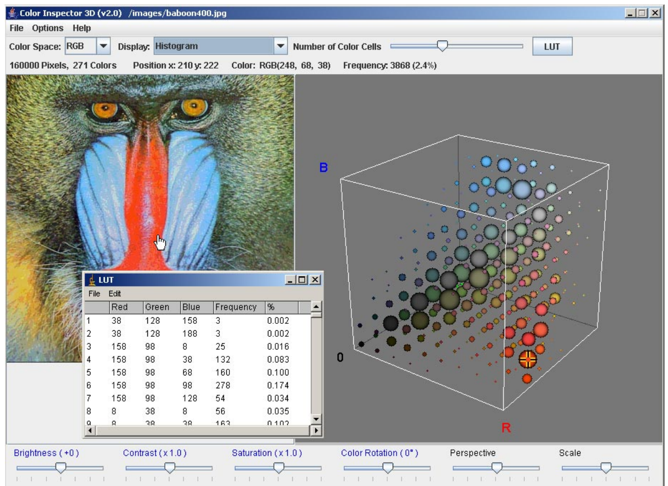
HSV Colorspace



- Perceptually meaningful dimensions: Hue, Saturation, Value (Intensity)

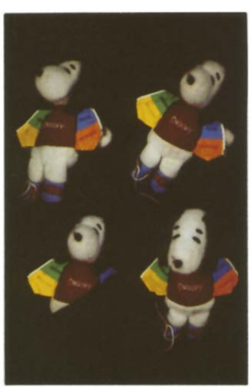
source: Svetlana La

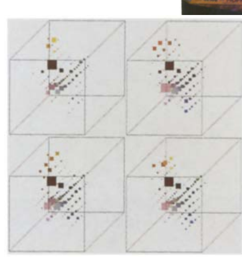
Color Histograms

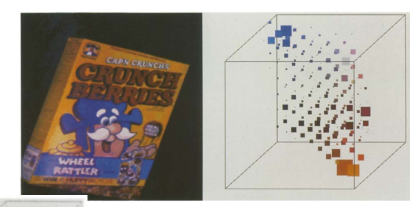


Uses of color in computer vision

Color histograms for indexing and retrieval



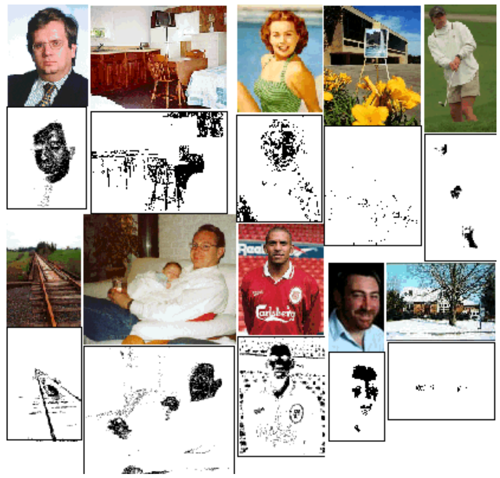




Swain and Ballard, Color Indexing, IJCV 1991.

Uses of color in computer vision

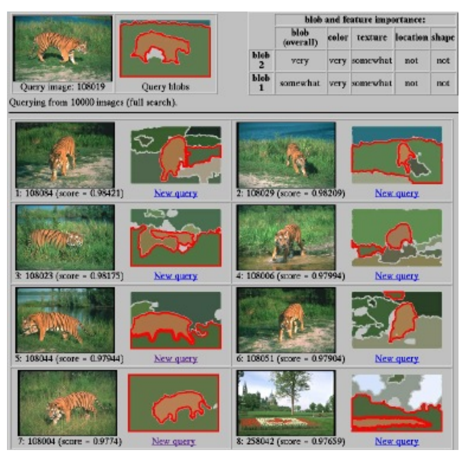
Skin detection



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

Uses of color in computer vision

Image segmentation
and retrieval



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

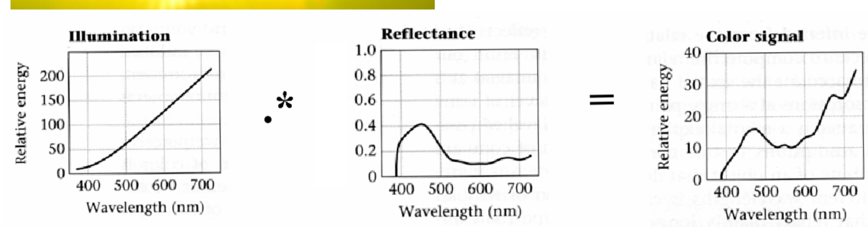
source: Svetlana Lazebnik

Color features

Pros/Cons?

Interaction of light and surfaces

- Reflected color is the result of interaction of light source spectrum with surface reflectance

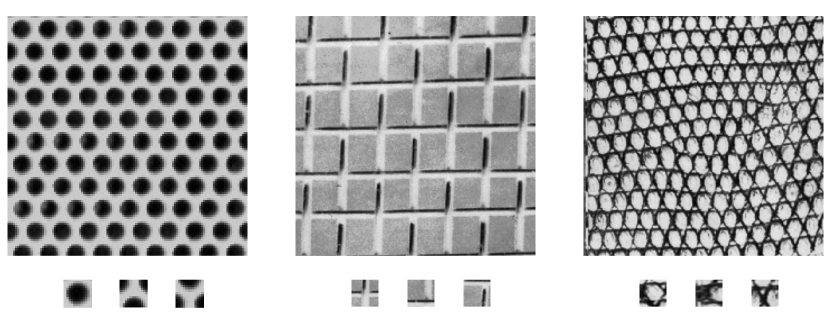


source: Svetlana Lazebnik

Texture Features

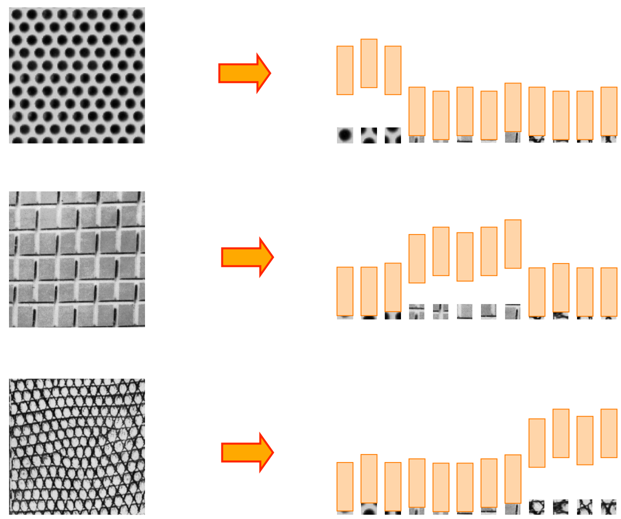
Texture Features

- Texture is characterized by the repetition of basic elements or *textons*
- For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Texture Histograms



Moving average

- Let's replace each pixel with a *weighted* average of its neighborhood
- The weights are called the *filter kernel*
- What are the weights for the average of a 3x3 neighborhood?

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

"box filter"

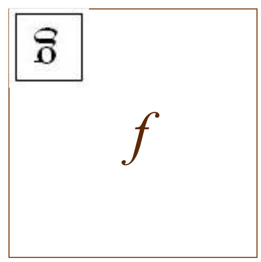
Source: D. Lowe

Convolution

- Let *f* be the image and *g* be the kernel. The output of convolving *f* with *g* is denoted *f* * *g*.

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

Convention:
kernel is "flipped"

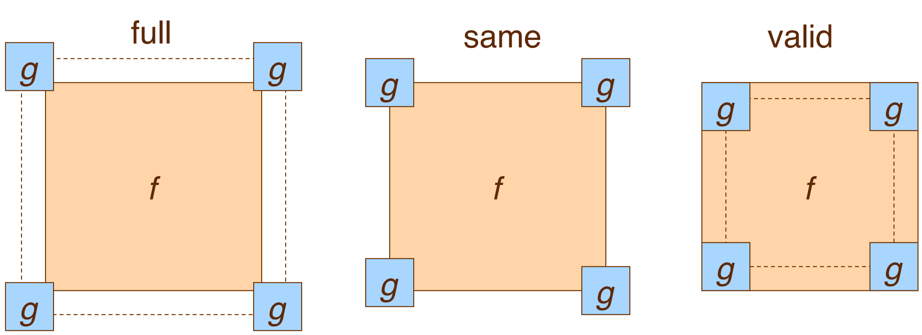


- MATLAB functions: conv2, filter2, imfilter

Source: F. Durand

Convolution

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



Convolution in 2D

$$g[x,y] = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}} \sum_{l=-\frac{w}{2}}^{\frac{w}{2}} f[k,l]h[x-k,y-l]$$

f

10	11	10	0	0	1
9	**10**	11	1	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	99	11
10	9	9	11	10	10

g

X	X	X	X	X	X
X	**10**				X
X					X
X					X
X					X
X	X	X	X	X	X

h

1/9

1	1	1
1	1	1
1	1	1

1/9.(10x1 + 11x1 + 10x1 + 9x1 + 10x1 + 11x1 + 10x1 + 9x1 + 10x1)
1/9.(90) = 10

Example:

I

10	11	10	0	0	1
9	10	11	**1**	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	99	11
10	9	9	11	10	10

O

X	X	X	X	X	X
X	10	7	**4**	1	X
X					X
X					X
X					X
X	X	X	X	X	X

F

1/9

1	1	1
1	1	1
1	1	1

1/9.(10x1 + 0x1 + 0x1 + 11x1 + 1x1 + 0x1 + 10x1 + 0x1 + 2x1) =
1/9.(34) = 3.7778

Example:

I

10	11	10	0	0	1
9	10	11	1	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	$\boxed{99}$	11
10	9	9	11	10	10

F

1/9

1	1	1
1	1	1
1	1	1

O

X	X	X	X	X	X
X	10	7	4	1	X
X					X
X					X
X				$\boxed{20}$	X
X	X	X	X	X	X

1/9.(10x1 + 9x1 + 11x1 + 9x1 + 99x1 + 11x1 + 11x1 + 10x1 + 10x1)

1/9.(180) = 20

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

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

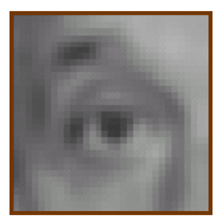
?

Source: D. Lowe

Practice with linear filters

Original

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



Blur (with a box filter)

Source: D. Lowe

Practice with linear filters

Original

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

?

(Note that filter sums to 1)

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

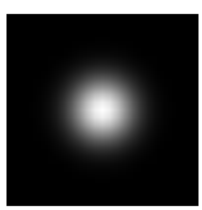
Example: Smoothing by Averaging

What is wrong with the picture ?

 Box filter

Smoothing with box filter revisited

- What' s wrong with this picture?
- What' s the solution?
  - To eliminate edge effects, weight contribution of neighborhood pixels according to their closeness to the center



"fuzzy blob"

Gaussian Filter

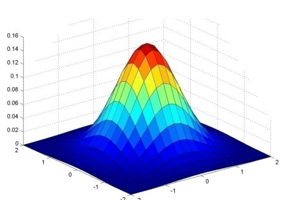
- A particular case of averaging
  - The coefficients are samples of a 1D Gaussian.
  - Gives more weight at the central pixel and less weights to the neighbors.
  - The further away the neighbors, the smaller the weight

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}},$$

  - Sample from the continuous Gaussian

Gaussian Filter

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

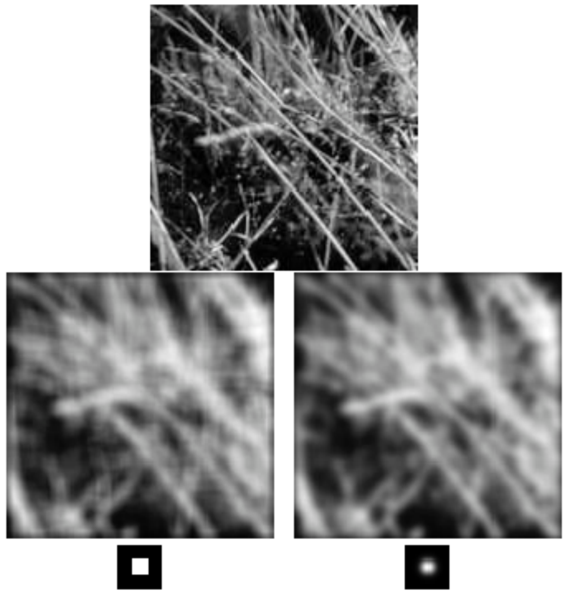


0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, σ = 1

- Constant factor at front makes volume sum to 1 (can be ignored when computing the filter values, as we should renormalize weights to sum to 1 in any case)

Source: C. Rasmussen

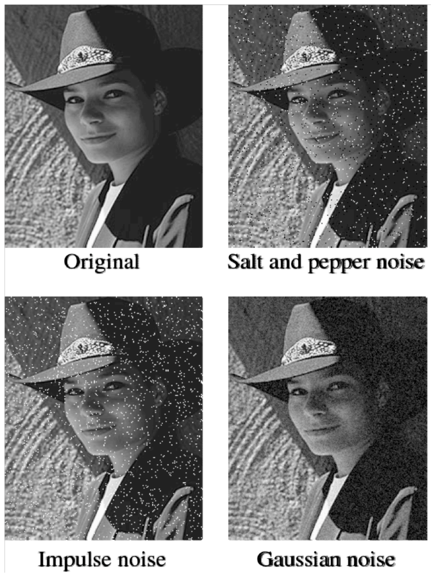


Non-linear Filtering

- Replace each pixel with the MEDIAN value of all the pixels in the neighborhood.

- Non-linear
- Does not spread the noise
- Can remove spike noise
- Expensive to run

Noise



- **Salt and pepper noise**: contains random occurrences of black and white pixels
- **Impulse noise:** contains random occurrences of white pixels
- **Gaussian noise**: variations in intensity drawn from a Gaussian normal distribution

Source: S. Seitz

Gaussian vs. median filtering

3x3 5x5 7x7

Gaussian

Median

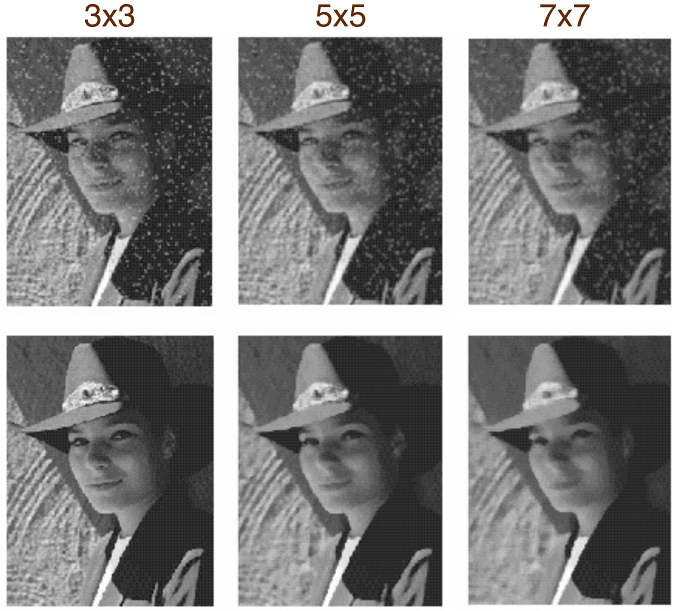


Image Smoothing With Gaussian (MATLAB)

```
figure(3);
sigma = 3;
width = 3 * sigma;
support = -width : width;
gauss2D = exp( - (support / sigma).^2 / 2);
gauss2D = gauss2D / sum(gauss2D);
smooth = conv2(conv2(bw, gauss2D, 'same'), gauss2D', 'same');
image(smooth);
colormap(gray(255));

gauss3D = gauss2D' * gauss2D;
tic ; smooth = conv2(bw,gauss3D, 'same'); toc
```

Demonstrates separabilty

Shape Features

We want invariance!!!

- Good features should be robust to all sorts of nastiness that can occur between images.

Types of invariance

- Illumination

Slide source: Tom Duerig

Types of invariance

- Illumination
- Scale

Slide source: Tom Duerig

Types of invariance

- Illumination
- Scale
- Rotation

Slide source: Tom Duerig

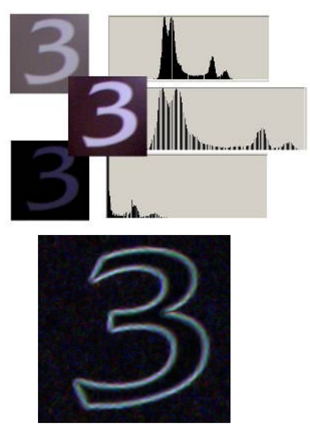
Types of invariance

- Illumination
- Scale
- Rotation
- Affine

Slide source: Tom Duerig

How to achieve illumination invariance

- Use edges instead of raw values



Slide source: Tom D

Edge detection

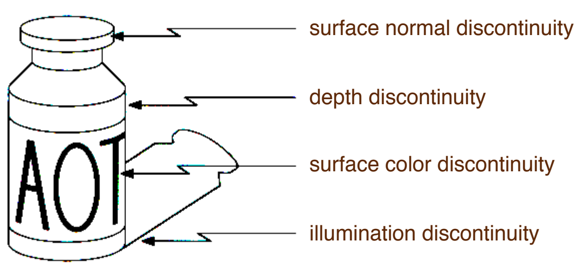
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Origin of edges

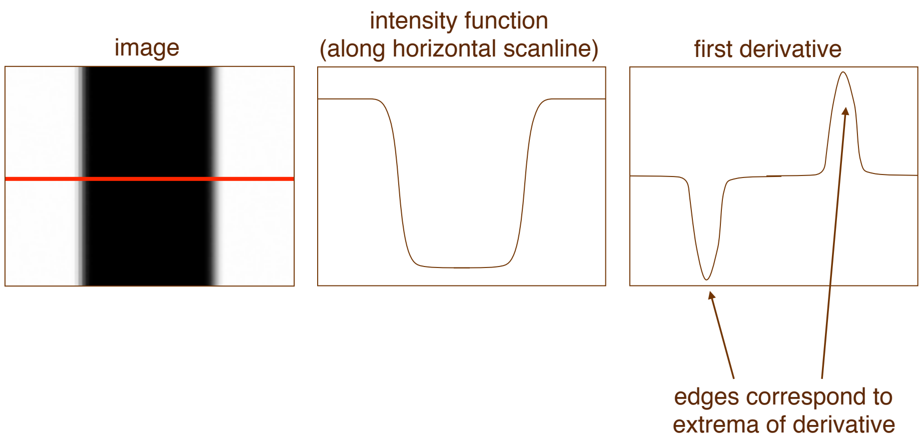
- Edges are caused by a variety of factors:



Source: Steve Seitz

Characterizing edges

- An edge is a place of rapid change in the image intensity function



source: Svetlana La

Finite difference filters

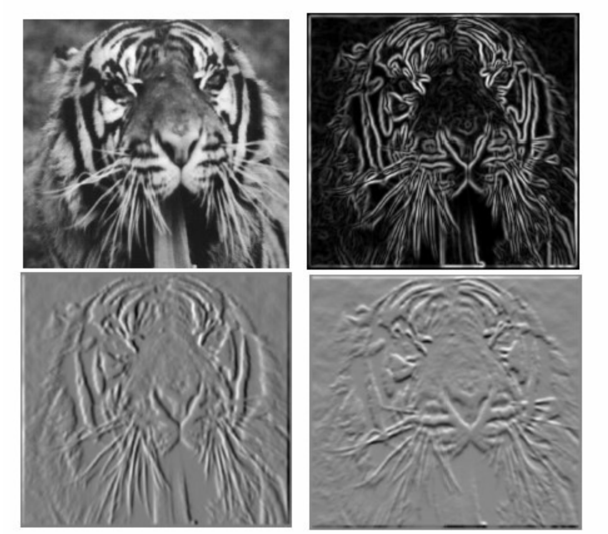
Prewitt: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$

Sobel: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$

Roberts: $M_x = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

Source: K. Grauman

Finite differences: example



- Which one is the gradient in the x-direction (resp. y-direction)?

source: Svetlana Lazebnik

Example

0	0	0	0	0
0	0	0	0	0
0	0	50	50	50
0	0	50	50	50
0	0	50	50	50

$I_x =$

0	0	0	0	0
0	50	100	150	150
0	50	100	150	150
0	0	0	0	0
0	0	0	0	0

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

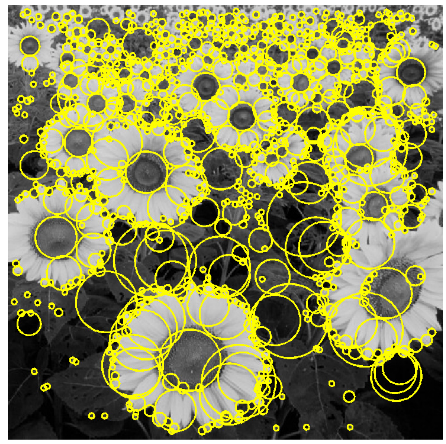
$I_y =$

0	0	0	0	0
0	50	50	0	0
0	100	100	0	0
0	150	150	0	0
0	150	150	0	0

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

Local Features

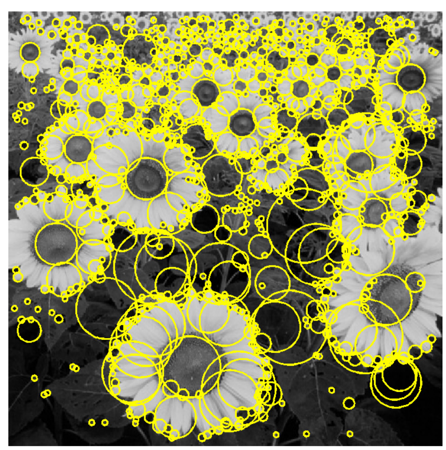
Feature points (locations) + feature descriptors

source: Svetlana Lazebnik

Local Features

Feature points (locations) + feature descriptors

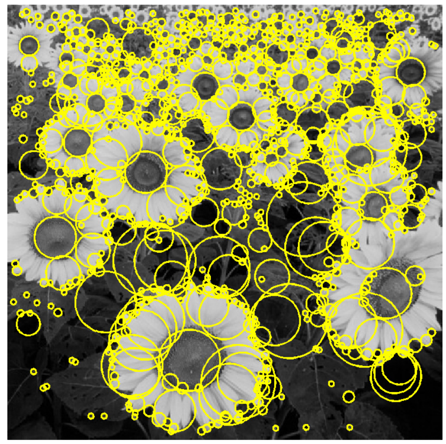



Where should we put features?

source: Svetlana Lazebnik

Local Features

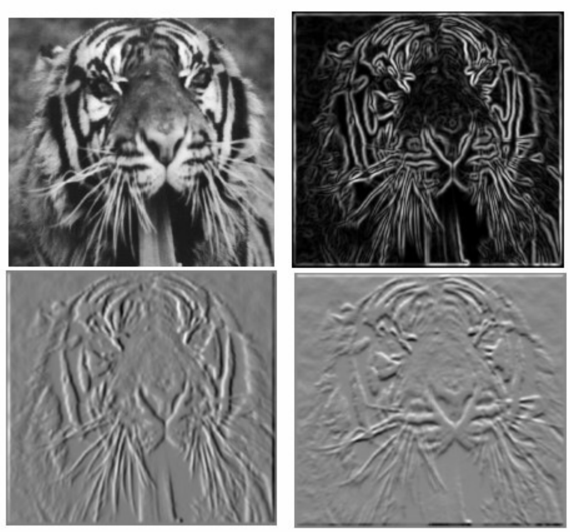
Feature points (locations) + feature descriptors



How should we describe them?

source: Svetlana Lazebnik

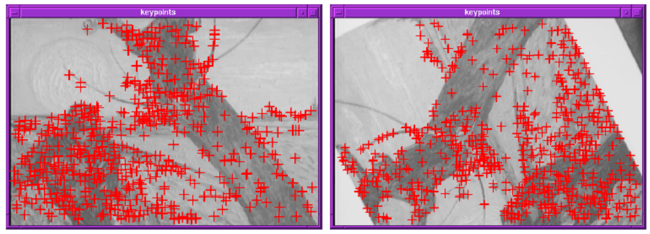
Where to put features?



Edges

source: Svetlana Lazebnik

Finding Corners



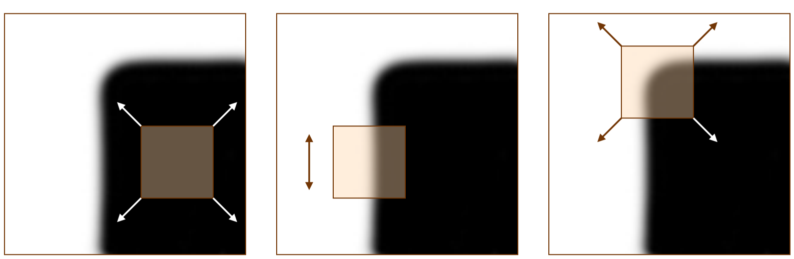
- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147--151.

source: Svetlana Lazebnik

Corner Detection: Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



“flat” region: no change in all directions

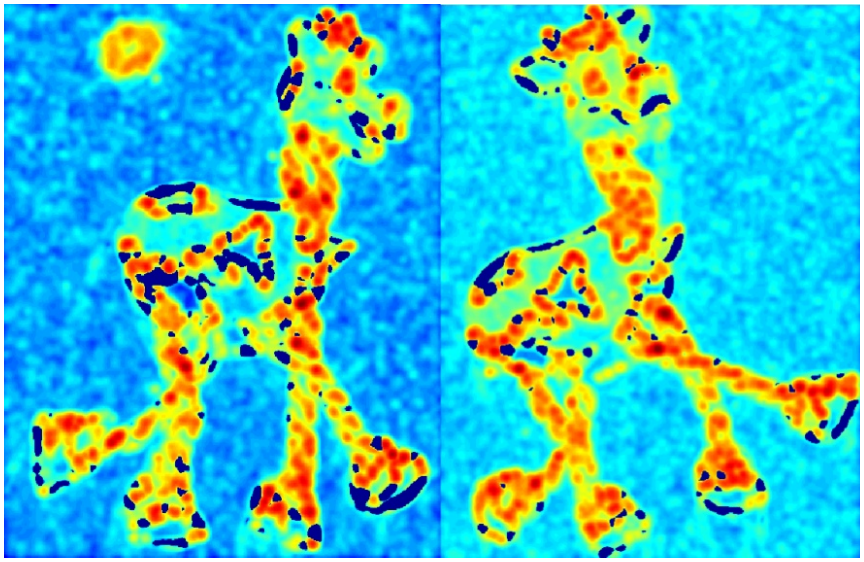
“edge”: no change along the edge direction

“corner”: significant change in all directions

Source: A. Efros

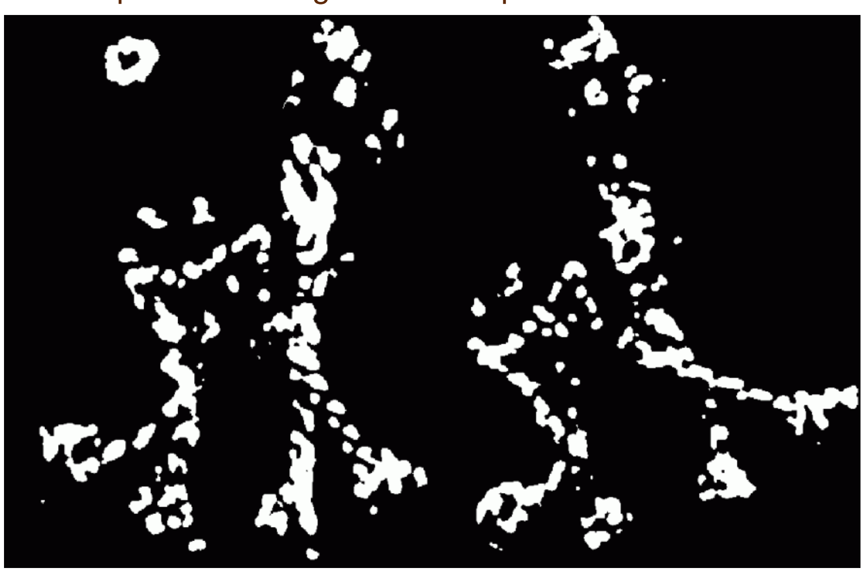
Harris Detector: Steps

Compute corner response *R*



Harris Detector: Steps

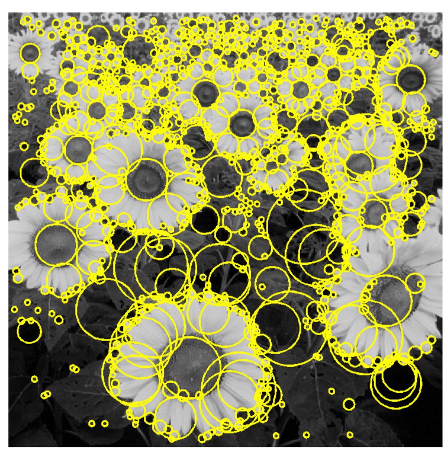
Find points with large corner response: *R*>threshold



Harris Detector: Steps

Local Features

Feature points (locations) + feature descriptors



How should we describe them?

source: Svetlana Lazebnik

SIFT

- SIFT (Scale Invariant Feature Transform) - stable robust and distinctive local features
- Current most popular shape based feature – description of local shape (oriented edges) around a keypoint

Scale Invariance

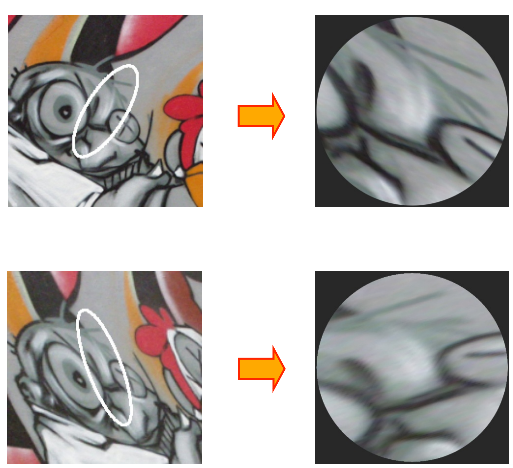
- Find the points, whose surrounding patches (with some scale) are distinctive find points that are distinctive in both position (x,y) and scale

Invariant description

- Geometrically transformed versions of the same neighborhood will give rise to regions that are related by the same transformation
- What to do if we want to compare the appearance of these image regions?
  - *Normalization*: transform these regions into same-size circles

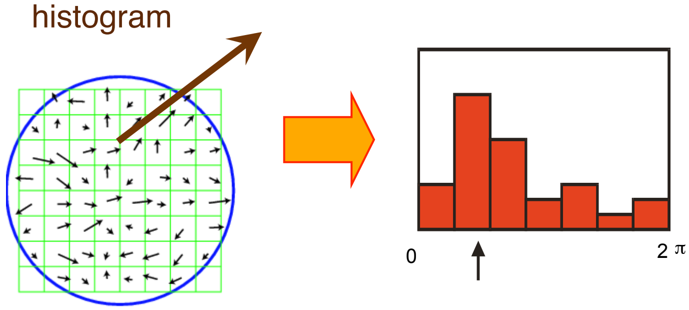
Affine normalization

- Problem: There is no unique transformation from an ellipse to a unit circle
  - We can rotate or flip a unit circle, and it still stays a unit circle



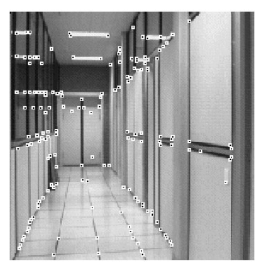
Eliminating rotation ambiguity

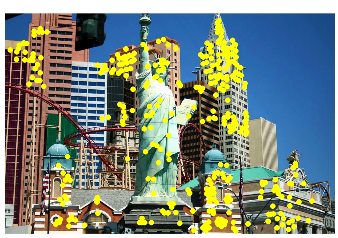
- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram

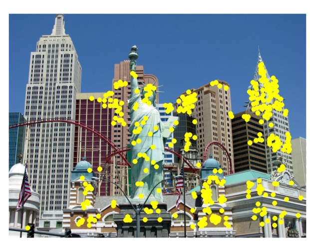


Applications

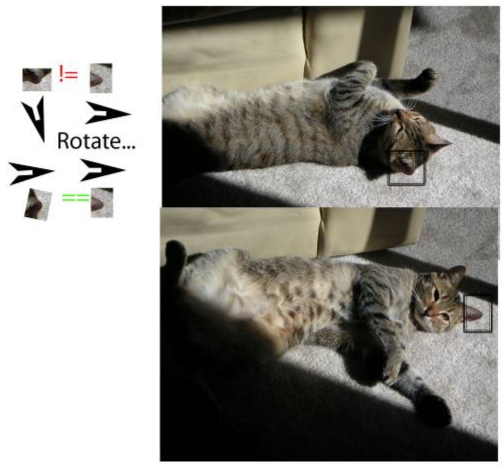
- Feature points are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Robot navigation
  - Indexing and database retrieval
  - Object recognition





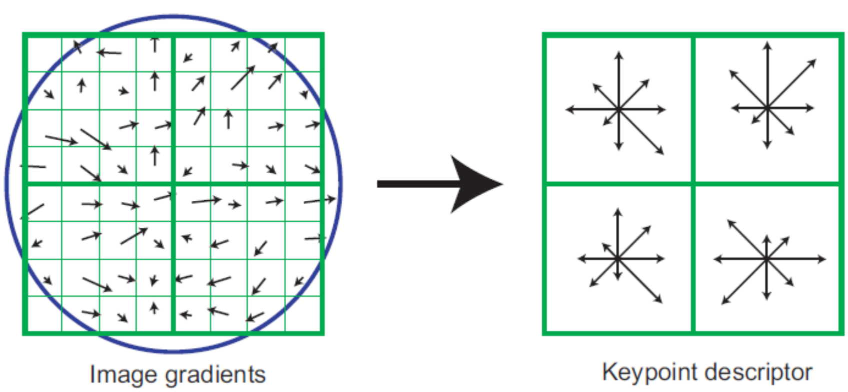


Rotation Invariance



Slide source: Tom Duerig

Feature descriptor

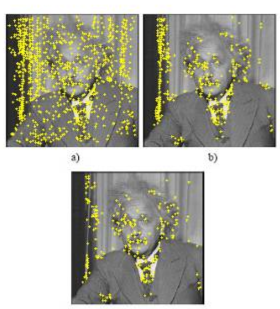


Feature descriptor

- Based on 16*16 patches
- 4*4 subregions
- 8 bins in each subregion
- 4*4*8=128 dimensions in total

Actual SIFT stage output

Keypoint detection



a) b) c)

Figure 5: a) Maxima of DoG across scales. b) Remaining keypoints after removal of low contrast points. C) Remaining keypoints after removal of edge responses (bottom).

SIFT matlab tutorial 17

Final keypoints with selected orientation and scale

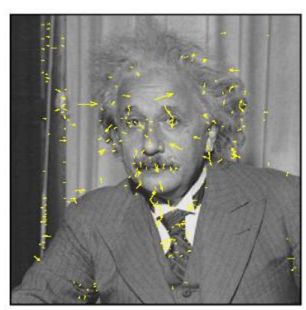


Figure 6: Extracted keypoints, arrows indicate scale and orientation.

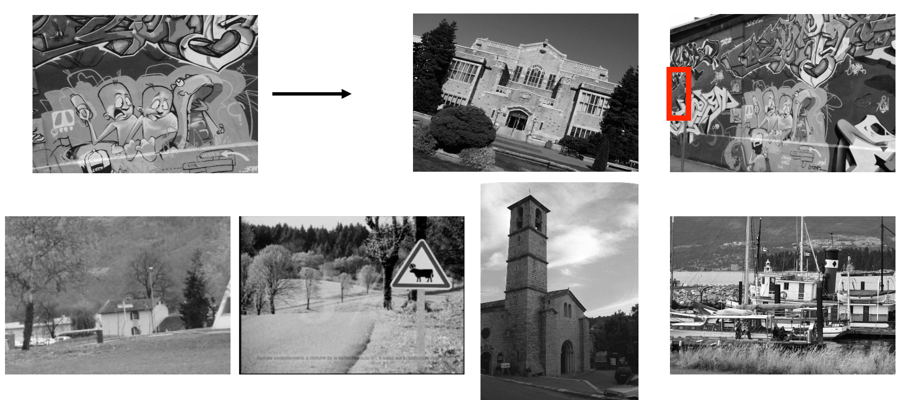
SIFT matlab tutorial 18

Application: object recognition

- The SIFT features of training images are extracted and stored
- For a query image

1. Extract SIFT feature
2. Efficient nearest neighbor indexing
3. 3 keypoints, Geometry verification

Recognition with local features



> 5000 images

[Local greyvalue invariants for image retrieval, C. Schmid and R. Mohr, PAMI 1997]

Semi-local constraints, neighboring points should match, angles, length ratios should be similar

Recognition with local features

- For each feature in the query
- Find the nearest neighbour in the database of features
- Check which object/model it belongs to
- That object/model gets one vote
- Repeat for all detected features in the query
- Model with largest number of votes wins

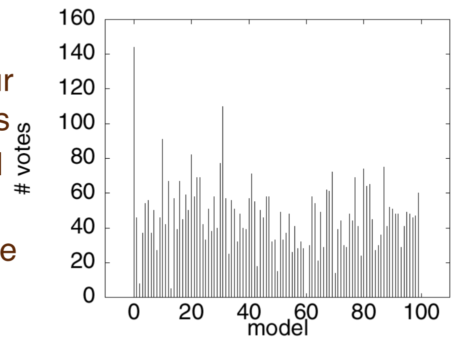


Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE

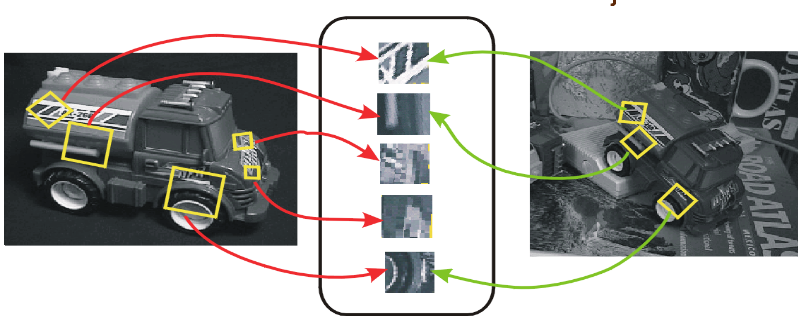
Local features for recognition of object instances

Local features for recognition of object instances

- Lowe, et al. 1999, 2003
- Mahamud and Hebert, 2000
- Ferrari, Tuytelaars, and Van Gool, 2004
- Rothganger, Lazebnik, and Ponce, 2004
- Moreels and Perona, 2005
- …

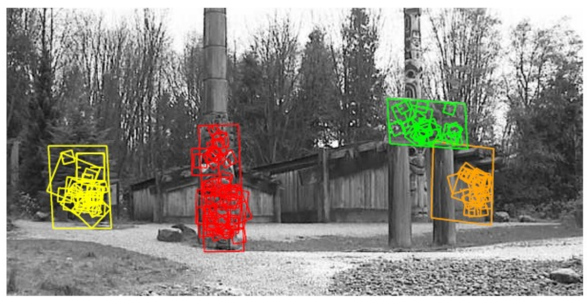
Invariant Local Features

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters
- SIFT features covered previously
- For each query view – count how many features can be matched with each of the database objects



SIFT Features

Local features for recognition of landmarks

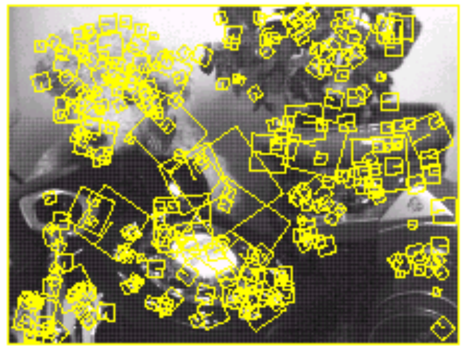


3D Object Recognition

- Only 3 keys are needed for recognition, so extra keys provide robustness
- Affine model is no longer as accurate

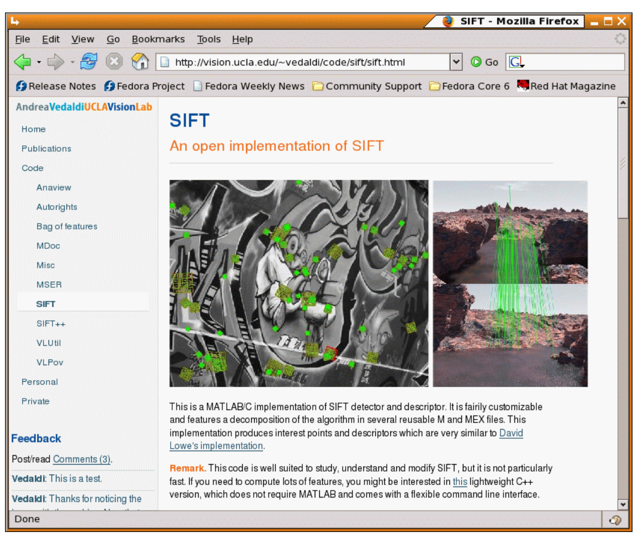
Test of illumination invariance

- Same image under differing illumination



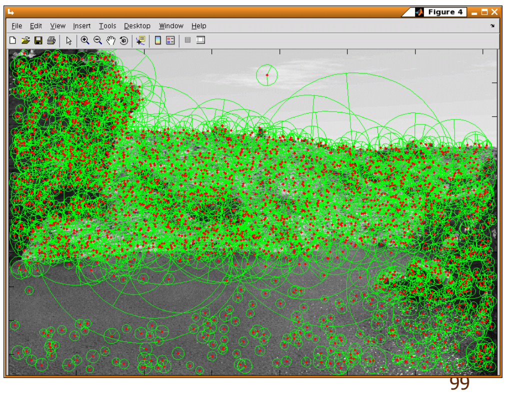
273 keys verified in final match

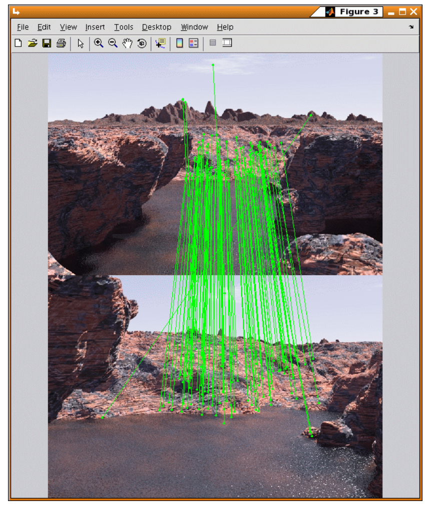
SOFTWARE for Matlab (at UCLA, Oxford) www.VLFeat.org



SIFT MATCHING DEMO

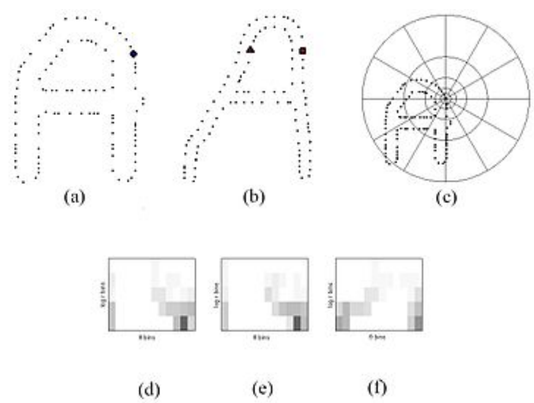
http://www.vlfeat.org/overview/sift.html



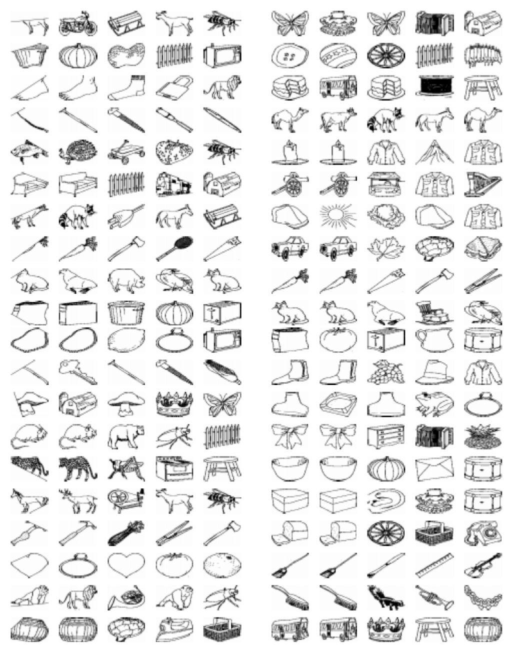


99

Shape Context

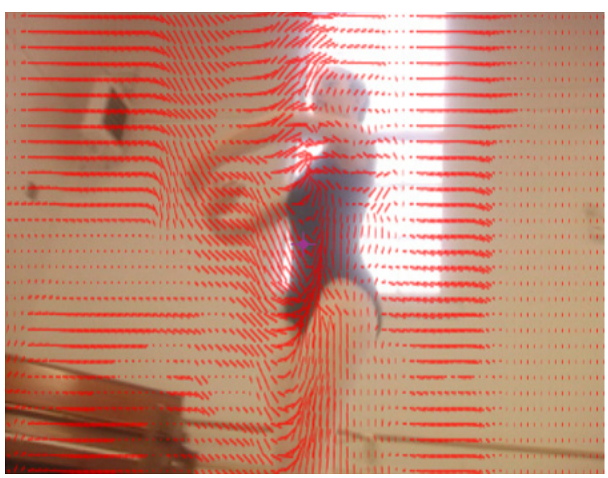


- (a) and (b) are the sampled edge points of the two shapes. (c) is the diagram of the log-polar bins used to compute the shape context. (d) is the shape context for the circle, (e) is that for the diamond, and (f) is that for the triangle. As can be seen, since (d) and (e) are the shape contexts for two closely related points, they are quite similar, while the shape context in (f) is very different.

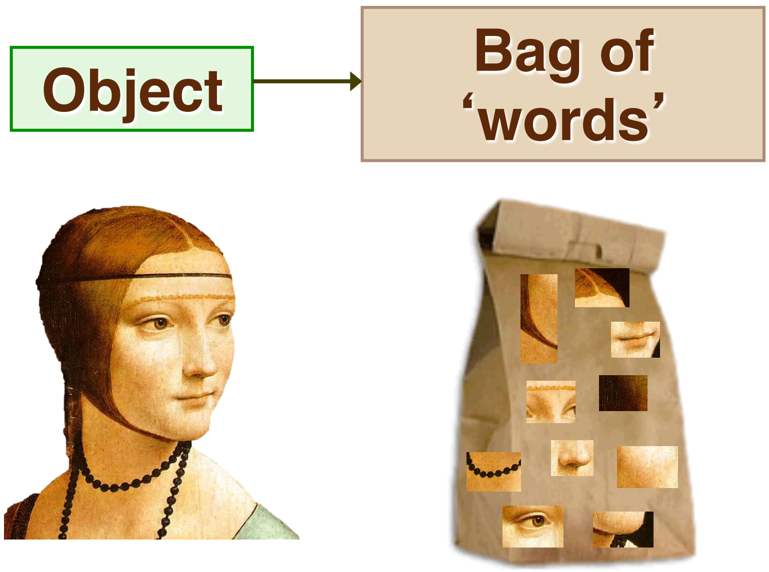


Motion/Optical Flow

- Feature tracking, optical flow – associate features in video

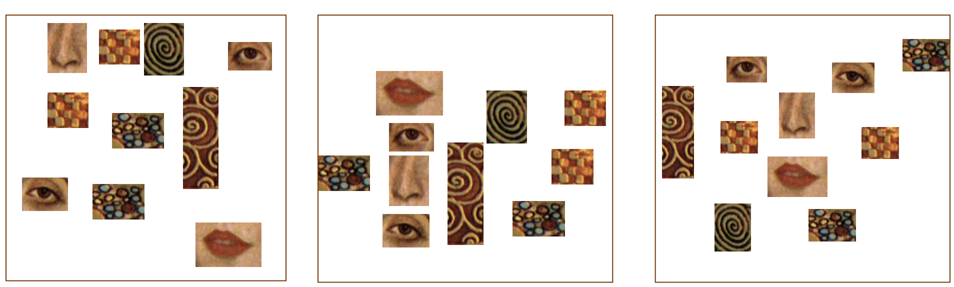


Bag-of-features models



Objects as texture

- All of these are treated as being the same

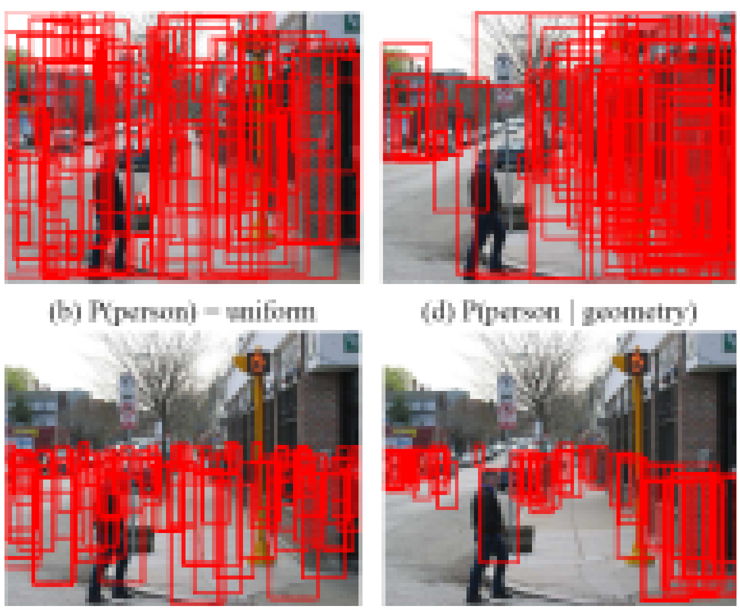


- No distinction between foreground and background: scene recognition?

Sliding window approaches

– Scale / orientation range to search over
– Speed
– Context

Context



(b) P(person) = uniform (d) P(person | geometry)

(f) P(person | viewpoint) (g) P(person|viewpoint,geometry)

Hoiem, Efros, Herbert, 2006