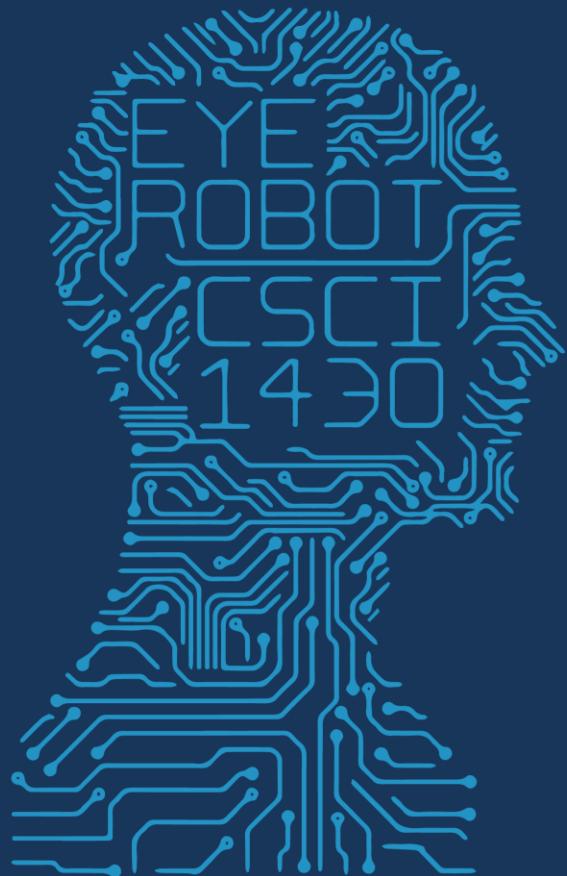
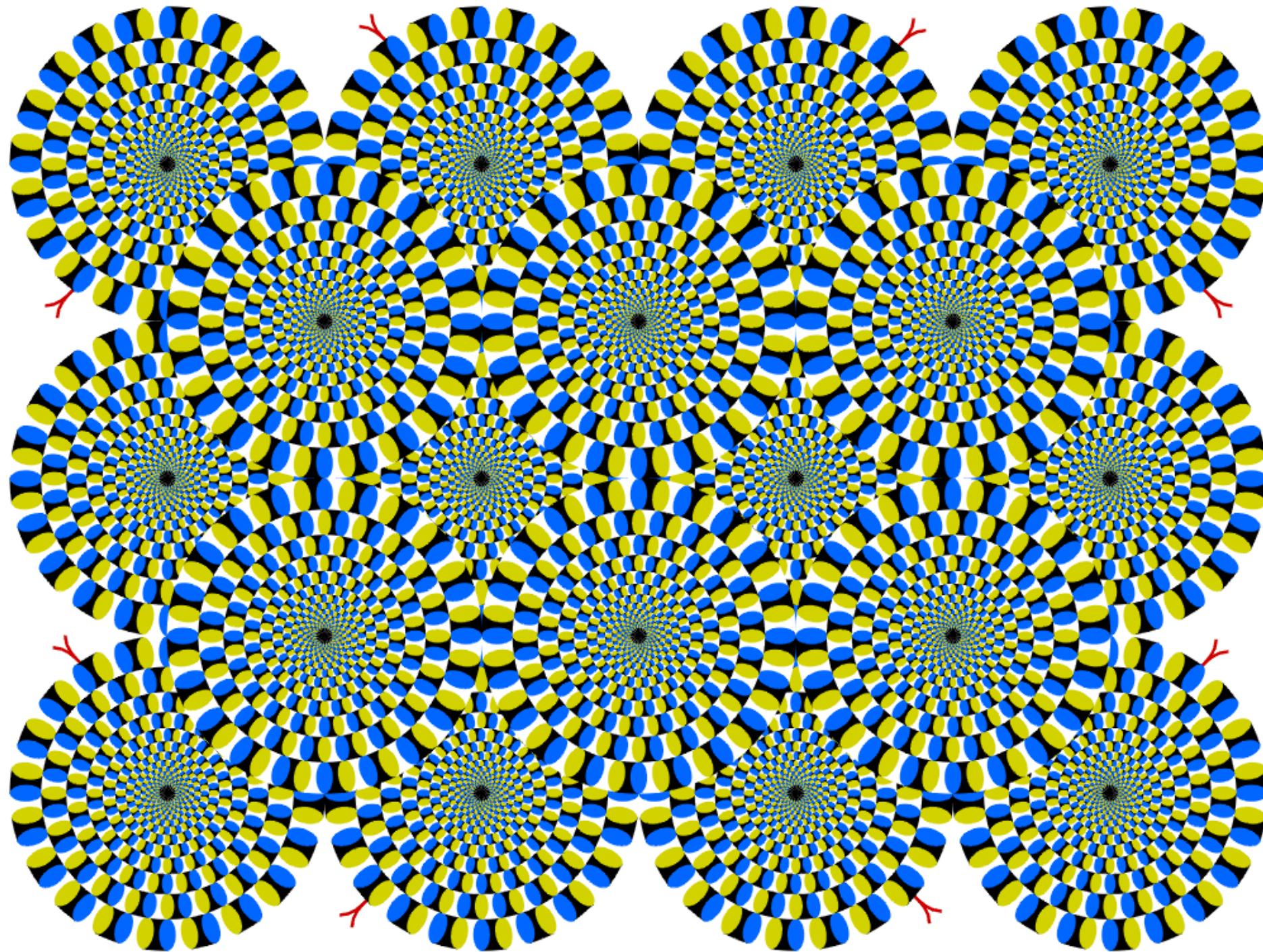


FUTURE VISION

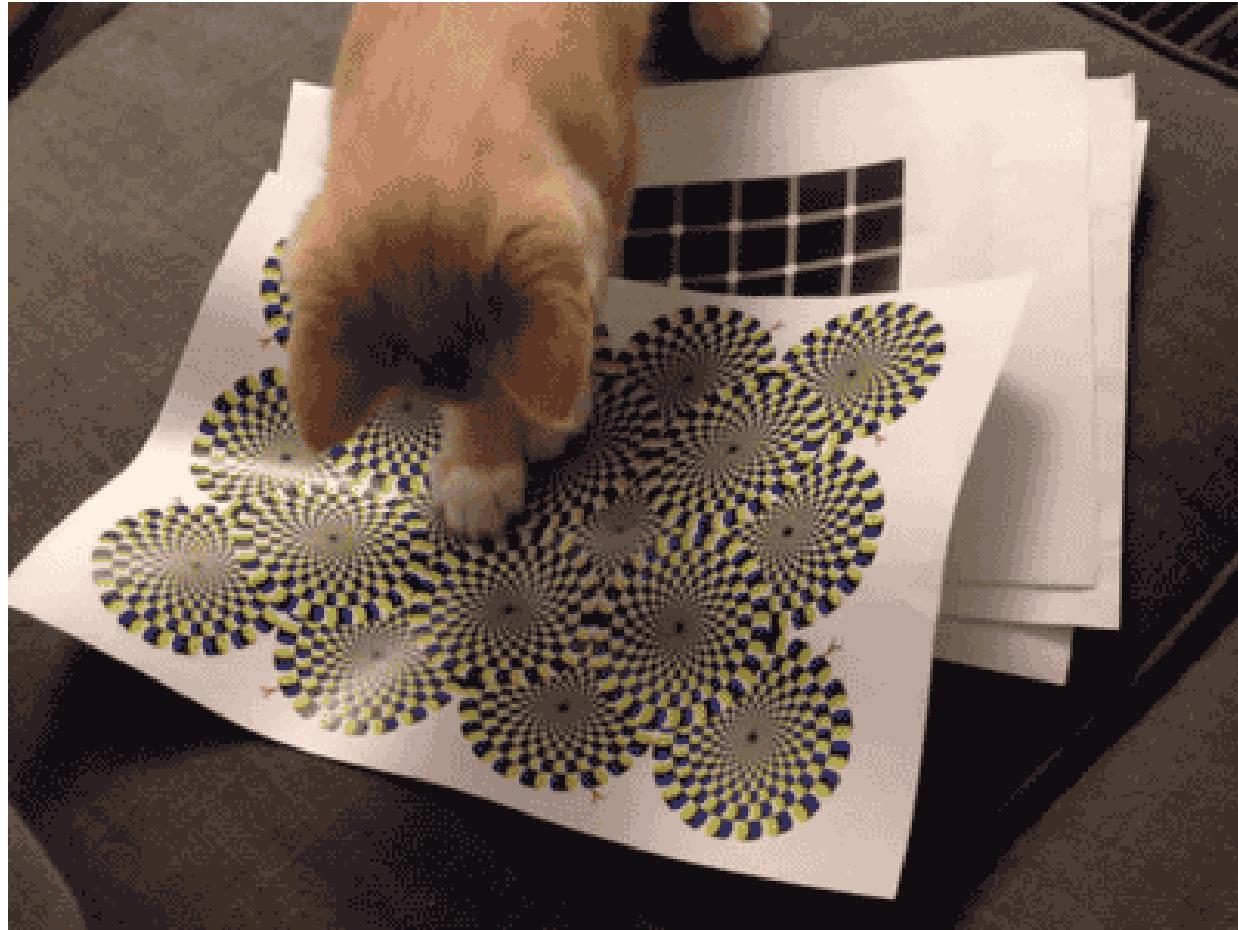
1950



30 December 2019
COMPUTER VISION



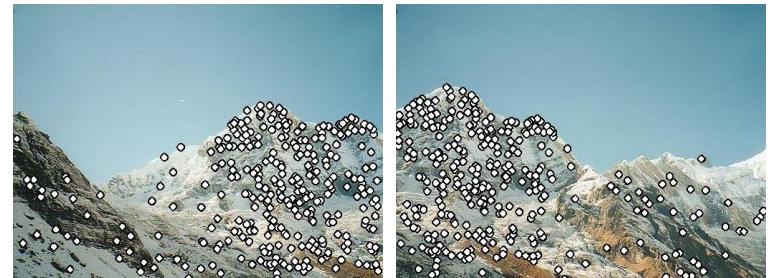
Does it work on other animals?



Local features: main components

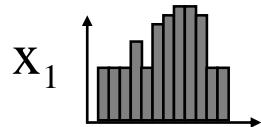
1) Detection:

Find a set of distinctive key points.

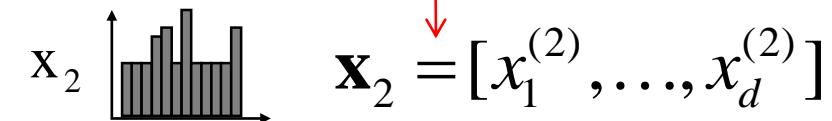
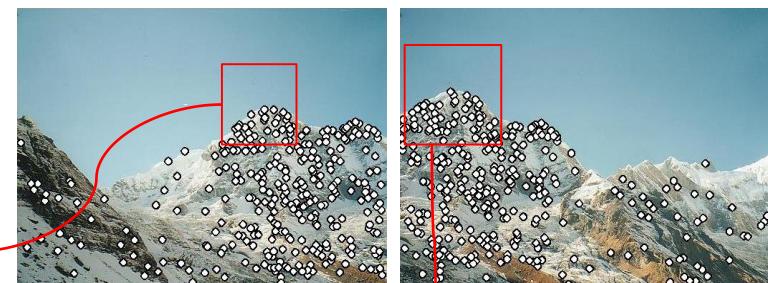


2) Description:

Extract feature descriptor around each interest point as vector.



$$\mathbf{x}_1 = [x_1^{(1)}, \dots, x_d^{(1)}]$$

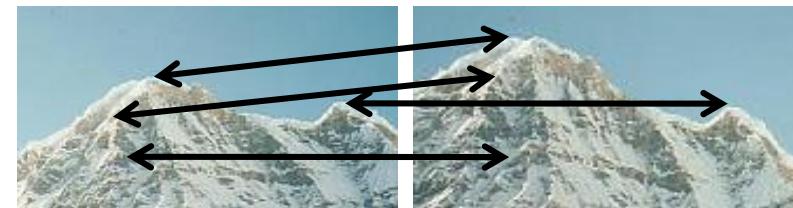


$$\mathbf{x}_2 = [x_1^{(2)}, \dots, x_d^{(2)}]$$

3) Matching:

Compute distance between feature vectors to find correspondence.

$$d(\mathbf{x}_1, \mathbf{x}_2) < T$$



**HOW INVARIANT ARE
HARRIS CORNERS?**

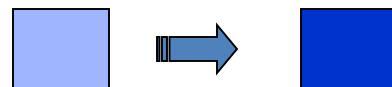
Invariance and covariance

Are locations *invariant* to photometric transformations
and *covariant* to geometric transformations?

- **Invariance:** image is transformed and corner locations do not change
- **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations

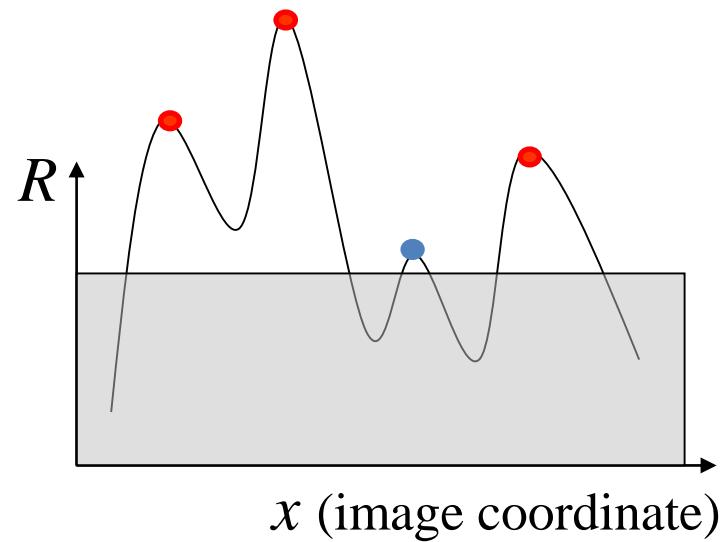
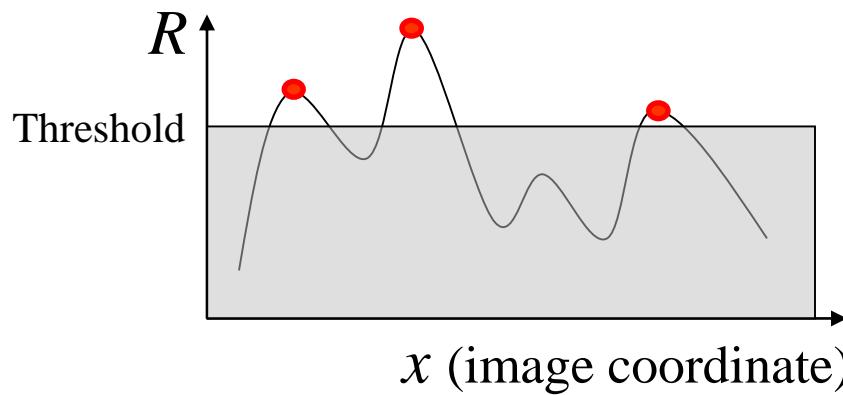


Affine intensity change



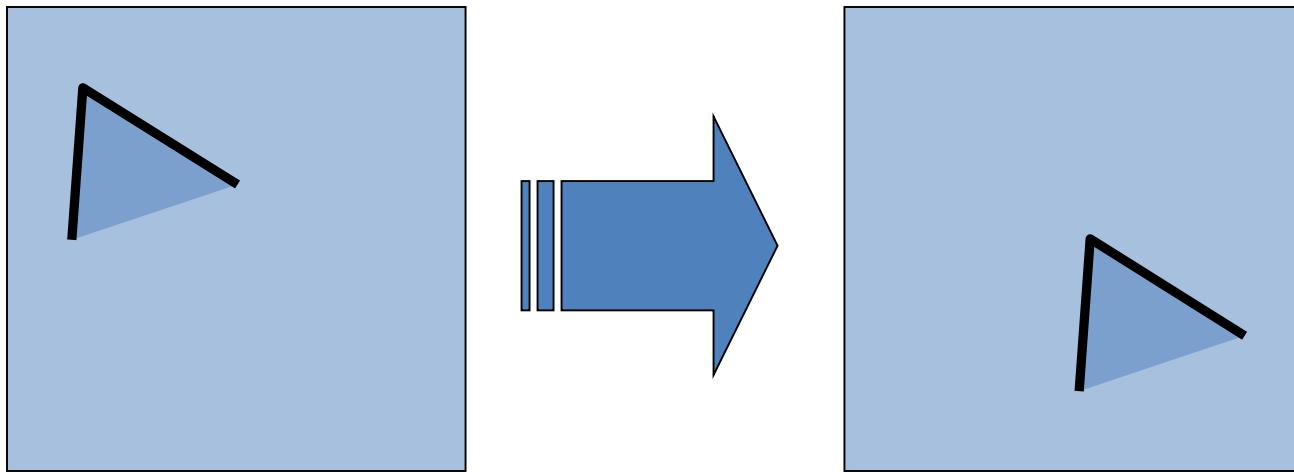
$$I \rightarrow a I + b$$

- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow a I$



Partially invariant to affine intensity change

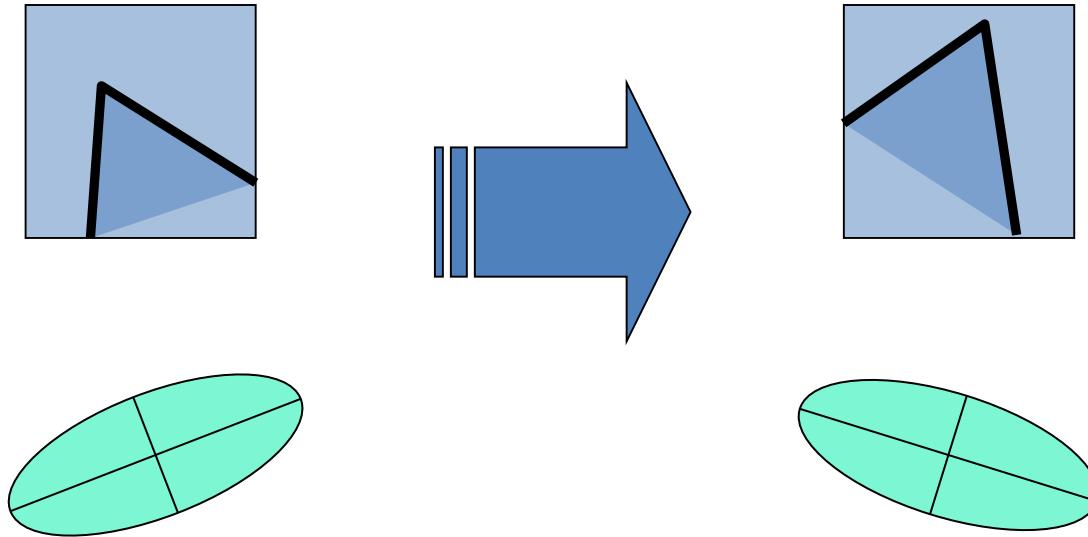
Image translation



- Derivatives and window function are shift-invariant.

Corner location is covariant w.r.t. translation

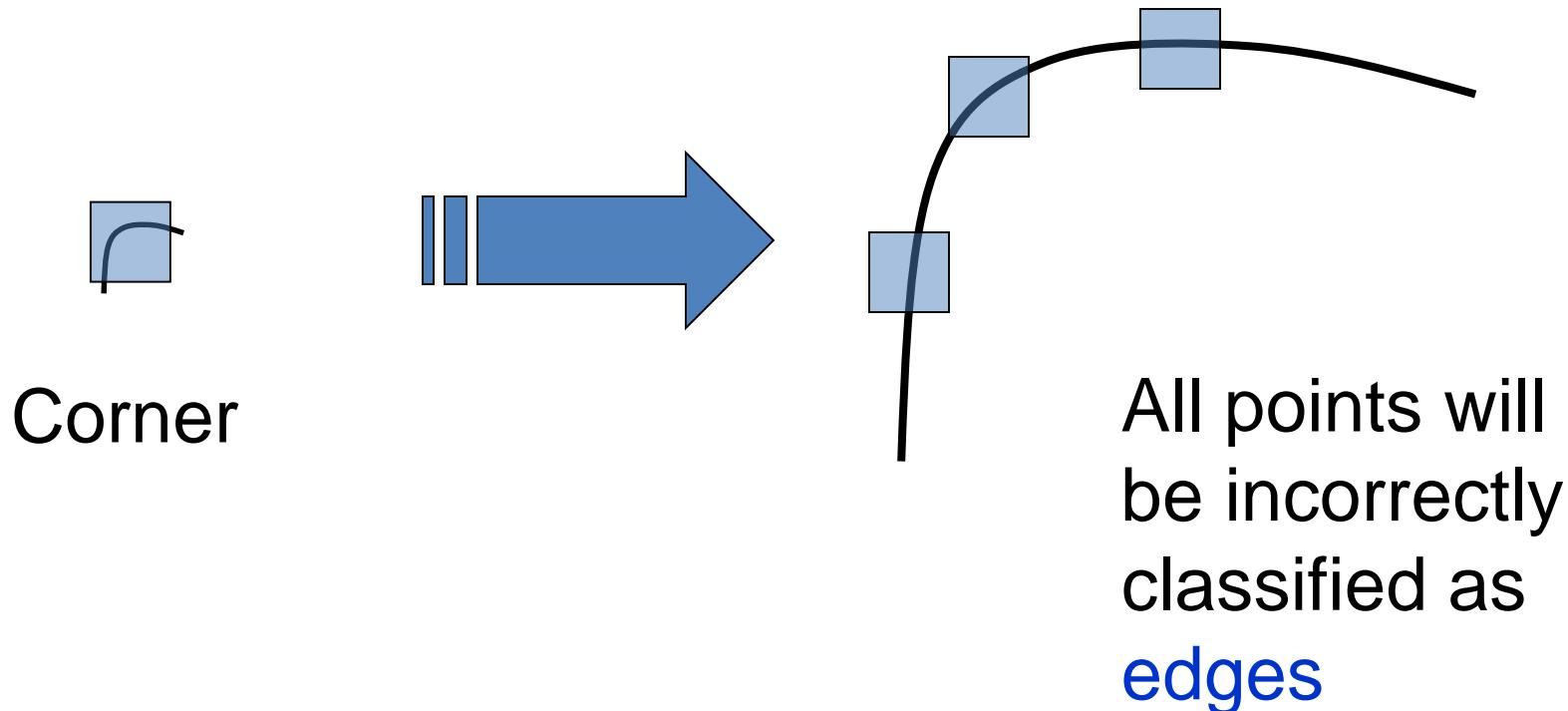
Image rotation



Second moment ellipse rotates but its shape (i.e., eigenvalues) remains the same.

Corner location is covariant w.r.t. rotation

Scaling



Corner location is not covariant to scaling!

**WHAT IS THE ‘SCALE’ OF A
FEATURE POINT?**

Automatic Scale Selection



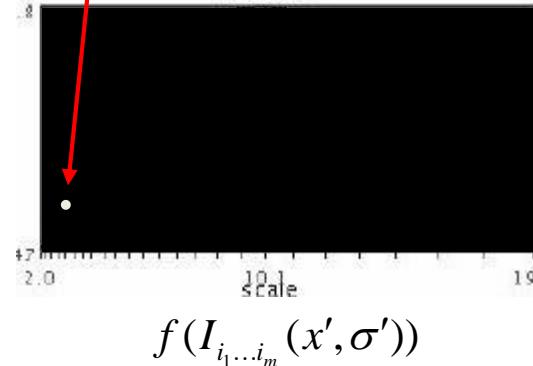
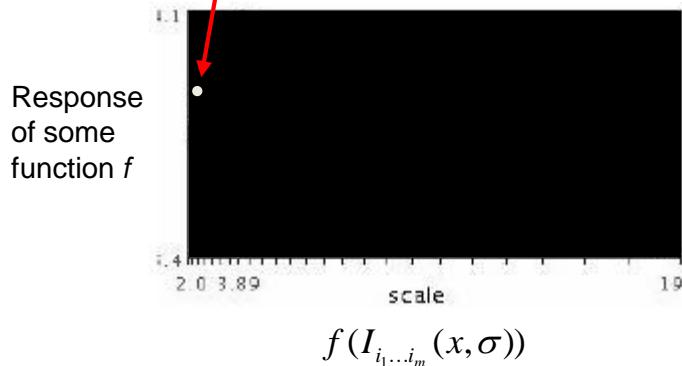
$$f(I_{i_1 \dots i_m}(x, \sigma)) = f(I_{i_1 \dots i_m}(x', \sigma'))$$

How to find patch sizes at which f response is equal?

What is a good f ?

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

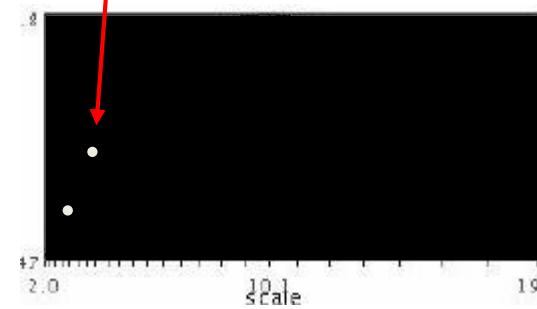
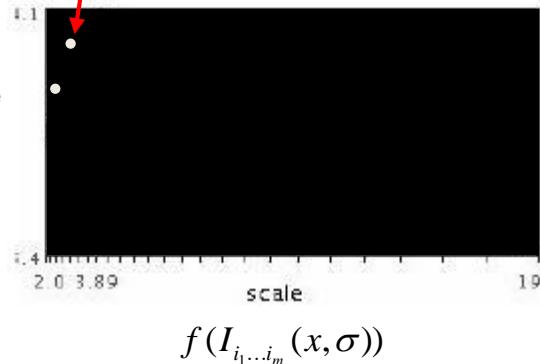


Automatic Scale Selection

- Function responses for increasing scale (scale signature)



Response
of some
function f

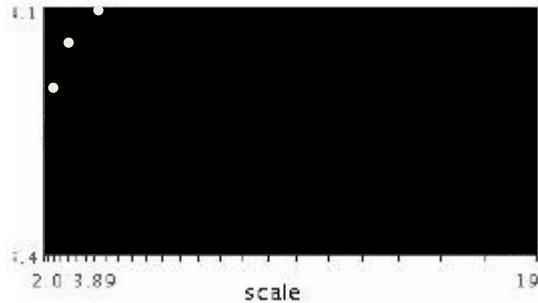


Automatic Scale Selection

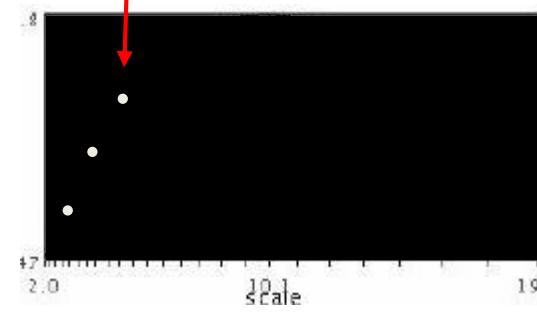
- Function responses for increasing scale (scale signature)



Response
of some
function f



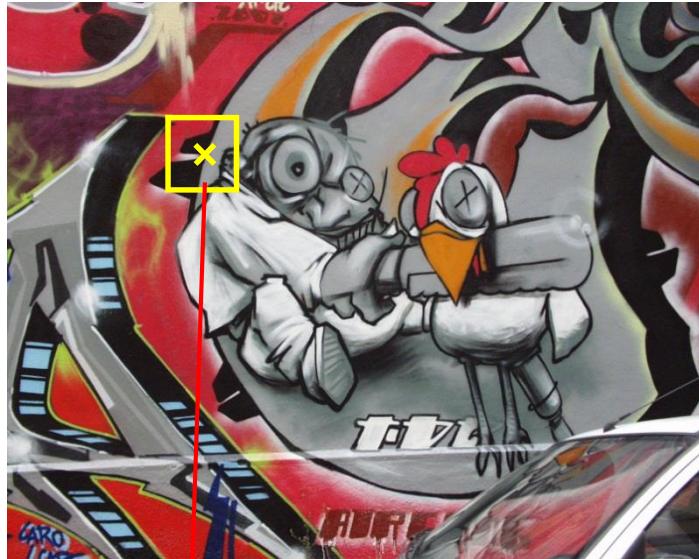
$$f(I_{i_1 \dots i_m}(x, \sigma))$$



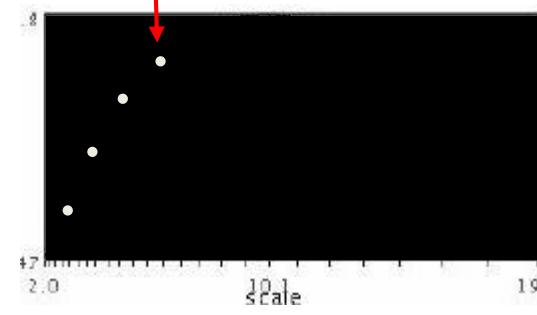
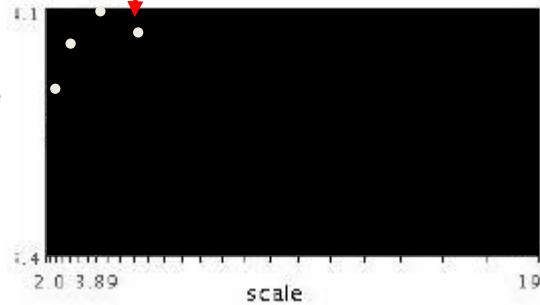
$$f(I_{i_1 \dots i_m}(x', \sigma'))$$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

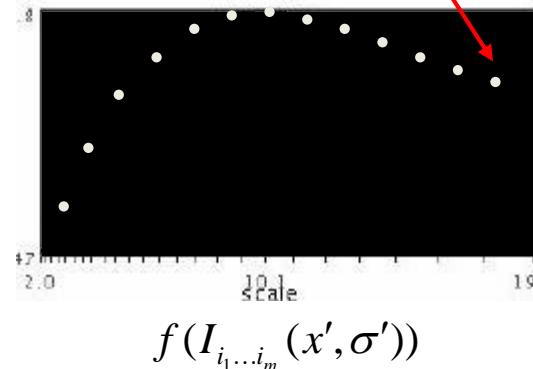
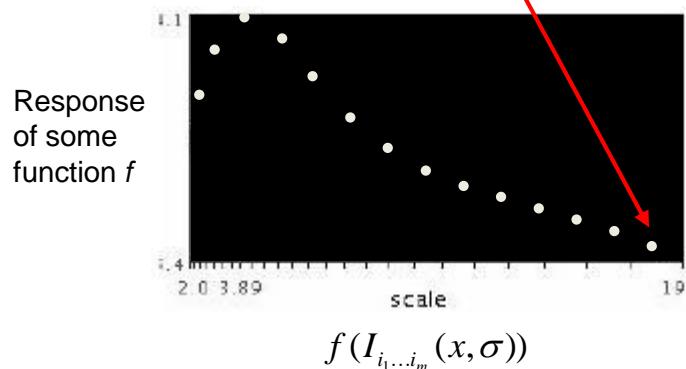
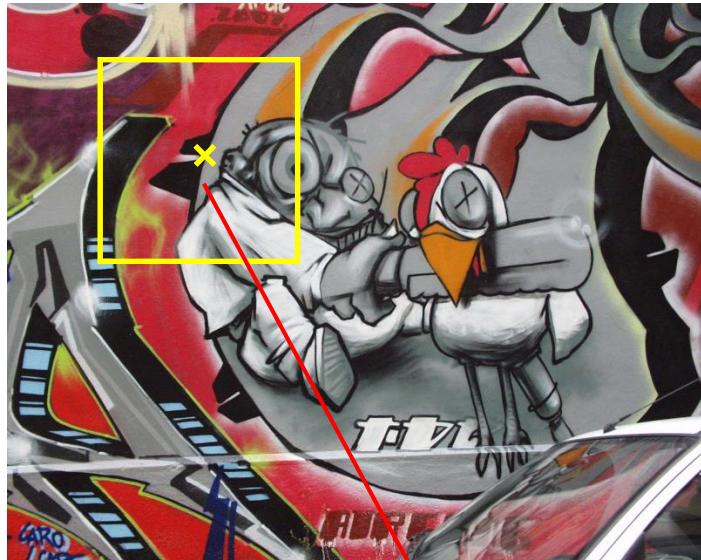


Response
of some
function f



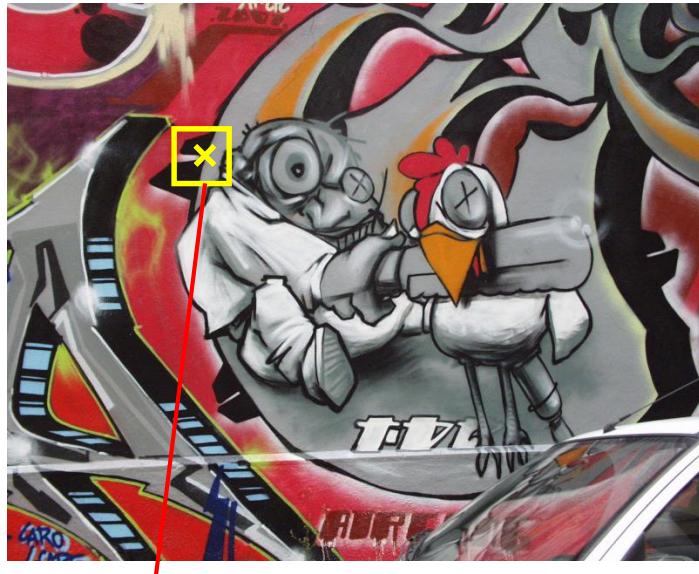
Automatic Scale Selection

- Function responses for increasing scale (scale signature)

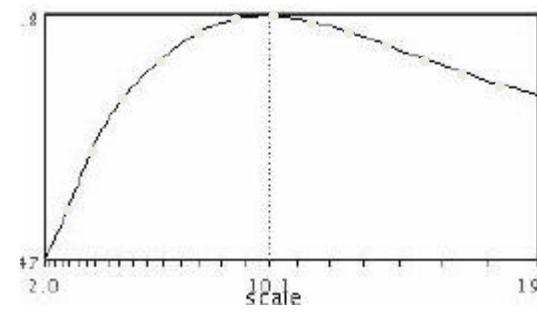
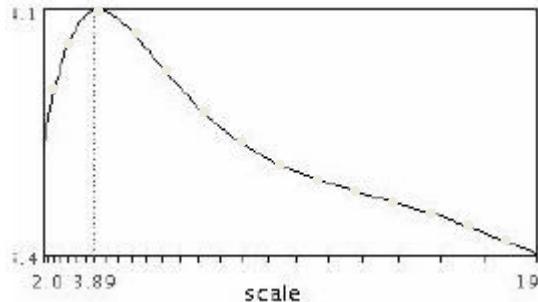


Automatic Scale Selection

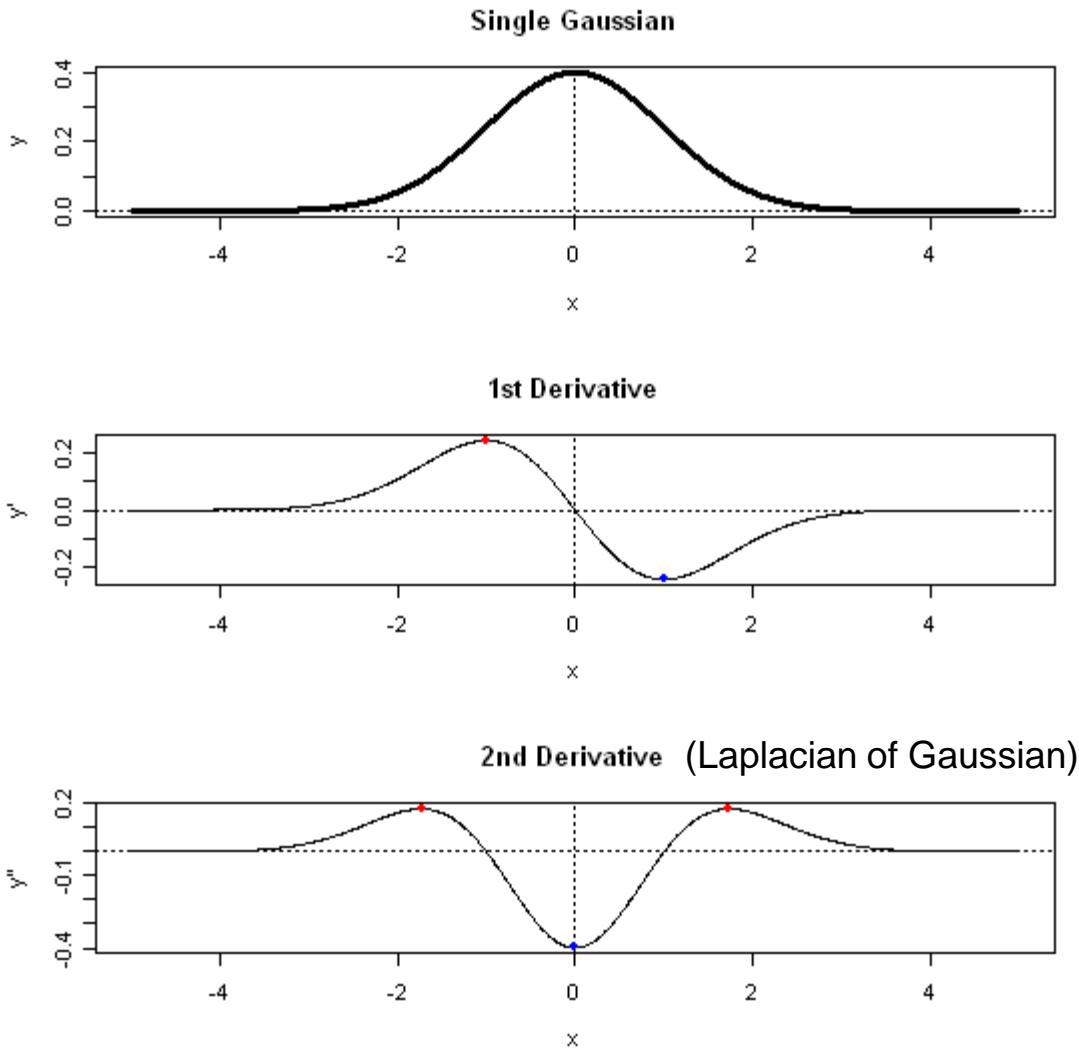
- Function responses for increasing scale (scale signature)



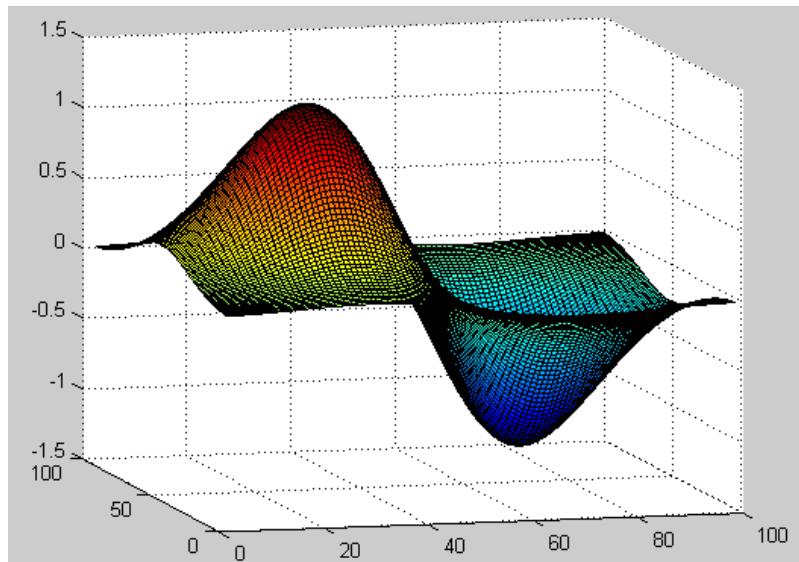
Response
of some
function f



What Is A Useful Signature Function f ?



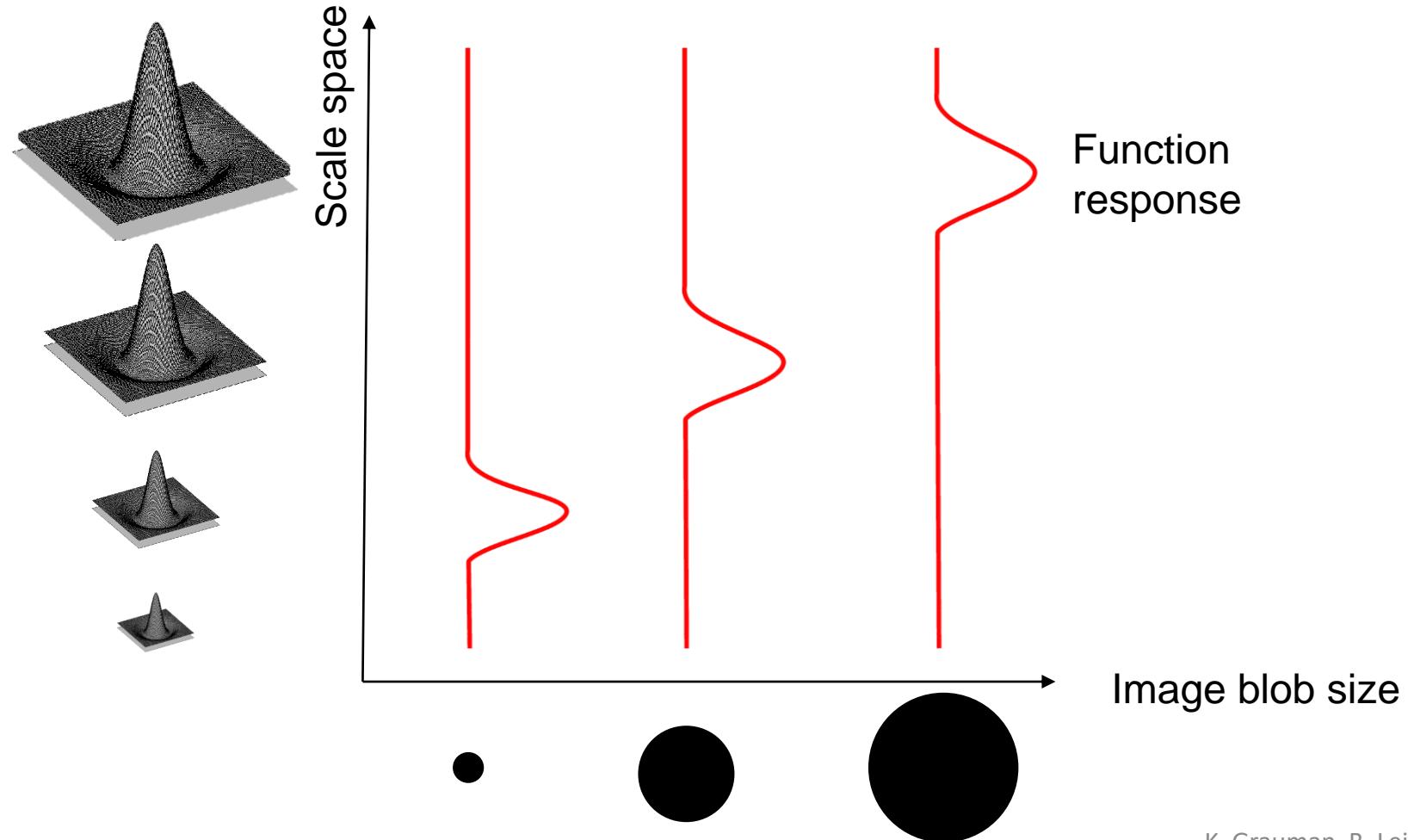
1st Derivative of Gaussian



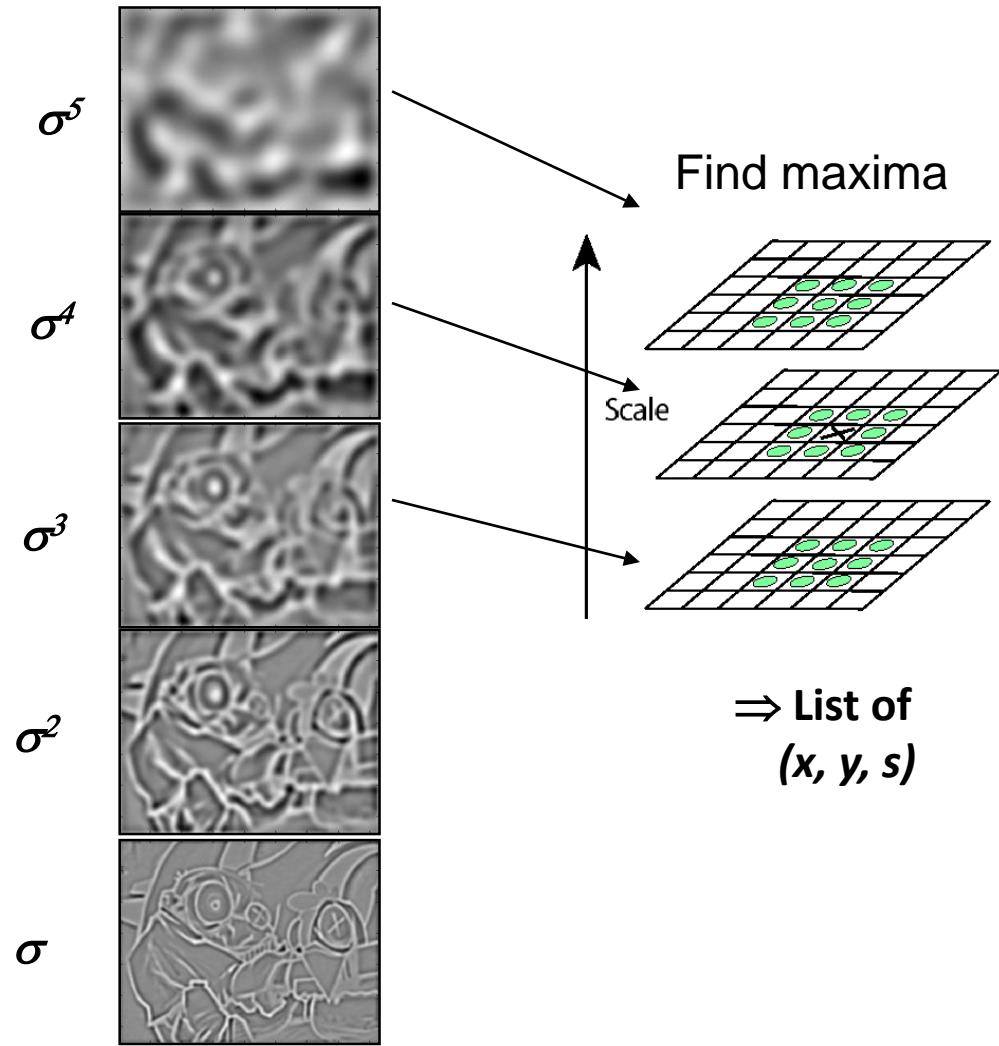
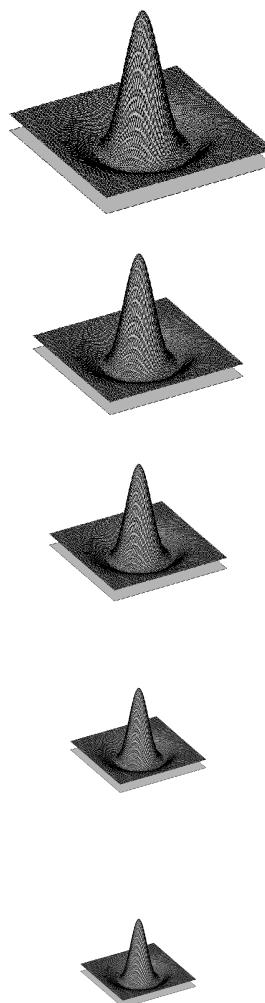
What Is A Useful Signature Function f ?

“Blob” detector is common for corners

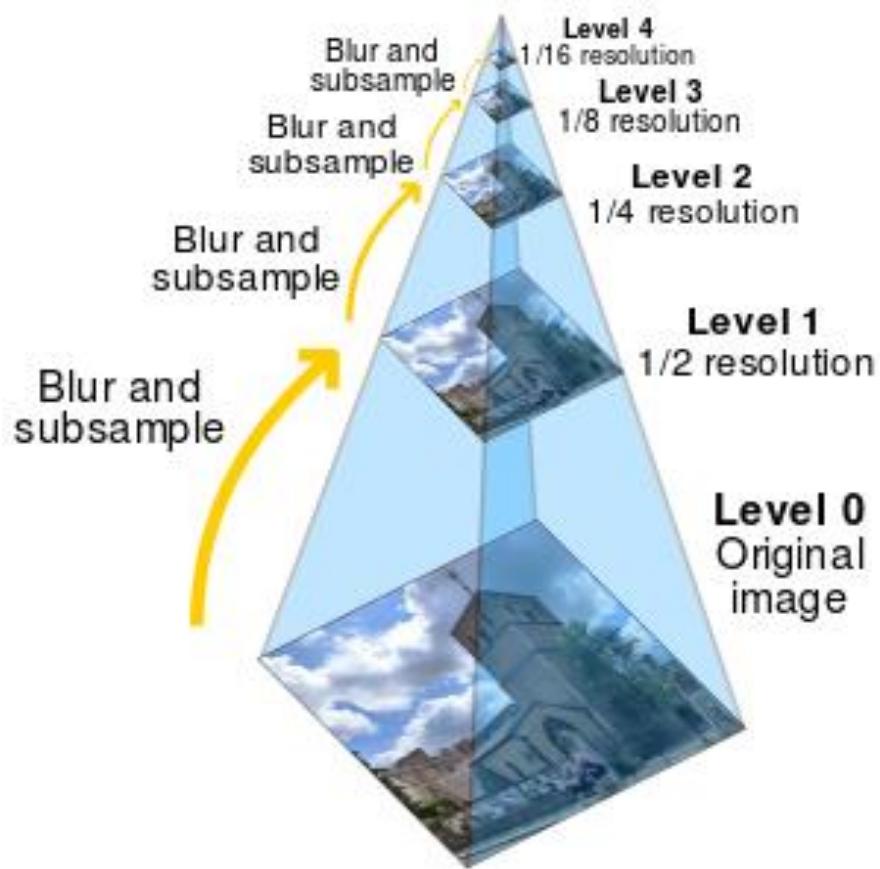
- Laplacian (2nd derivative) of Gaussian (LoG)



Find local maxima in position-scale space



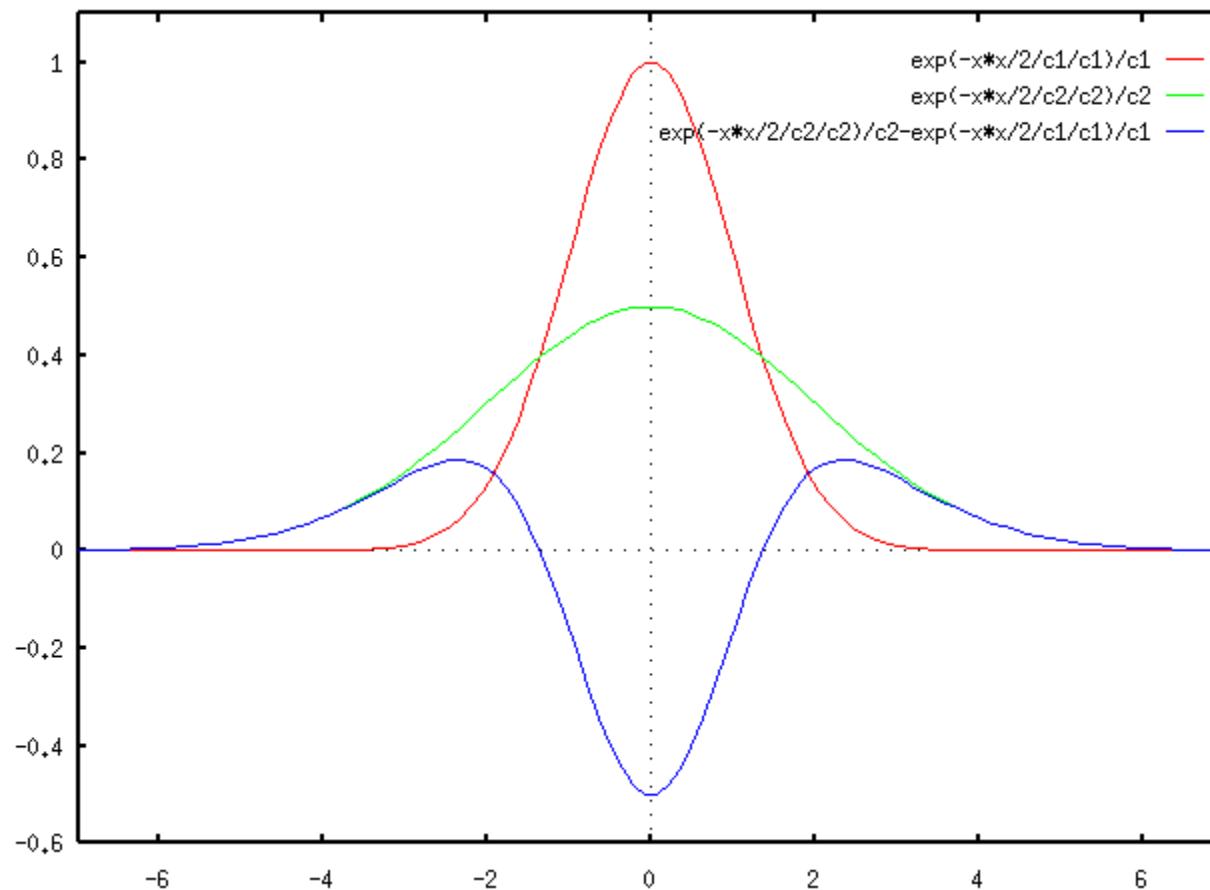
Laplacian pyramid



Like a Gaussian pyramid (left), but subtract layers from each other before subsampling.

Alternative kernel

Approximate LoG with Difference-of-Gaussian (DoG).



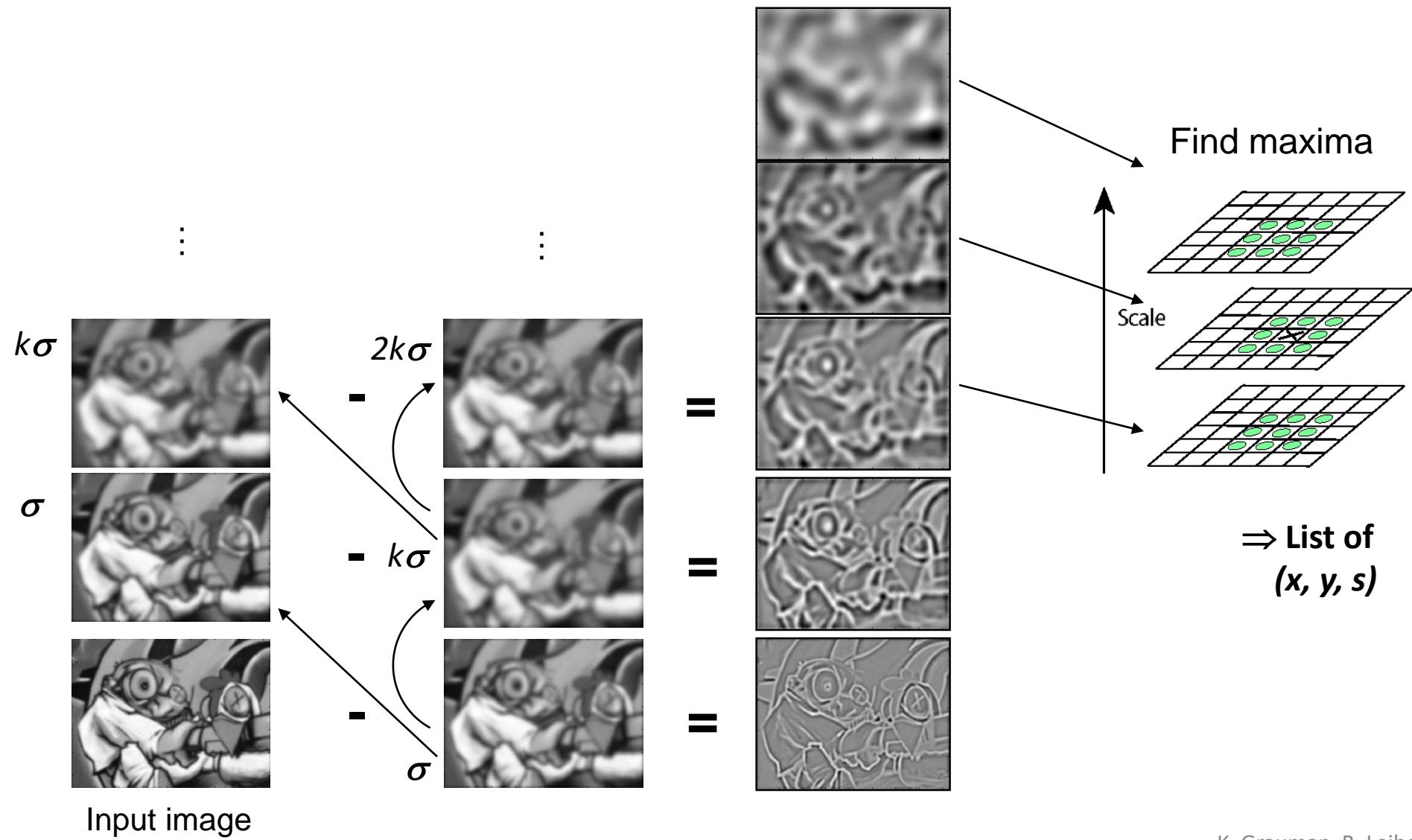
Alternative kernel

Approximate LoG with Difference-of-Gaussian (DoG).

1. Blur image with σ Gaussian kernel
2. Blur image with $k\sigma$ Gaussian kernel
3. Subtract 2. from 1.

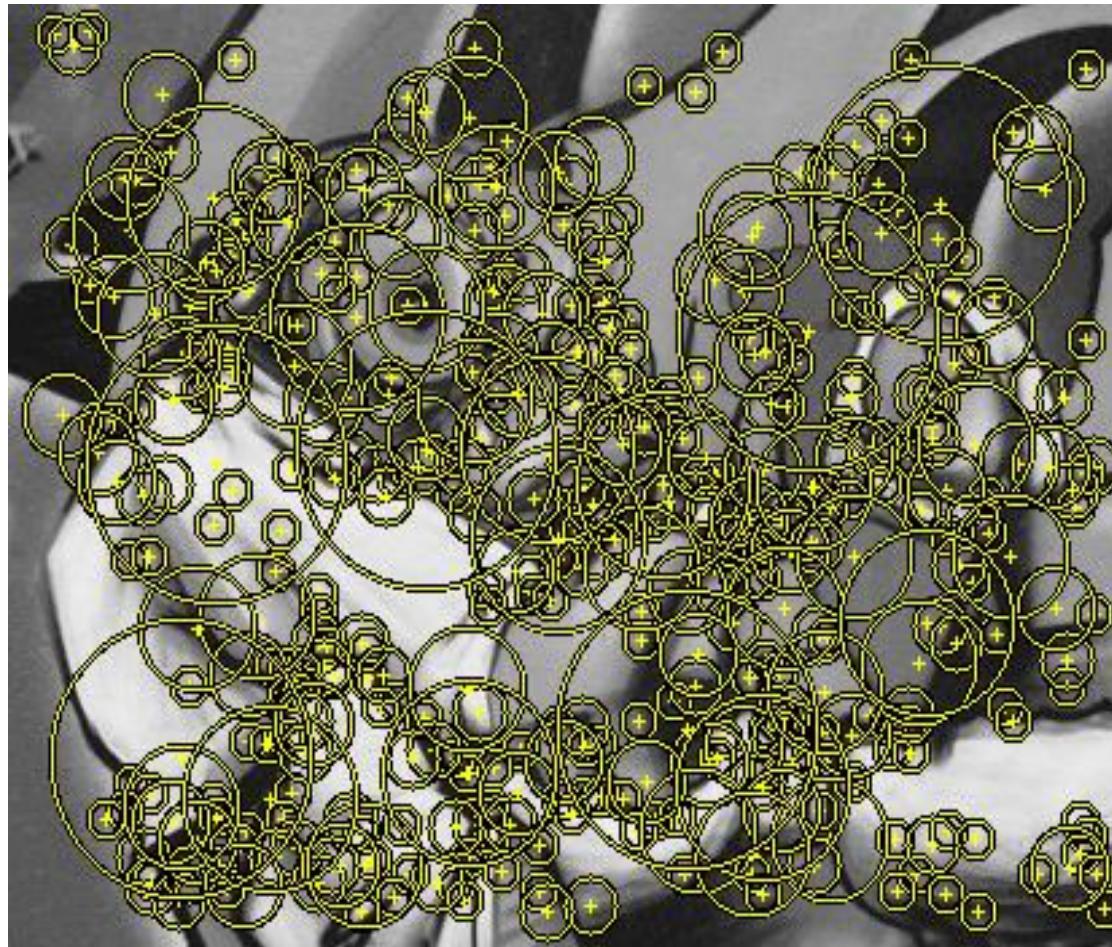


Find local maxima in position-scale space of DoG



Results: Difference-of-Gaussian

- Larger circles = larger scale
- Descriptors with maximal scale response



Comparison of Keypoint Detectors

Table 7.1 Overview of feature detectors.

Feature Detector	Corner	Blob	Region	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	✓			✓			+++	+++	+++	++
Hessian		✓		✓			++	++	++	+
SUSAN	✓			✓			++	++	++	+++
Harris-Laplace	✓	(✓)		✓	✓		+++	+++	++	+
Hessian-Laplace	(✓)	✓		✓	✓		+++	+++	+++	+
DoG	(✓)	✓		✓	✓		++	++	++	++
SURF	(✓)	✓		✓	✓		++	++	++	+++
Harris-Affine	✓	(✓)		✓	✓	✓	+++	+++	++	++
Hessian-Affine	(✓)	✓		✓	✓	✓	+++	+++	+++	++
Salient Regions	(✓)	✓		✓	✓	(✓)	+	+	++	+
Edge-based	✓			✓	✓	✓	+++	+++	+	+
MSER		✓		✓	✓	✓	+++	+++	++	+++
Intensity-based		✓		✓	✓	✓	++	++	++	++
Superpixels		✓		✓	(✓)	(✓)	+	+	+	+

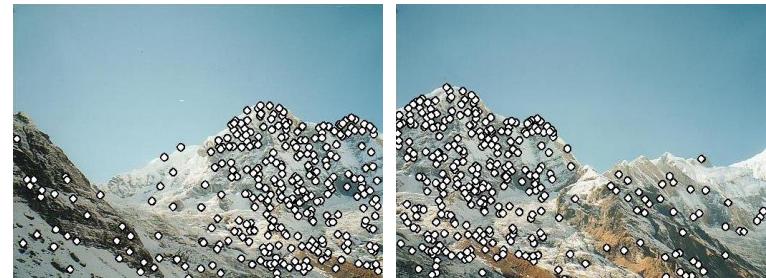
Local Image Descriptors

Read Szeliski 4.1

Local features: main components

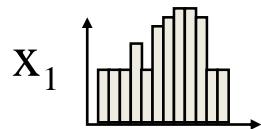
1) Detection:

Find a set of distinctive key points.

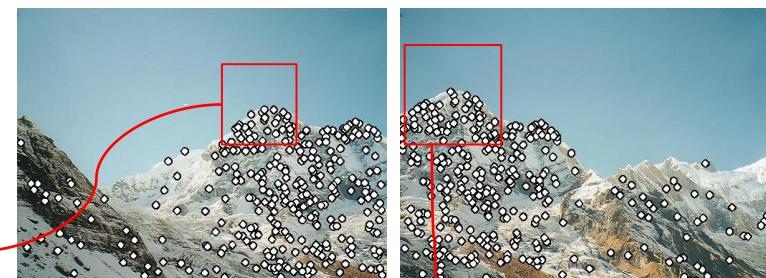


2) Description:

Extract feature descriptor around each interest point as vector.



$$\mathbf{x}_1 = [x_1^{(1)}, \dots, x_d^{(1)}]$$



3) Matching:

Compute distance between feature vectors to find correspondence.

$$d(\mathbf{x}_1, \mathbf{x}_2) < T$$

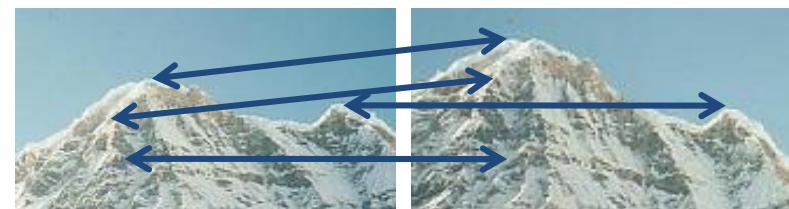
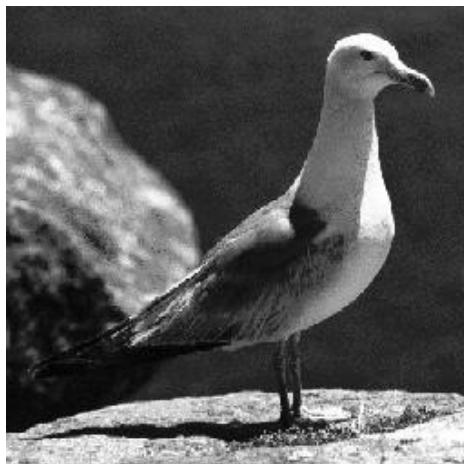
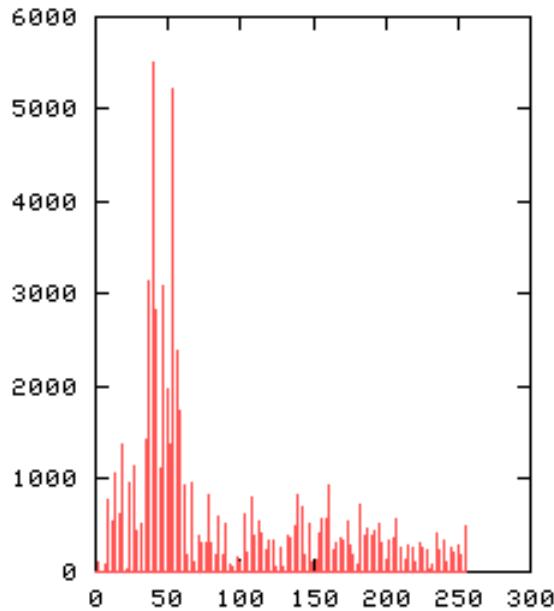


Image Representations: Histograms



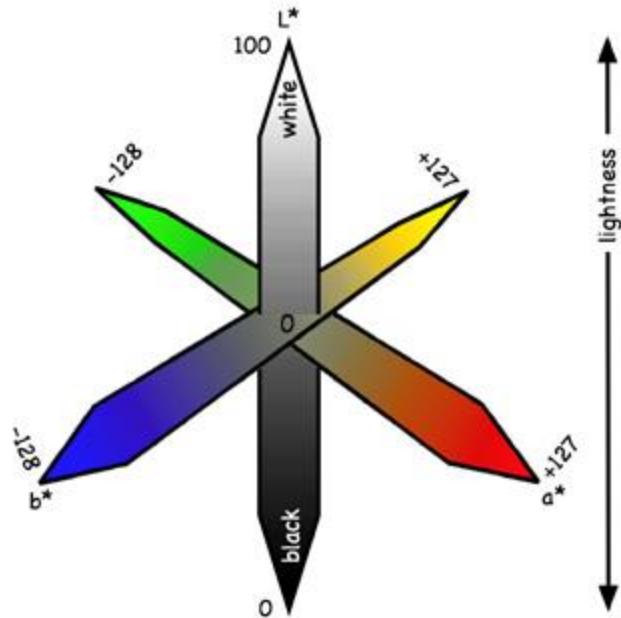
Global histogram to represent distribution of features
– Color, texture, depth, ...

Local histogram per detected point

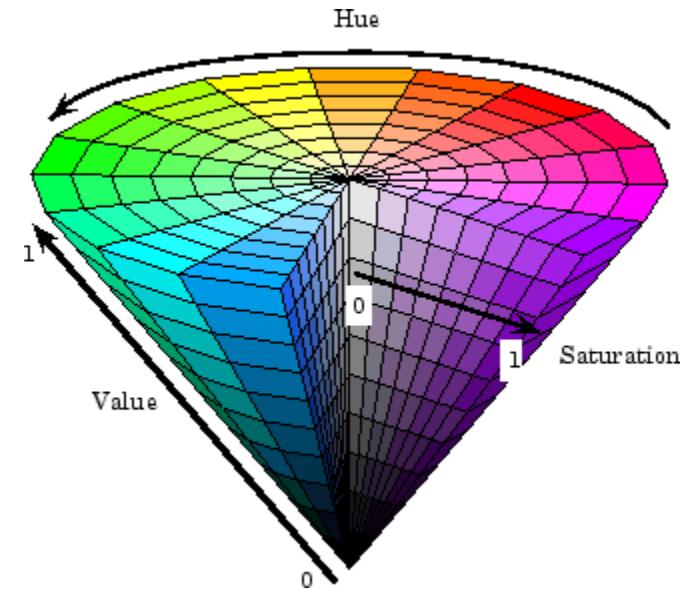


For what things might we compute histograms?

- Color



L^{*}a^{*}b^{*} color space

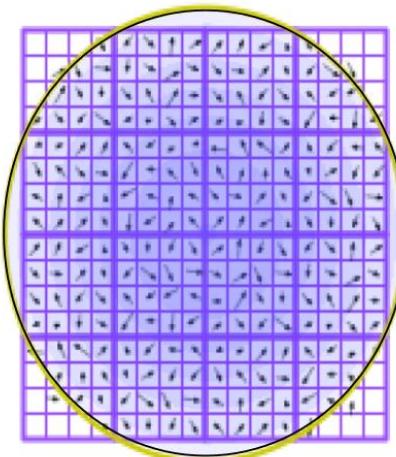
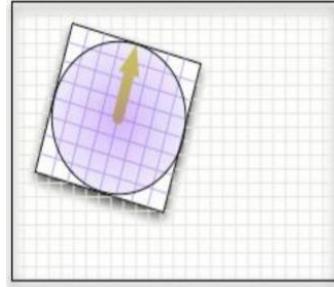
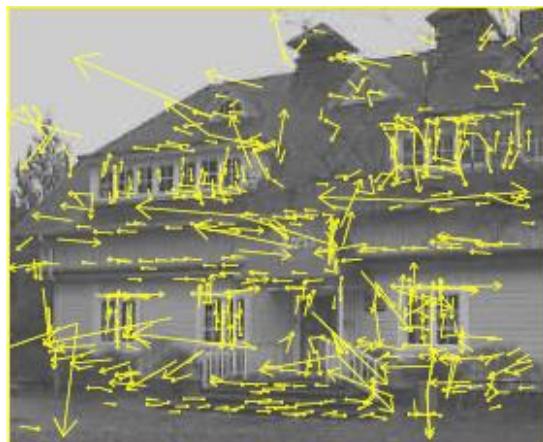


HSV color space

- Model local appearance

For what things might we compute histograms?

- Texture
- Local histograms of oriented gradients
- SIFT: Scale Invariant Feature Transform
 - Extremely popular (40k citations)



*	*	*	*
*	*	*	*
*	*	*	*
*	*	*	*

Keypoint descriptor

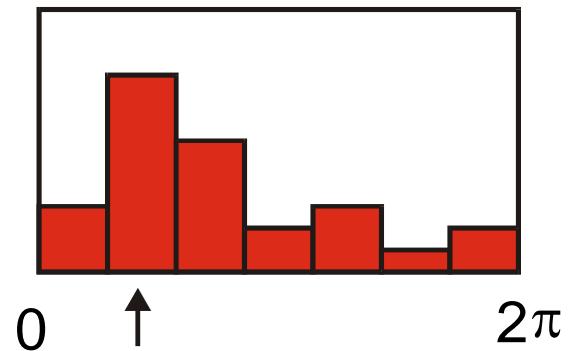
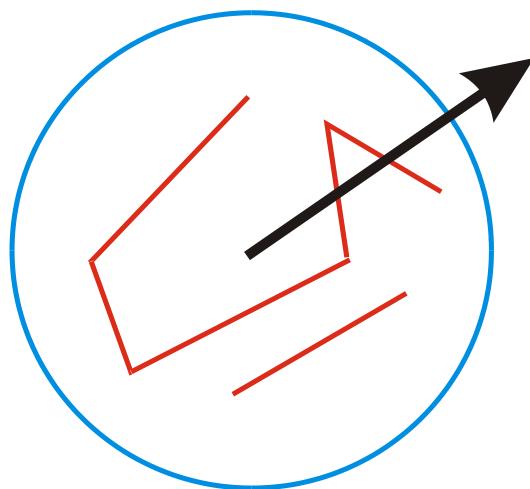
SIFT – Lowe IJCV 2004

SIFT preprocessing

- Find Difference of Gaussian scale-space extrema as feature point locations
- Post-processing
 - Subpixel position interpolation
 - Discard low-contrast points
 - Eliminate points along edges
- Orientation estimation per feature point

SIFT Orientation estimation

- Compute gradient orientation histogram
- Select dominant orientation Θ

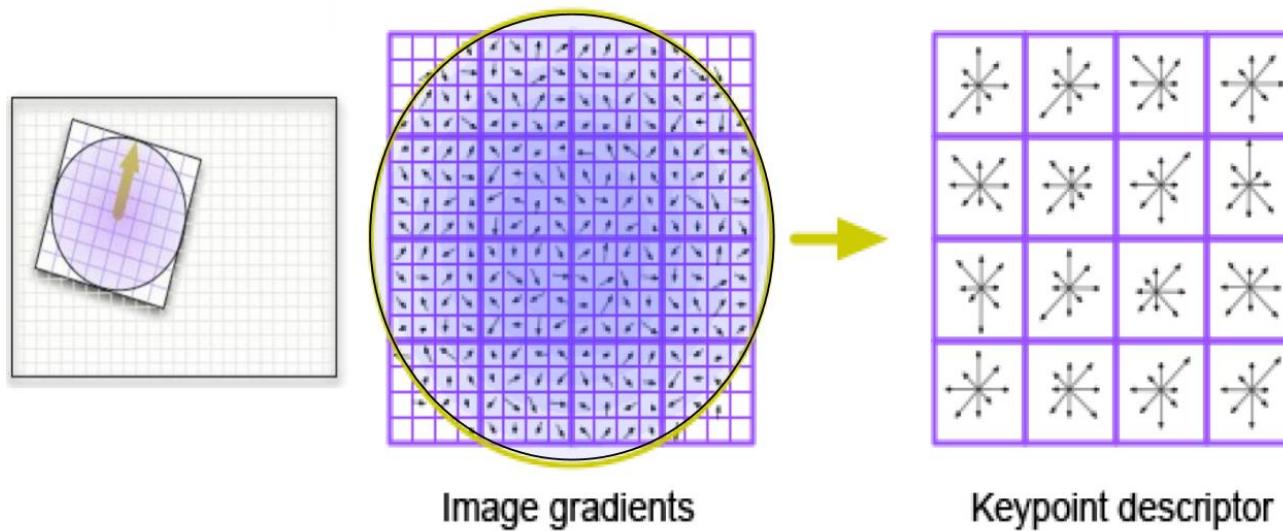


SIFT

- Find Difference of Gaussian scale-space extrema
- Post-processing
 - Position interpolation
 - Discard low-contrast points
 - Eliminate points along edges
- Orientation estimation
- Descriptor extraction
 - Motivation: We want some sensitivity to spatial layout, but not too much, so blocks of histograms give us that.

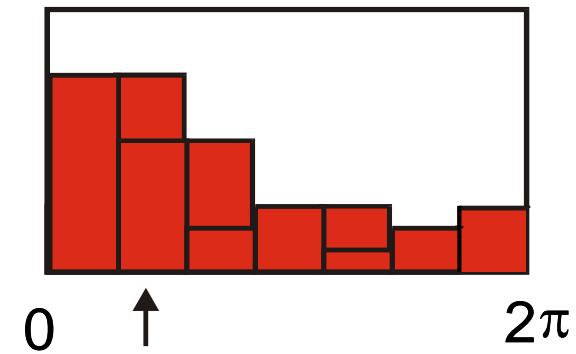
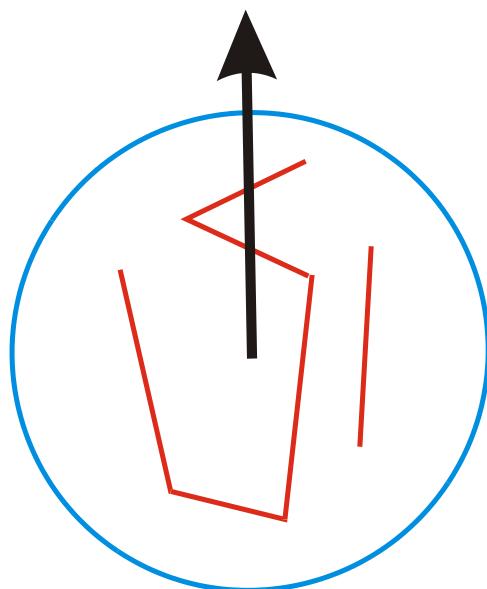
SIFT Descriptor Extraction

- Given a keypoint with scale and orientation:
 - Pick scale-space image which most closely matches estimated scale
 - Resample image to match orientation OR
 - Subtract detector orientation from vector to give invariance to general image rotation.



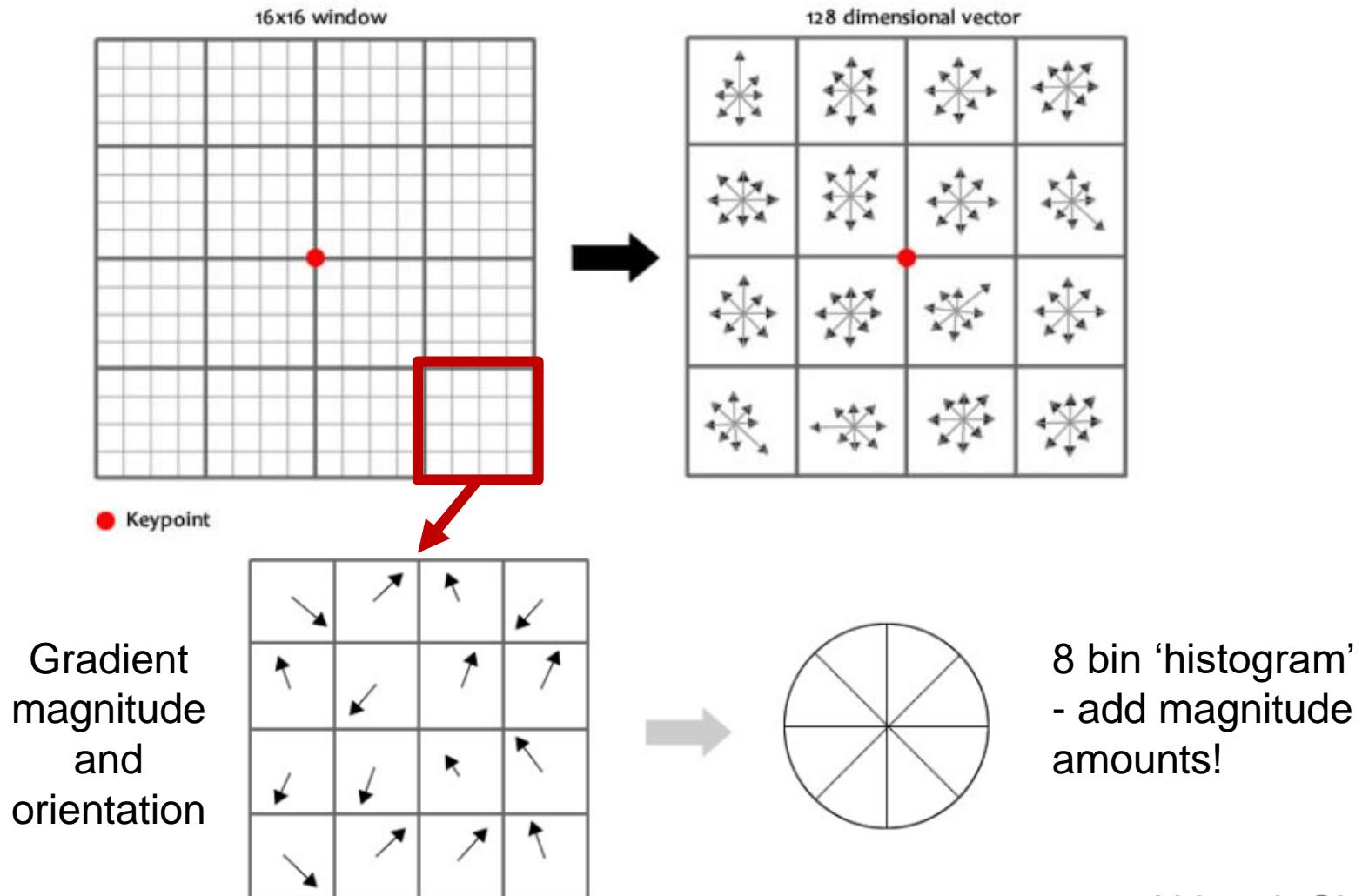
SIFT Orientation Normalization

- Compute orientation histogram
- Select dominant orientation Θ
- Normalize: rotate to fixed orientation



SIFT Descriptor Extraction

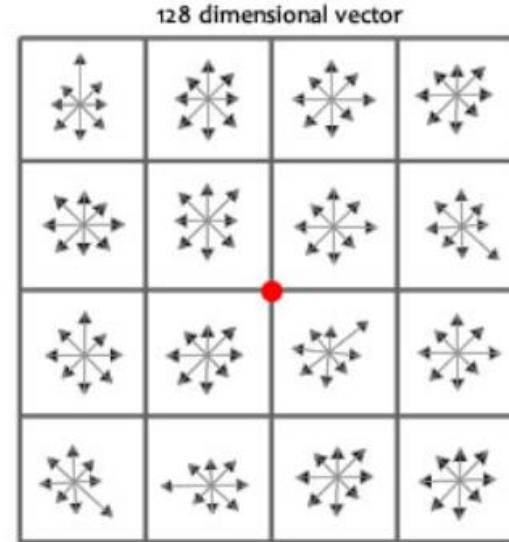
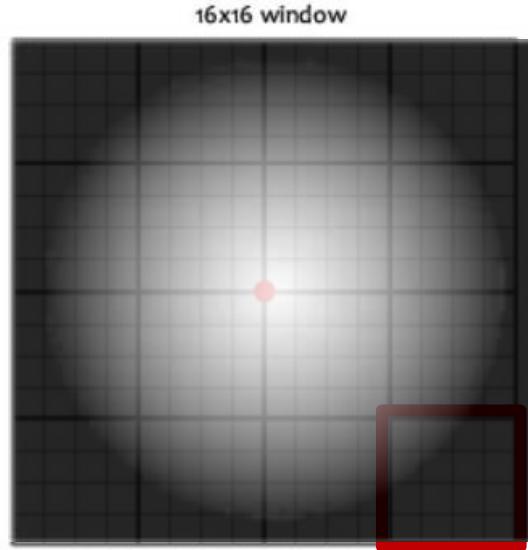
- Given a keypoint with scale and orientation



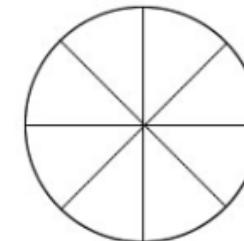
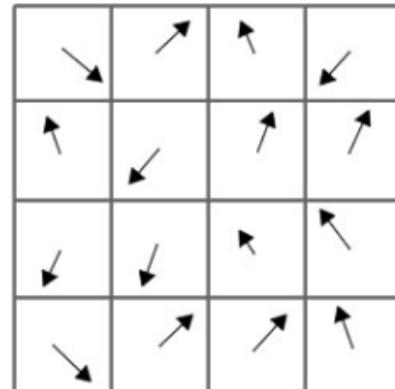
SIFT Descriptor Extraction

Weight 16x16 grid by Gaussian to add location robustness and reduce effect of outer regions

σ = half window width



Gradient magnitude and orientation



8 bin 'histogram'
- add magnitude amounts!

SIFT Descriptor Extraction

- Extract 8×16 values into 128-dim vector
- Illumination invariance:
 - Working in gradient space, so robust to $I = I + b$
 - Normalize vector to [0...1]
 - Robust to $I = \alpha I$ brightness changes
 - Clamp all vector values > 0.2 to 0.2.
 - Robust to “non-linear illumination effects”
 - Image value saturation / specular highlights
 - Renormalize

Specular highlights move between image pairs!

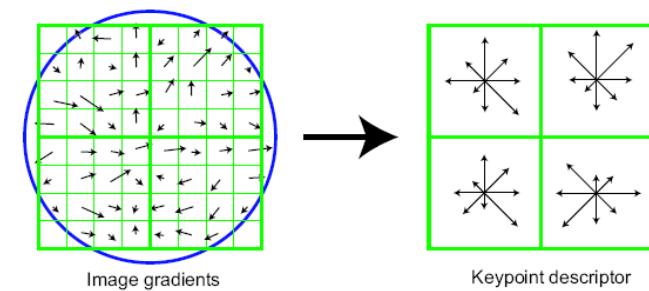


Efficient Implementation

- Filter using oriented kernels based on directions of histogram bins.
- Called ‘steerable filters’

Review: Local Descriptors

- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
 - Robust and Distinctive
 - Compact and Efficient
- Most available descriptors focus on edge/gradient information
 - Capture texture information
 - Color rarely used





The top 100 most confident local feature matches from a baseline implementation of project 2. In this case, 93 were correct (highlighted in green) and 7 were incorrect (highlighted in red).

Project 2: Local Feature Matching

CSCI 1430: Introduction to Computer Vision

Logistics

- Files: [proj2.zip](#) (6.9 MB).
- Part 1: Questions
 - Questions + template: Now in the zip: questions/
 - Hand-in process: Gradescope as PDF. Submit anonymous materials please!
- Part 2: Code
 - Writeup template: In the zip: writeup/
 - Required files: code/, writeup/writeup.pdf
 - Hand-in process: Gradescope as ZIP file. Submit anonymous materials please!

SIFT-like descriptor in Project 2

- SIFT is hand designed based on intuition
- You implement your own SIFT-like descriptor
 - Ignore scale/orientation to start.
- Parameters: stick with defaults + minor tweaks
- Feel free to look at papers / resources for inspiration

2017 Spring TA Martin Zhu recommends this tutorial:

<http://aishack.in/tutorials/sift-scale-invariant-feature-transform-features/>

- Note an error in here: Gaussian weight is applied to the larger 16x16 block and not the smaller 4x4 blocks. See Lowe's original paper, Sec. 6.1 par 2.

Lowe's original paper: <http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>

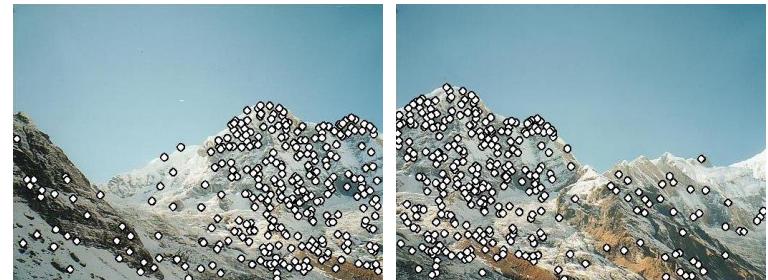
Feature Matching

Read Szeliski 4.1

Local features: main components

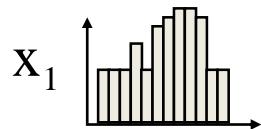
1) Detection:

Find a set of distinctive key points.

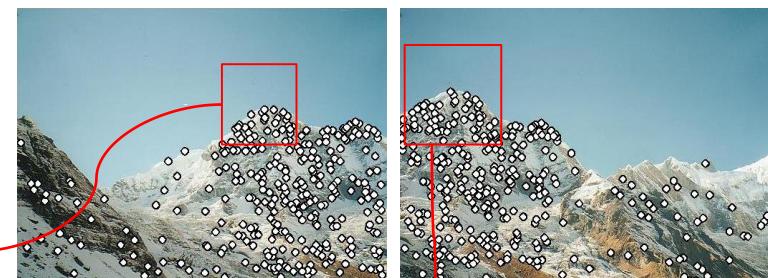


2) Description:

Extract feature descriptor around each interest point as vector.

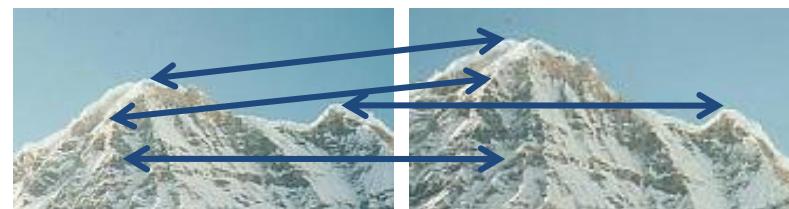


$$\mathbf{x}_1 = [x_1^{(1)}, \dots, x_d^{(1)}]$$

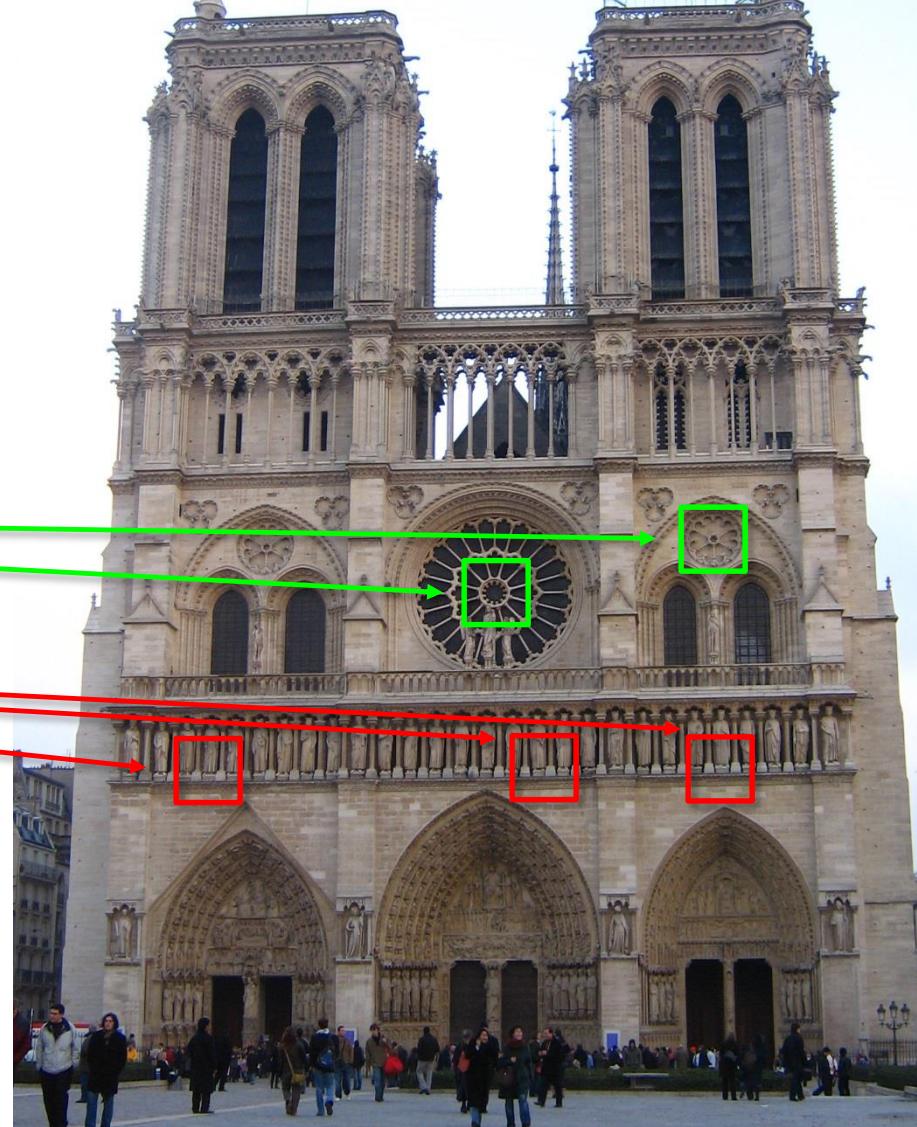
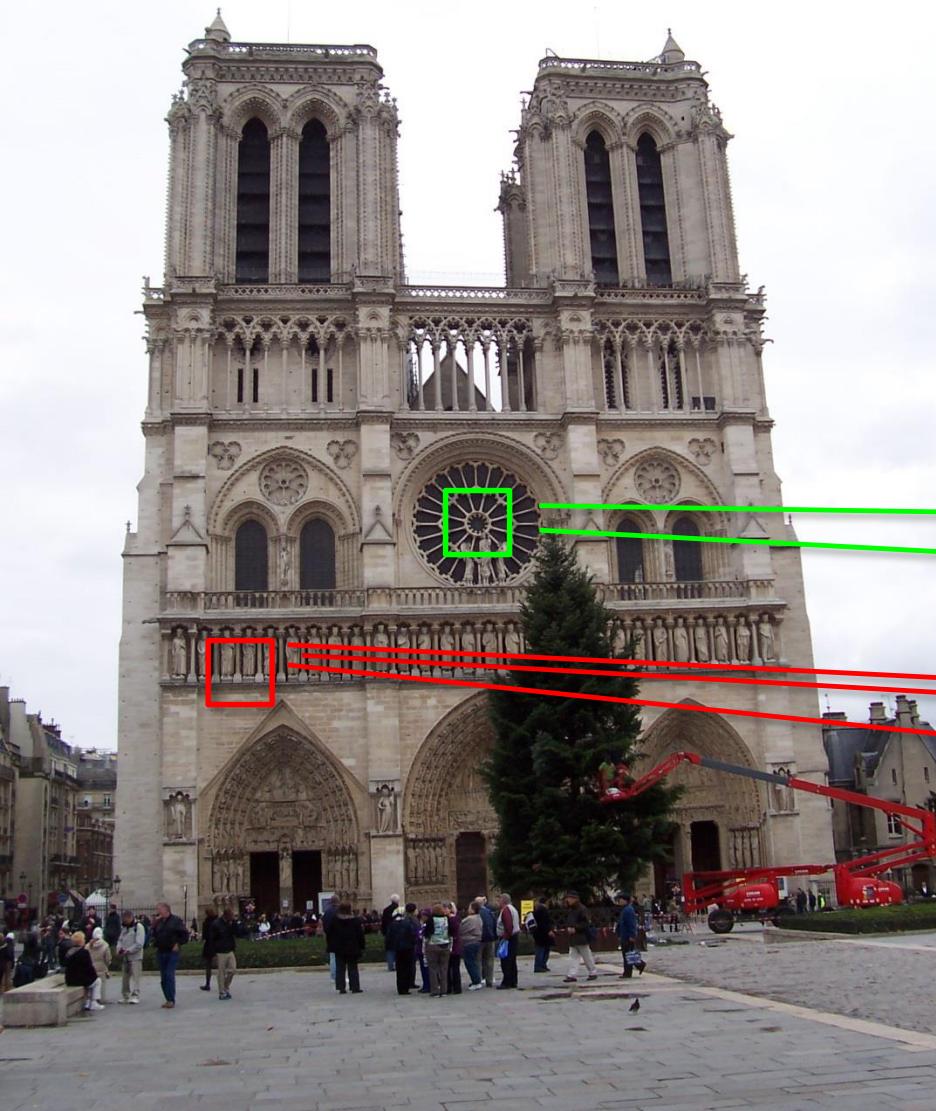


3) Matching:

Compute distance between feature vectors to find correspondence.



How do we decide which features match?



Distance: 0.34, 0.30, 0.40

Distance: 0.61, 1.22

Euclidean distance vs. Cosine Similarity

- Euclidean distance:

$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

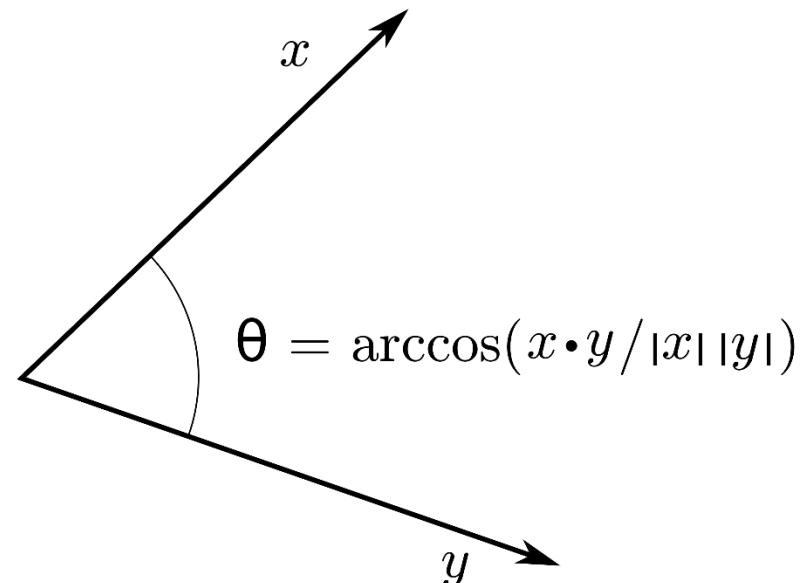
$$= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}.$$

$$\|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{q} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}.$$

- Cosine similarity:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2 \cos \theta$$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2}$$



Feature Matching

- Criteria 1:
 - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
 - Match point to lowest distance (nearest neighbor)
- Problems:
 - Does everything have a match?

Feature Matching

- Criteria 2:
 - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
 - Match point to lowest distance (nearest neighbor)
 - Ignore anything higher than threshold (no match!)
- Problems:
 - Threshold is hard to pick
 - Non-distinctive features could have lots of close matches, only one of which is correct

Nearest Neighbor Distance Ratio

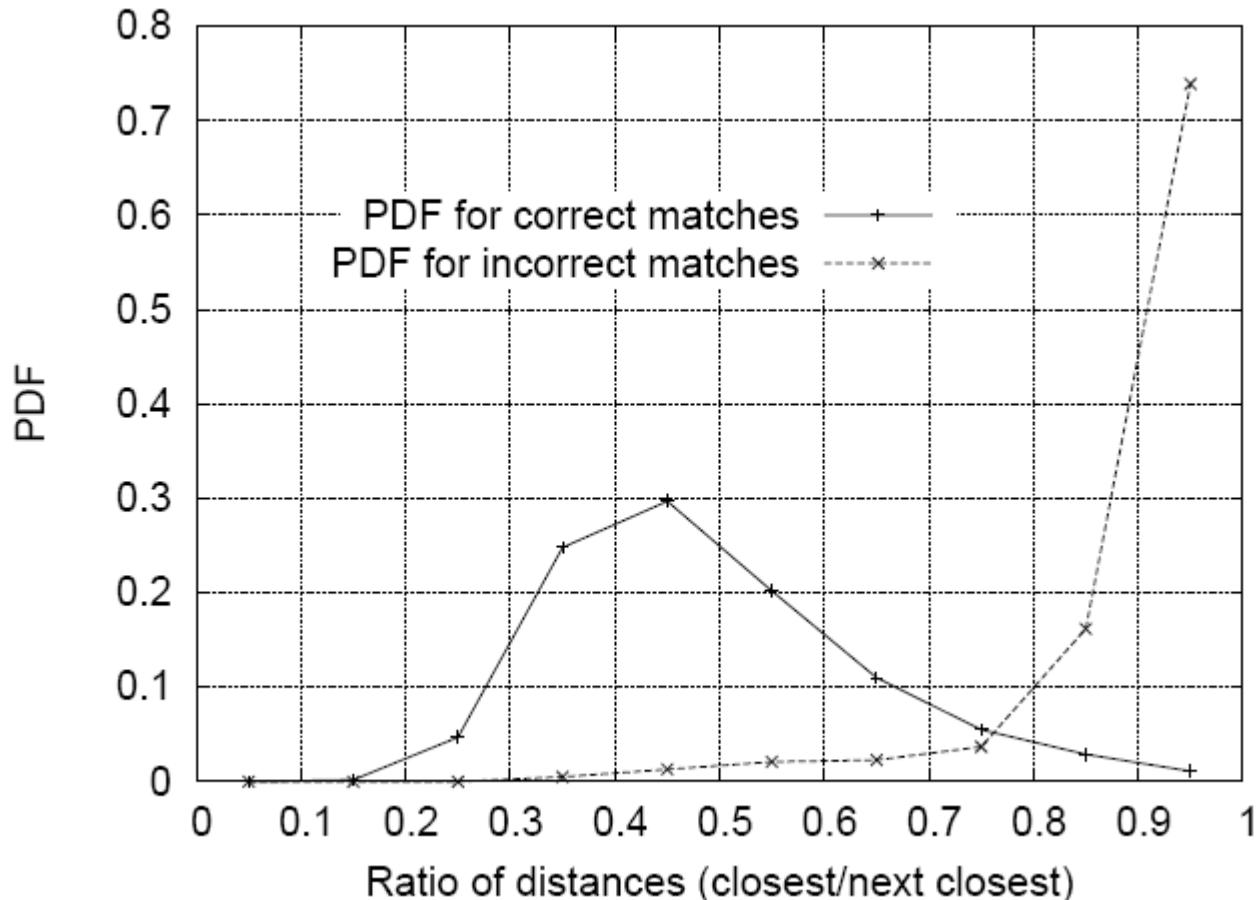
Compare distance of closest (NN1) and second-closest (NN2) feature vector neighbor.

- If $NN1 \approx NN2$, ratio $\frac{NN1}{NN2}$ will be ≈ 1 -> matches too close.
- As $NN1 \ll NN2$, ratio $\frac{NN1}{NN2}$ tends to 0.

Sorting by this ratio puts matches in order of confidence.
Threshold ratio – but how to choose?

Nearest Neighbor Distance Ratio

- Lowe computed a probability distribution functions of ratios
- 40,000 keypoints with hand-labeled ground truth



Ratio threshold depends on your application's view on the trade-off between the number of false positives and true positives!

Efficient compute cost

- Naïve looping: Expensive
- Operate on matrices of descriptors
- E.g., for row vectors,

```
features_image1 * features_image2T
```

produces matrix of dot product results
for all pairs of features (cosine similarity).
What can we do for Euclidean distance?

Live SIFT Demo

Actually a similar alternative, called SURF.

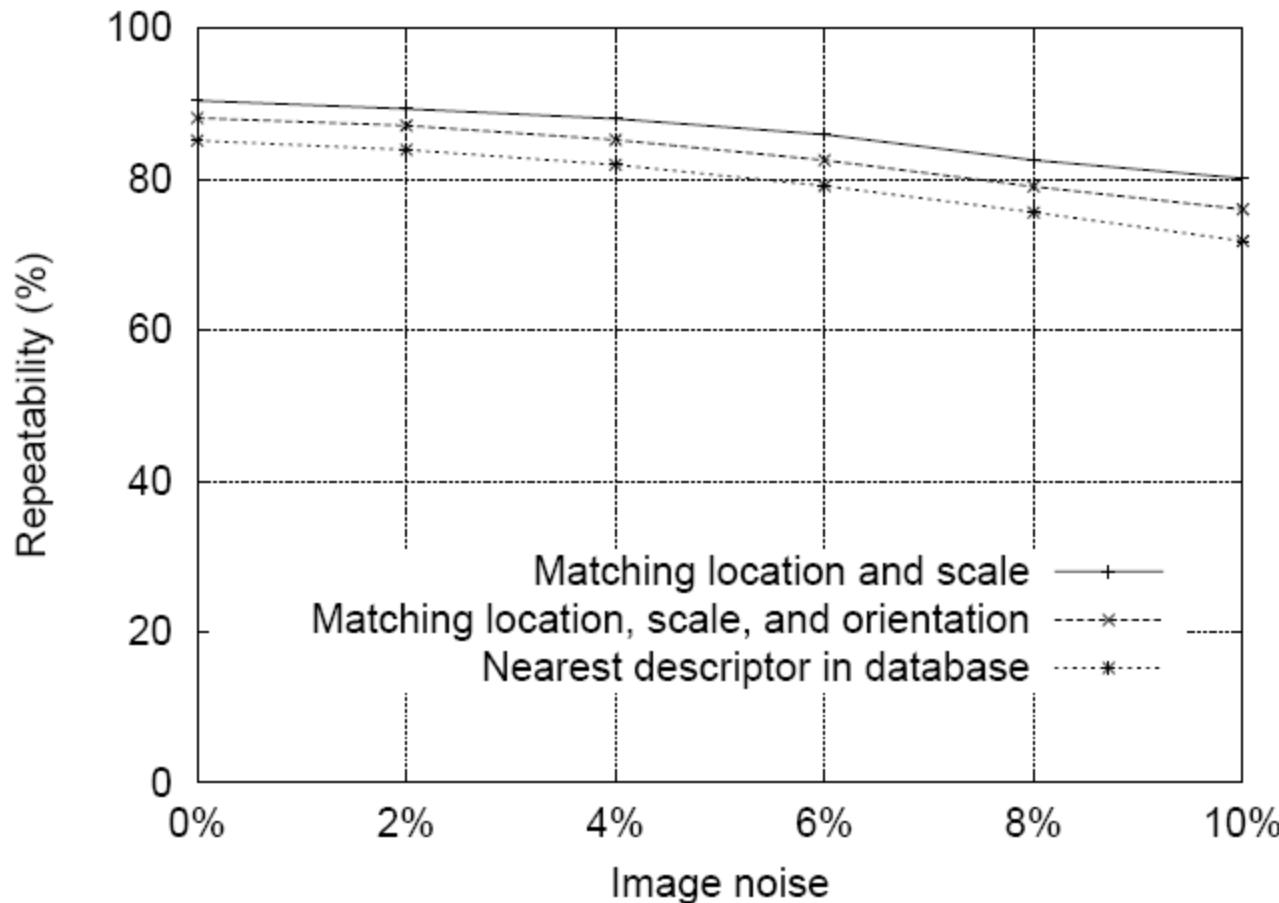
Speeded-Up Robust Features

Bay et al., ECCV 2006

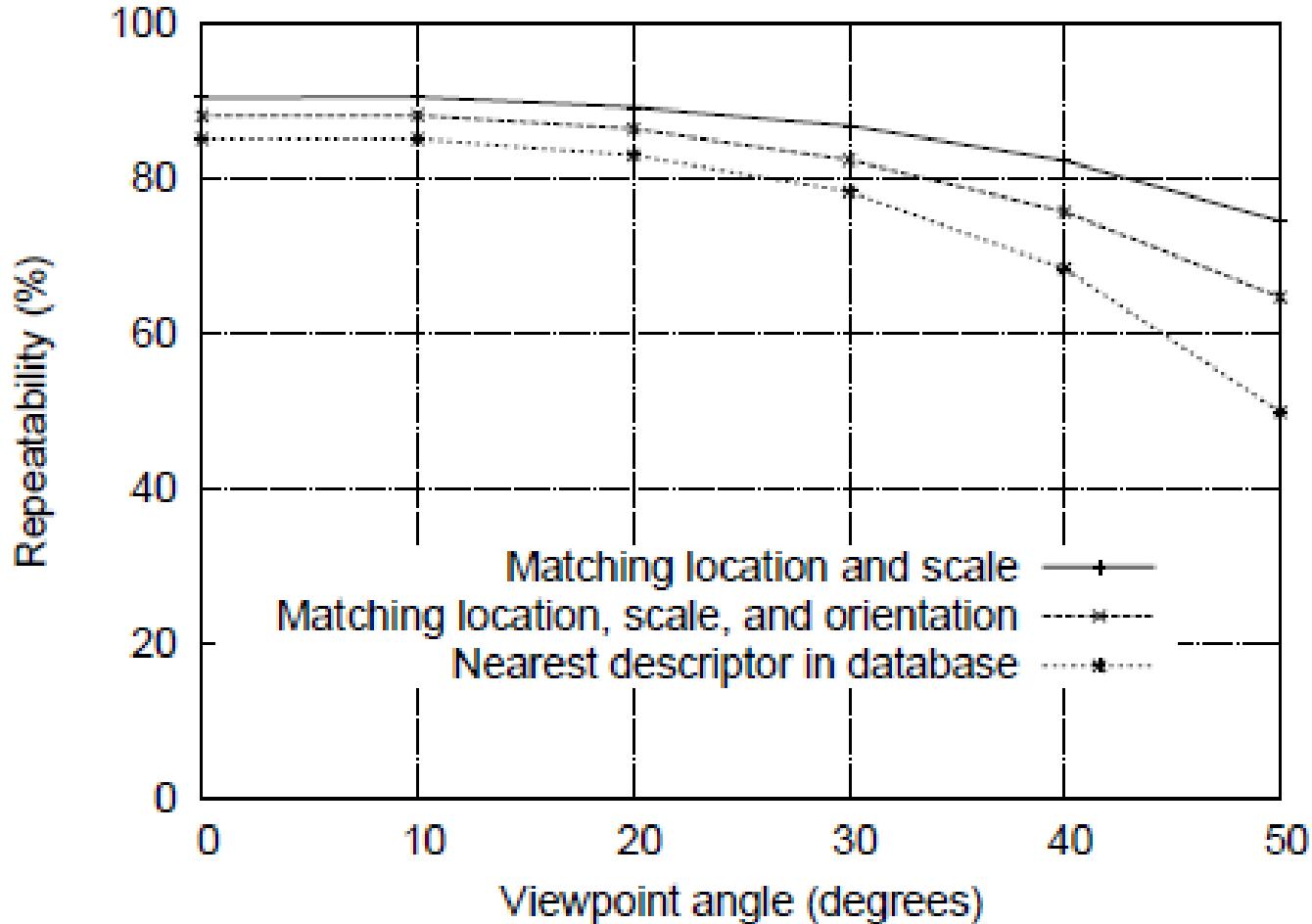
(Both are patented; SURF has non-commercial license.)

HOW GOOD IS SIFT?

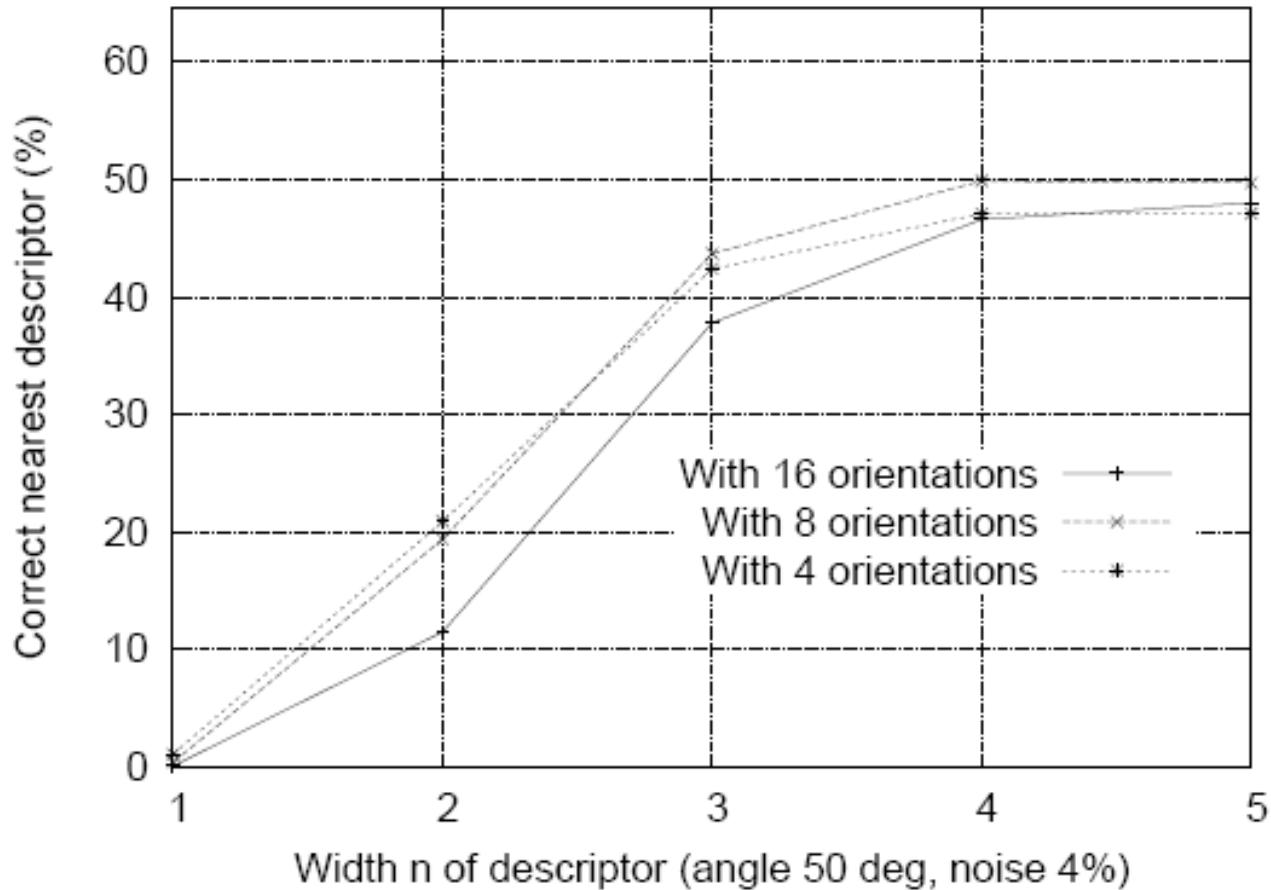
SIFT Repeatability



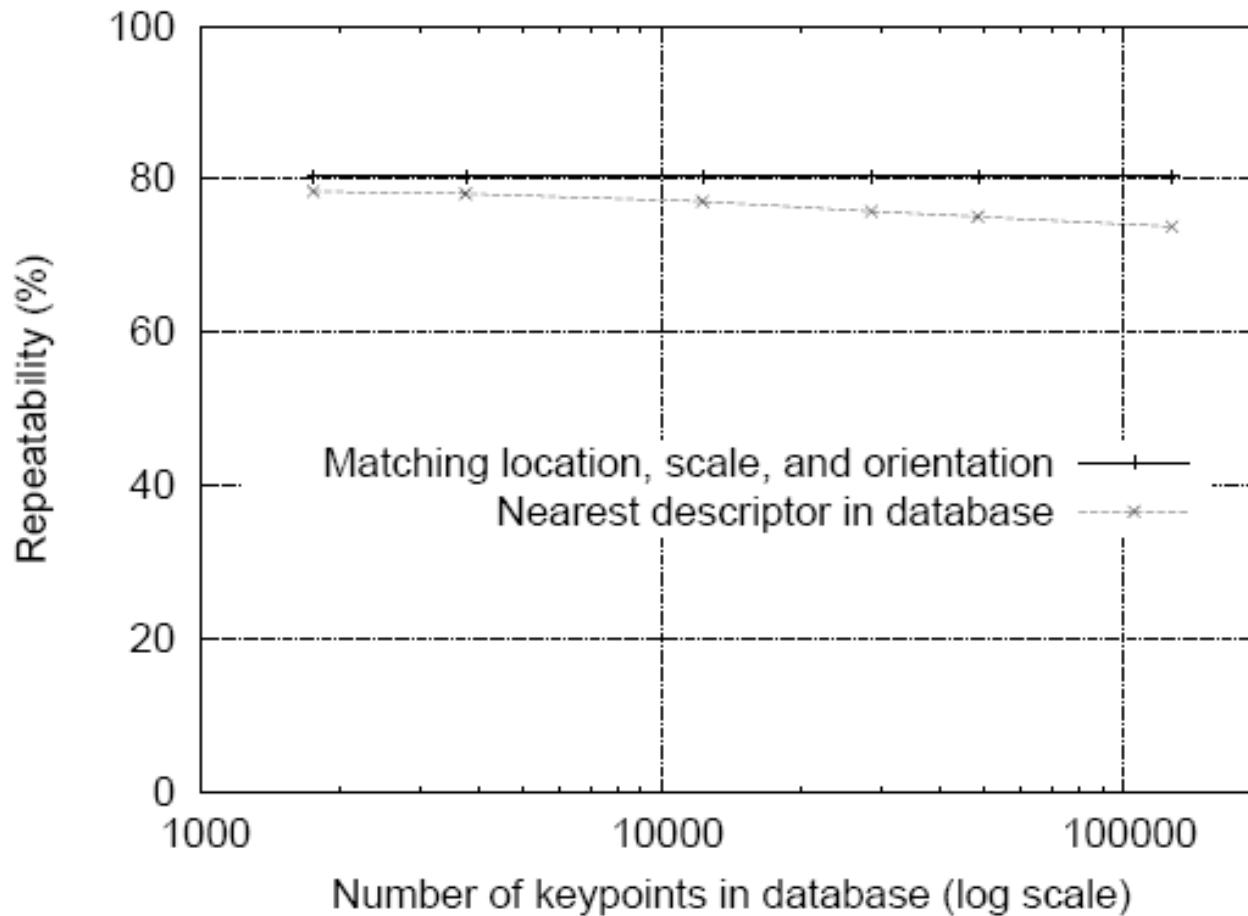
SIFT Repeatability



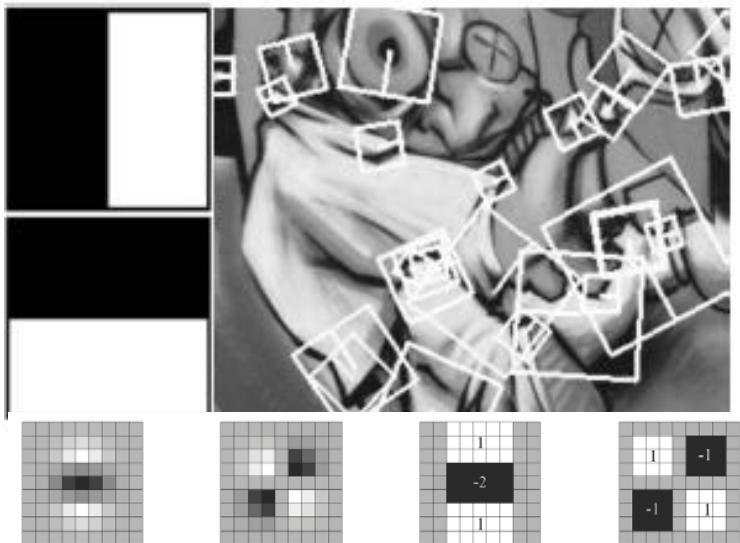
SIFT Repeatability



SIFT Repeatability



Local Descriptors: SURF



Fast approximation of SIFT idea

Efficient computation by 2D box filters & integral images

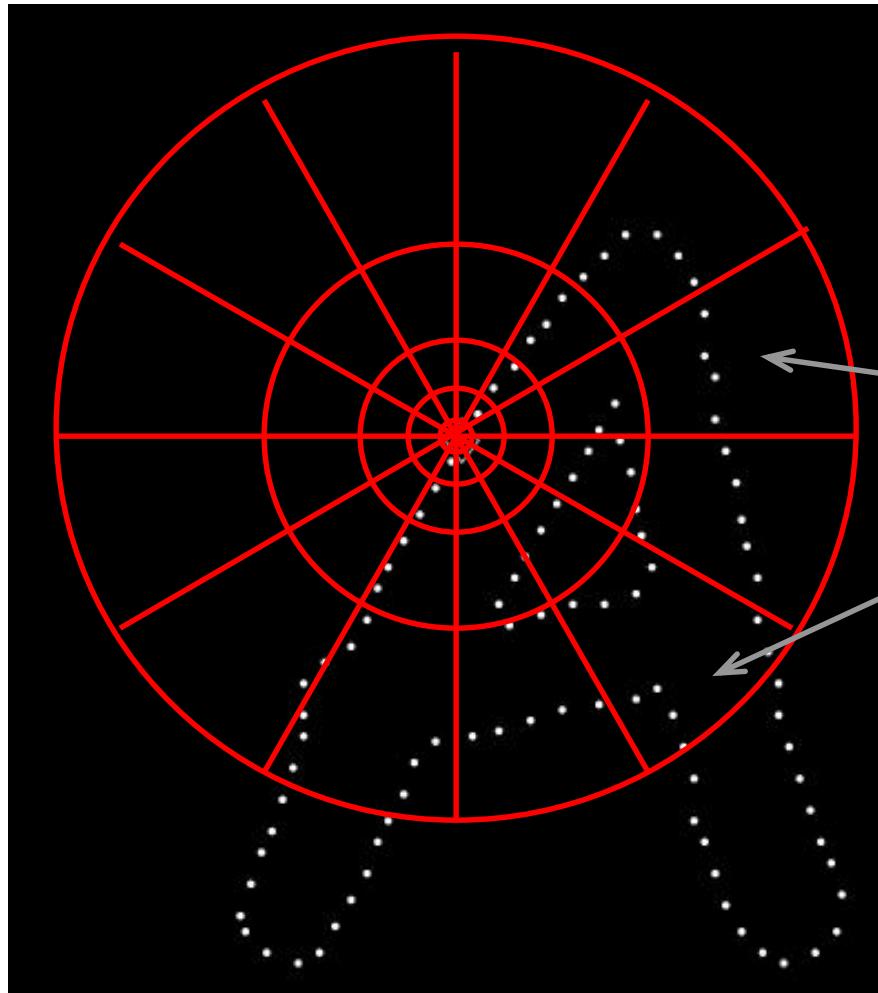
⇒ 6 times faster than SIFT

Equivalent quality for object identification

GPU implementation available

Feature extraction @ 200Hz
(detector + descriptor, 640×480 img)
<http://www.vision.ee.ethz.ch/~surf>

Local Descriptors: Shape Context



Count the number of points
inside each bin, e.g.:

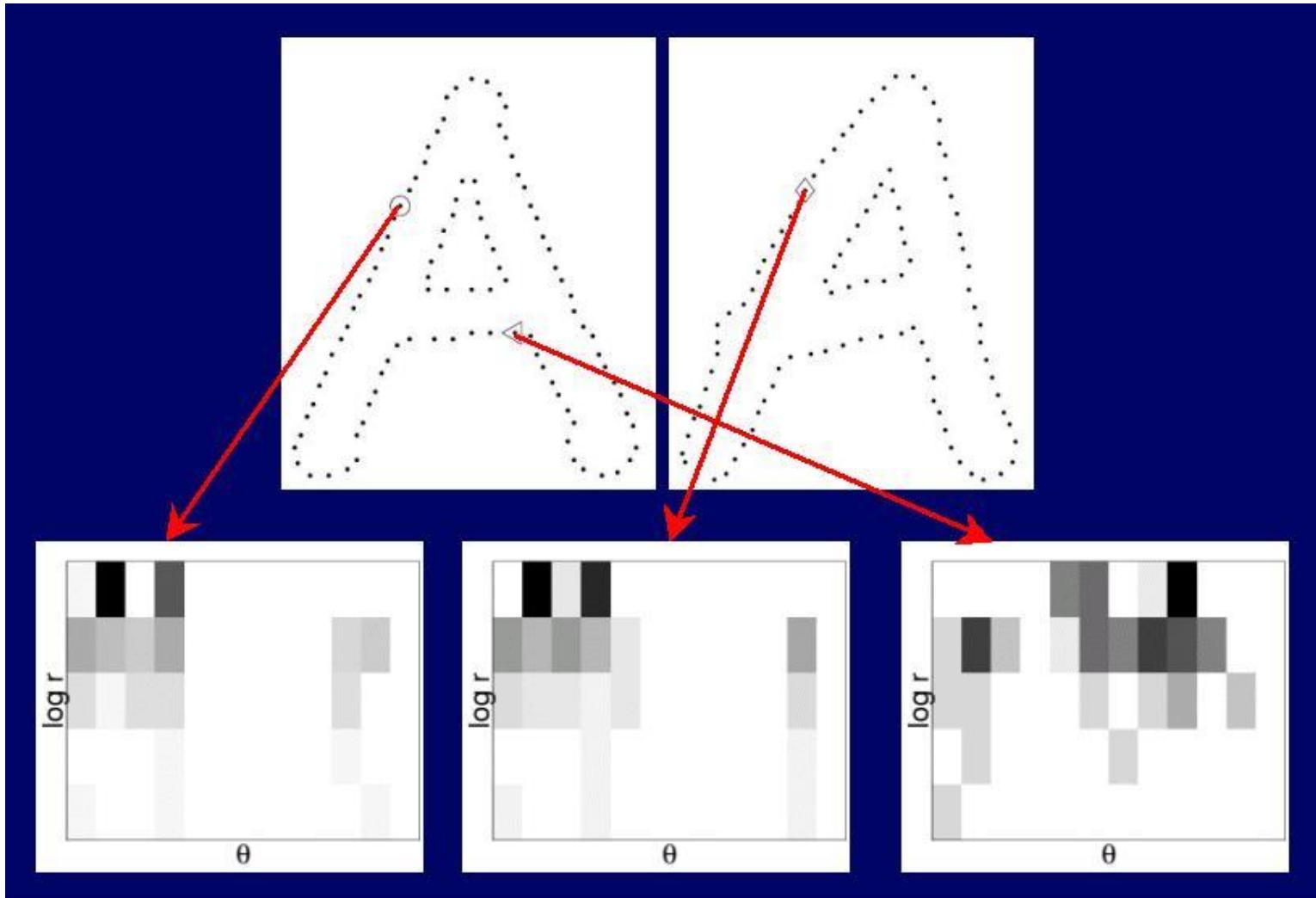
Count = 4

:

Count = 10

Log-polar binning:
More precision for nearby
points, more flexibility for
farther points.

Shape Context Descriptor



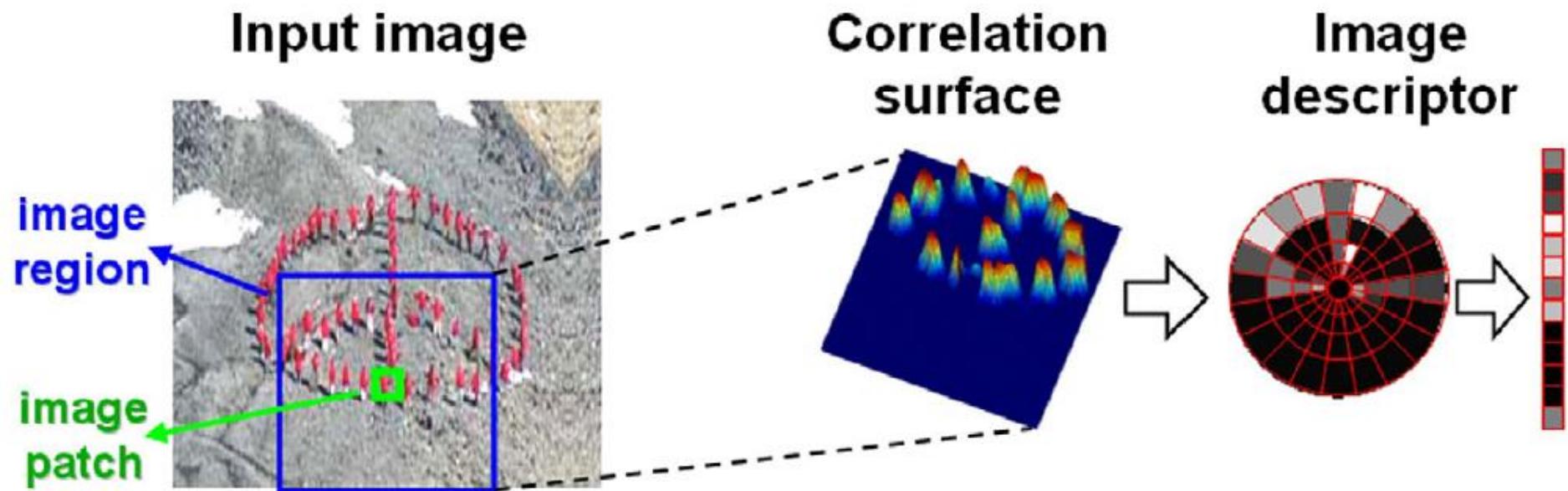
Self-similarity Descriptor



Figure 1. *These images of the same object (a heart) do NOT share common image properties (colors, textures, edges), but DO share a similar geometric layout of local internal self-similarities.*

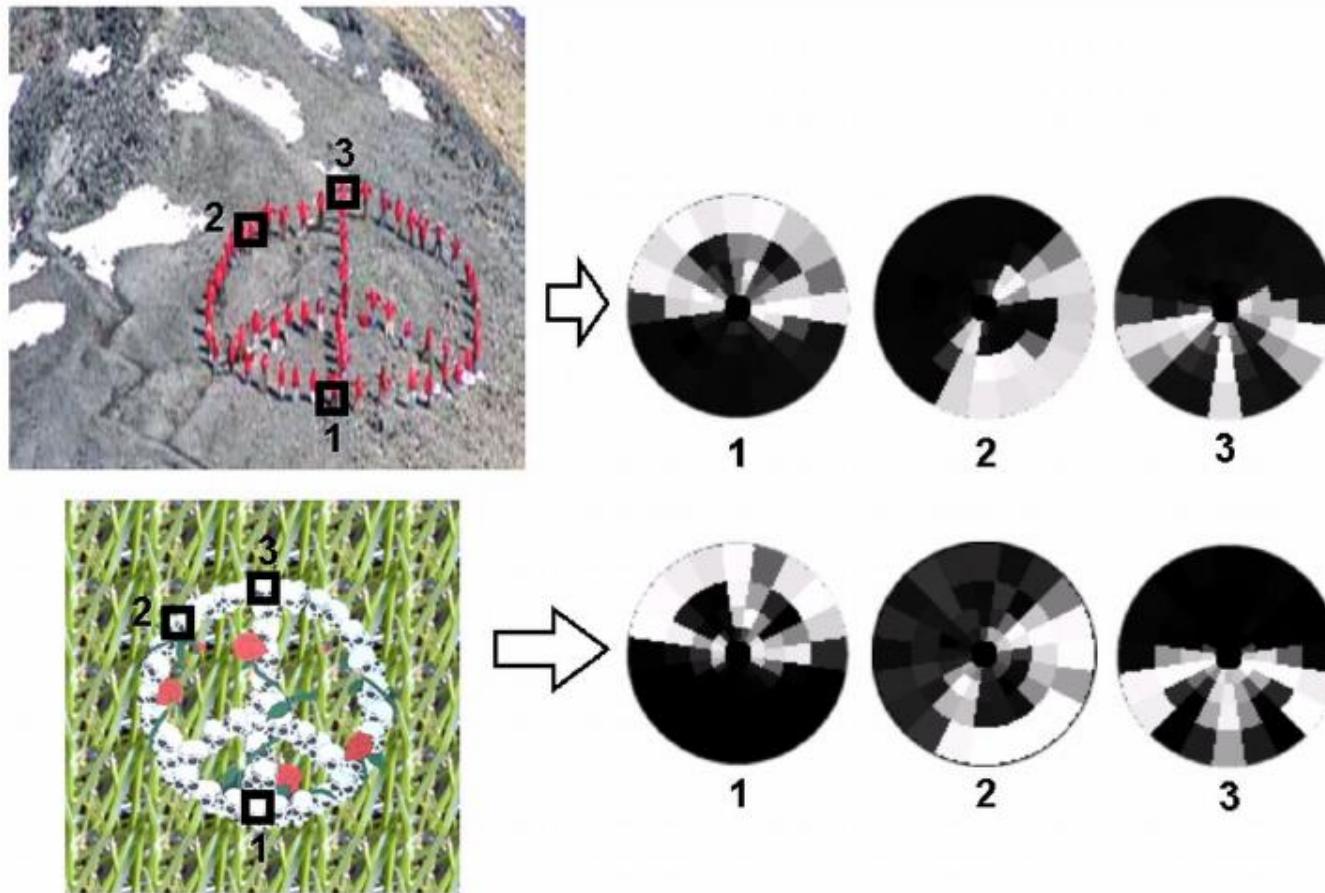
Matching Local Self-Similarities across Images
and Videos, Shechtman and Irani, 2007

Self-similarity Descriptor



Matching Local Self-Similarities across Images
and Videos, Shechtman and Irani, 2007

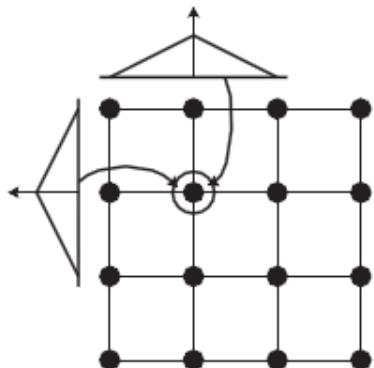
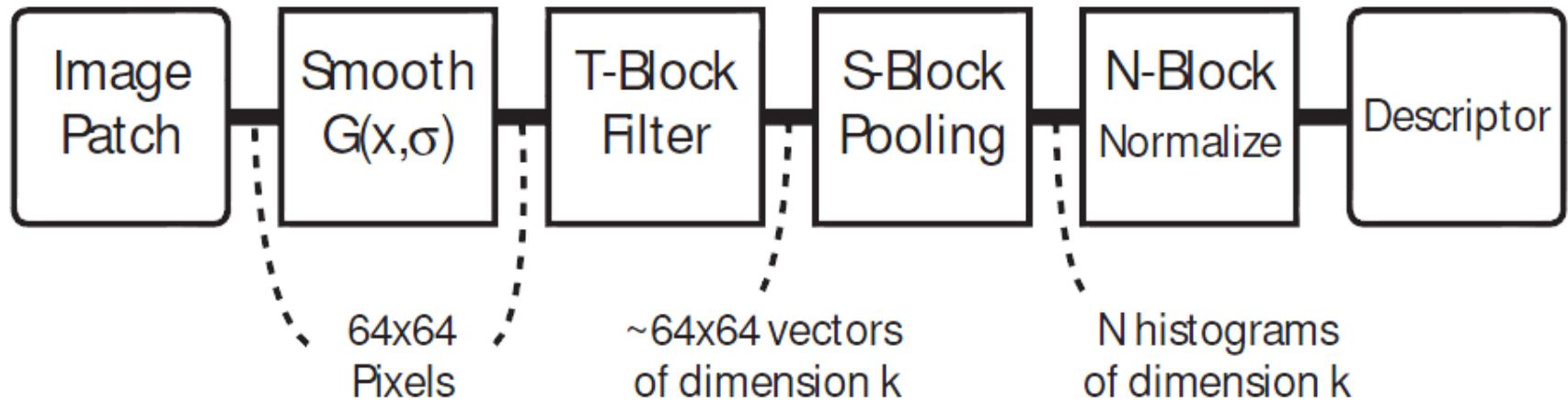
Self-similarity Descriptor



Matching Local Self-Similarities across Images
and Videos, Shechtman and Irani, 2007

Learning Local Image Descriptors

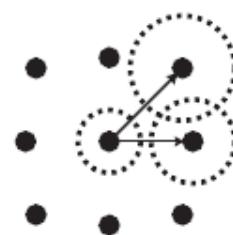
Winder and Brown, 2007



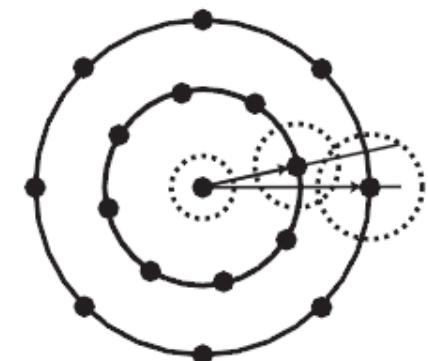
S1: SIFT grid with
bilinear weights



S2: GLOH polar grid
with bilinear radial
and angular weights



S3: 3x3 grid with
Gaussian weights



S4: 17 polar samples
with Gaussian weights

Right features are application specific

- Shape: scene-scale, object-scale, detail-scale
 - 2D form, shading, shadows, texture, linear perspective
- Material properties: albedo, feel, hardness, ...
 - Color, texture
- Motion
 - Optical flow, tracked points
- Distance
 - Stereo, position, occlusion, scene shape
 - If known object: size, other objects

Available at a web site near you...

- Many local feature detectors have executables available online:
 - <http://www.robots.ox.ac.uk/~vgg/research/affine>
 - <http://www.cs.ubc.ca/~lowe/keypoints/>
 - <http://www.vision.ee.ethz.ch/~surf>

Review: Interest points

- Keypoint detection: repeatable and distinctive
 - Corners, blobs, stable regions
 - Harris, DoG
- Descriptors: robust and selective
 - Spatial histograms of orientation
 - SIFT

