Feature Detection

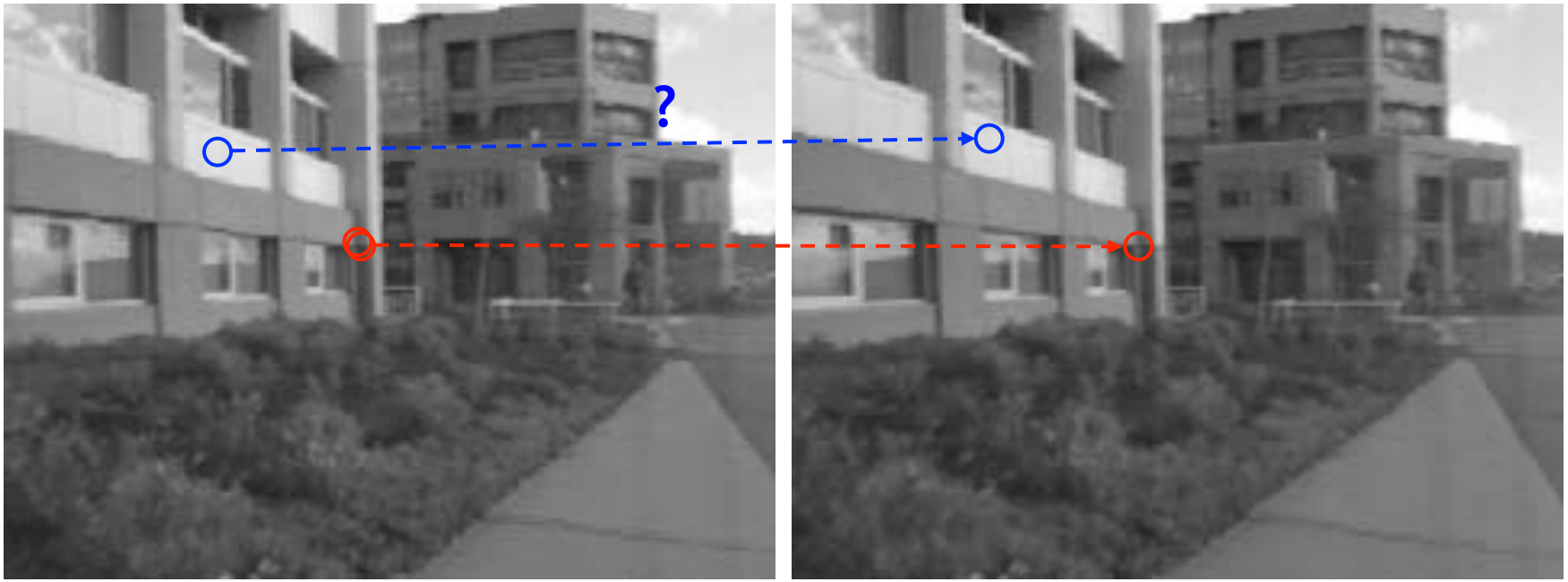
Instructor
LE Thanh Sach, Ph.D.
Ltsach@cse.hcmut.edu.vn
http://www.cse.hcmut.edu.vn/~ltsach/

Outline

- Image features
- Corner feature:
  - Moravec
  - Harris
- Scale Invariant Feature Transform (**SIFT**)
- Speed Up Robust Features (**SURF**)
- Gradient Location Orientation Histogram (**GLOH**)
- Multi-Scale Oriented Patches (**MSOP**)
- Feature Matching

Feature: Concept

- Feature can be defined as the information which is extracted (**detect** + **describe**) from images and is aimed to be used for further tasks in computer vision, for examples, classification, matching, and tracking.



Feature: Assessment Criteria

- **Distinctive**: a single feature can be correctly matched with high probability.
- **Invariant**: invariant to scale, rotation, affine, illumination and noise for robust matching across a substantial range of affine distortion, viewpoint change and so on.

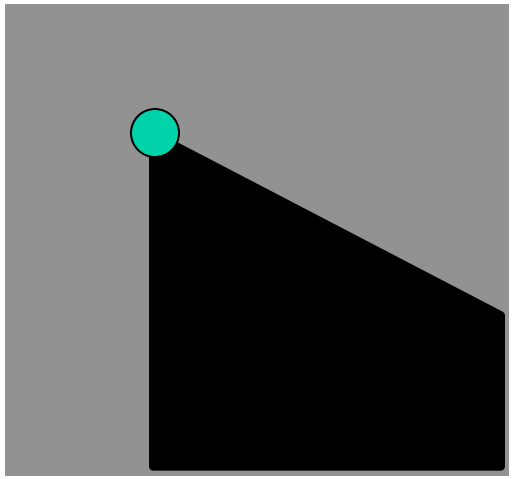
Feature: Tasks

- **Feature detection** locates where they are
- **Feature description** describes what they are
- **Feature matching** decides whether two are the same one

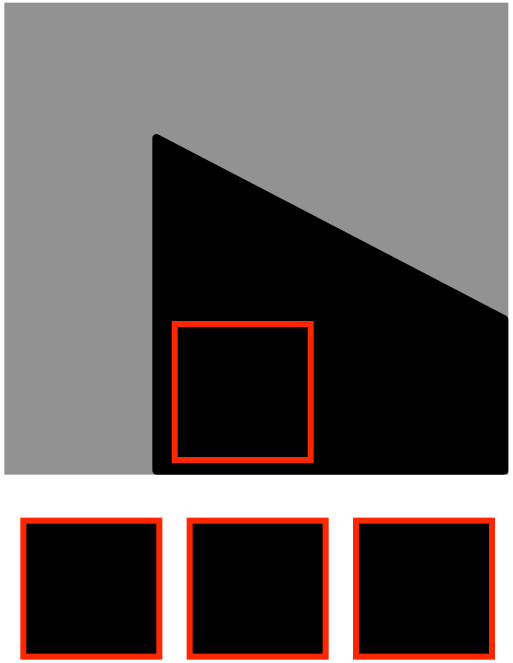
Corner detector

Moravec corner detector (1980)

- Corner point:
  - Be easily recognized by looking through a small window
  - Shift a window in *any direction* should give *a large change* in intensity

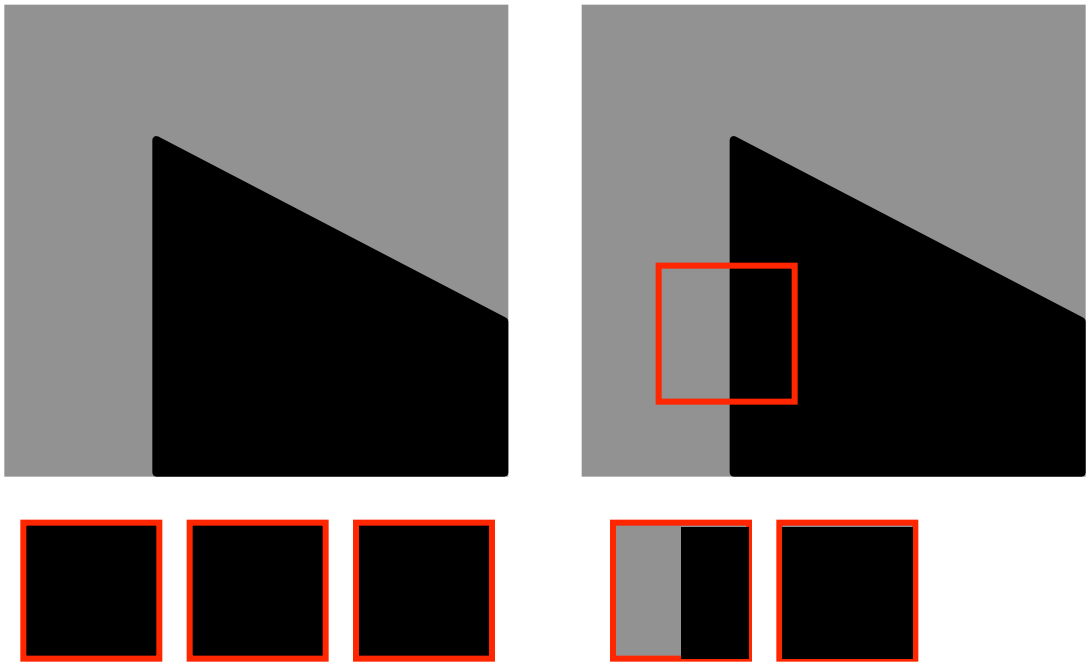


Moravec corner detector



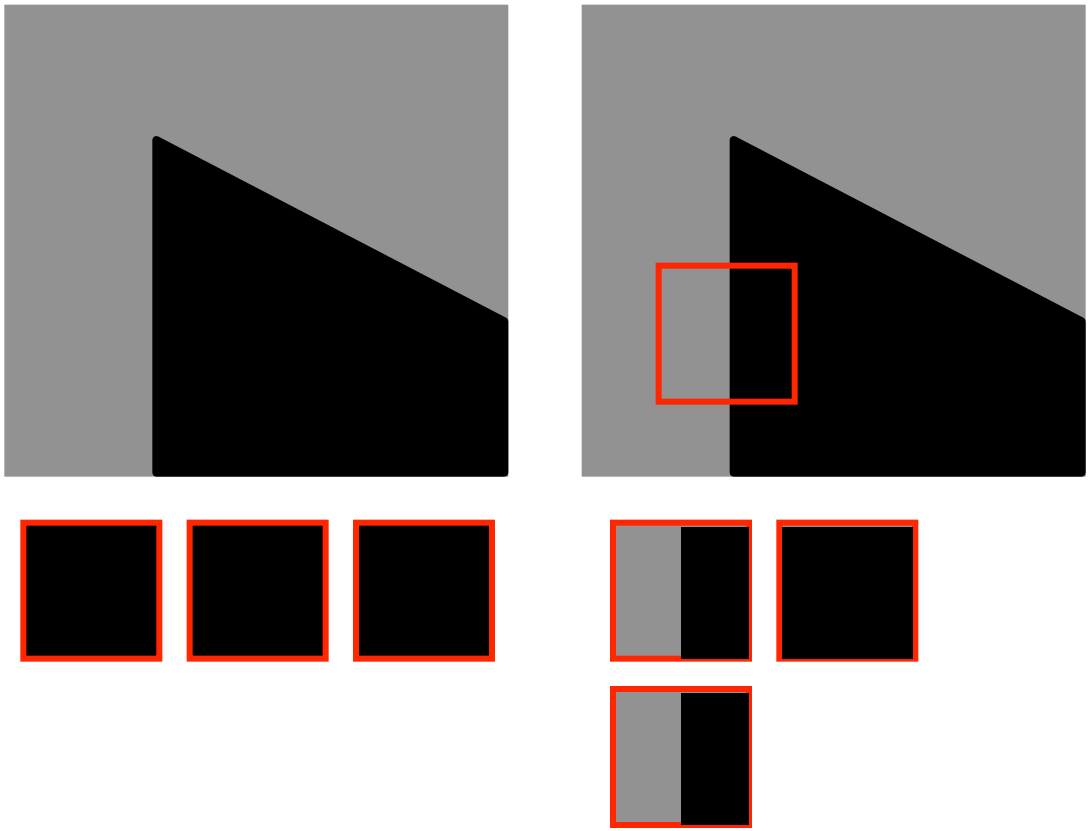
flat

Moravec corner detector



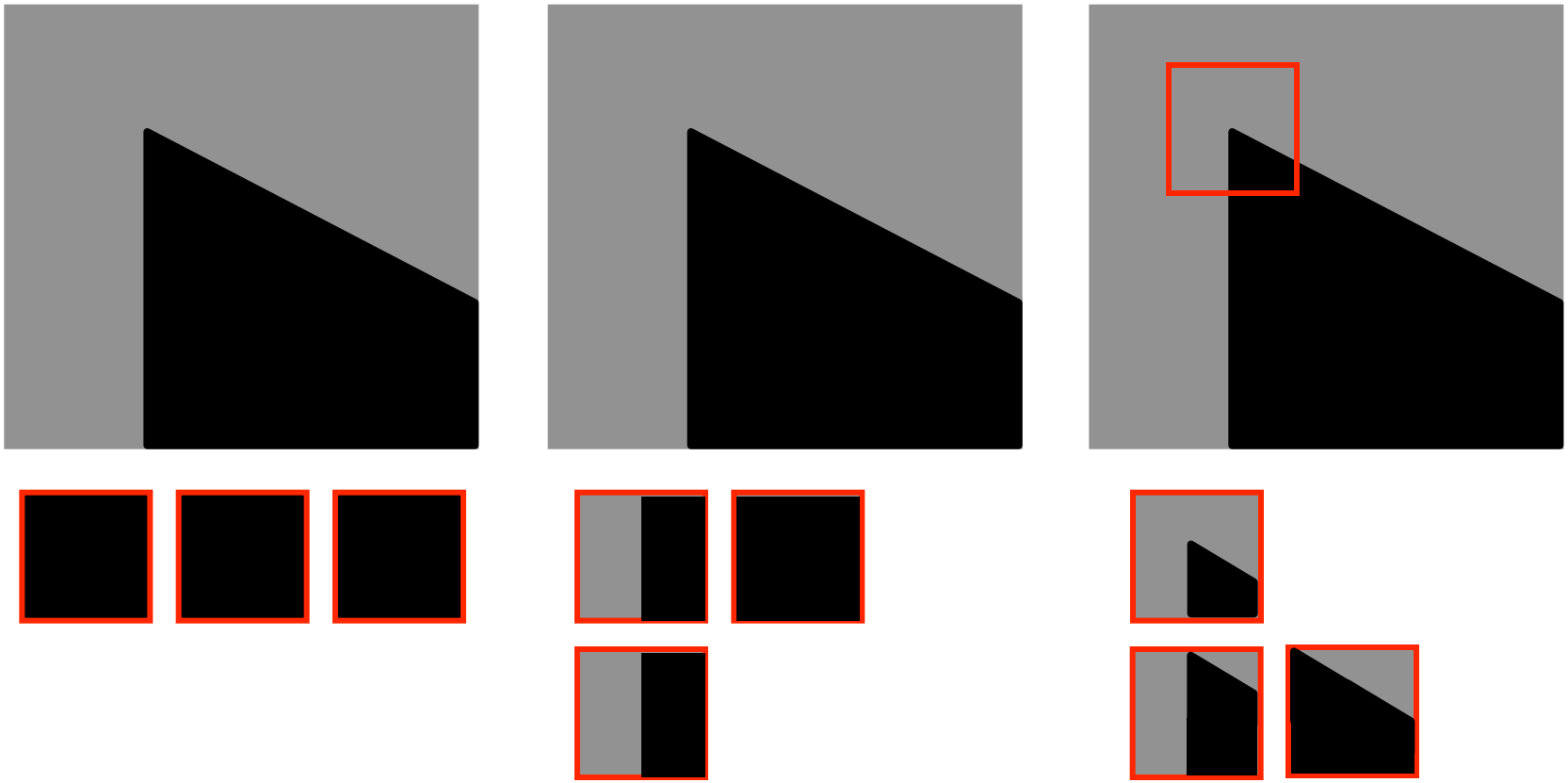
flat

Moravec corner detector



flat edge

Moravec corner detector



flat

edge

corner
isolated point

Moravec corner detector

Change of intensity for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

window function

shifted intensity

intensity

Window function $w(x,y)$ =

1 in window, 0 outside

Moravec corner detector

Change of intensity for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

window function

shifted intensity

intensity

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)

➔Look for local maxima in $\equiv min\{E(u,v)\}$

Problems of Moravec detector

- (Problem 1): Noisy response
  - due to a binary window function
- (Problem 2): Only a set of shifts at every 45 degree is considered
- (Problem 3): Only minimum of E is taken into account

⇨Harris corner detector (1988) solves these problems.

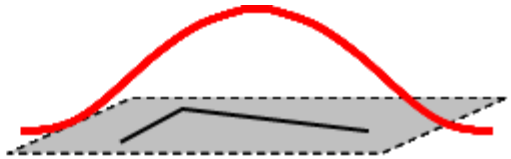
Harris corner detector

(Problem 1): Noisy response due to a binary window function

➔Use a Gaussian function

$$w(x,y) = \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

Window function $w(x,y)$ =



Gaussian

Harris corner detector

(Problem 2): Only a set of shifts at every 45 degree is considered

➔ Consider all small shifts by Taylor's expansion

Harris corner detector

Taylor expansion:

$$f(x+u,y+v) = f(x,y) + uf_x(x,y) + vf_y(x,y) + \quad \text{(1st order)}$$

$$\frac{1}{2!}\left[u^2 f_{xx}(x,y) + uv f_{xy}x,y + v^2 f_{yy}(x,y)\right] + \quad \text{(2nd order)}$$

$$\frac{1}{3!}\left[u^3 f_{xxx}(x,y) + u^2 v f_{xxy}(x,y) + uv^2 f_{xyy}(x,y) + v^3 f_{yyy}(x,y)\right] \quad \text{(3rd order)}$$

$+ \ldots$ (higher order)

Harris corner detector

1st order approximation:

$$f(x+u, y+v) \approx f(x,y) + u f_x(x,y) + v f_y(x,y)$$

Harris corner detector

$$\sum [I(x+u, y+v) - I(x,y)]^2$$

$$\approx \sum [I(x,y) + uI_x + vI_y - I(x,y)]^2 \qquad (1^{st} \text{ approximation})$$

$$= \sum u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2$$

$$= \sum \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \qquad \text{(matrix form)}$$

$$= \begin{bmatrix} u & v \end{bmatrix} \left(\sum \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}$$

Harris corner detector

Equivalently, for small shifts [*u*,*v*] we have a *bilinear* approximation:

$$E(u,v) \cong \begin{bmatrix} u & v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

Where $\mathbf{M}$ is a 2×2 matrix computed from image derivatives:

$$\mathbf{M} = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Harris corner detector (matrix form)

$$E(\mathbf{u}) = \sum_{\mathbf{x}_0 \in W(\mathbf{p})} w(\mathbf{x}_0) \left| I(\mathbf{x}_0 + \mathbf{u}) - I(\mathbf{x}_0) \right|^2$$

$$\left| I(\mathbf{x}_0 + \mathbf{u}) - I(\mathbf{x}_0) \right|^2$$

$$= \left| \left(I_0 + \frac{\partial I}{\partial \mathbf{x}}^T \mathbf{u} \right) - I_0 \right|^2$$

$$= \left| \frac{\partial I}{\partial \mathbf{x}}^T \mathbf{u} \right|^2$$

$$= \mathbf{u}^T \frac{\partial I}{\partial \mathbf{x}} \frac{\partial I}{\partial \mathbf{x}}^T \mathbf{u}$$

$$= \mathbf{u}^T \mathbf{M} \mathbf{u}$$

***u*=(■*u*@*v*)**

Harris corner detector

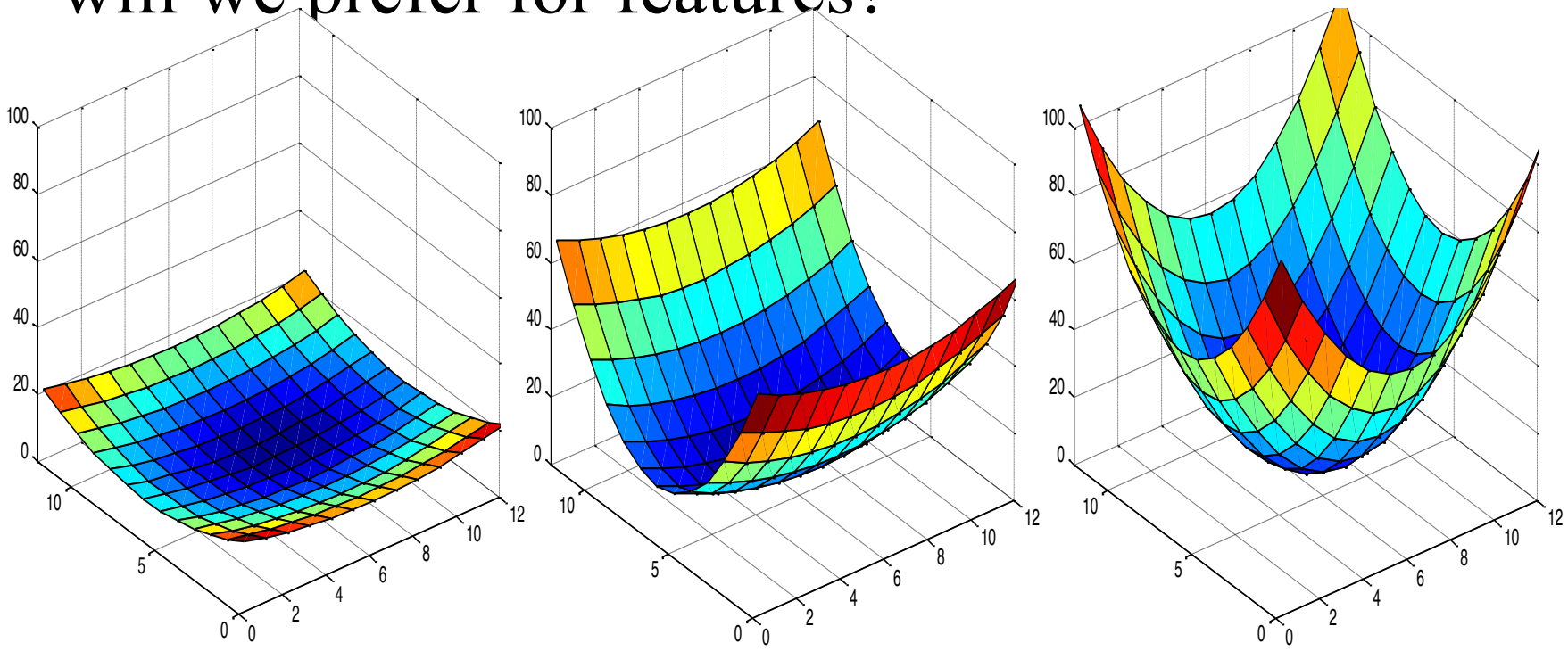
Only minimum of E is taken into account

➢A new corner measurement by investigating the shape of the error function

$\mathbf{u}^T\mathbf{M}\mathbf{u}$ represents a **quadratic function**; Thus, we can analyze E's shape by looking at the property of $\mathbf{M}$

Harris corner detector

High-level idea: what shape of the error function will we prefer for features?



flat edge corner

Quadratic forms

- Quadratic form (homogeneous polynomial of degree two) of n variables x_i

$$\sum_{\substack{i=1 \\ i \le j}}^{n} \sum_{j=1}^{n} c_{ij} x_i x_j$$

- $4x_1^2 + 5x_2^2 + 3x_3^2 + 2x_1x_2 + 4x_1x_3 + 6x_2x_3$

$$= \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} 4 & 1 & 2 \\ 1 & 5 & 3 \\ 2 & 3 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Symmetric matrices

- Quadratic forms can be represented by a real symmetric matrix **A** where

$$a_{ij} = \begin{cases} c_{ij} & \text{if } i = j, \\ \frac{1}{2}c_{ij} & \text{if } i < j, \\ \frac{1}{2}c_{ji} & \text{if } i > j. \end{cases}$$

Symmetric matrices

- Quadratic forms can be represented by a real symmetric matrix **A** where

$$\sum_{\substack{i=1 \\ i \le j}}^{n} \sum_{j=1}^{n} c_{ij} x_i x_j = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j$$

$$= \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$= \mathbf{x}^t A \mathbf{x}$$

Eigenvalues of symmetric matrices

suppose $A \in \mathbf{R}^{n \times n}$ is symmetric, *i.e.*, $A = A^T$

fact: the eigenvalues of A are real

suppose $Av = \lambda v$, $v \neq 0$, $v \in \mathbf{C}^n$

$$\overline{v}^T A v = \overline{v}^T (Av) = \lambda \overline{v}^T v = \lambda \sum_{i=1}^{n} |v_i|^2$$

$$\overline{v}^T A v = \overline{(Av)}^T v = \overline{(\lambda v)}^T v = \overline{\lambda} \sum_{i=1}^{n} |v_i|^2$$

we have $\lambda = \overline{\lambda}$, *i.e.*, $\lambda \in \mathbf{R}$

(hence, can assume $v \in \mathbf{R}^n$)

Eigenvectors of symmetric matrices

suppose $A \in \mathbf{R}^{n \times n}$ is symmetric, $i.e.$, $A = A^T$

fact: there is a set of orthonormal eigenvectors of A

$$A = Q \Lambda Q^T$$

Eigenvectors of symmetric matrices

suppose $A \in \mathbf{R}^{n \times n}$ is symmetric, *i.e.*, $A = A^T$

fact: there is a set of orthonormal eigenvectors of A

$$A = Q \Lambda Q^T$$

$$\mathbf{x}^T \mathbf{A} \mathbf{x}$$

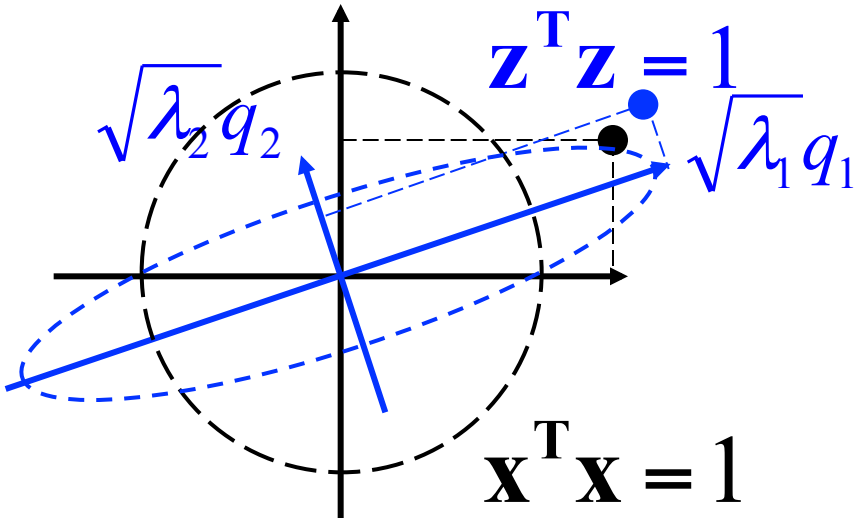
$$= \mathbf{x}^T \mathbf{Q} \, \mathbf{\Lambda} \, \mathbf{Q}^T \mathbf{x}$$

$$= \left(\mathbf{Q}^T \mathbf{x}\right)^T \mathbf{\Lambda} \left(\mathbf{Q}^T \mathbf{x}\right)$$

$$= \mathbf{y}^T \mathbf{\Lambda} \mathbf{y}$$

$$= \left(\mathbf{\Lambda}^{\frac{1}{2}} \mathbf{y}\right)^T \left(\mathbf{\Lambda}^{\frac{1}{2}} \mathbf{y}\right)$$

$$= \mathbf{z}^T \mathbf{z}$$



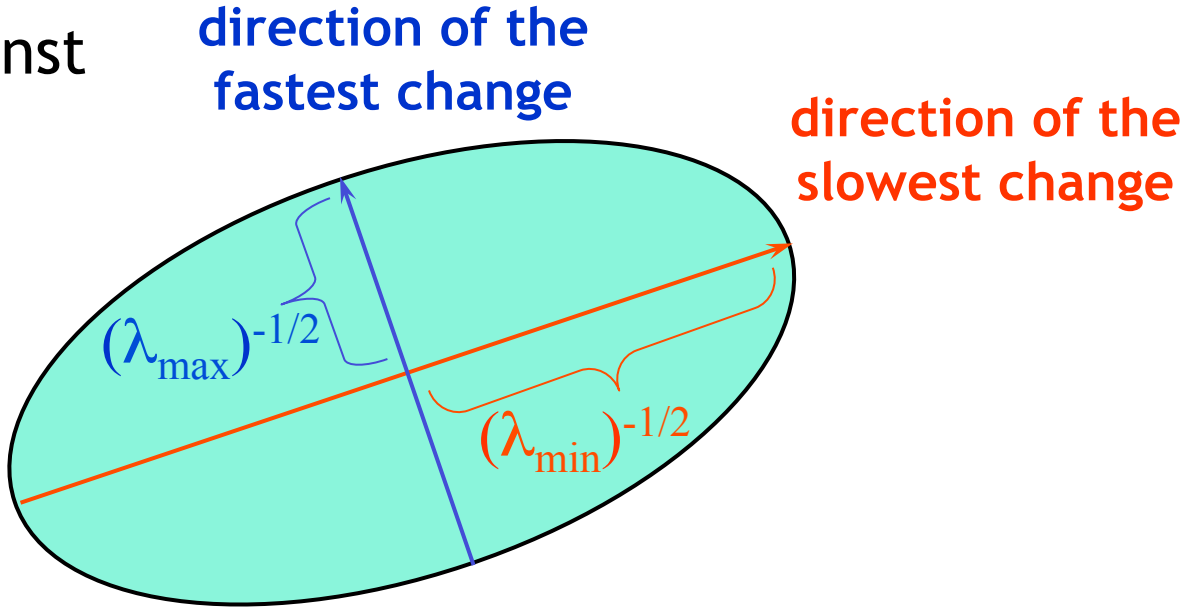
Harris corner detector

Intensity change in shifting window: eigenvalue analysis

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

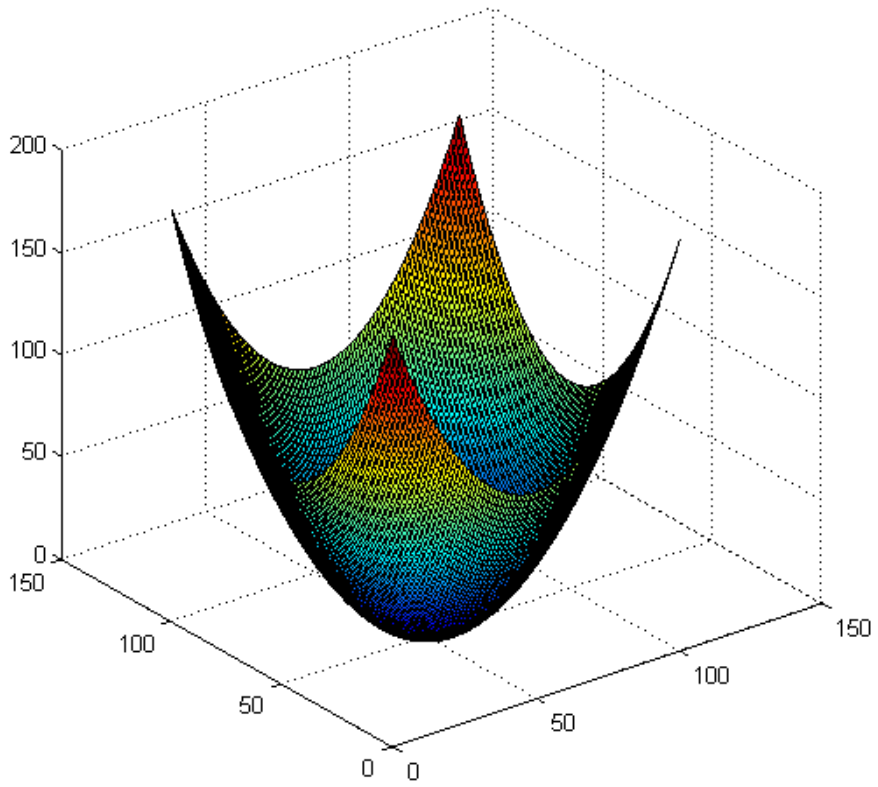
λ_1, λ_2 – eigenvalues of $\mathbf{M}$

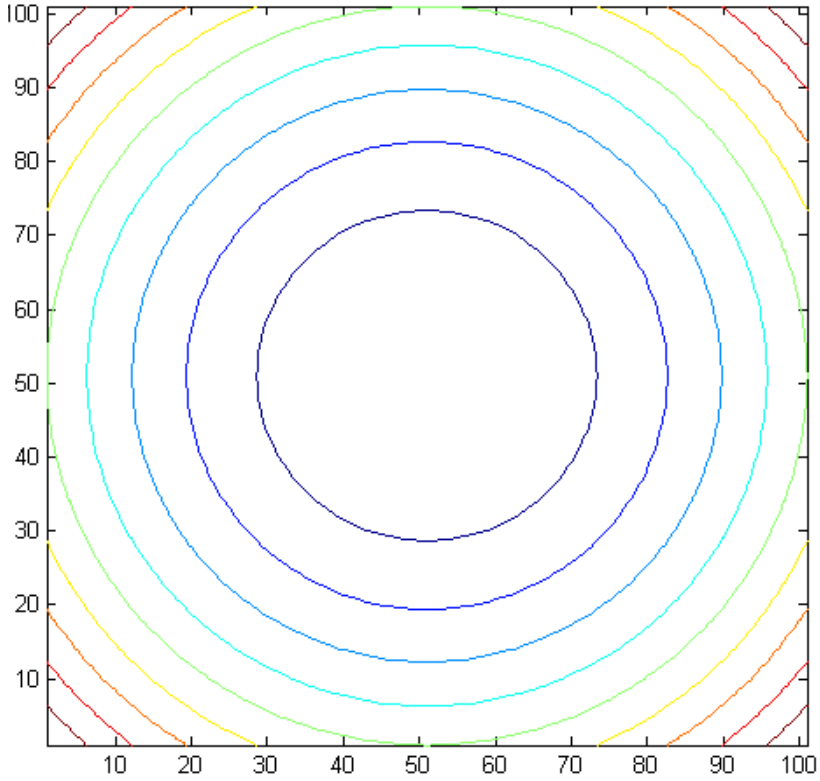
Ellipse $E(u,v)$ = const



Visualize quadratic functions

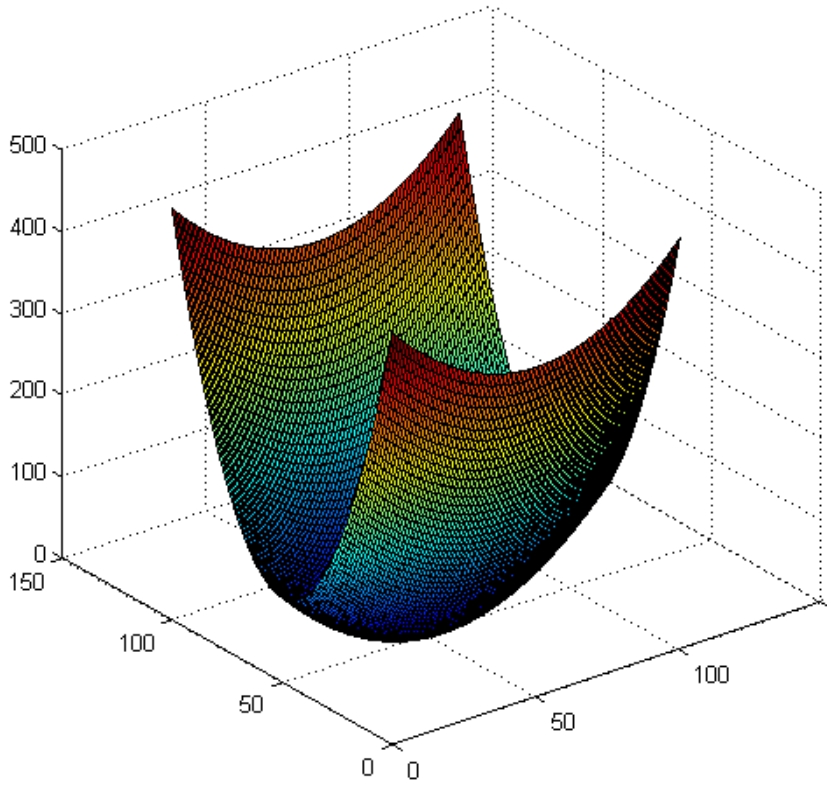
$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

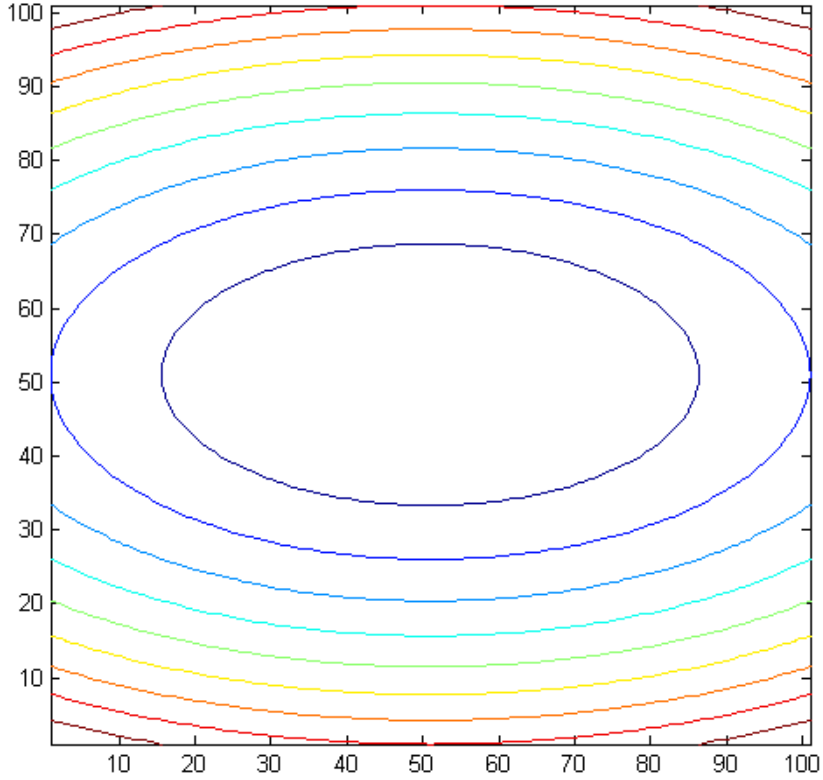




Visualize quadratic functions

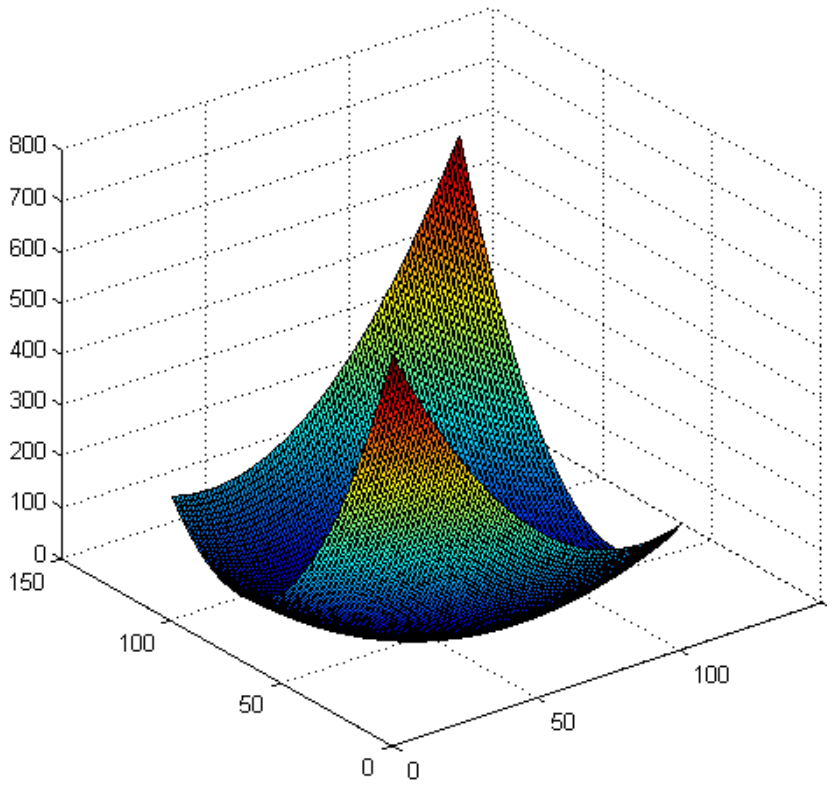
$$\mathbf{A} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

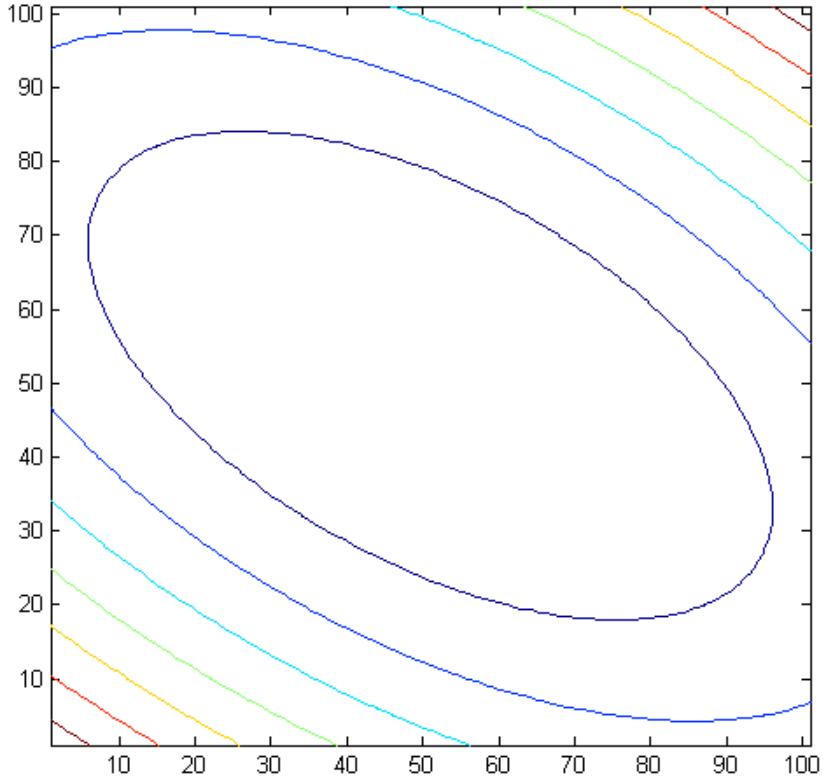




Visualize quadratic functions

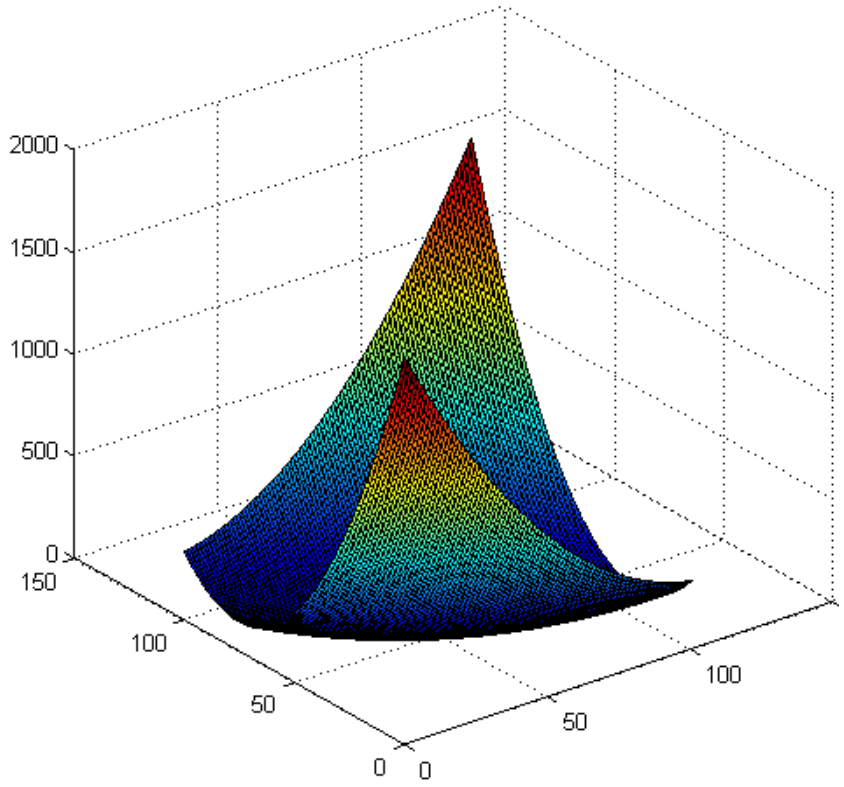
$$\mathbf{A} = \begin{bmatrix} 3.25 & 1.30 \\ 1.30 & 1.75 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$

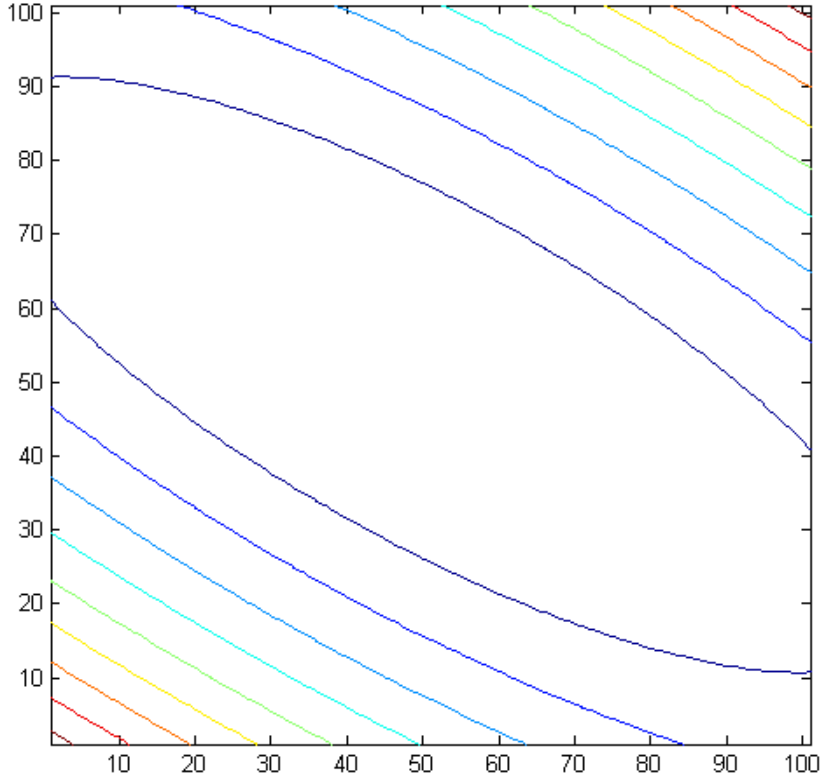




Visualize quadratic functions

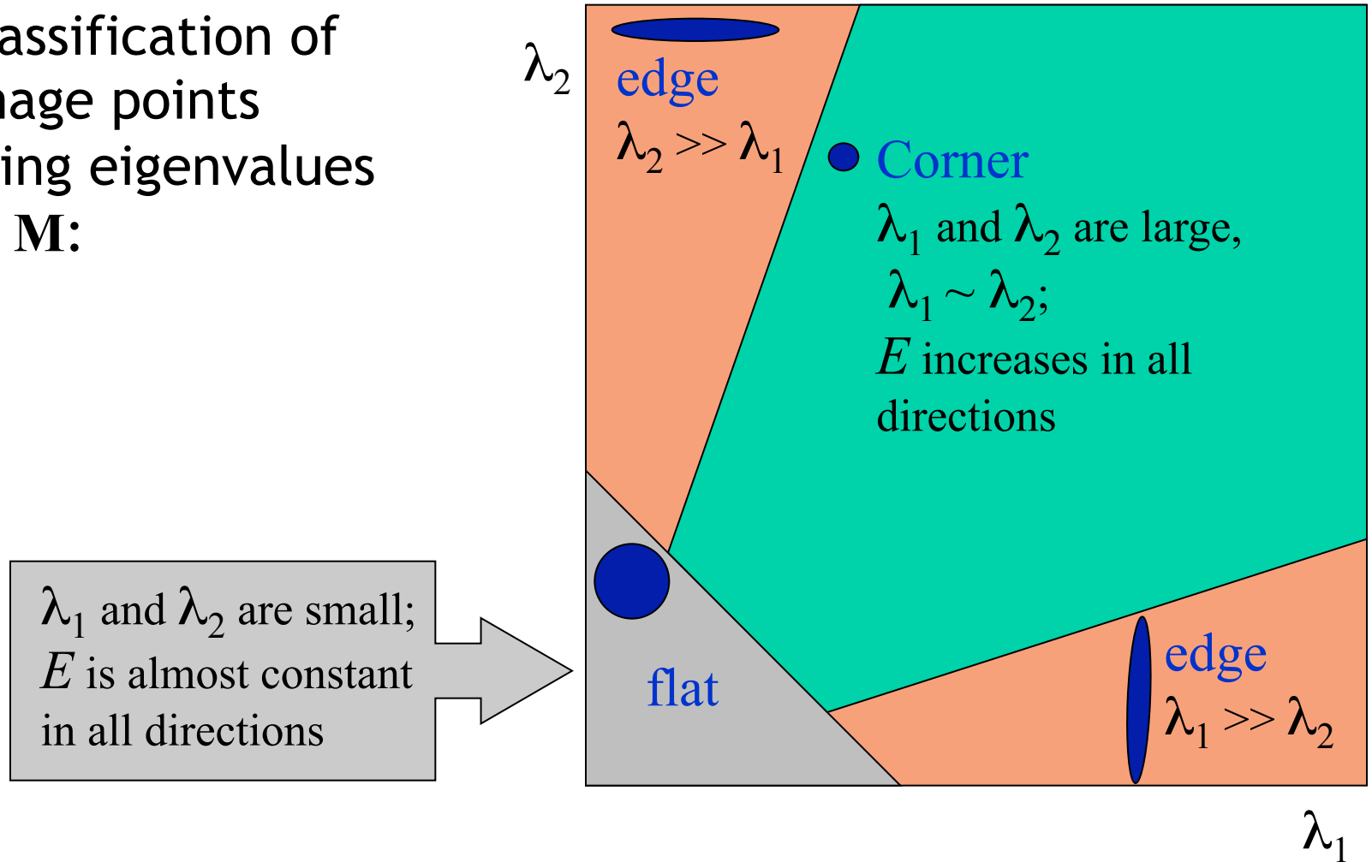
$$\mathbf{A} = \begin{bmatrix} 7.75 & 3.90 \\ 3.90 & 3.25 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$





Harris corner detector

Classification of image points using eigenvalues of **M**:



Harris corner detector

$$\lambda = \frac{a_{00} + a_{11} \pm \sqrt{(a_{00} - a_{11})^2 + 4a_{10}a_{01}}}{2}$$

Only for reference, you do not need them to compute R

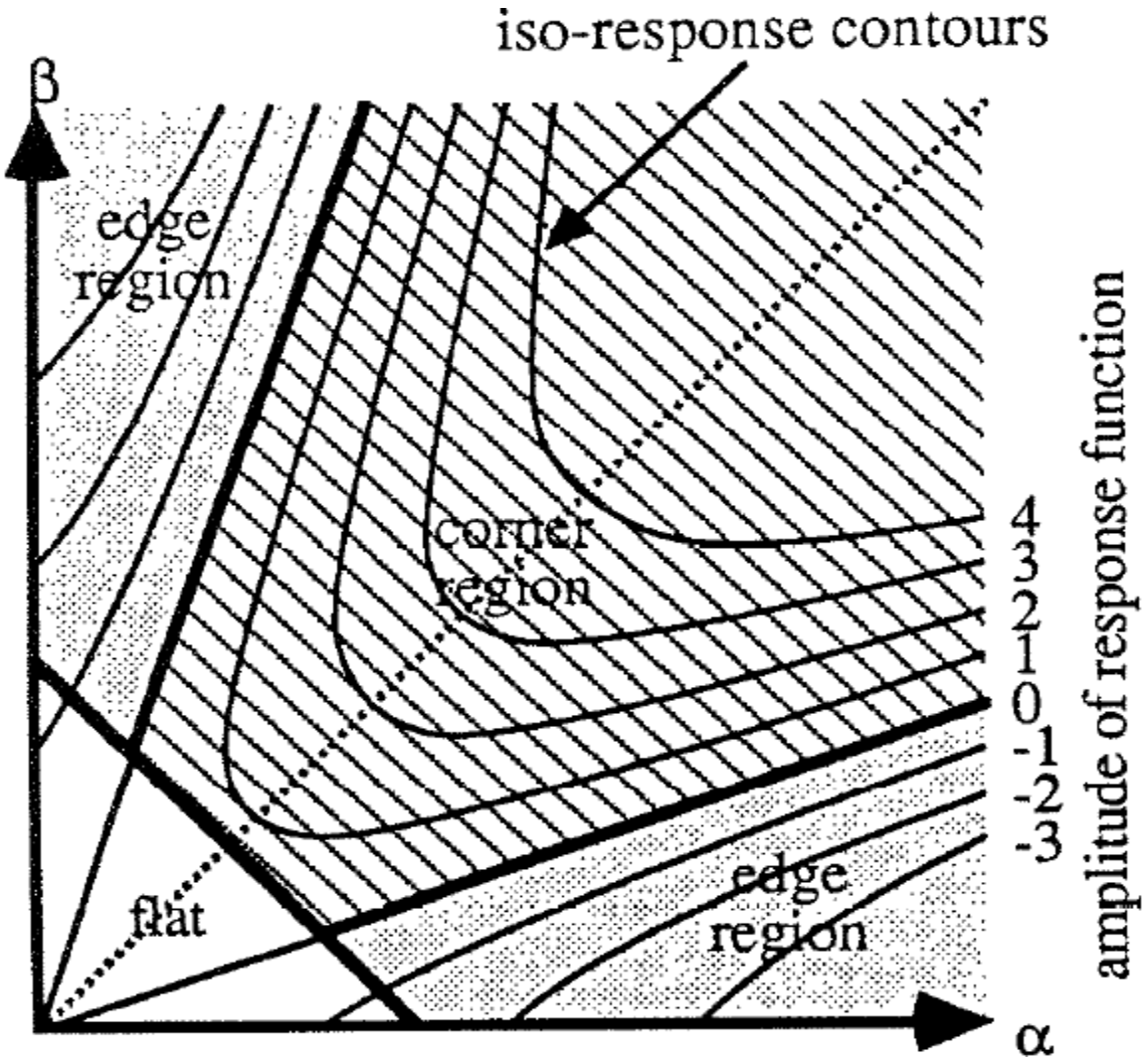
Measure of corner response:

$$R = \det\mathbf{M} - k\left(\text{trace}\mathbf{M}\right)^2$$

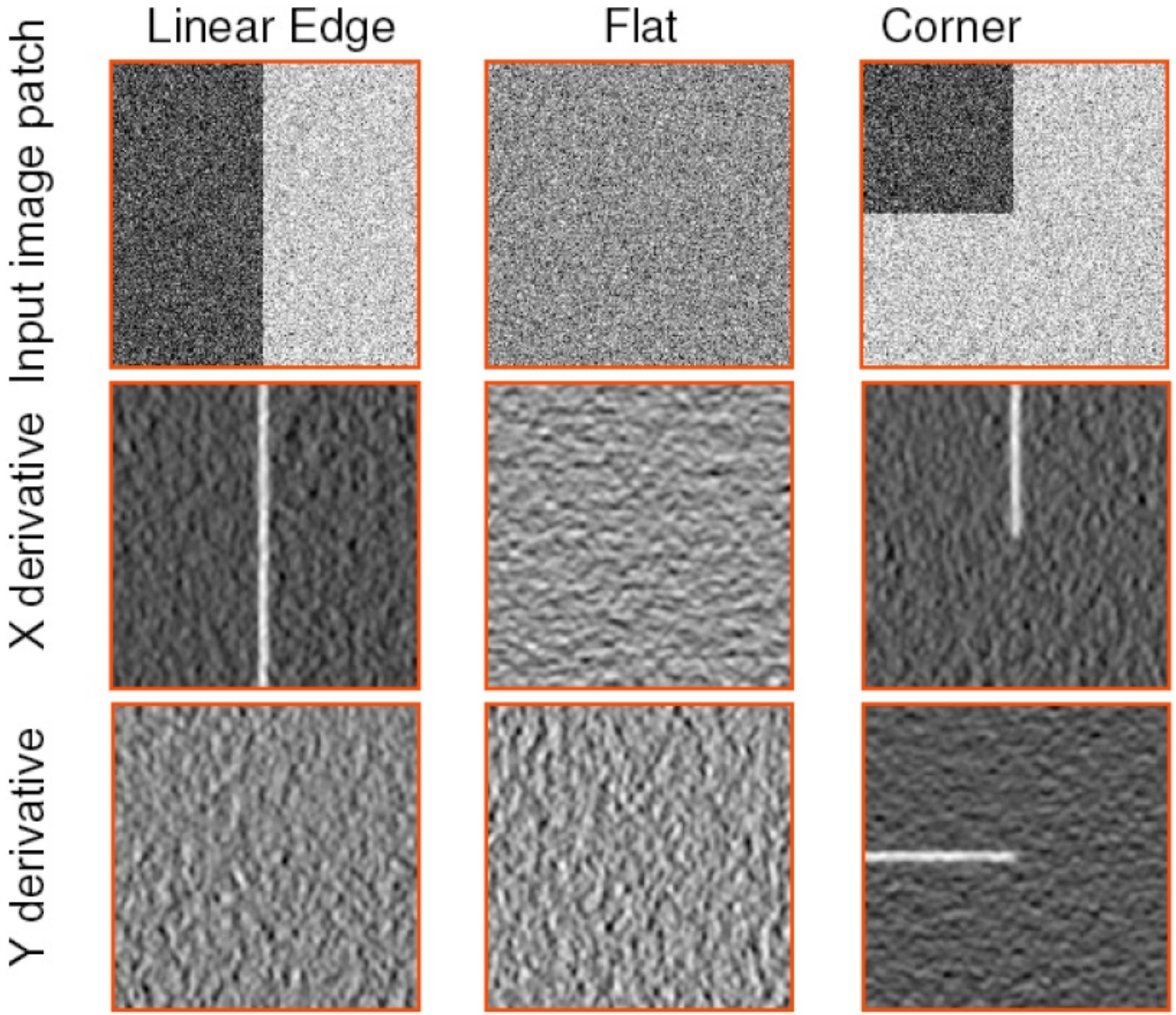
$$\det \mathbf{M} = \lambda_1\lambda_2$$
$$\text{trace}\,\mathbf{M} = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

Harris corner detector

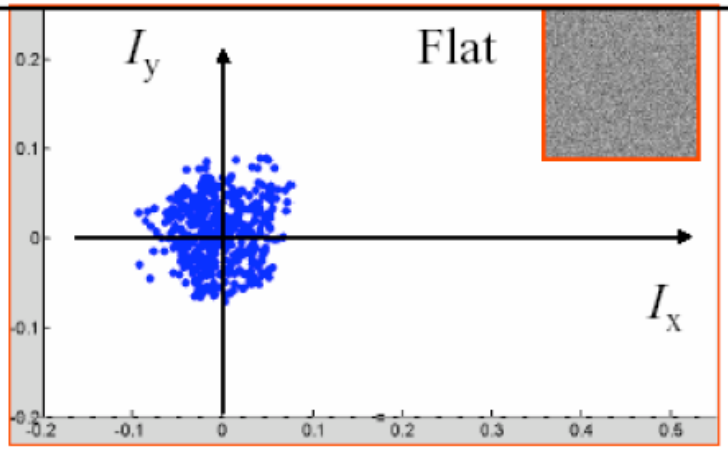


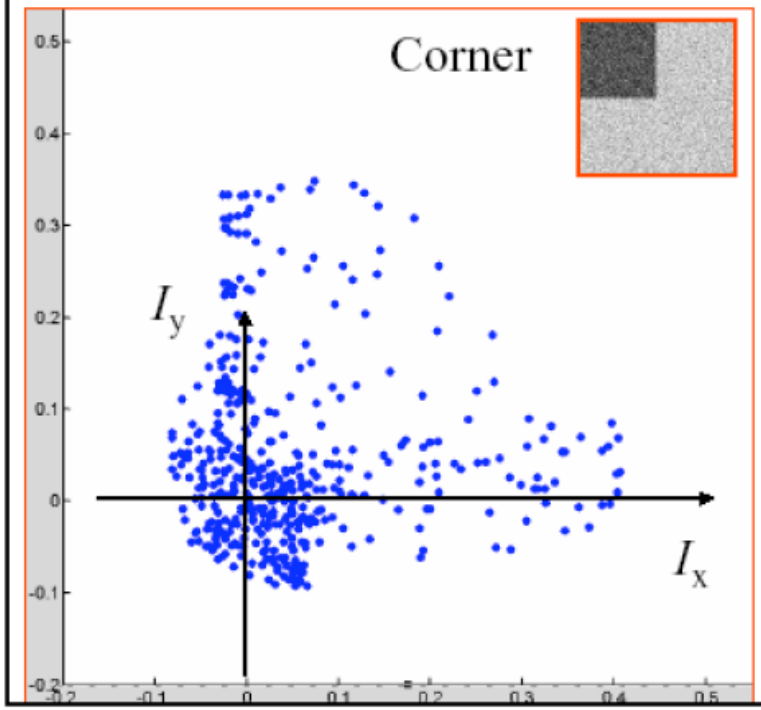
Another view

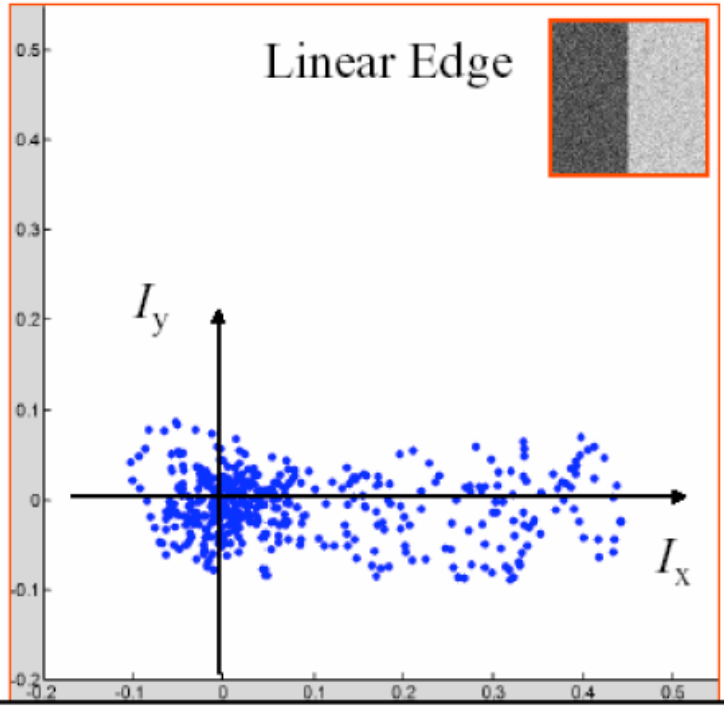


Another view

The distribution of the x and y derivatives is very different for all three types of patches

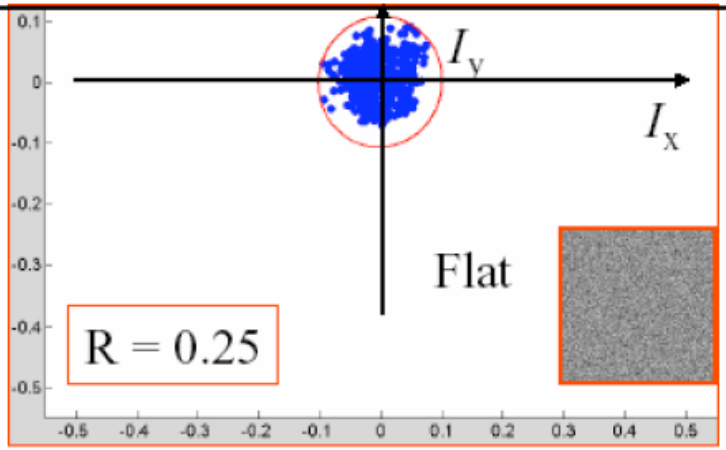


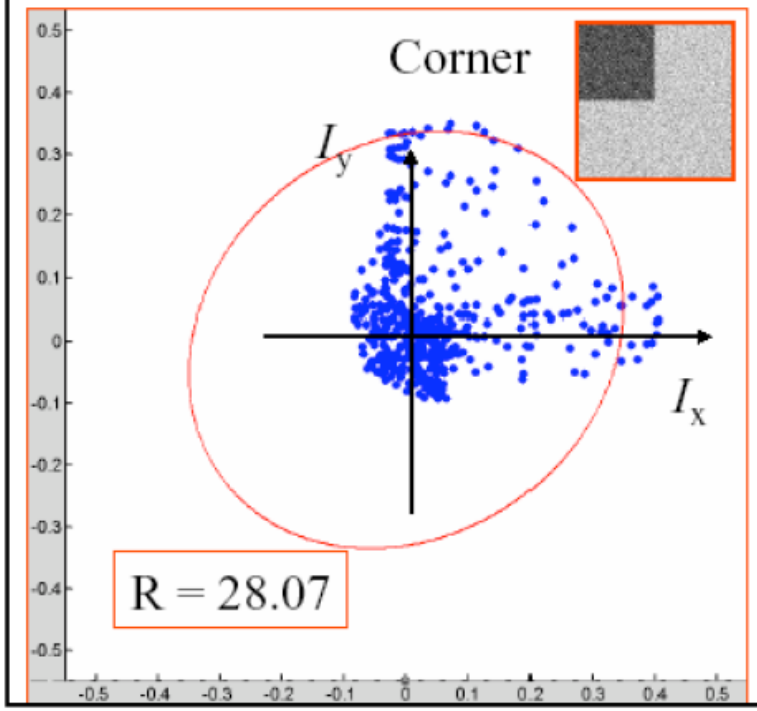


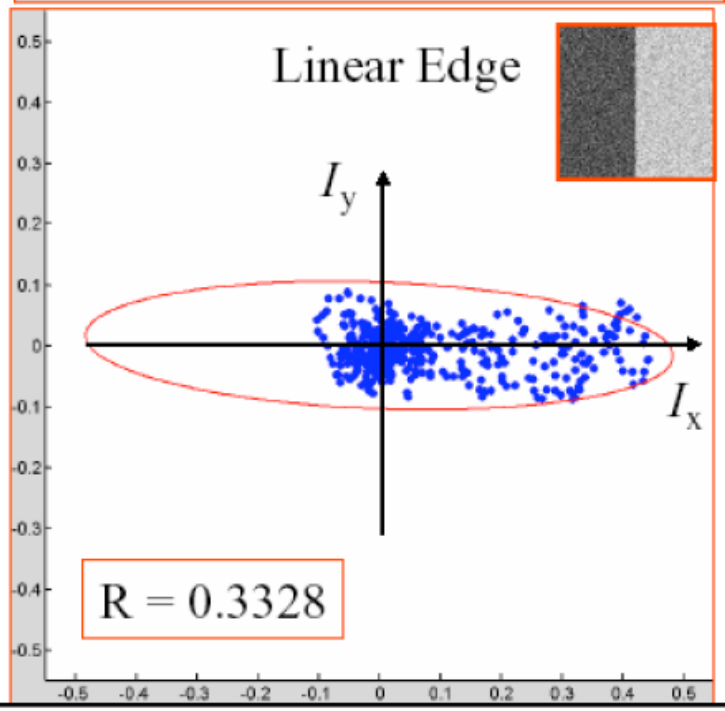


Another view

The distribution of x and y derivatives can be characterized by the shape and size of the principal component ellipse







Summary of Harris detector

1. **Compute x and y derivatives of image**

$$I_x = G_\sigma^x * I \qquad I_y = G_\sigma^y * I$$

2. **Compute products of derivatives at every pixel**

$$I_{x^2} = I_x \cdot I_x \qquad I_{y^2} = I_y \cdot I_y \qquad I_{xy} = I_x \cdot I_y$$

3. **Compute the sums of the products of derivatives at each pixel**

$$S_{x^2} = G_{\sigma'} * I_{x^2} \qquad S_{y^2} = G_{\sigma'} * I_{y^2} \qquad S_{xy} = G_{\sigma'} * I_{xy}$$

Summary of Harris detector

4. **Define the matrix at each pixel**

$$M(x,y) = \begin{bmatrix} S_{x^2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y^2}(x,y) \end{bmatrix}$$

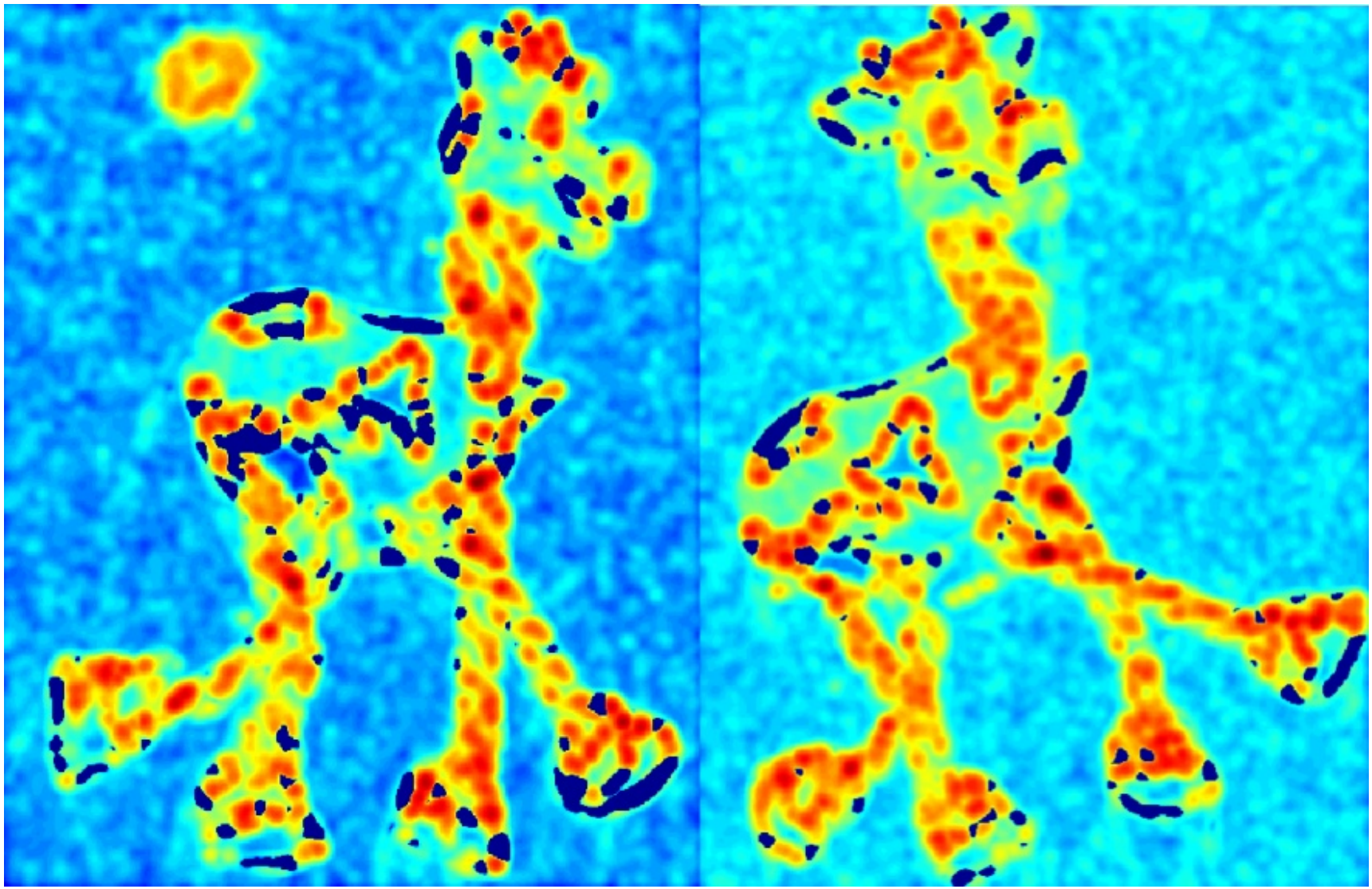
5. **Compute the response of the detector at each pixel**

$$R = \det M - k\left(\text{trace}M\right)^2$$

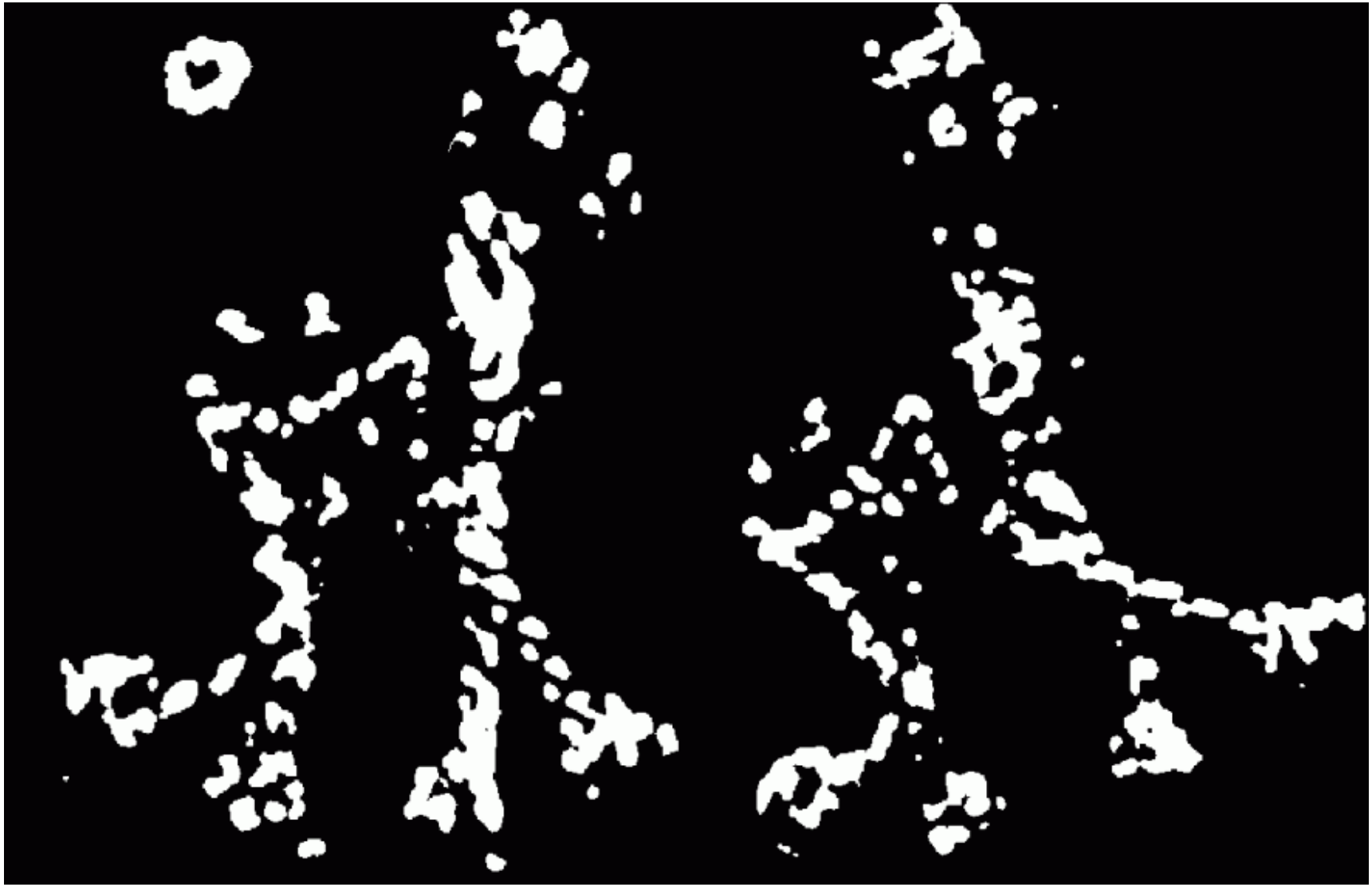
6. **Threshold on value of R; compute nonmax suppression.**

Harris corner detector (input)

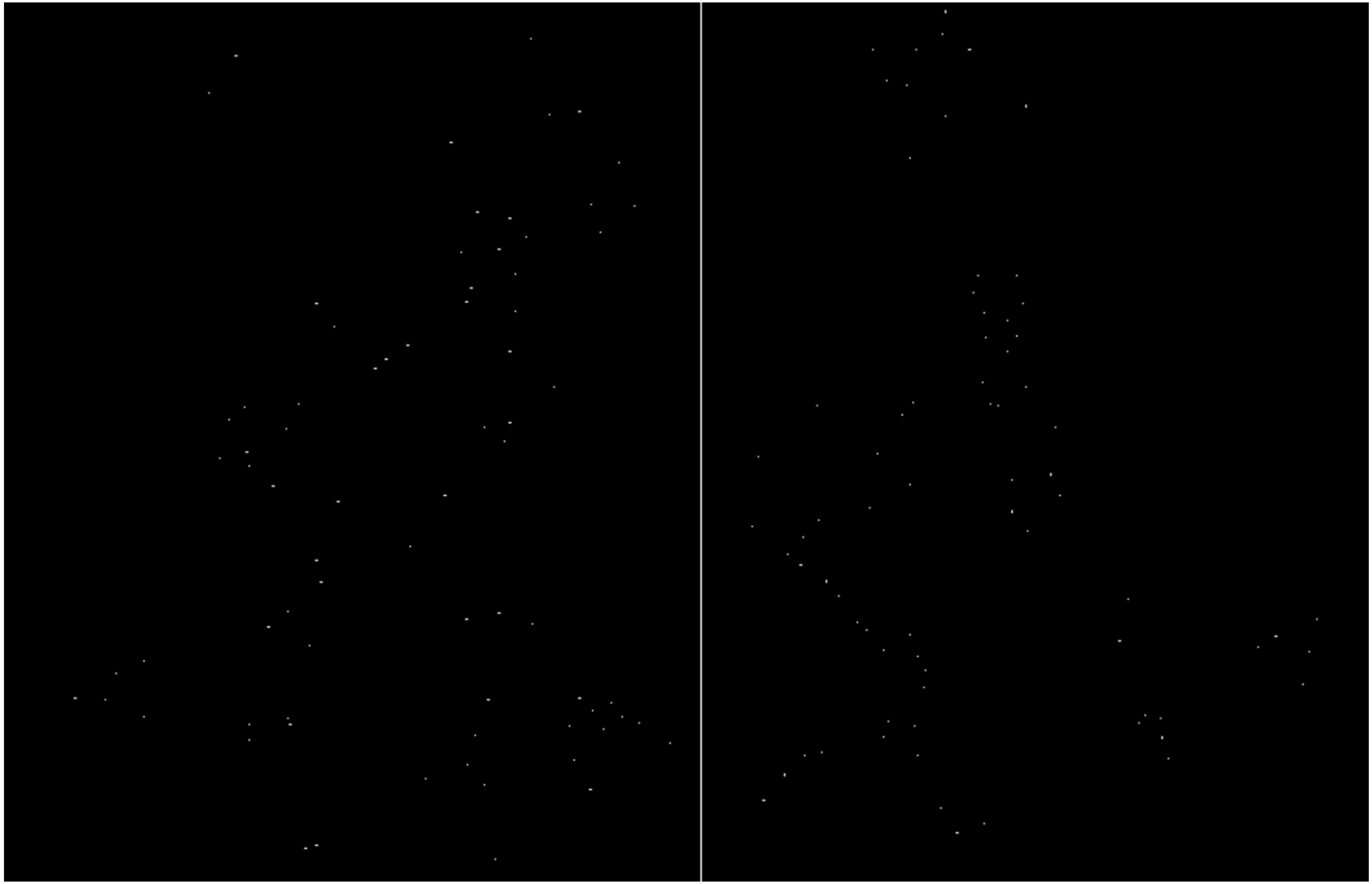
Corner response R



Threshold on R



Local maximum of R



Harris corner detector

Harris detector: summary

- Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

$$E(u,v) \cong \begin{bmatrix} u,v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

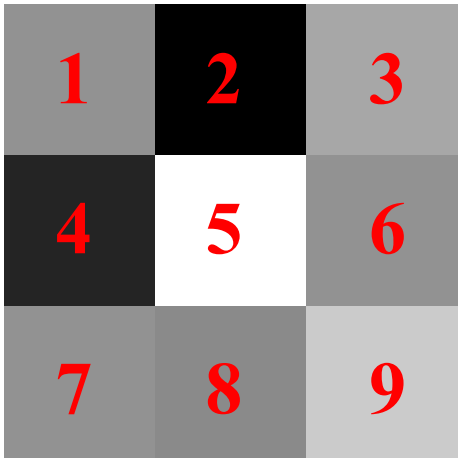
- Describe a point in terms of eigenvalues of M: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

- A good (corner) point should have a *large intensity change* in *all directions*, i.e. R should be large positive

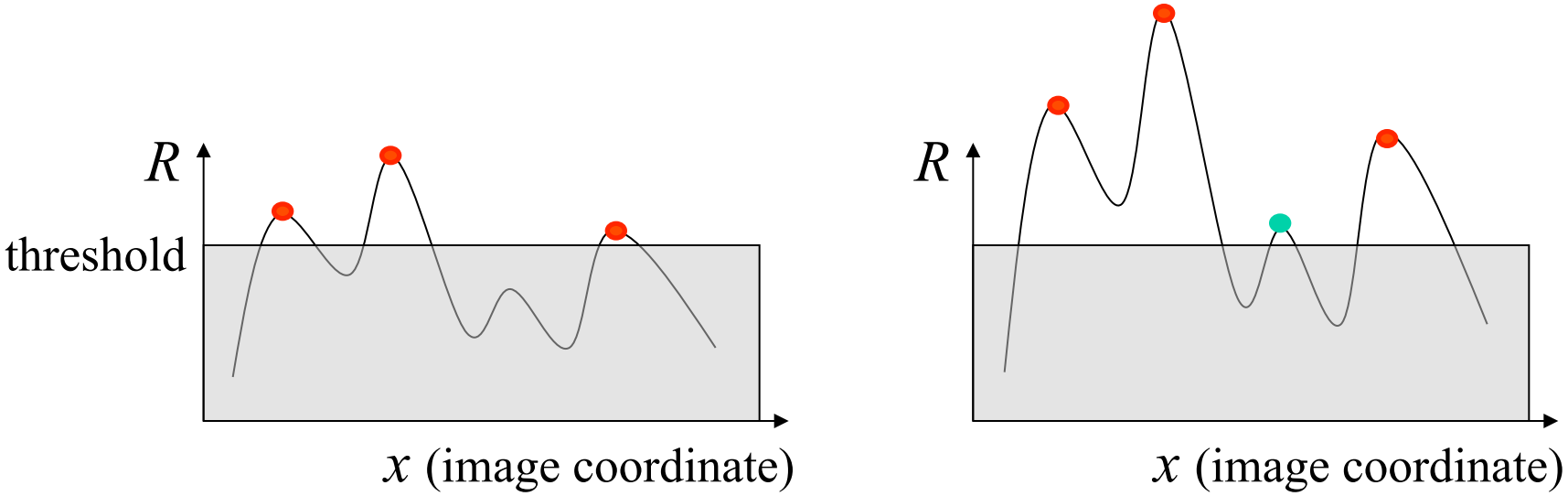
Now we know where features are

- But, how to match them?
- What is the descriptor for a feature? The simplest solution is the intensities of its spatial neighbors. This might not be robust to brightness change or small shift/rotation.



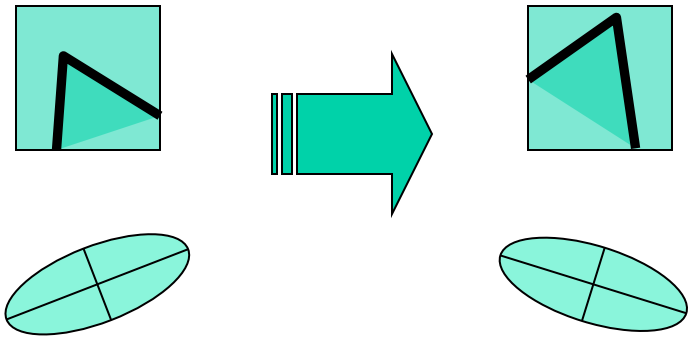
Harris detector: some properties

- Partial invariance to *affine intensity* change
  - ✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
  - ✓ Intensity scale: $I \rightarrow a\, I$



Harris Detector: Some Properties

❖Rotation invariance



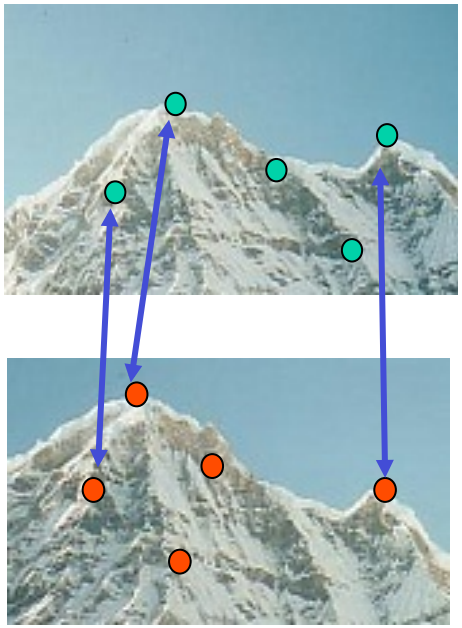
Ellipse rotates but its shape (i.e. eigenvalues) remains the same

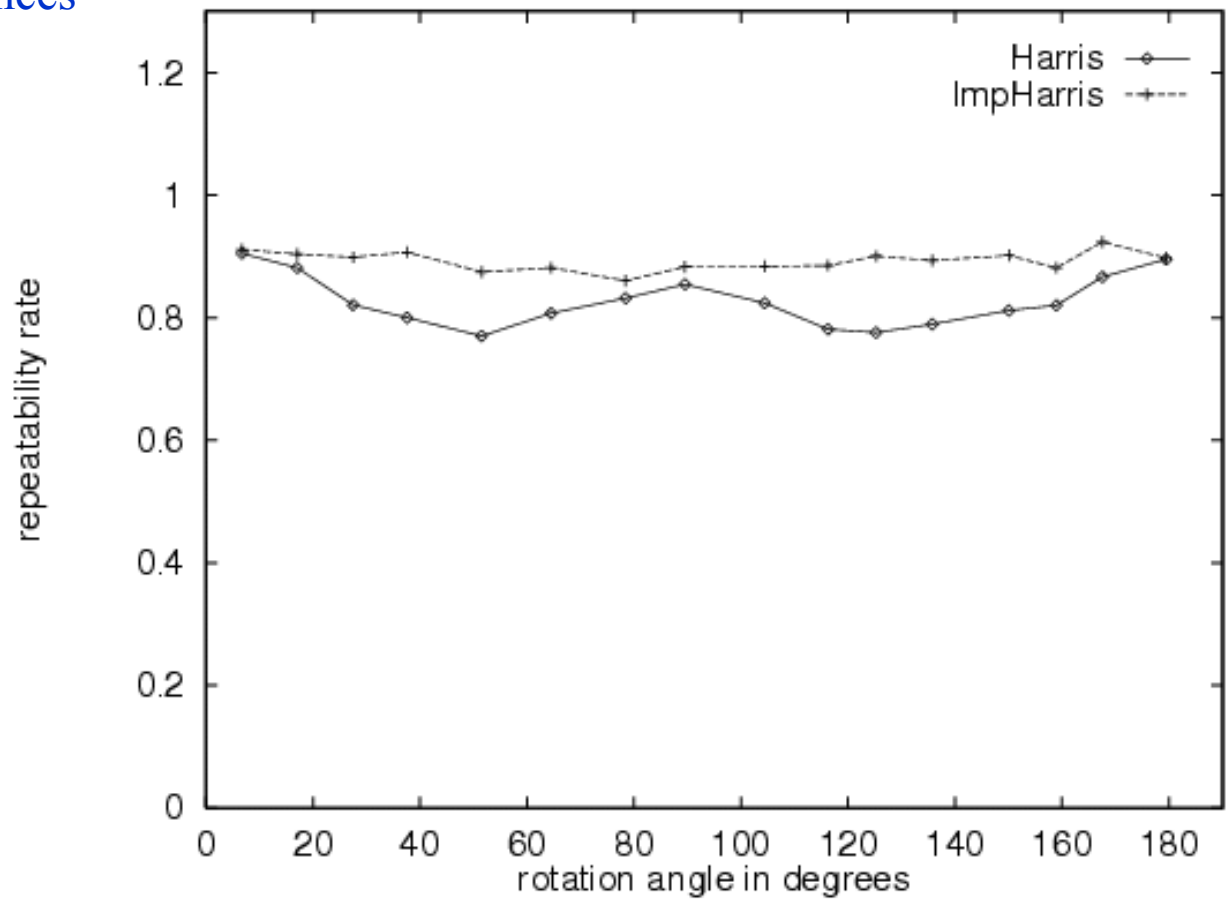
Corner response R is invariant to image rotation

Harris Detector is rotation invariant

Repeatability rate:

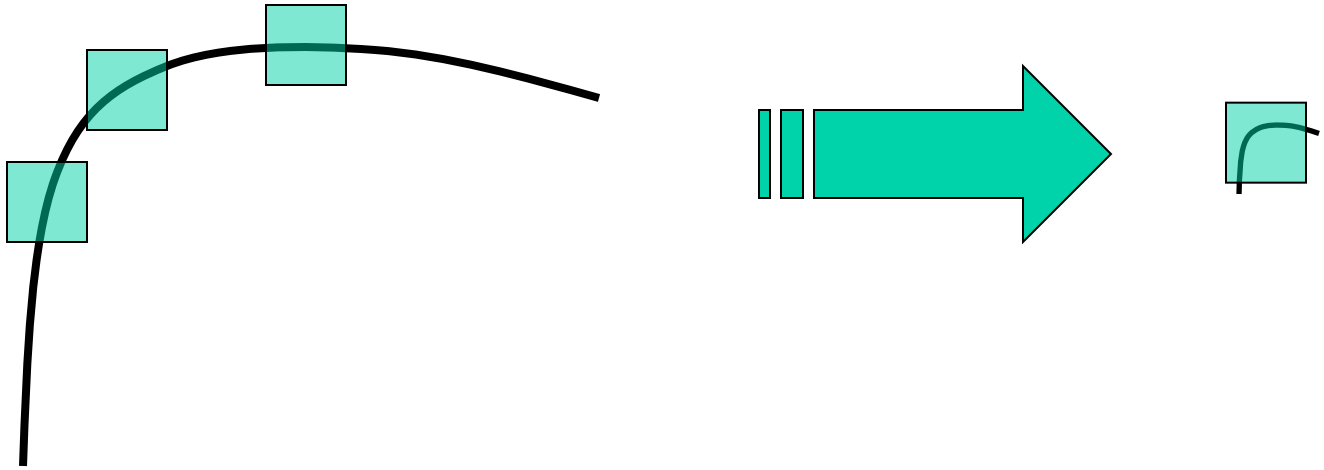
$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





Harris Detector: Some Properties

❖But: not invariant to *image scale*!



All points will be classified as edges

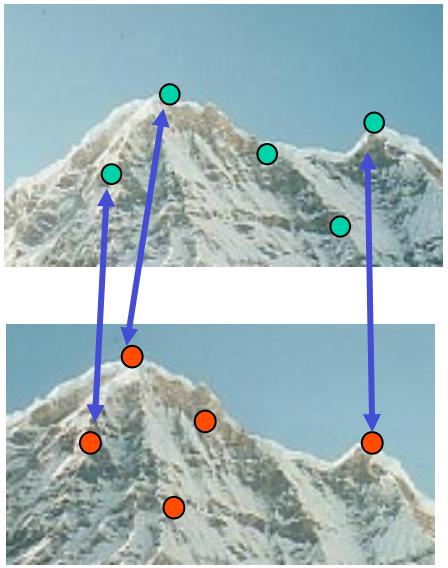
Corner !

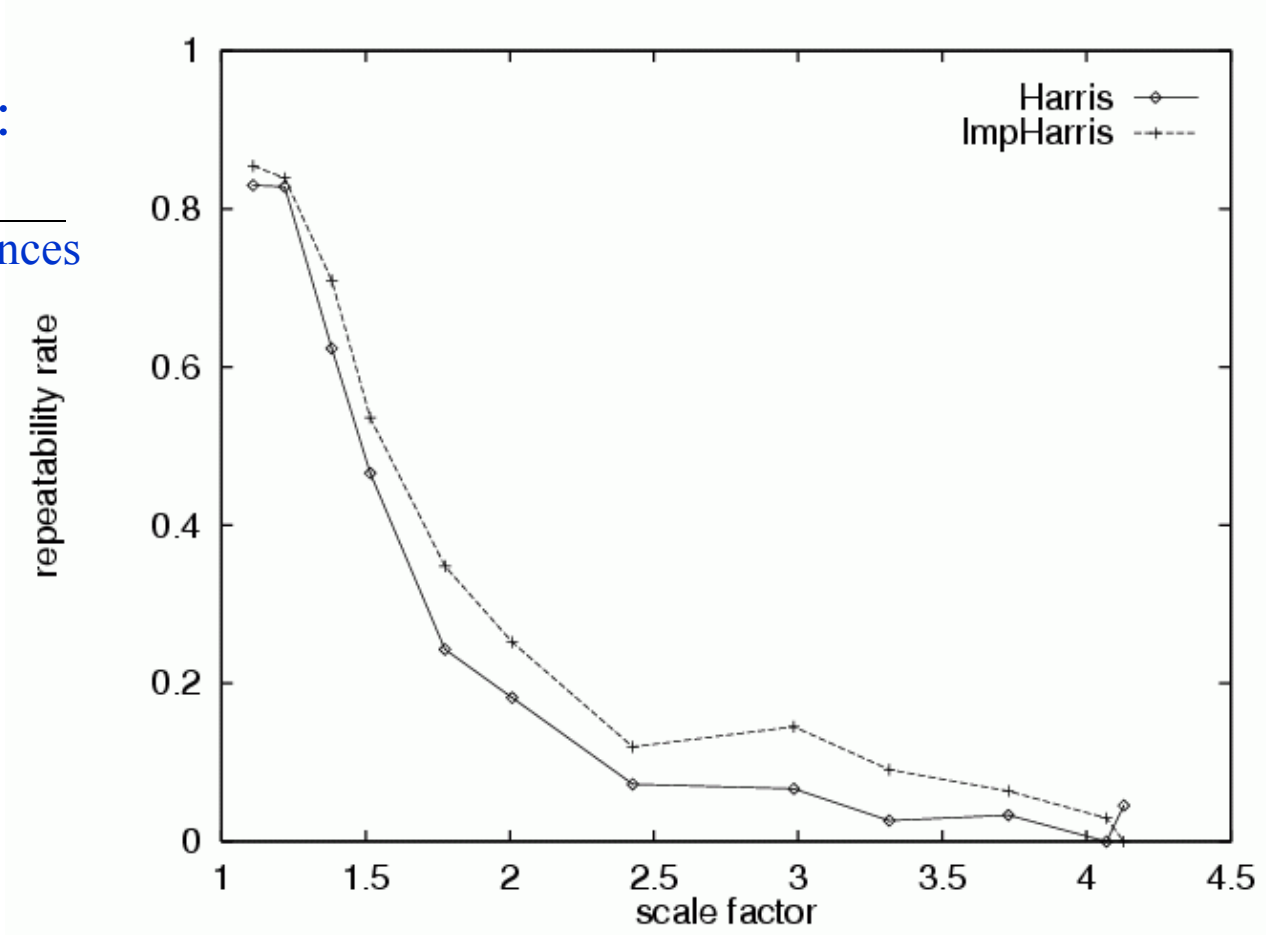
Harris detector: some properties

- Quality of Harris detector for different scale changes

Repeatability rate:

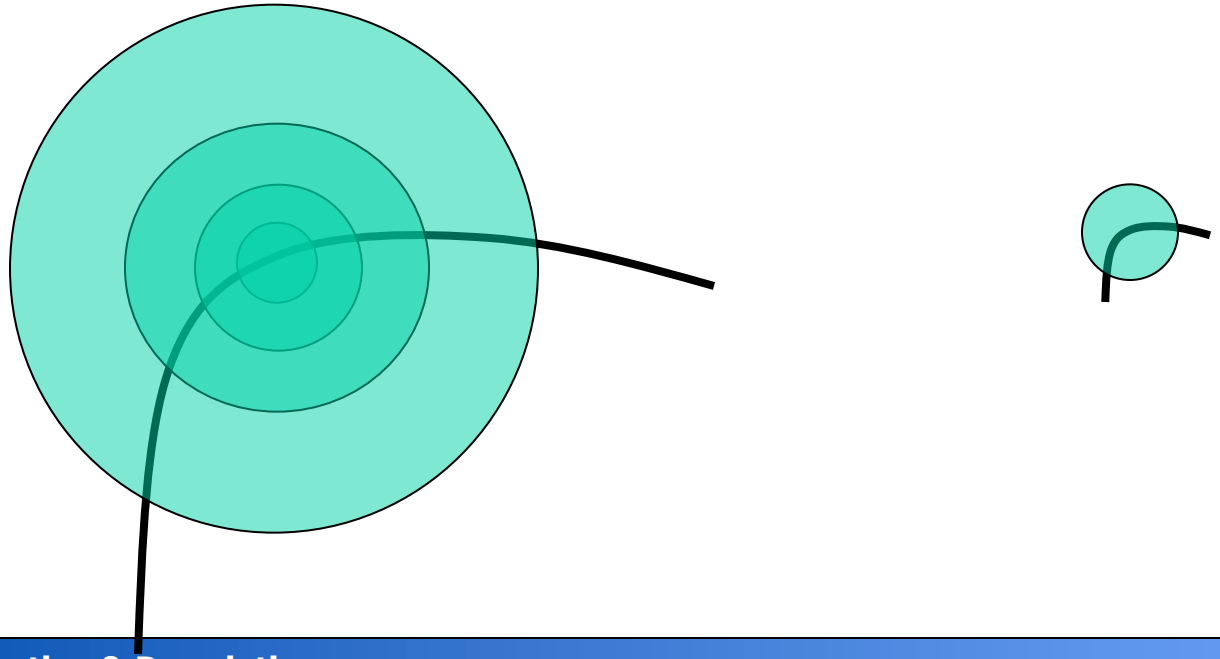
$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





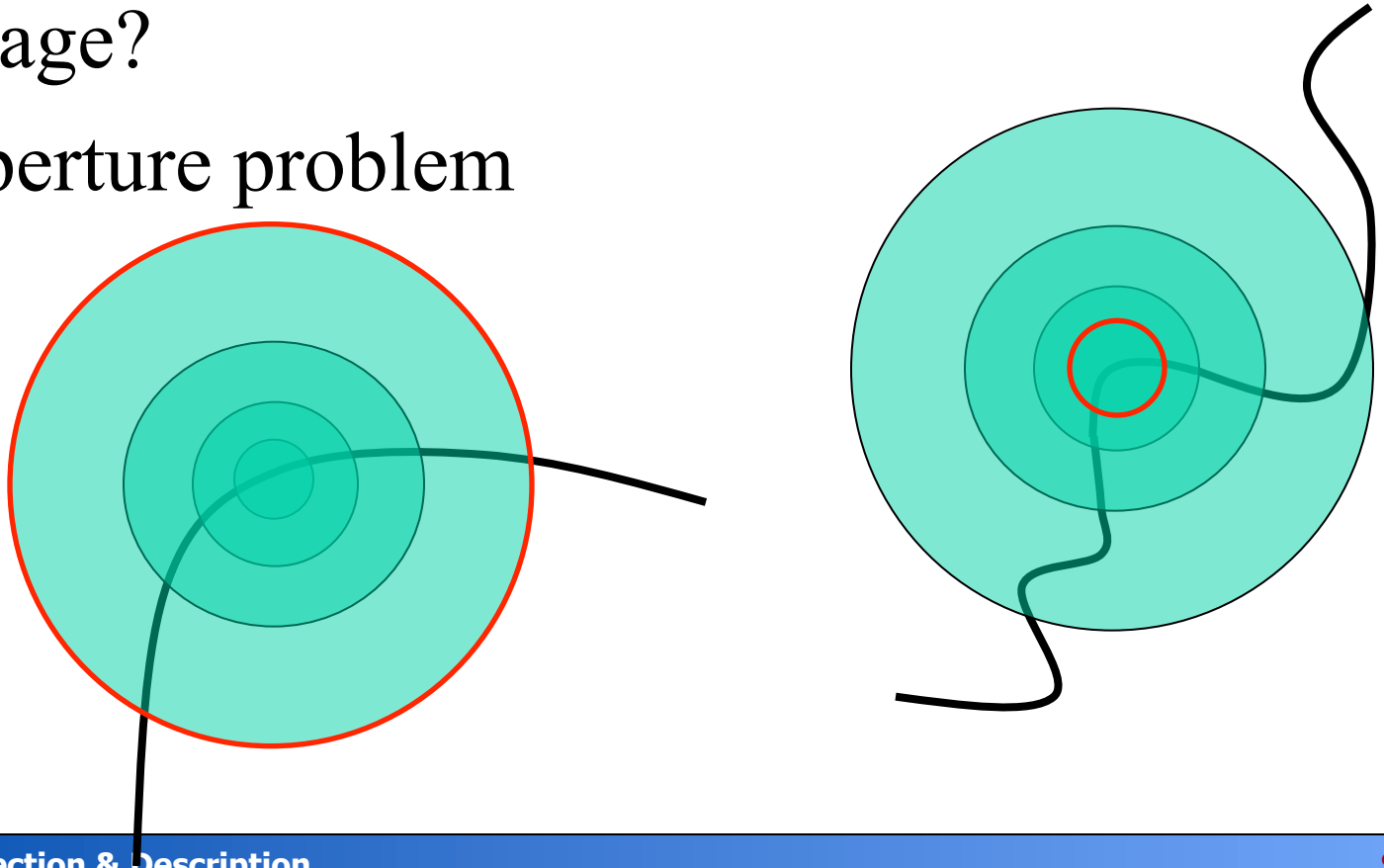
Scale invariant detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale invariant detection

- The problem: how do we choose corresponding circles ***independently*** in each image?
- Aperture problem



SIFT
(Scale Invariant Feature Transform)

SIFT

- SIFT is an carefully designed procedure with empirically determined parameters for the invariant and distinctive features.

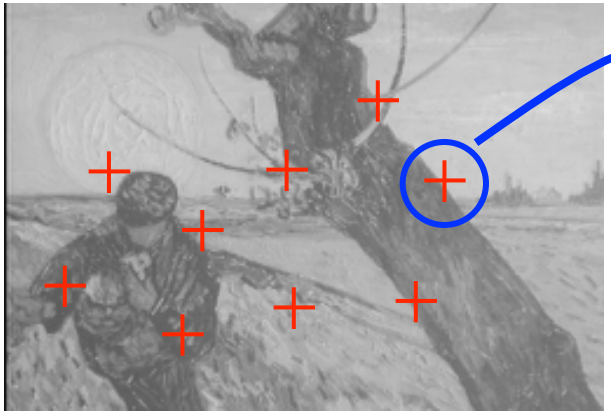
SIFT stages:

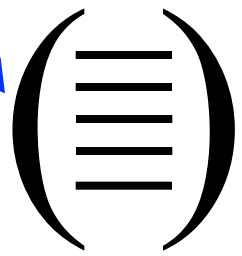
- ❖Scale-space extrema detection
- ❖Keypoint localization

detector

- ❖Orientation assignment
- ❖Keypoint descriptor

descriptor





local descriptor

A 500x500 image gives about 2000 features

1. Detection of scale-space extrema

- For scale invariance, search for stable features across all possible scales using a continuous function of scale, scale space.
- SIFT uses DoG filter for scale space because it is efficient and as stable as scale-normalized Laplacian of Gaussian.

1. Detection of scale-space extrema

DoG filtering

Convolution with a variable-scale Gaussian

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

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

Difference-of-Gaussian (DoG) filter

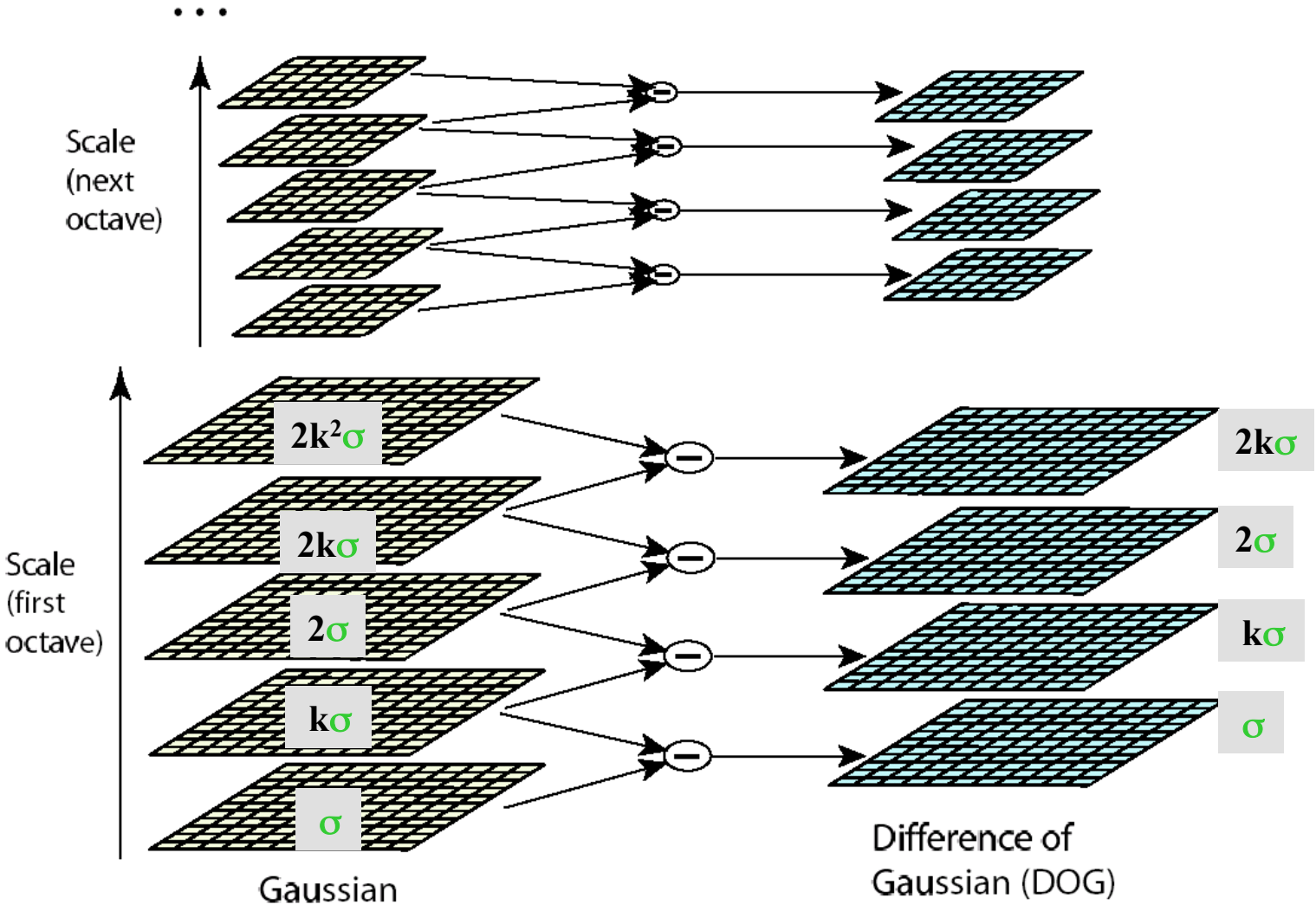
$$G(x, y, k\sigma) - G(x, y, \sigma)$$

Convolution with the DoG filter

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

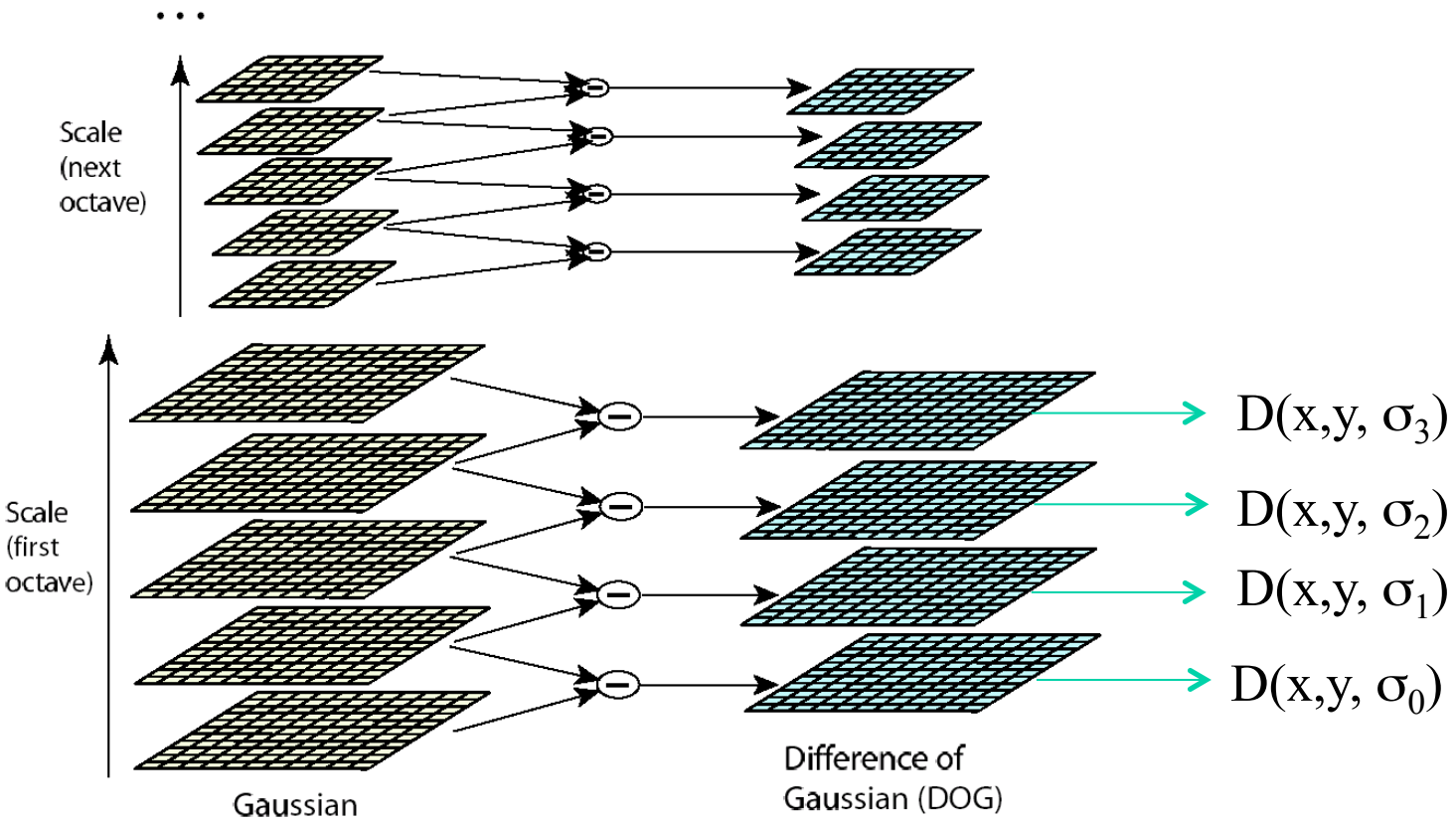
1. Detection of scale-space extrema

Scale space



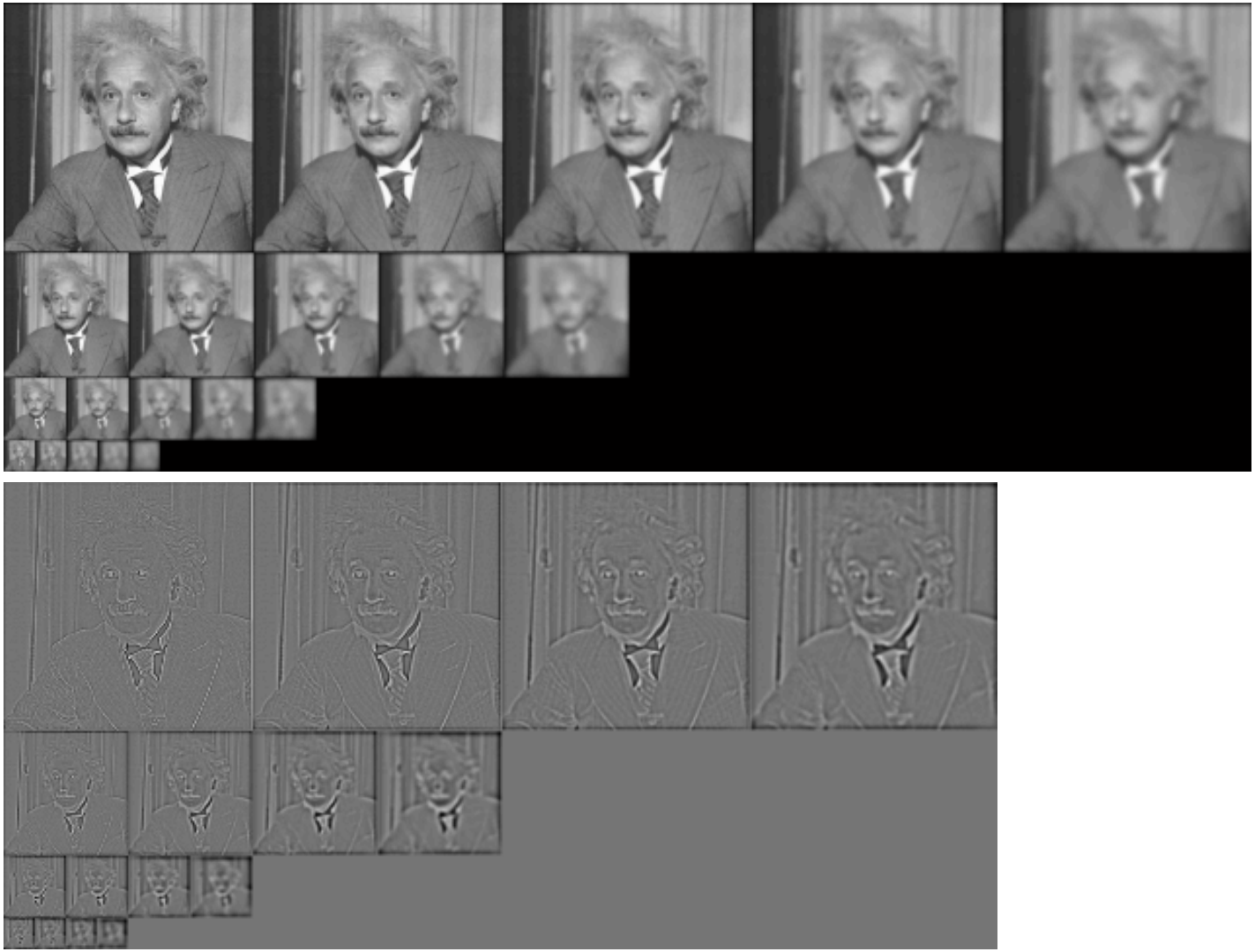
1. Detection of scale-space extrema

Scale space



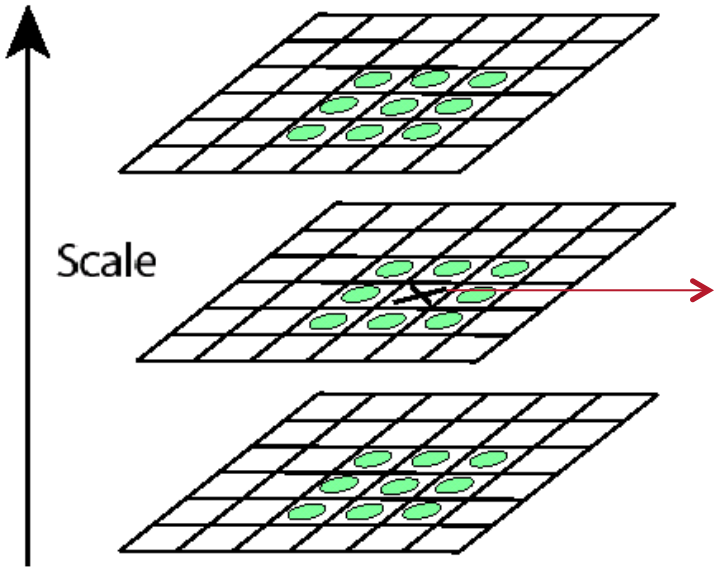
1. Detection of scale-space extrema

Examples



1. Detection of scale-space extrema

Key-points localization



X is selected as a **key-point**,

if it is larger or smaller than all 26 neighbors

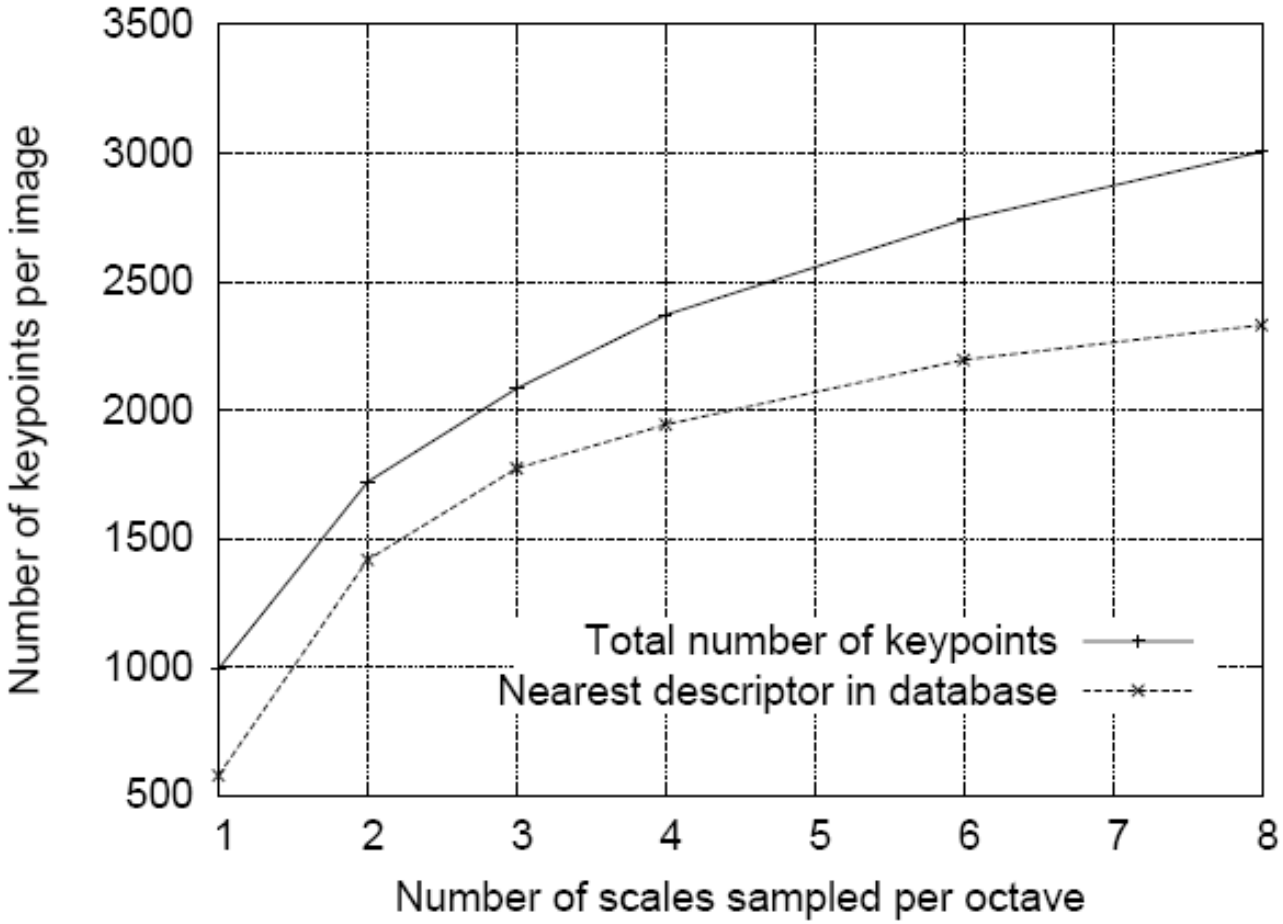
1. Detection of scale-space extrema

Decide scale sampling frequency

- It is impossible to sample the whole space, tradeoff efficiency with completeness.
- Decide the best sampling frequency by experimenting on 32 real image subject to synthetic transformations. (rotation, scaling, affine stretch, brightness and contrast change, adding noise…)

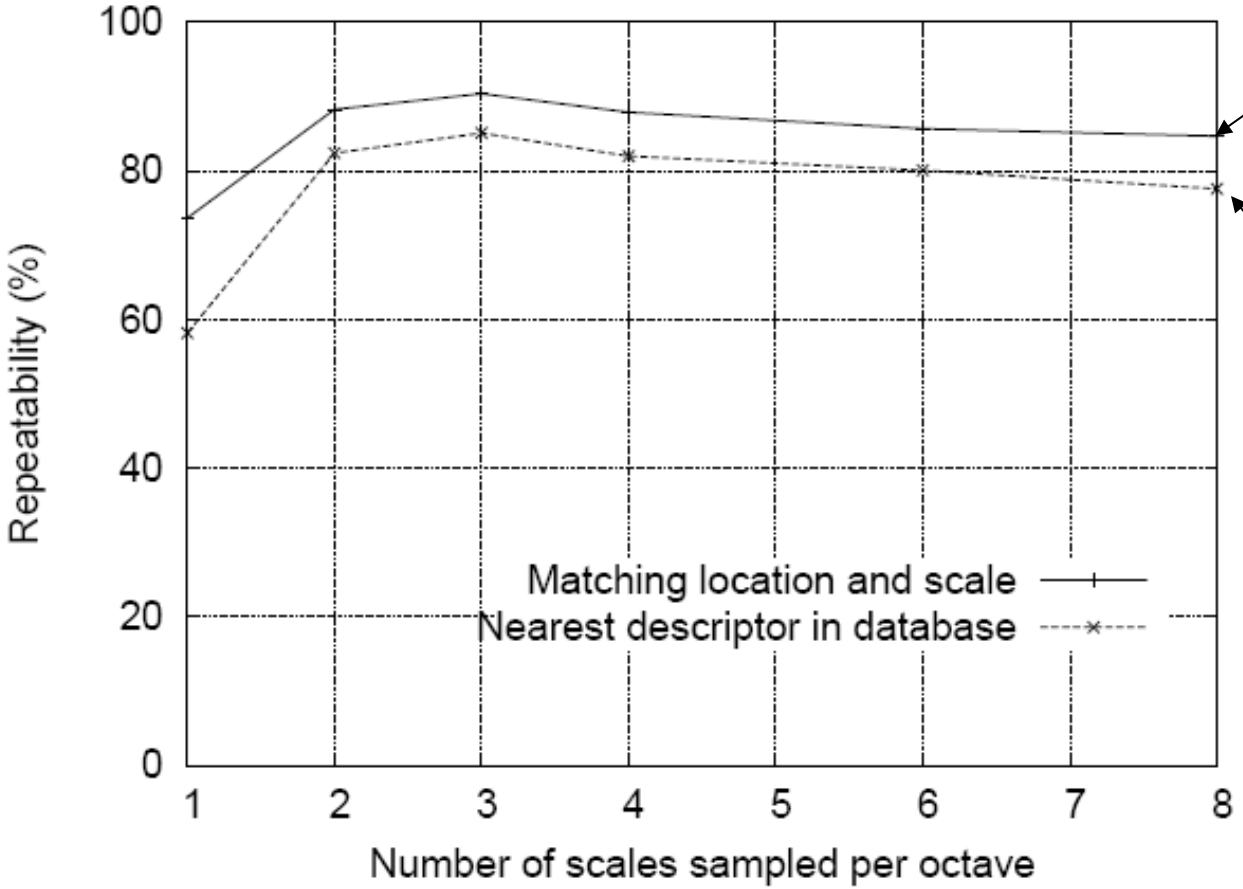
1. Detection of scale-space extrema

Decide scale sampling frequency



1. Detection of scale-space extrema

Decide scale sampling frequency



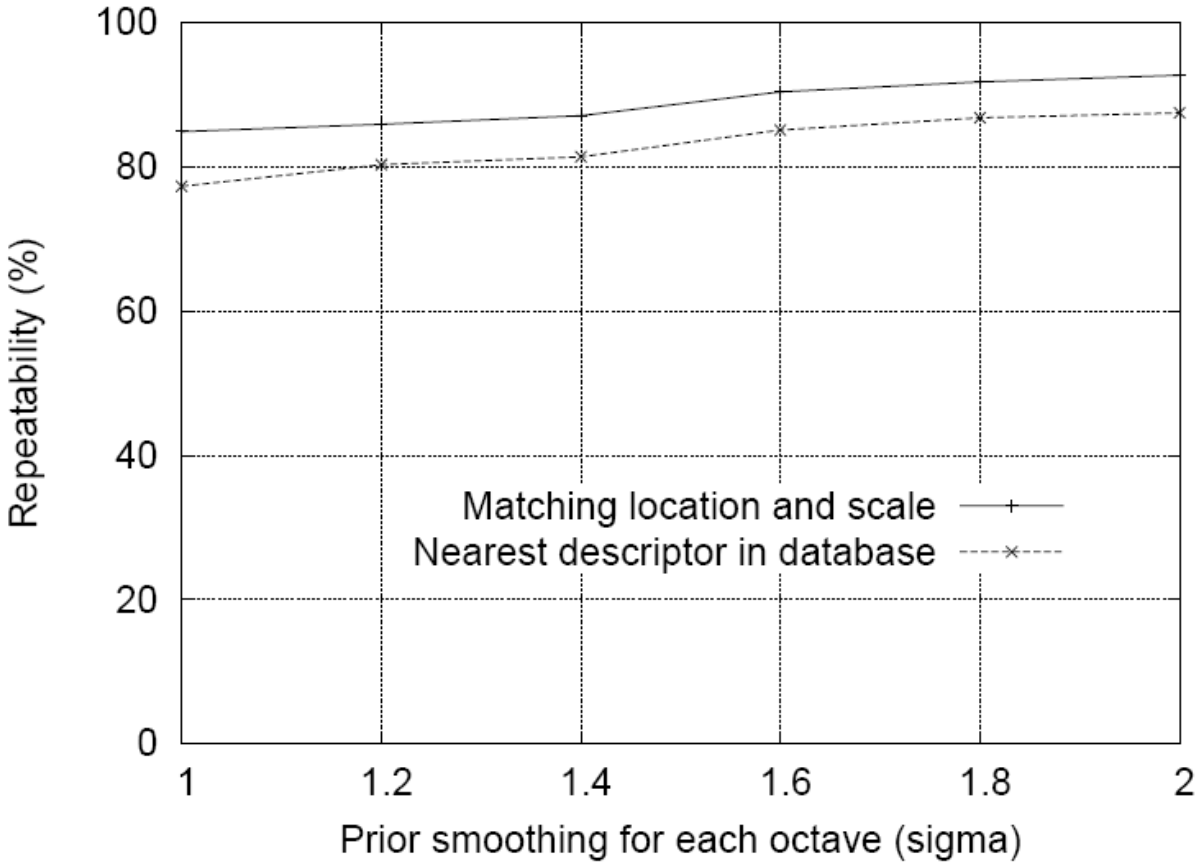
for detector, repeatability

for descriptor, distinctiveness

s=3 is the best, for larger s, too many unstable features

1. Detection of scale-space extrema

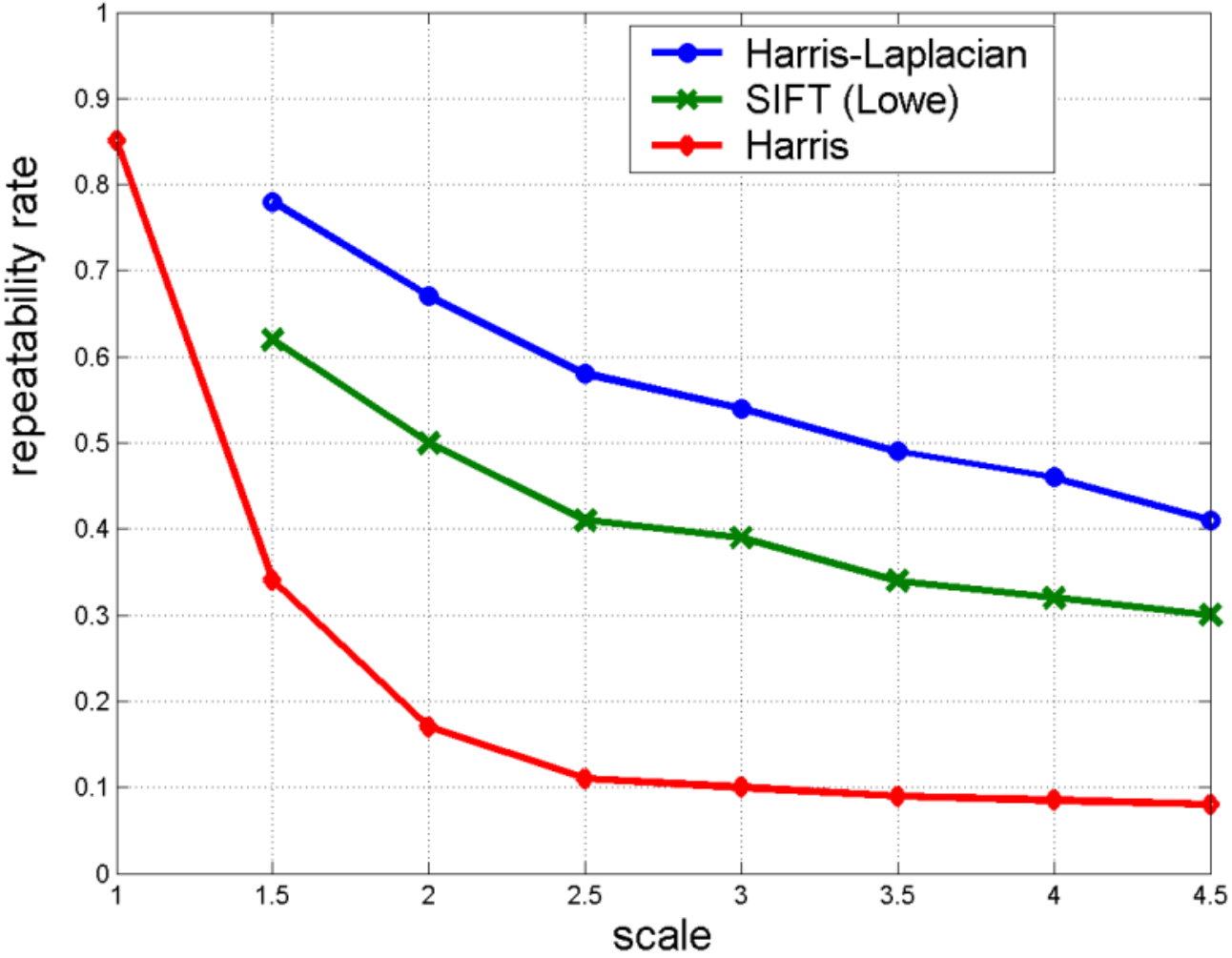
Pre-smoothing



σ =1.6, plus a double expansion

1. Detection of scale-space extrema

Scale Invariance



2. Accurate key-point localization

- **Fit a 3D quadratic function** for finding sub-pixel maxima

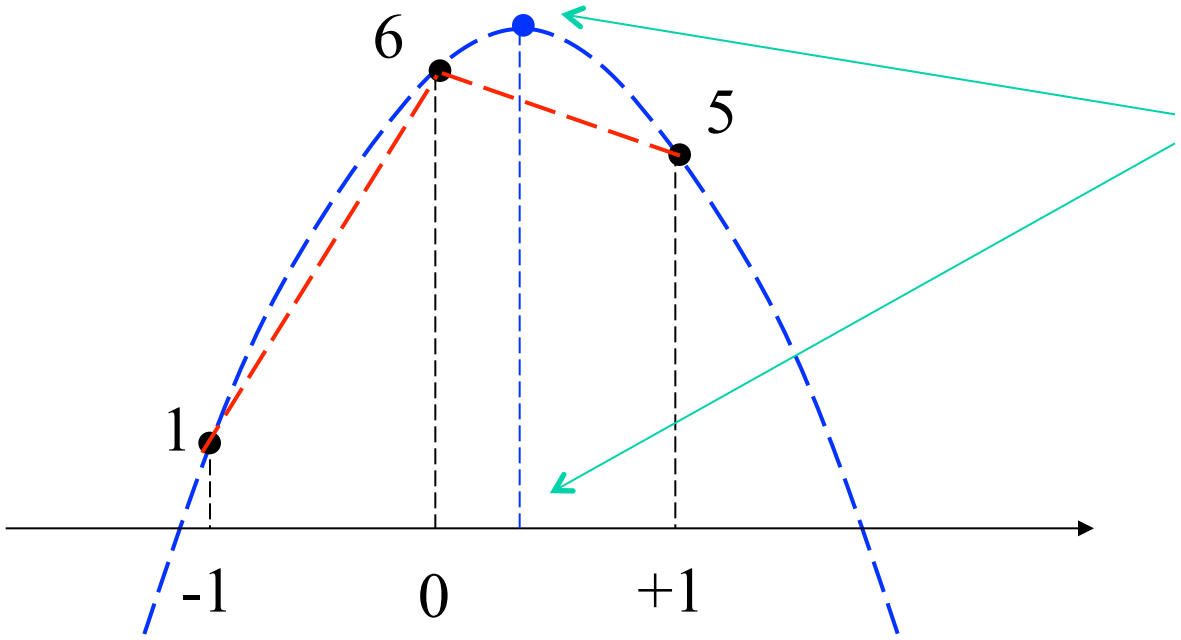
- **Reject key-points** with low contrast areas

- **Reject key-points** on edges

2. Accurate keypoint localization:

Fit a 3D quadratic function

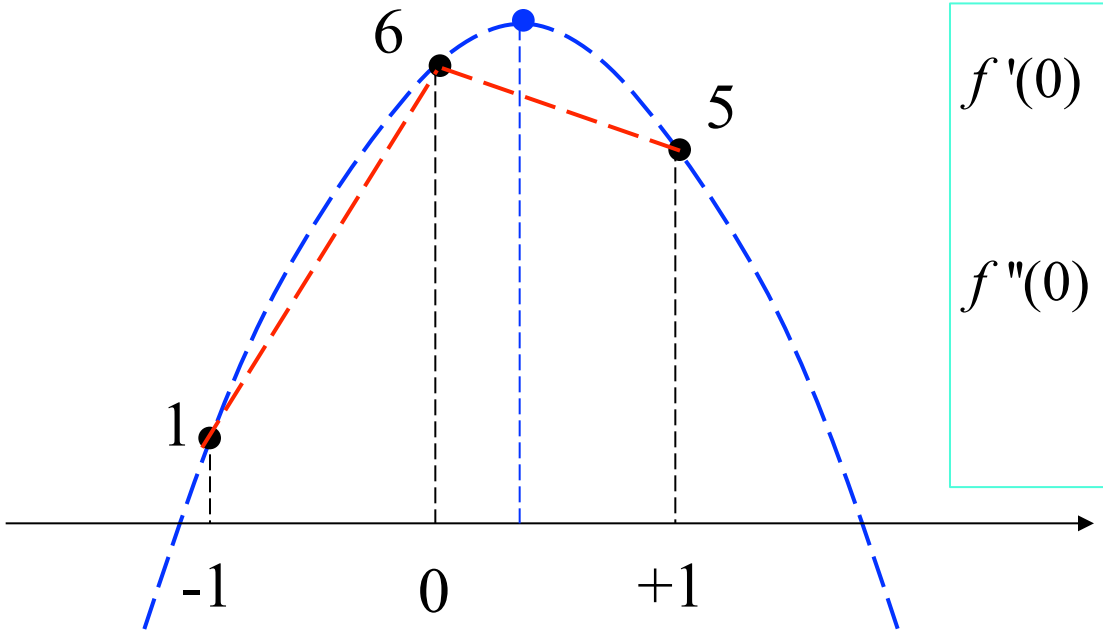
❖Idea:



2. Accurate keypoint localization:

Fit a 3D quadratic function

❖Idea:



$$f'(0) = \frac{f(1) - f(-1)}{2} = \frac{5-1}{2}$$

$$= 2$$

$$f''(0) = f(1) - 2f(0) + f(-1)$$

$$= 5 - 2*6 + 1$$

$$= -6$$

2. Accurate keypoint localization:

Fit a 3D quadratic function

- Idea:
  - Use Taylor Expansion:

$$f(x) \approx \hat{f}(x) \; @f(0) + f'(0)x + \frac{f''(0)}{2}x^2$$

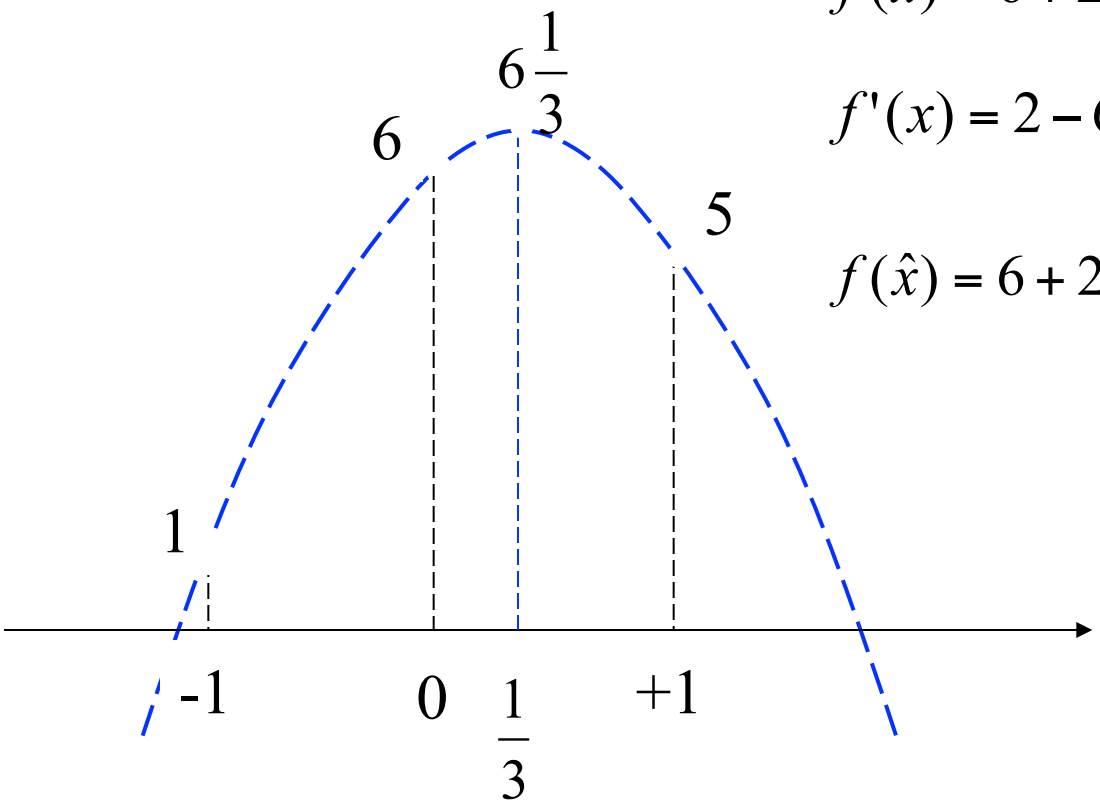
If Minimum/Maximum is at x then
x is the solution of the following equation:

$$f'(x) = 0$$

Minimum/Maximum is: $f(x)$

2. Accurate key-point localization:

Fit a 3D quadratic function



$$f(x) \approx 6 + 2x + \frac{-6}{2}x^2 = 6 + 2x - 3x^2$$

$$f'(x) = 2 - 6x = 0 \longrightarrow \hat{x} = \frac{1}{3}$$

$$f(\hat{x}) = 6 + 2 \cdot \frac{1}{3} - 3 \cdot \left(\frac{1}{3}\right)^2 = 6\frac{1}{3}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = D(x_0, y_0, k_0) + \frac{\partial D^T}{\partial \mathrm{X}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T\frac{\partial^2 D}{\partial \mathrm{X}^2}\mathrm{X}$$

Where, $\mathrm{X} = \begin{bmatrix} x \\ y \\ \sigma \end{bmatrix}$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = \underbrace{D(x_0, y_0, k_0)}_{\text{Scalar}} + \underbrace{\frac{\partial D^T}{\partial \mathrm{X}}}_{\text{Vector: 1×3}} \mathrm{X} + \frac{1}{2} \mathrm{X}^T \underbrace{\frac{\partial^2 D}{\partial \mathrm{X}^2}}_{\text{Matrix: 3×3}} \mathrm{X}$$

$$\frac{\partial D}{\partial \mathrm{X}} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} @\mathbf{g}$$

$$\frac{\partial^2 D}{\partial^2 \mathrm{X}} = \begin{bmatrix} \frac{\partial^2 D}{\partial^2 x} & \frac{\partial^2 D}{\partial x \partial y} & \frac{\partial^2 D}{\partial x \partial \sigma} \\ \frac{\partial^2 D}{\partial y \partial x} & \frac{\partial^2 D}{\partial^2 y} & \frac{\partial^2 D}{\partial y \partial \sigma} \\ \frac{\partial^2 D}{\partial \sigma \partial x} & \frac{\partial^2 D}{\partial \sigma \partial y} & \frac{\partial^2 D}{\partial^2 \sigma} \end{bmatrix} @A$$

2. Accurate key-point localization:

Fit a 3D quadratic function

How to compute derivatives, for example, 2D function f(x,y):

$f_{-1,1}$	$f_{0,1}$	$f_{1,1}$
$f_{-1,0}$	$f_{0,0}$	$f_{1,0}$
$f_{-1,-1}$	$f_{0,-1}$	$f_{1,-1}$

f(x,y)

Derivatives:

$$\frac{\partial f}{\partial x} = (f_{1,0} - f_{-1,0})/2$$

$$\frac{\partial f}{\partial y} = (f_{0,1} - f_{0,-1})/2$$

$$\frac{\partial^2 f}{\partial x^2} = f_{1,0} - 2f_{0,0} + f_{-1,0}$$

$$\frac{\partial^2 f}{\partial y^2} = f_{0,1} - 2f_{0,0} + f_{0,-1}$$

$$\frac{\partial^2 f}{\partial x \partial y} = (f_{-1,-1} - f_{-1,1} - f_{1,-1} + f_{1,1})/4$$

2. Accurate key-point localization:

Fit a 3D quadratic function

Derivatives:

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

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

$$\frac{\partial^2 D}{\partial x \partial y} = \frac{[D(x+1,y+1,k) - D(x+1,y-1,k)] - [D(x-1,y+1,k) - D(x-1,y-1,k)]}{4}$$

$$\frac{\partial^2 D}{\partial x \partial \sigma} = \frac{[D(x+1,y,k+1) - D(x+1,y,k-1)] - [D(x-1,y,k+1) - D(x-1,y,k-1)]}{4}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

Derivatives:

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

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

$$\frac{\partial^2 D}{\partial y \partial x} = \frac{[D(x+1, y+1, k) - D(x-1, y+1, k)] - [D(x+1, y-1, k) - D(x-1, y-1, k)]}{4}$$

$$\frac{\partial^2 D}{\partial y \partial \sigma} = \frac{[D(x, y+1, k+1) - D(x, y+1, k-1)] - [D(x, y-1, k+1) - D(x, y-1, k-1)]}{4}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

Derivatives:

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

$$\frac{\partial^2 D}{\partial \sigma^2} = \frac{D(x, y, k+1) - 2D(x, y, k) + D(x, y, k-1)}{1}$$

$$\frac{\partial^2 D}{\partial \sigma \partial y} = \frac{[D(x+1, y, k+1) - D(x-1, y, k+1)] - [D(x+1, y, k-1) - D(x-1, y, k-1)]}{4}$$

$$\frac{\partial^2 D}{\partial \sigma \partial y} = \frac{[D(x, y+1, k+1) - D(x, y-1, k+1)] - [D(x, y+1, k-1) - D(x, y-1, k-1)]}{4}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = D(x_0, y_0, k_0) + \frac{\partial D^T}{\partial \mathrm{X}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T\frac{\partial^2 D}{\partial \mathrm{X}^2}\mathrm{X}$$

Finding maximum/minimum by solving the following equation:

$$\frac{\partial \hat{D}}{\partial \mathrm{X}} = 0$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = D(x_0, y_0, k_0) + \frac{\partial D^T}{\partial \mathrm{X}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \frac{\partial^2 D}{\partial \mathrm{X}^2}\mathrm{X}$$

$$= D(x_0, y_0, k_0) + \mathbf{g}^{\mathrm{T}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \mathrm{A}\mathrm{X}$$

Question:

How to compute derivative of the following forms?

$$h_1(X) = \mathbf{g}^T X$$

$$h_2(X) = X^T A X$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$h_1(\mathbf{x}) = \mathbf{g}^{\mathrm{T}}\mathbf{x}$$

$$= \begin{pmatrix} g_1 & \mathrm{L} & g_n \end{pmatrix} \begin{pmatrix} x_1 \\ \mathrm{M} \\ x_n \end{pmatrix}$$

$$= \sum_{i=1}^{n} g_i x_i$$

$$\frac{\partial h_1}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial h_1}{\partial x_1} \\ \mathrm{M} \\ \frac{\partial h_1}{\partial x_n} \end{pmatrix} = \begin{pmatrix} g_1 \\ \mathrm{M} \\ g_n \end{pmatrix} = \mathbf{g}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$h_2(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}$$

$$= \begin{pmatrix} x_1 & \mathrm{L} & x_n \end{pmatrix} \begin{pmatrix} a_{11} & \mathrm{L} & a_{1n} \\ \mathrm{M} & \mathrm{O} & \mathrm{M} \\ a_{n1} & \mathrm{L} & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \mathrm{M} \\ x_n \end{pmatrix}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\frac{\partial h_2}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial h_2}{\partial x_1} \\ \mathrm{M} \\ \frac{\partial h_2}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} a_{i1} x_i + \sum_{j=1}^{n} a_{1j} x_j \\ \mathrm{M} \\ \sum_{i=1}^{n} a_{in} x_i + \sum_{j=1}^{n} a_{nj} x_j \end{pmatrix}$$

$$= \mathbf{A}^{\mathrm{T}}\mathbf{x} + \mathbf{A}\mathbf{x}$$

$$= (\mathbf{A}^{\mathrm{T}} + \mathbf{A})\mathbf{x}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = D(x_0, y_0, k_0) + \frac{\partial D^T}{\partial \mathrm{X}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \frac{\partial^2 D}{\partial \mathrm{X}^2}\mathrm{X}$$

$$= D(x_0, y_0, k_0) + \mathrm{g}^{\mathrm{T}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \mathrm{A}\mathrm{X}$$

Differentiate two sides

$$\frac{\partial \hat{D}}{\partial \mathrm{X}} = \mathbf{g} + \frac{1}{2}(A^T + A)X \qquad \textbf{(A is symmetric)}$$

$$= \mathbf{g} + AX$$

2. Accurate key-point localization:

Fit a 3D quadratic function

$$\hat{D}(\mathrm{X}) = D(x_0, y_0, k_0) + \frac{\partial D^T}{\partial \mathrm{X}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \frac{\partial^2 D}{\partial \mathrm{X}^2}\mathrm{X}$$

$$= D(x_0, y_0, k_0) + \mathrm{g}^{\mathrm{T}}\mathrm{X} + \frac{1}{2}\mathrm{X}^T \mathrm{A}\mathrm{X}$$

Maximum/Minimum is the solution of:

$$\mathbf{g} + AX = 0$$

$$\hat{\mathbf{X}} = -A^{-1}\mathbf{g}$$

2. Accurate key-point localization:

Fit a 3D quadratic function

Maximum/Minimum at:

$$\hat{X} = -\begin{bmatrix} \frac{\partial^2 D}{\partial^2 x} & \frac{\partial^2 D}{\partial x \partial y} & \frac{\partial^2 D}{\partial x \partial \sigma} \\ \frac{\partial^2 D}{\partial y \partial x} & \frac{\partial^2 D}{\partial^2 y} & \frac{\partial^2 D}{\partial y \partial \sigma} \\ \frac{\partial^2 D}{\partial \sigma \partial x} & \frac{\partial^2 D}{\partial \sigma \partial y} & \frac{\partial^2 D}{\partial^2 \sigma} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix}$$

Maximum/Minimum is:

$$\hat{D}(\hat{\mathrm{X}}) = D(x_0, y_0, k_0) + \mathrm{g}^{\mathrm{T}} \hat{\mathrm{X}} + \frac{1}{2} \hat{\mathrm{X}}^T \mathrm{A} \hat{\mathrm{X}}$$

2. Accurate key-point localization:

Reject key-points at low contrast areas

- If any dimension of X is larger than **0.5**: change sample point
- If $D(X) < 0.03$: Throw out the minimum/maximum

2. Accurate key-point localization:

Reject key-points on edges

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$ Hessian matrix at keypoint location

$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$

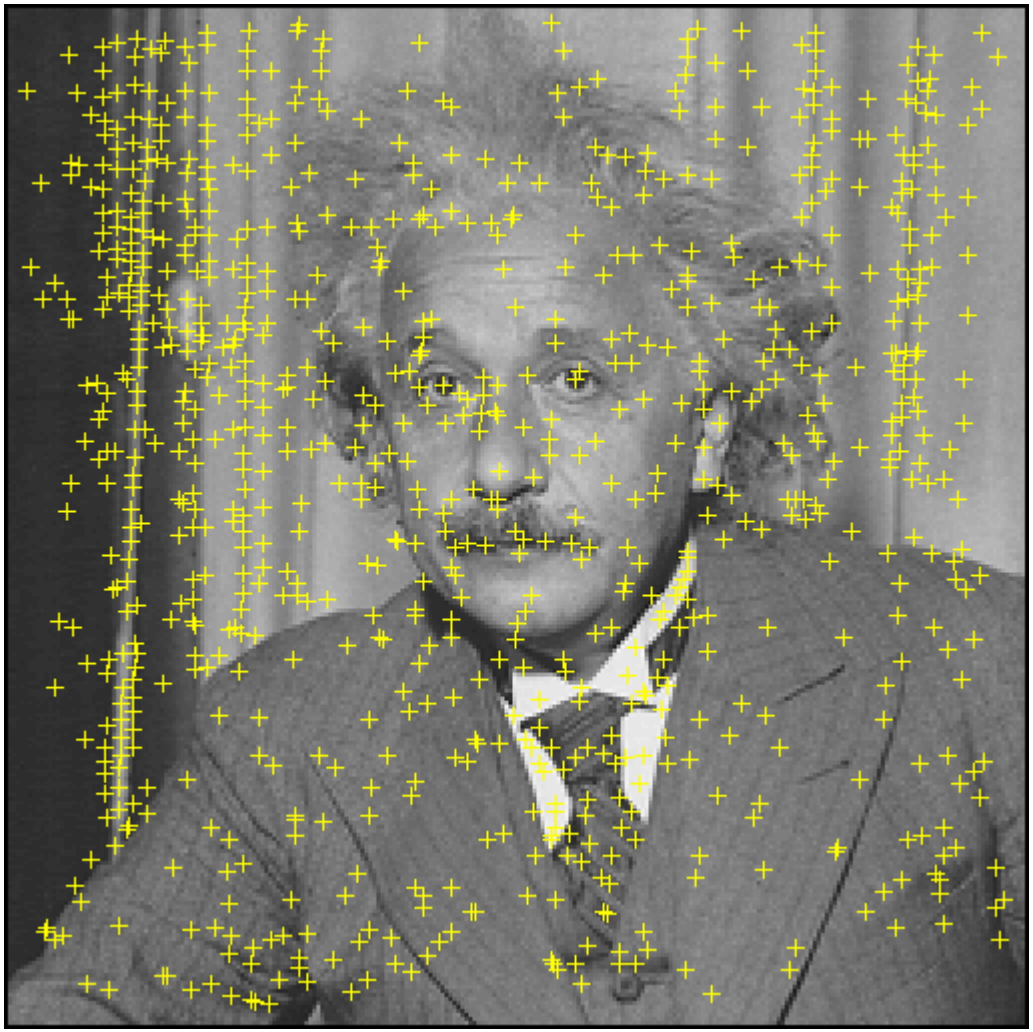
$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

Let $\alpha = r\beta$ $\quad \dfrac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} = \dfrac{(\alpha+\beta)^2}{\alpha\beta} = \dfrac{(r\beta+\beta)^2}{r\beta^2} = \dfrac{(r+1)^2}{r}$

Keep the points with $\dfrac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \dfrac{(r+1)^2}{r}.$ r=10

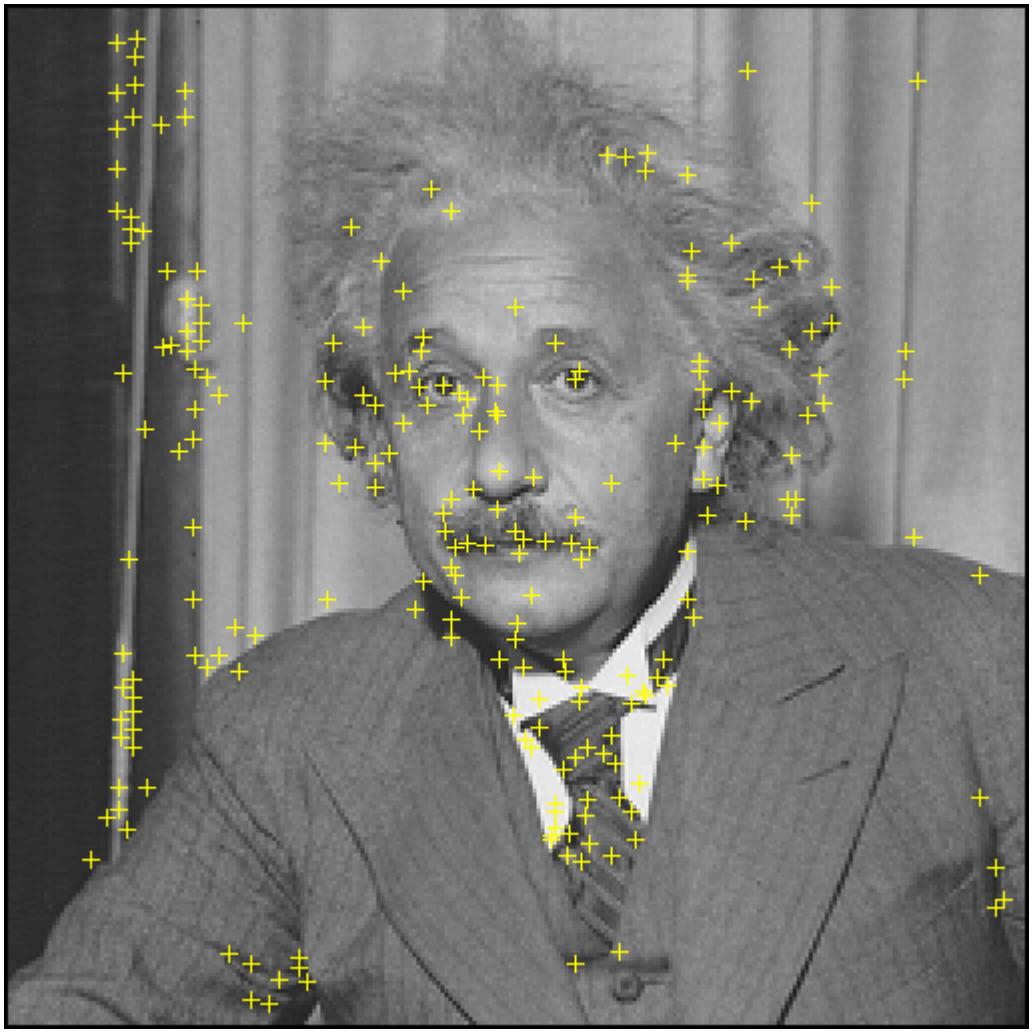
2. Accurate key-point localization:

Examples



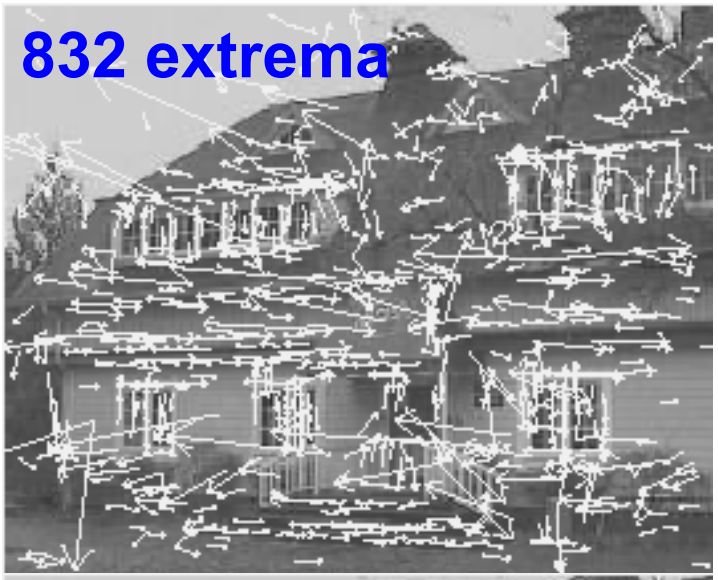
2. Accurate key-point localization:

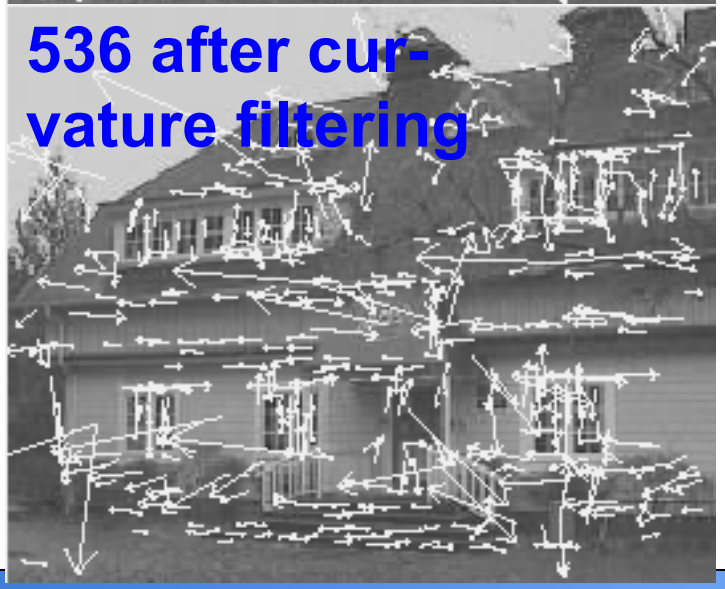
After removing points at low contrast areas or on edges



2. Accurate key-point localization:

Other examples





3. Orientation assignment

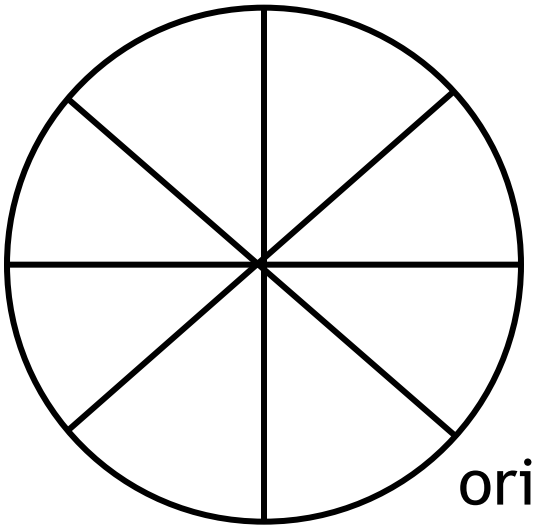
- By assigning a consistent orientation, the keypoint descriptor can be orientation invariant.
- For a keypoint, L is the Gaussian-smoothed image with the closest scale,

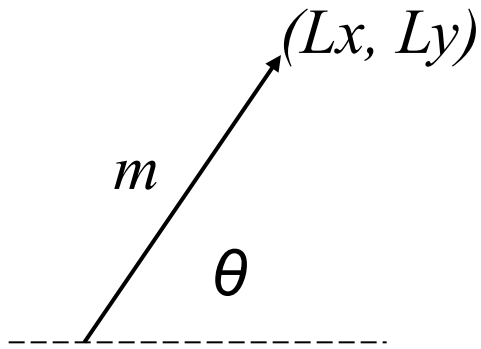
3. Orientation assignment

Compute orientation

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

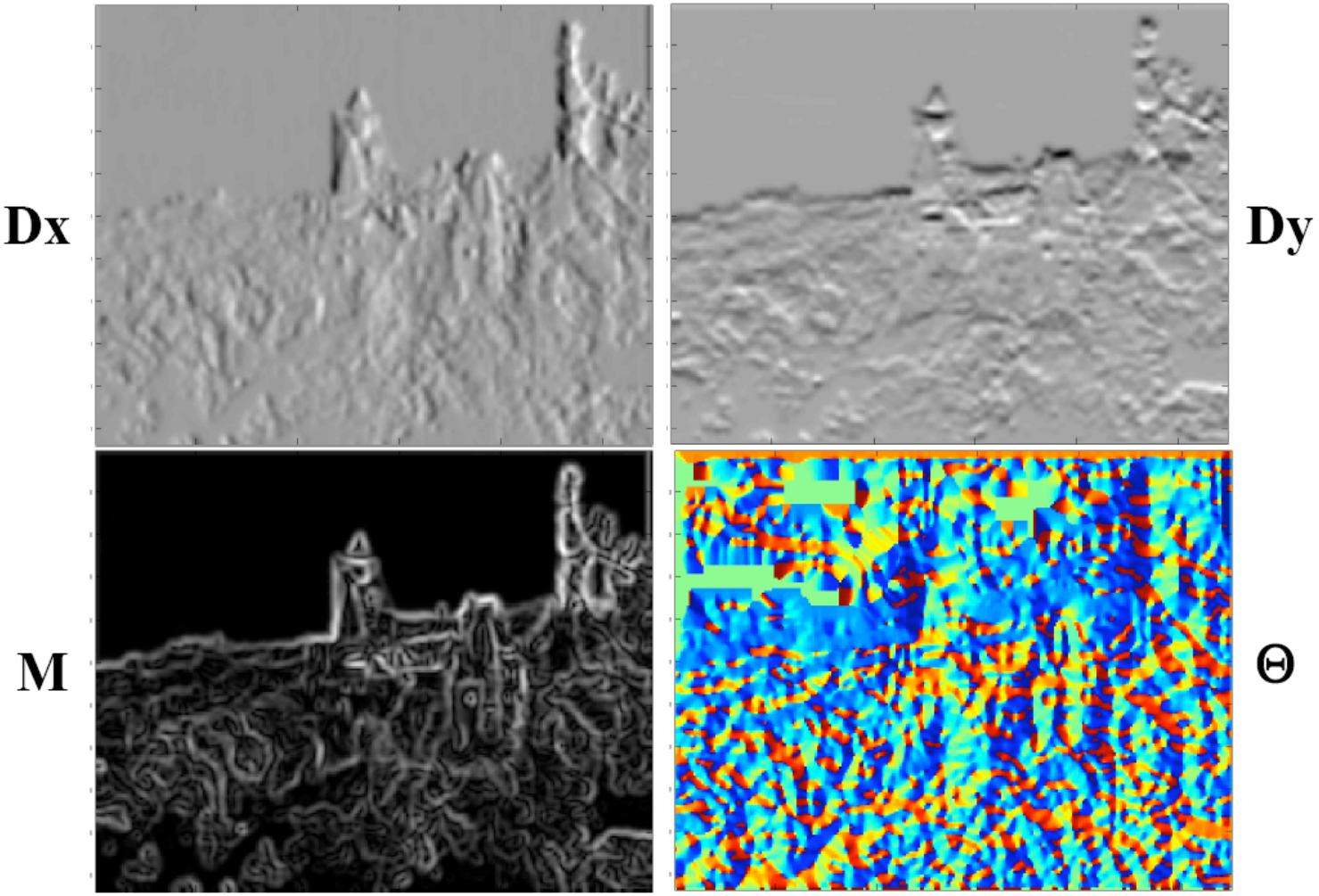




orientation histogram (36 bins)

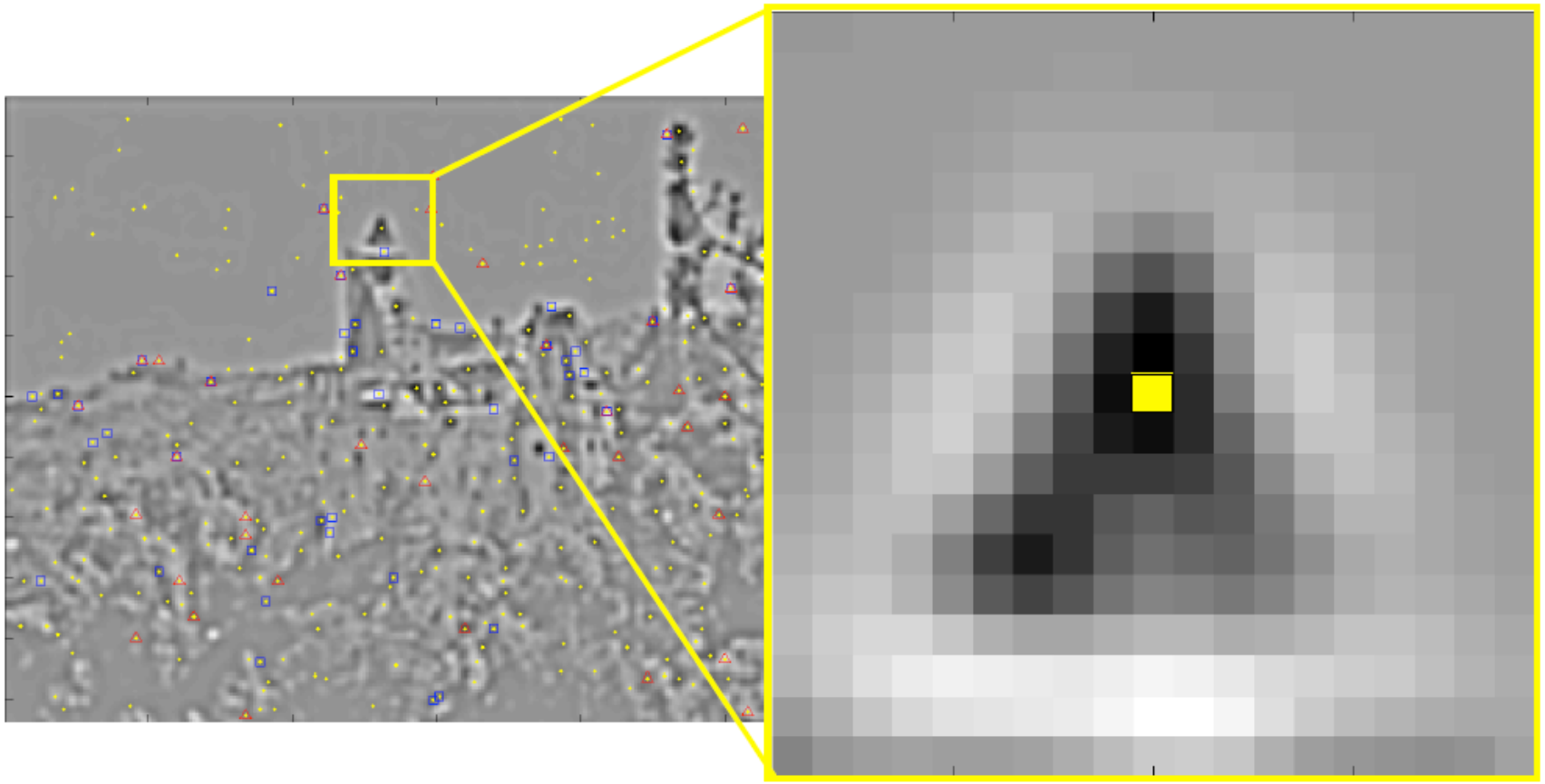
3. Orientation assignment

Examples



3. Orientation assignment

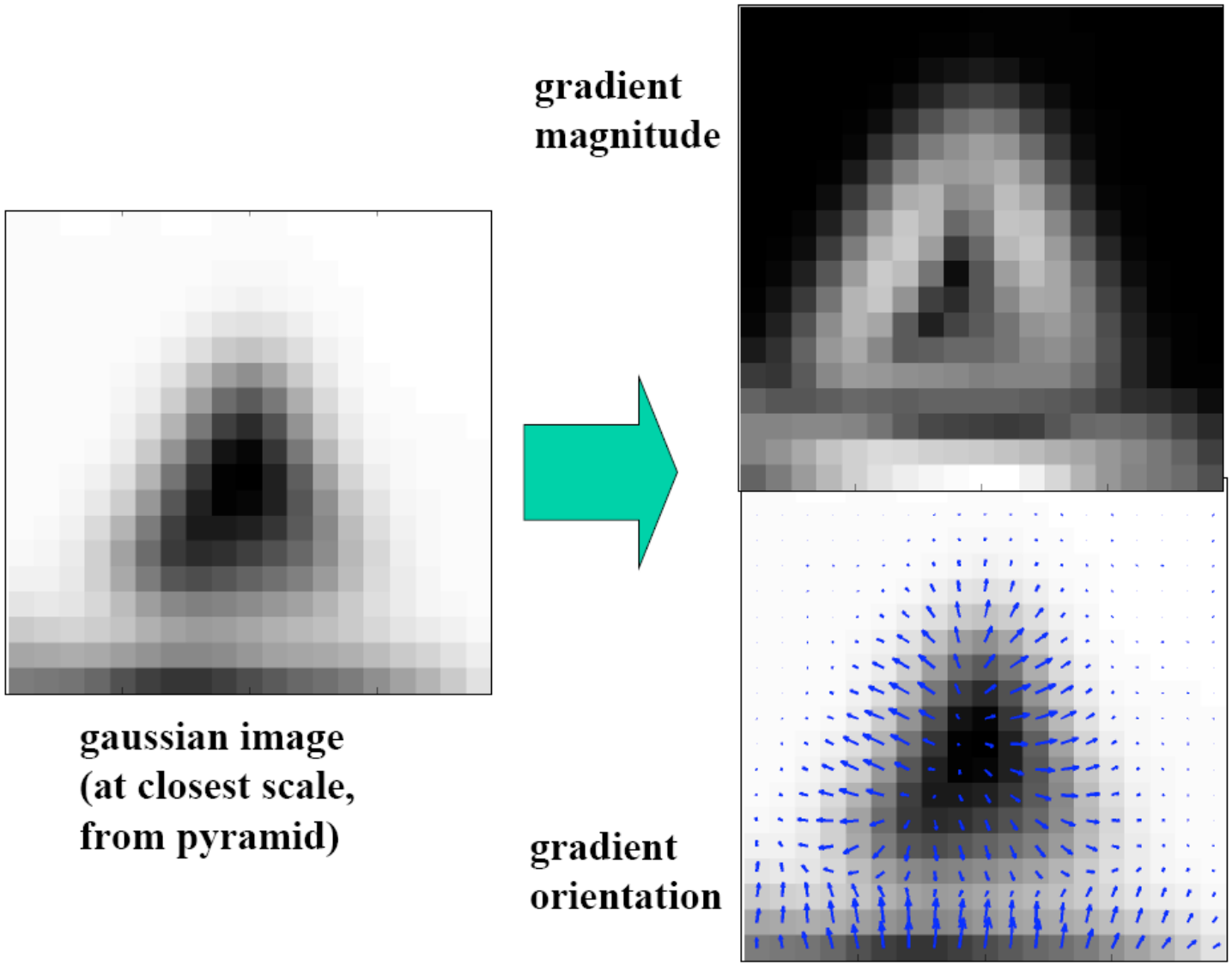
Demonstration



- Keypoint location = extrema location
- Keypoint scale is scale of the DOG image

3. Orientation assignment

Demonstration

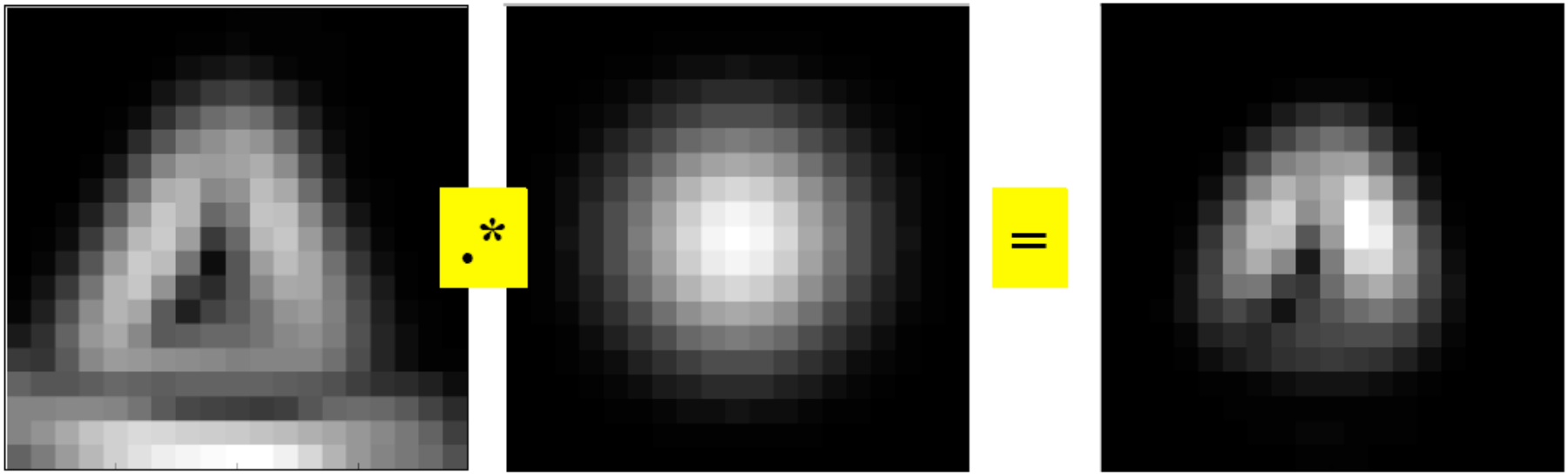


3. Orientation assignment

Determine the most important orientation

a. Weighting the gradient's magnitude of key-points, using Gaussian function.

σ = 1.5*scale of the key-point



gradient magnitude

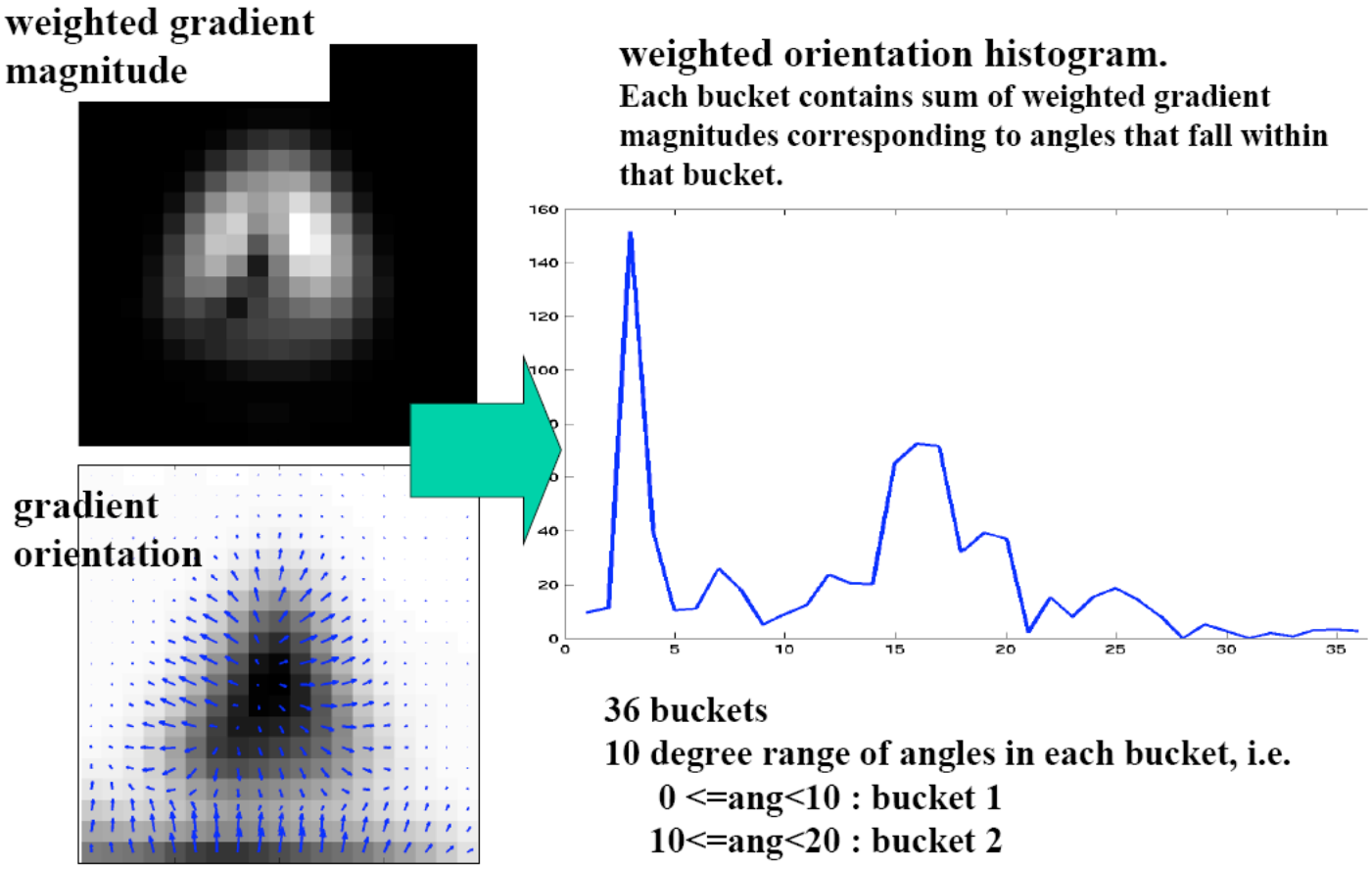
weighted by 2D gaussian kernel

weighted gradient magnitude

3. Orientation assignment

Determine the most important orientation

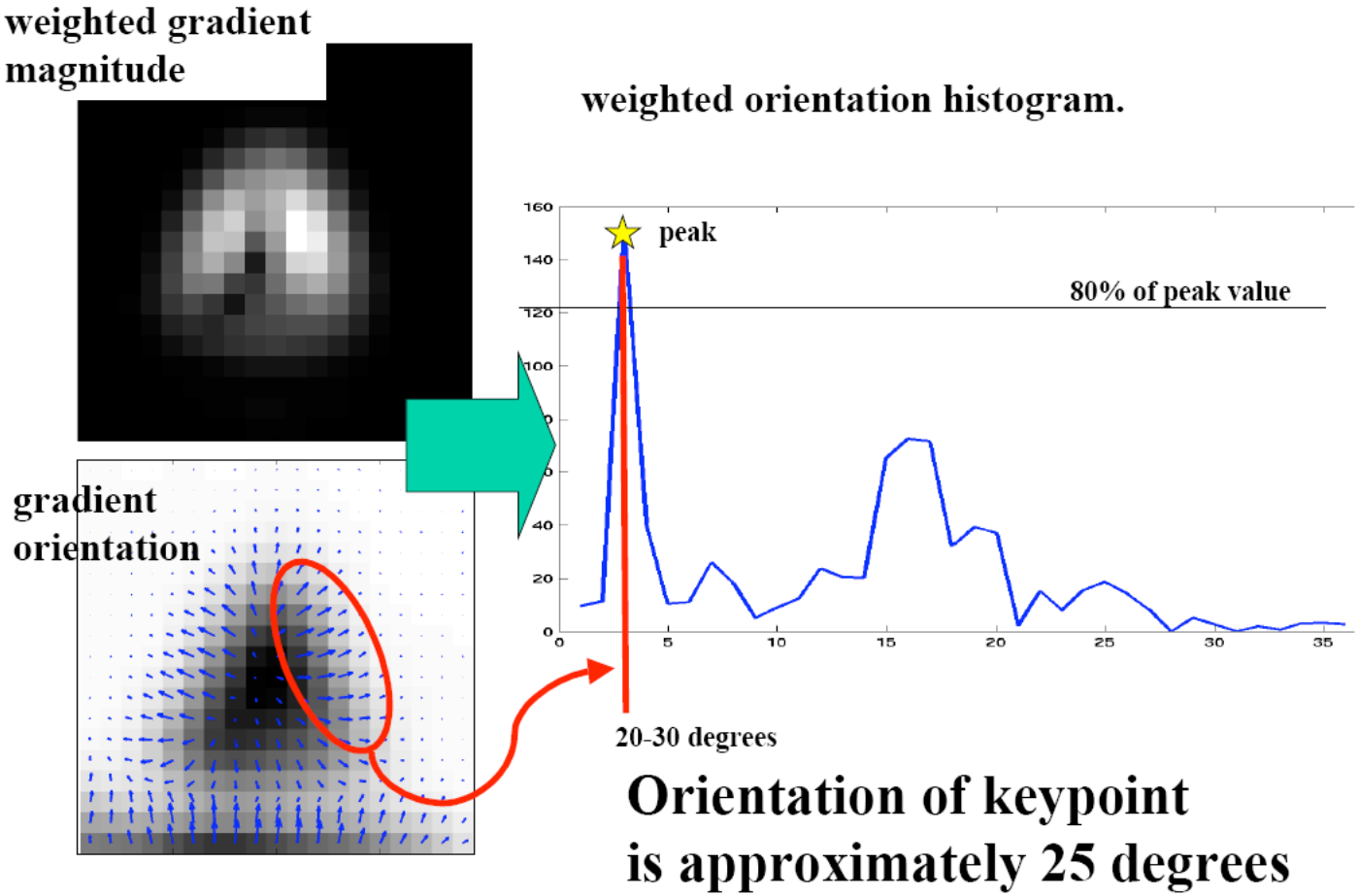
b. Computing the histogram of the gradient's magnitude of key-points.



3. Orientation assignment

Determine the most important orientation

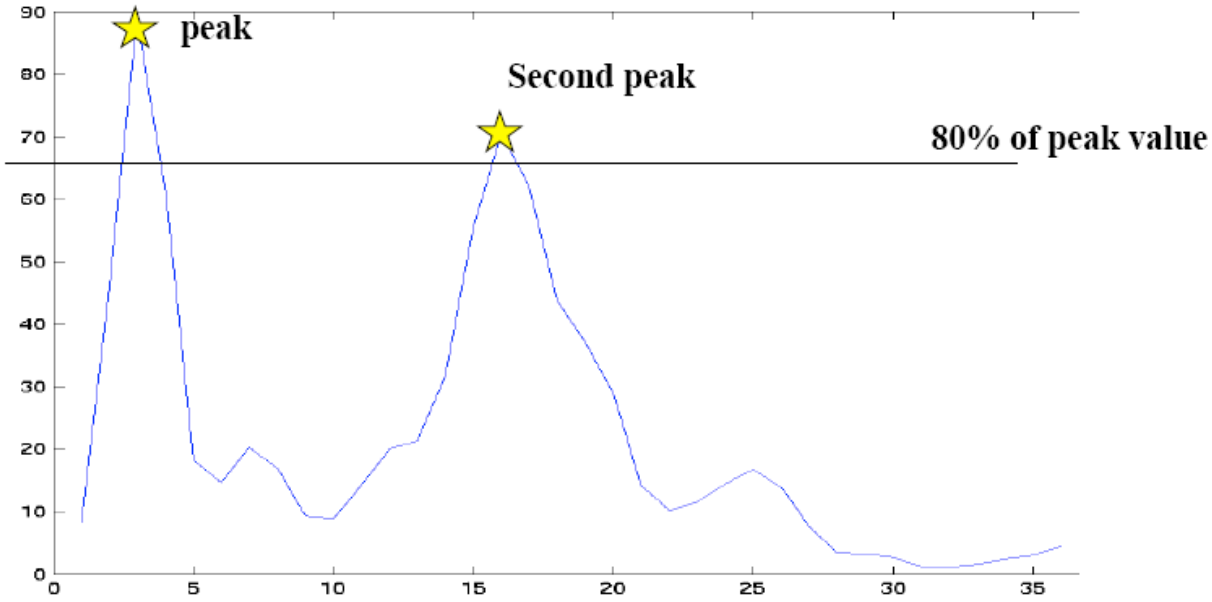
c. Finding the histogram's peaks. Using fitting to obtain accurate the peaks' position



3. Orientation assignment

Determine the most important orientation

There may be multiple orientations.



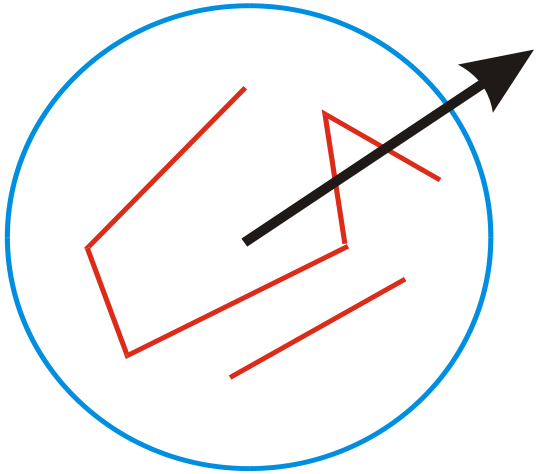
In this case, generate duplicate keypoints, one with orientation at 25 degrees, one at 155 degrees.

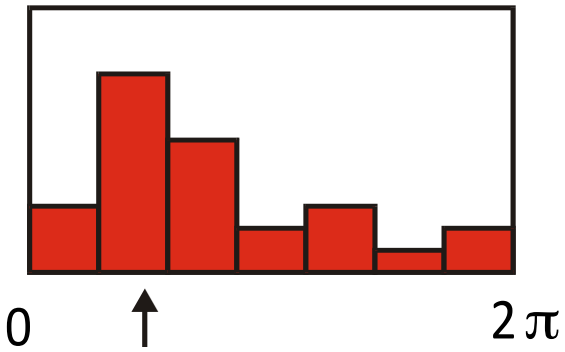
Design decision: you may want to limit number of possible multiple peaks to two.

3. Orientation assignment

Determine the most important orientation

Summarization

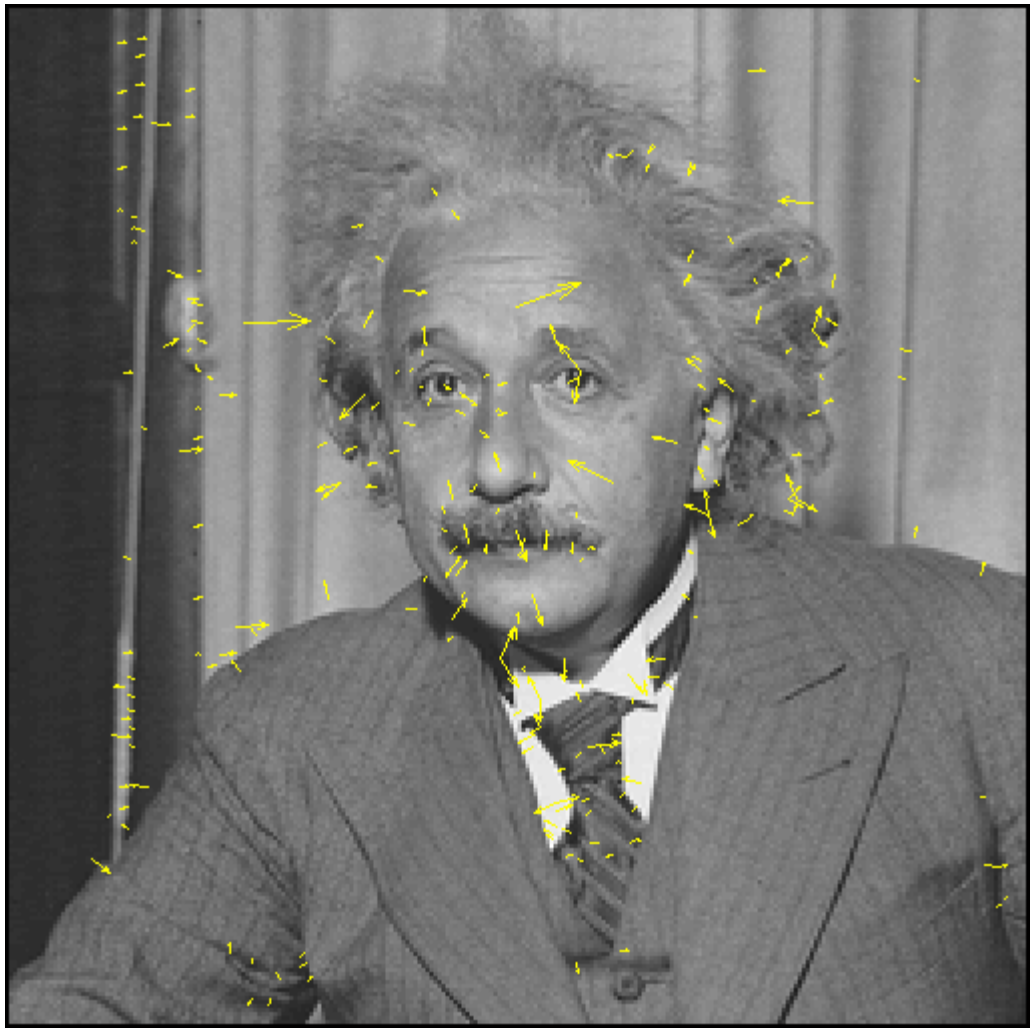




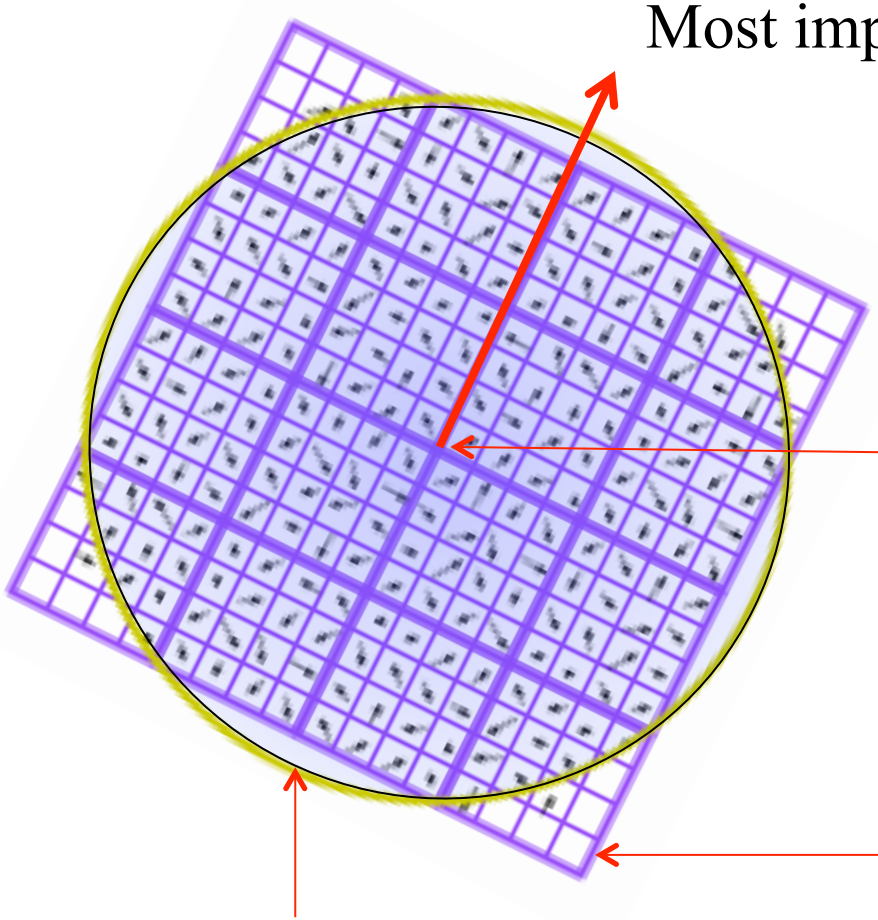
- 36-bin orientation histogram over 360°, weighted by m and 1.5*scale falloff
- Peak is the orientation
- Local peak within 80% creates multiple orientations
- About 15% has multiple orientations and they contribute a lot to stability

3. Orientation assignment

The most important orientation: Demo



4. Local image descriptor



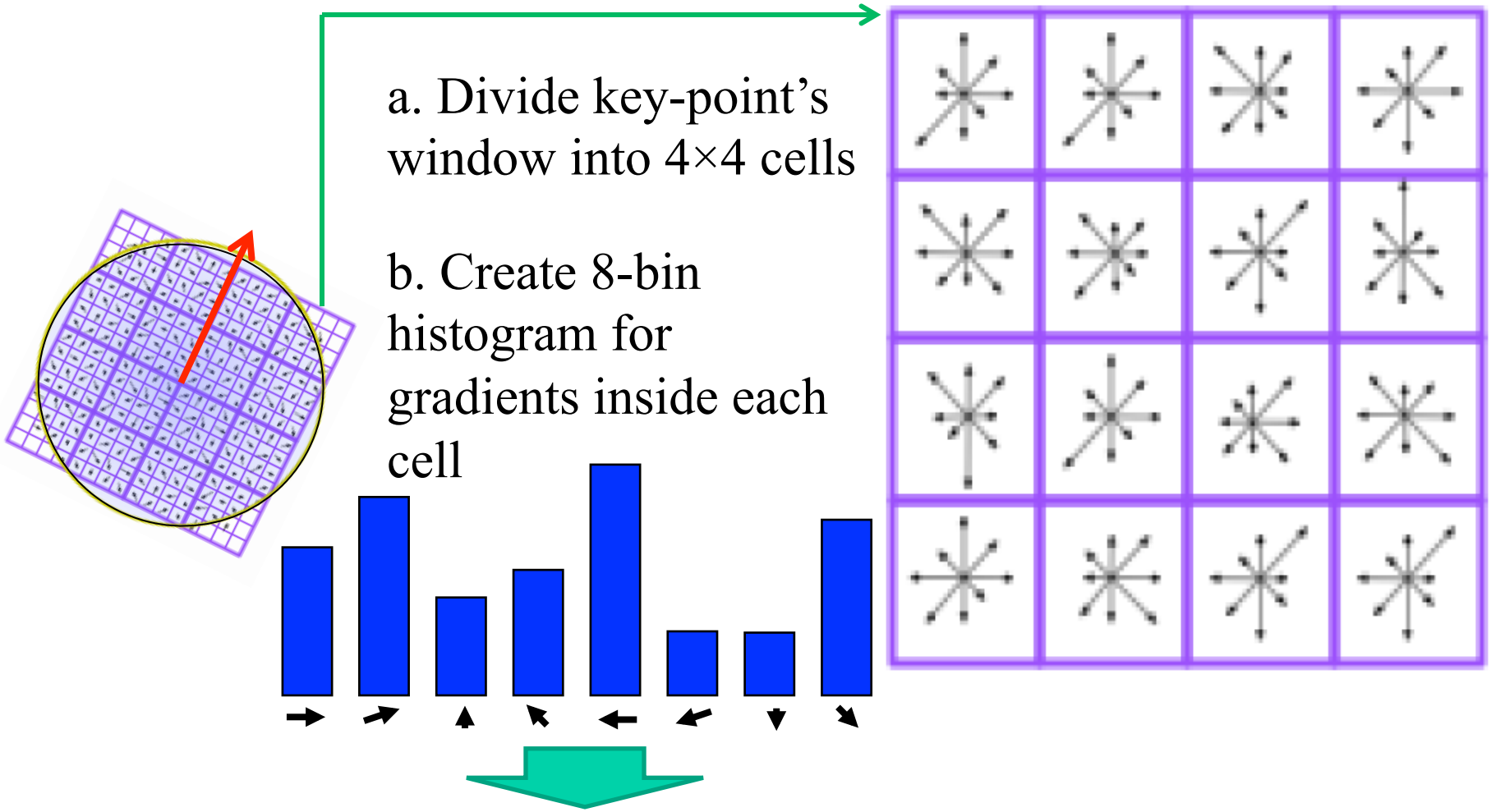
Most important orientation

A Key-point

Gradient sub-image
Size: 16×16 cells
Cell: pixel or resampled cell

Gaussian function for
weighting magnitudes
σ = 0.5 window' size

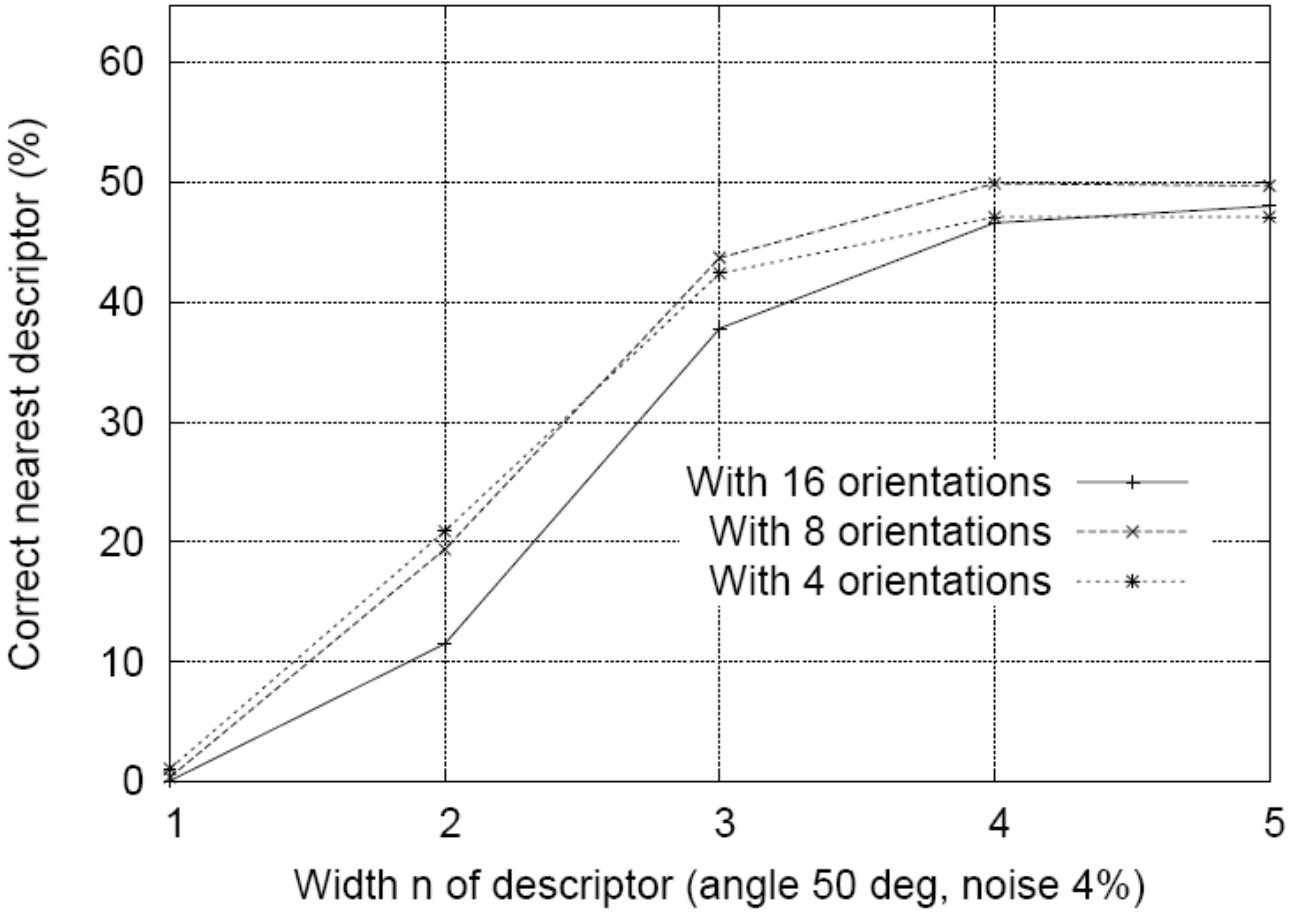
4. Local image descriptor



A feature vector has 4×4×8 = 128 components

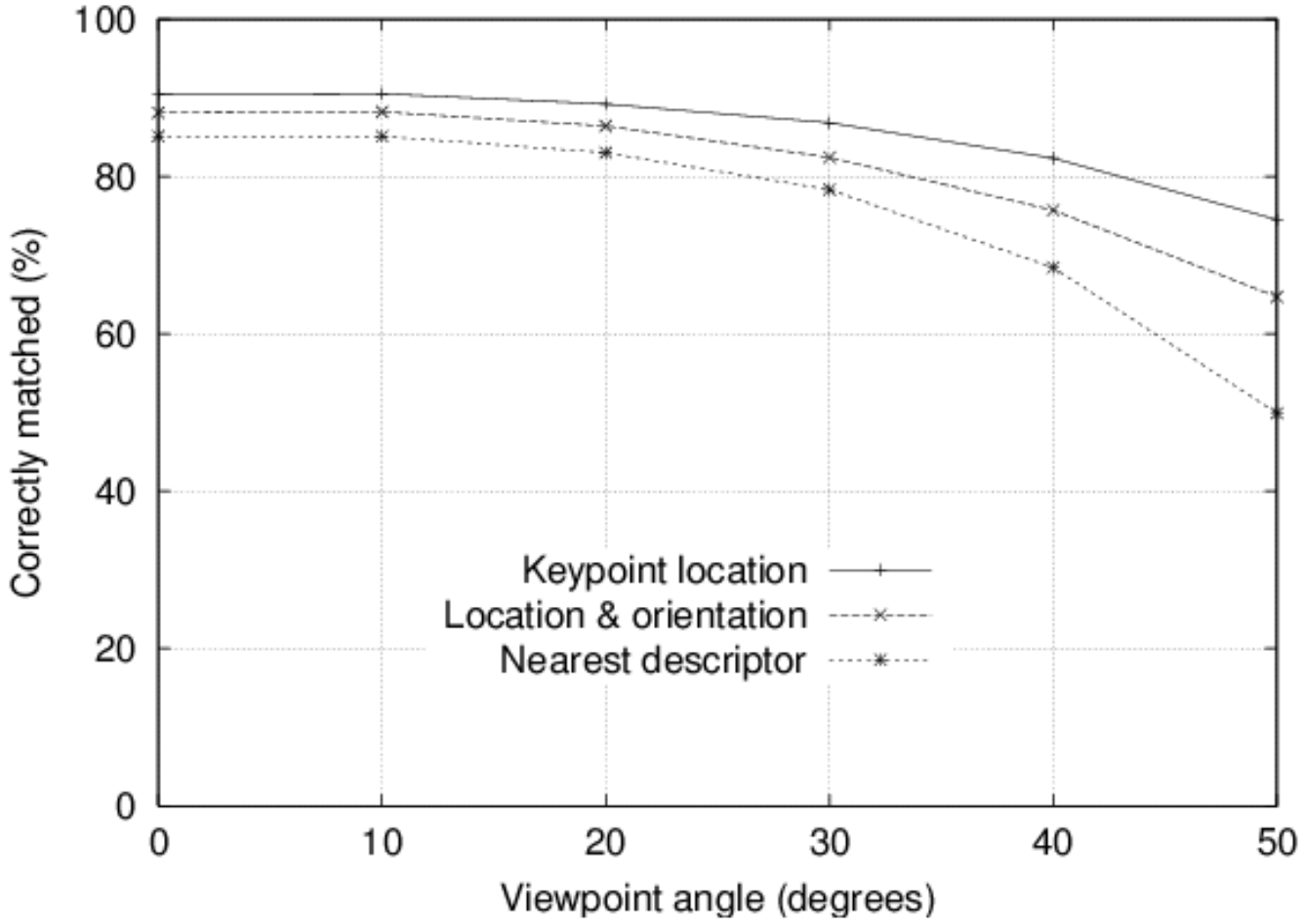
4. Local image descriptor

Why 4x4x8?



4. Local image descriptor

Sensitivity to affine change

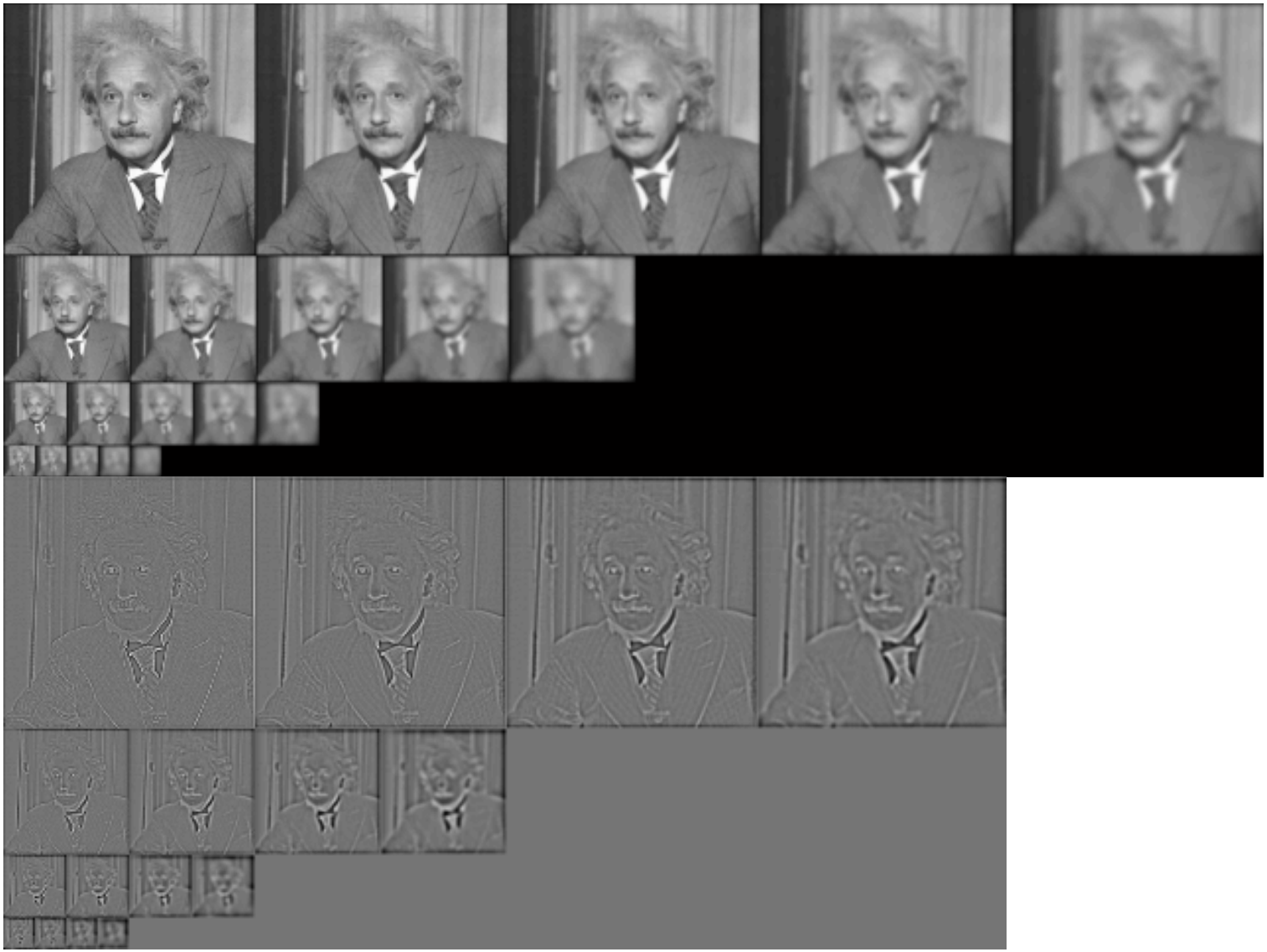


Feature matching

- for a feature x, he found the closest feature x_1 and the second closest feature x_2. If the distance ratio of $d(x, x_2)$ and $d(x, x_1)$ is smaller than 0.8, then it is accepted as a match.

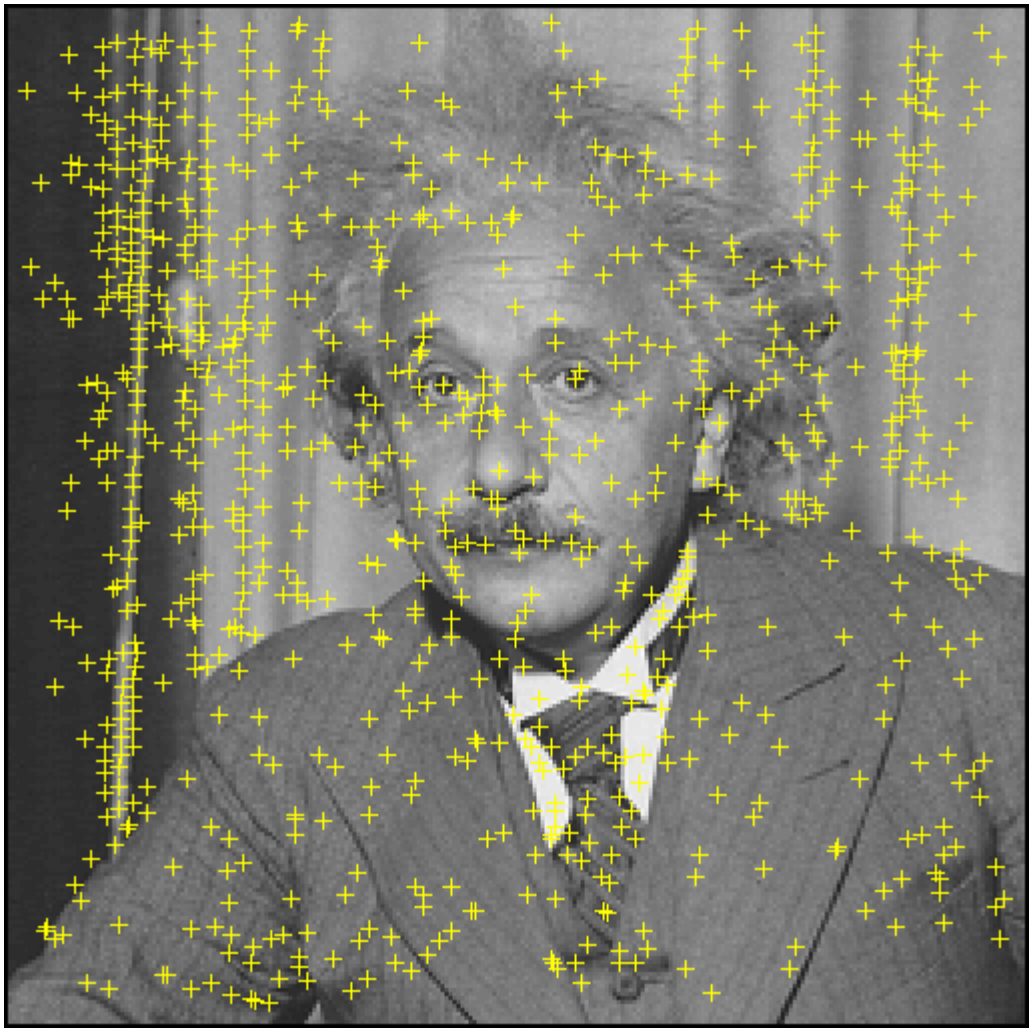
SIFT: Summary

Create scale-space



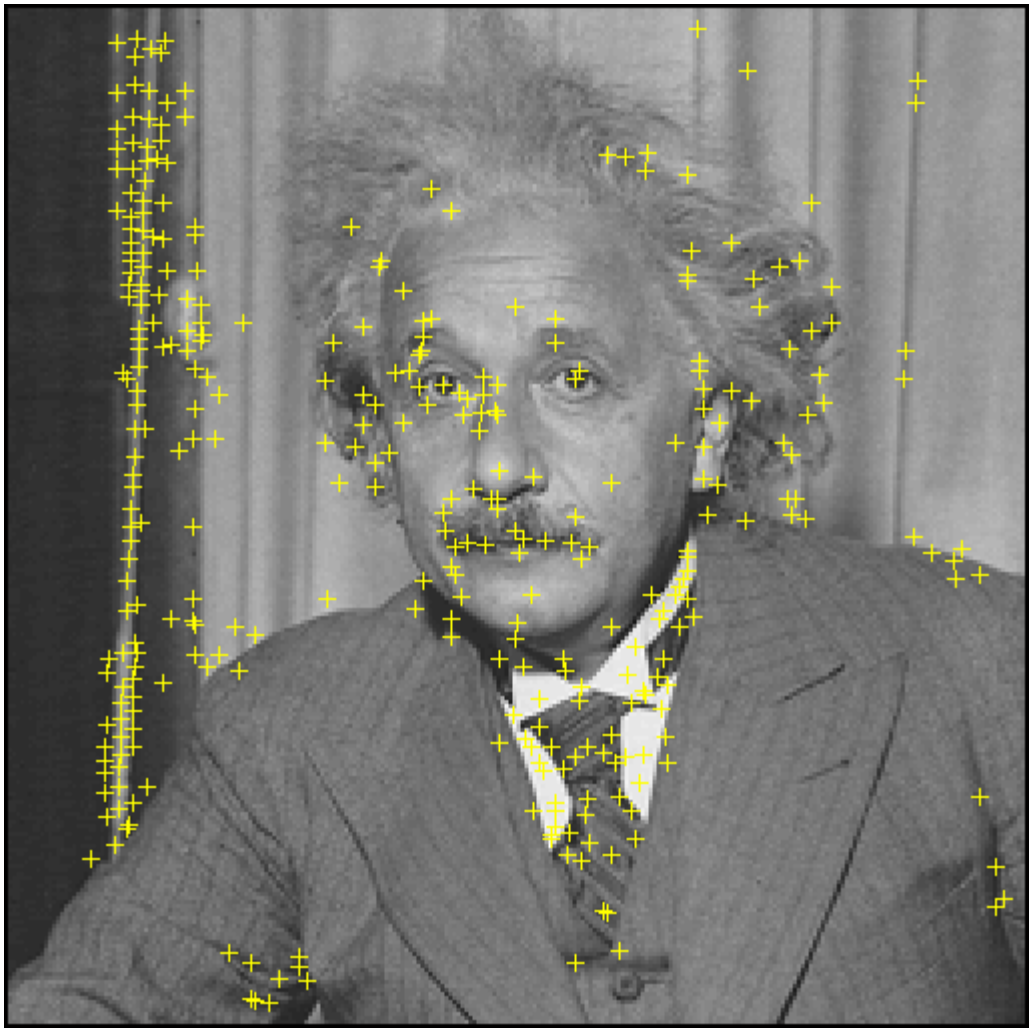
SIFT: Summary

Determine extrema



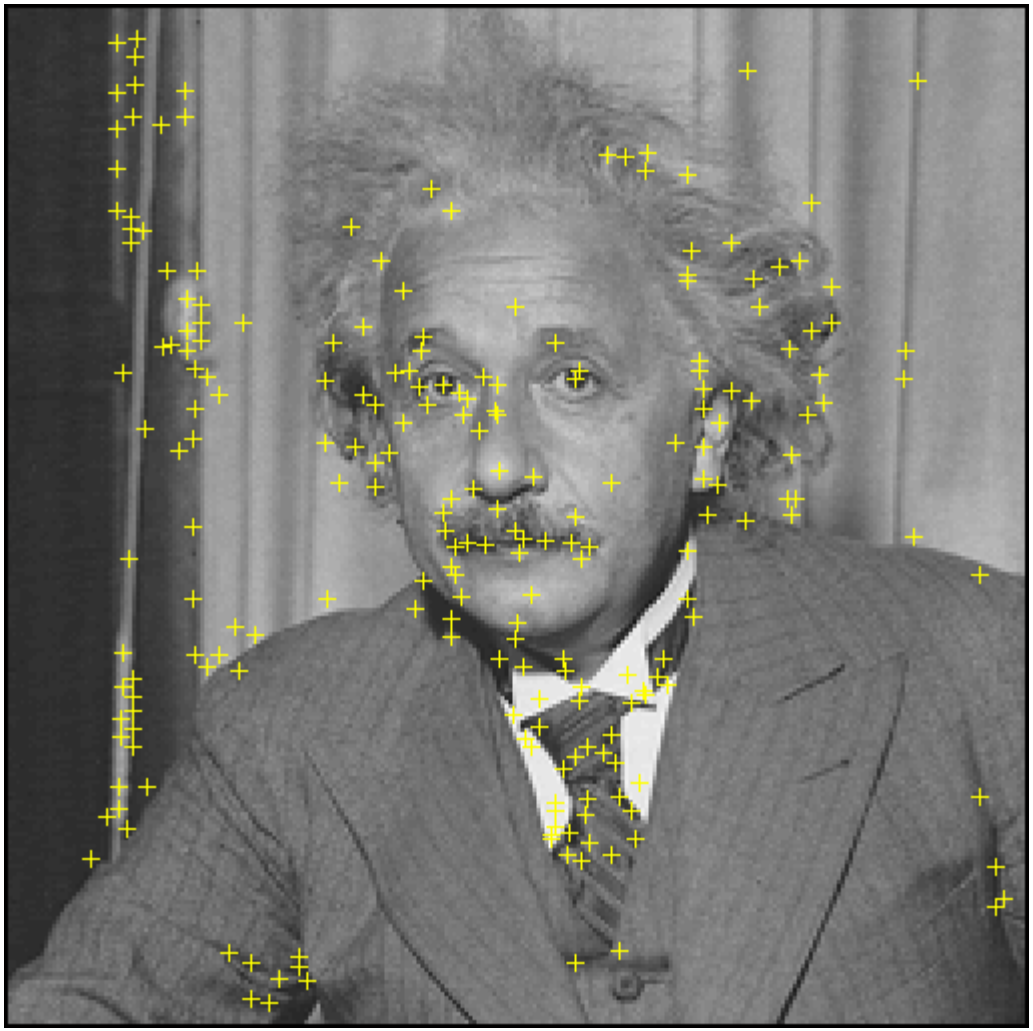
SIFT: Summary

Remove low contrast points



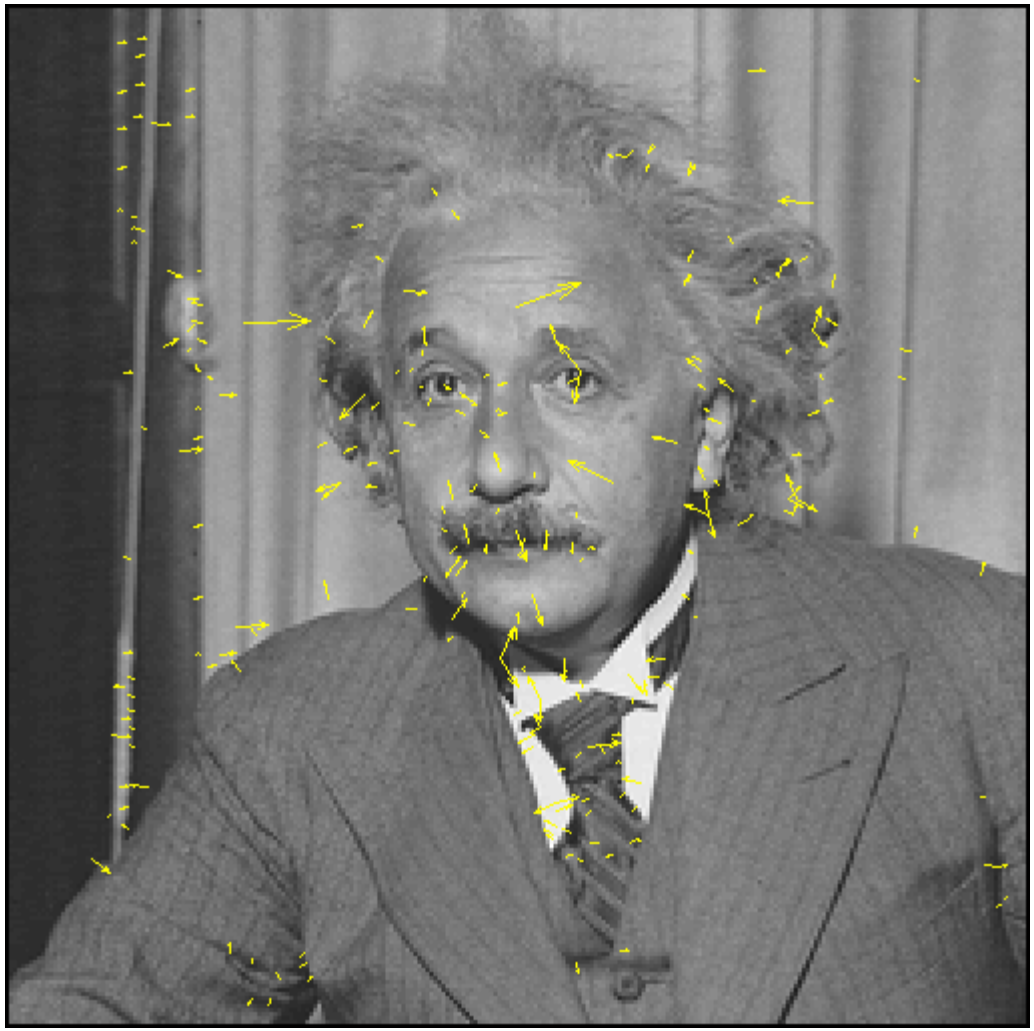
SIFT: Summary (Vfeat)

Remove edge points



SIFT: Summary

Generate descriptor



SURF
Speeded Up Robust Features

SURF: Speeded Up Robust Features

- Speed-up computations by fast approximation of (i) Hessian matrix and (ii) descriptor using "integral images".

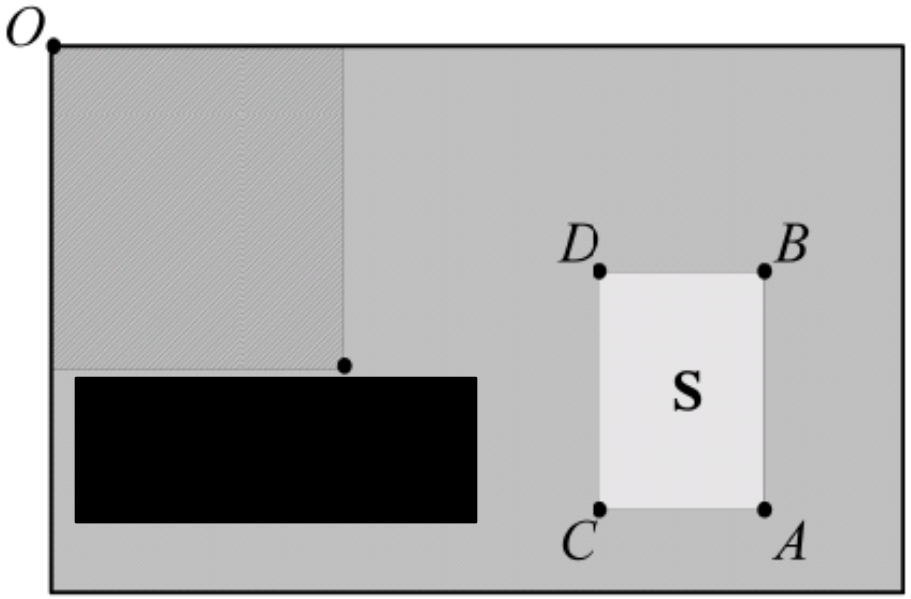
$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

- What is an "integral image"?

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, "SURF: Speeded Up Robust Features", **European Computer Vision Conference (ECCV),** 2006.

Integral Image

- The integral image $I_{\Sigma}(x,y)$ of an image I(x, y) represents the sum of all pixels in I(x,y) of a rectangular region **formed by (0,0) and (x,y).**



Using integral images, it takes only four array references to calculate the sum of pixels over a rectangular region of any size.

$$\mathbf{S}=A-B-C+D$$

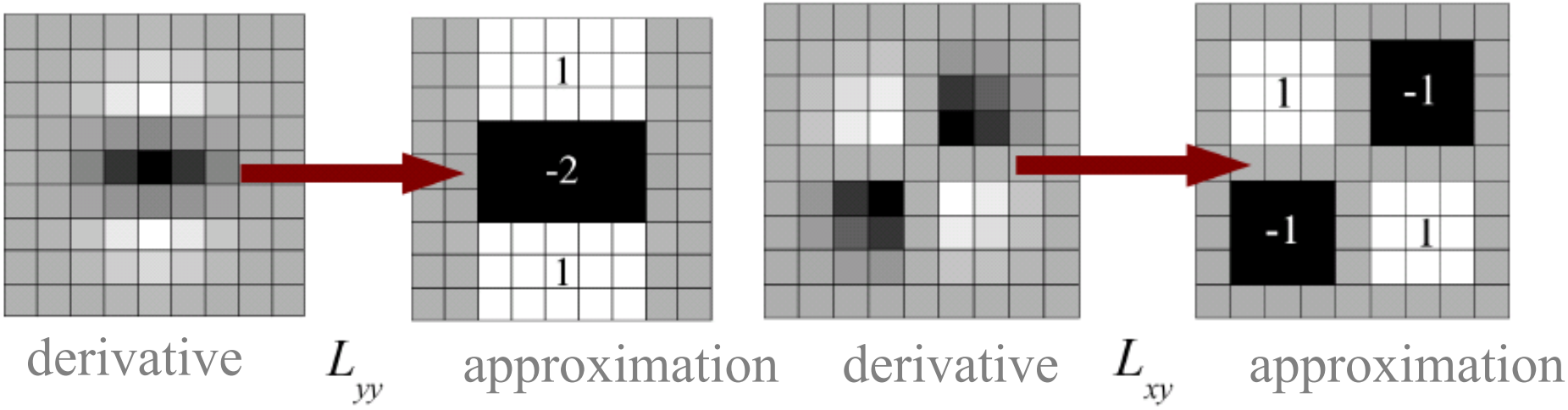
SURF: Speeded Up Robust Features (cont'd)

- Approximate L_{xx}, L_{yy}, and L_{xy} using box filters

- Those values can be computed very fast using integral images!

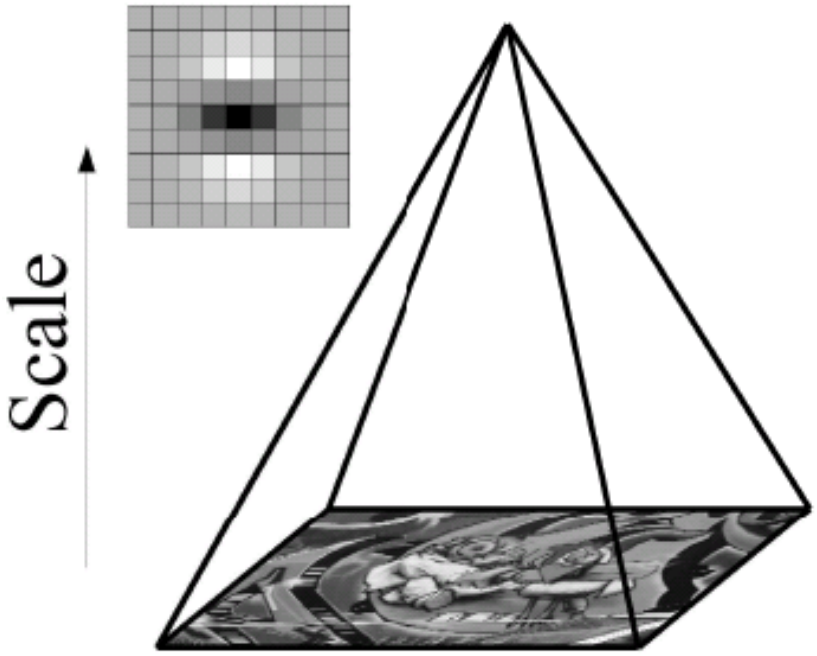
SURF: Speeded Up Robust Features (cont'd)

(box filters shown are 9 x 9
– good approximations for a Gaussian with σ=1.2)



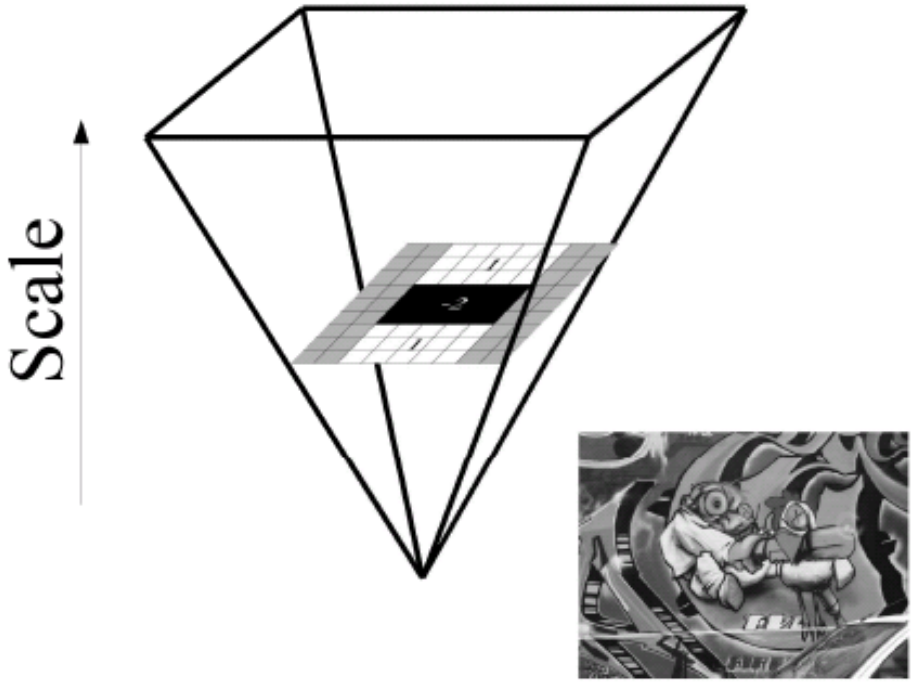
SURF

- In SIFT
  - Original images are repeatedly smoothed with a Gaussian
  - Then, sub-sampled to achieve a higher level of the pyramid.



SURF

- In SURF:
  - use filters of larger size on the original image.
  - SPEED-UP: use integral images to compute the filter's response



SURF vs. SIFT

- Approximation of H:

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

Using DoG

$$SIFT: \; H_{approx}^{SIFT} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

Using box filters

$$SURF: \; H_{approx}^{SURF} = \begin{bmatrix} \hat{L}_{xx} & \hat{L}_{xy} \\ \hat{L}_{yx} & \hat{L}_{yy} \end{bmatrix}$$

SURF

- Instead of using a different measure for selecting the location and scale of interest points (e.g., Hessian and DOG in SIFT), SURF uses the determinant of H_{approx}^{SURF} to find both.

- Determinant elements must be weighted to obtain a good approximation:

$$\det(H_{approx}^{SURF}) = \hat{L}_{xx}\hat{L}_{yy} - (0.9\hat{L}_{xy})^2$$

SURF

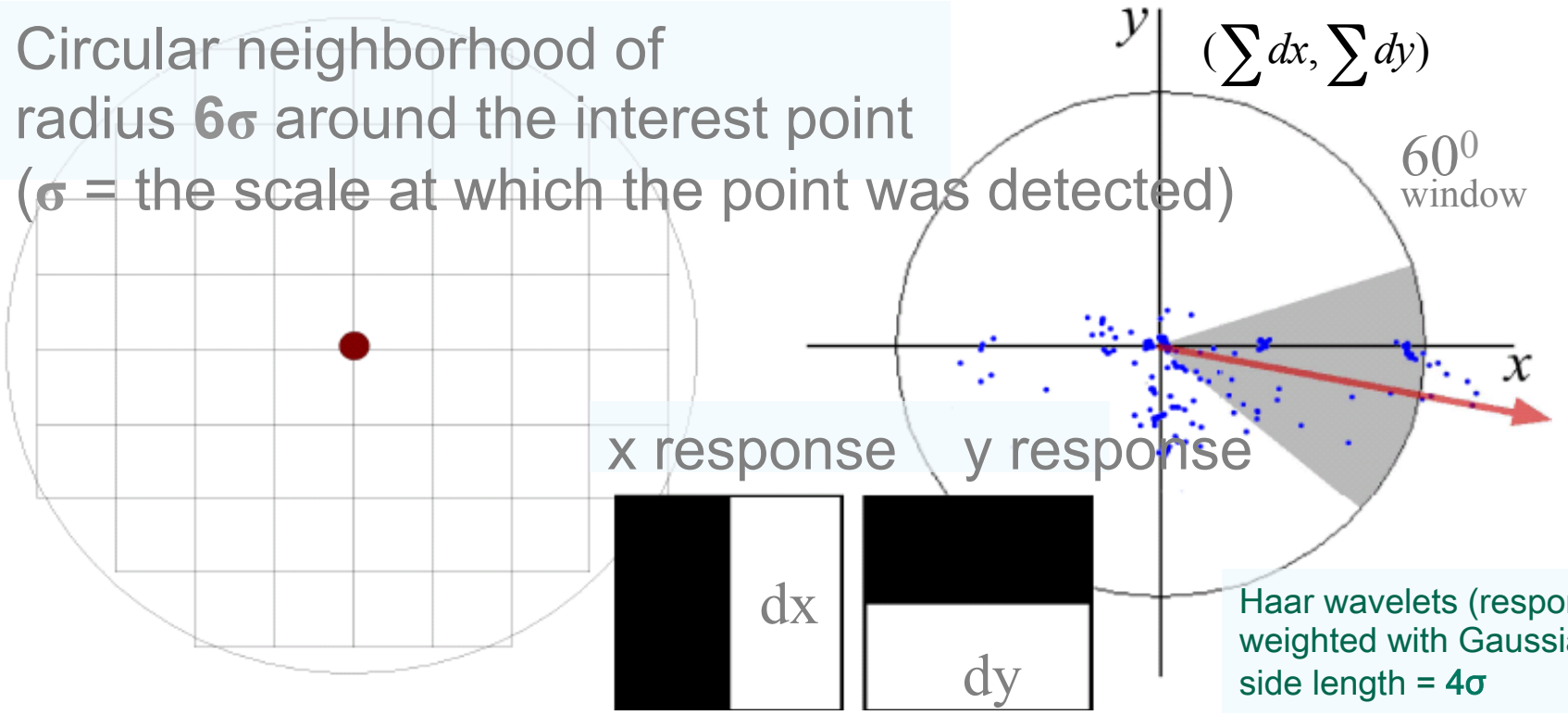
- Once interest points have been localized both in space and scale, the next steps are:

  (1) Orientation assignment

  (2) Keypoint descriptor

SURF

- Orientation assignment



Can be computed very fast using integral images!

SURF

- Orientation assignment

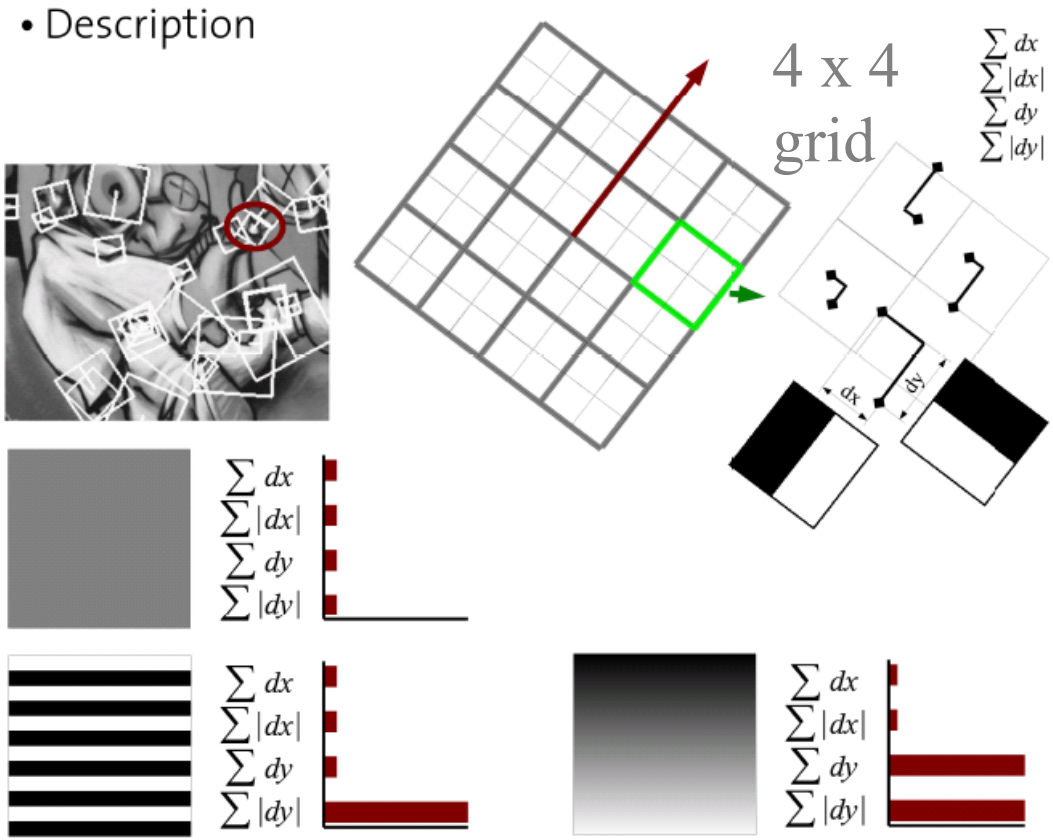
  - The image is convoluted with two first-order Haar wavelets.

  - The filter responses at certain sampling points around the keypoint are represented as a vector in a two-dimensional space.

  - A rotating window of 60^0 is used to sum up all vectors within its range, and the longest resulting vector determines the orientation.

SURF

- Keypoint descriptor (square region of size 20σ)



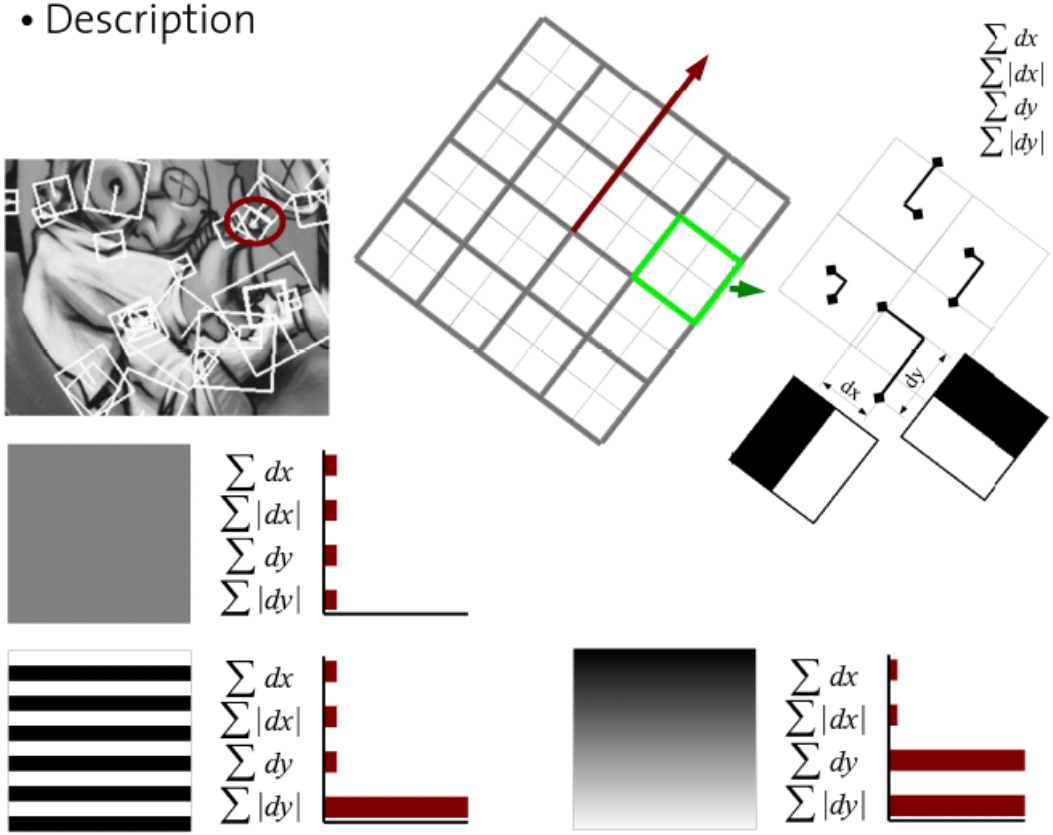
- Sum the response over each sub-region for dx and dy separately.
- To bring in information about the polarity of the intensity changes, extract the sum of absolute value of the responses too.

Feature vector size:

4 x 16 = 64

SURF

• Description



❖SURF-128

- The sum of d_x and $|d_x|$ are computed separately for points where $d_y < 0$ and $d_y > 0$
- Similarly for the sum of d_y and $|d_y|$
- **More discriminatory!**

SURF: Speeded Up Robust Features

- Has been reported to be 3 times faster than SIFT.
- Less robust to illumination and viewpoint changes compared to SIFT.

K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", **IEEE Transactions on Pattern Analysis and Machine Intelligence**, vol. 27, no. 10, pp. 1615-1630, 2005.

SIFT extensions

PCA

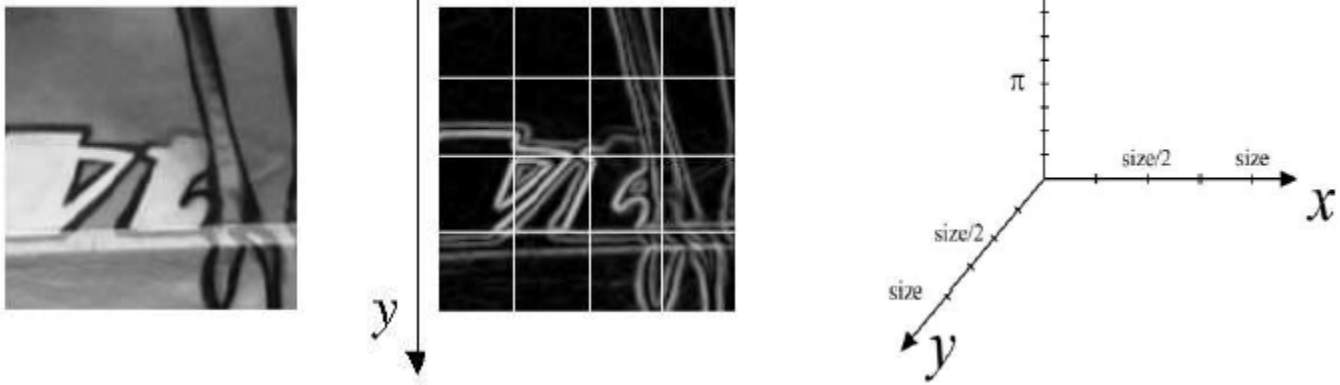
Average face:

Top ten eigenfaces (left = highest eigenvalue, right = lowest eigenvalue):

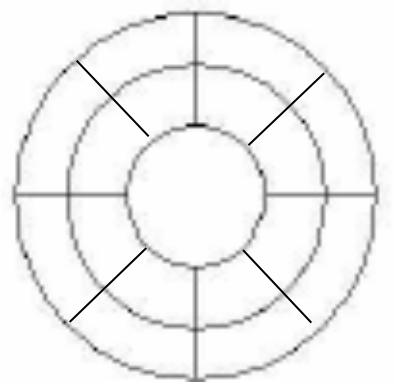
PCA-SIFT

- Only change step 4
- Pre-compute an eigen-space for local gradient patches of size 41x41
- 2x39x39=3042 elements
- Only keep 20 components
- A more compact descriptor

GLOH (Gradient location-orientation histogram)



SIFT



17 location bins
16 orientation bins
Analyze the 17x16=272-d eigen-space, keep 128 components

SIFT is still considered the best.

Multi-Scale Oriented Patches

- Simpler than SIFT. Designed for image matching. [Brown, Szeliski, Winder, CVPR'2005]
- Feature detector
  - Multi-scale Harris corners
  - Orientation from blurred gradient
  - Geometrically invariant to rotation
- Feature descriptor
  - Bias/gain normalized sampling of local patch (8x8)
  - Photometrically invariant to affine changes in intensity

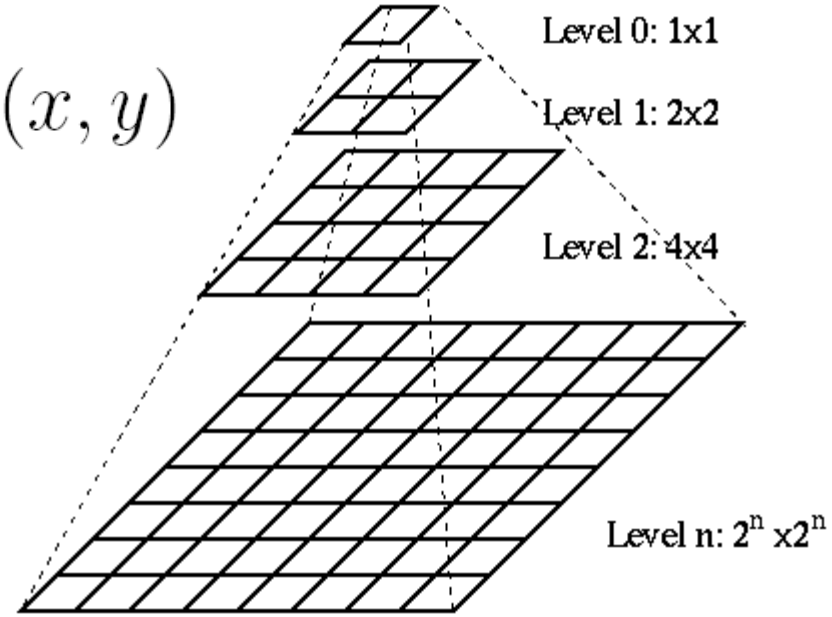
Multi-Scale Harris corner detector

$$P_0(x,y) = I(x,y)$$

$$P'_l(x,y) = P_l(x,y) * g_{\sigma_p}(x,y)$$

$$P_{l+1}(x,y) = P'_l(sx,sy)$$

$$s = 2 \quad \sigma_p = 1.0$$



- Image stitching is mostly concerned with matching images that have the same scale, so sub-octave pyramid might not be necessary.

Multi-Scale Harris corner detector

$$\mathbf{H}_l(x,y) = \nabla_{\sigma_d} P_l(x,y) \nabla_{\sigma_d} P_l(x,y)^T * g_{\sigma_i}(x,y)$$

$$\nabla_\sigma f(x,y) \triangleq \nabla f(x,y) * g_\sigma(x,y)$$

smoother version of gradients

$$\sigma_i = 1.5 \qquad \sigma_d = 1.0$$

Corner detection function:

$$f_{HM}(x,y) = \frac{\det \mathbf{H}_l(x,y)}{\operatorname{tr} \mathbf{H}_l(x,y)} = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

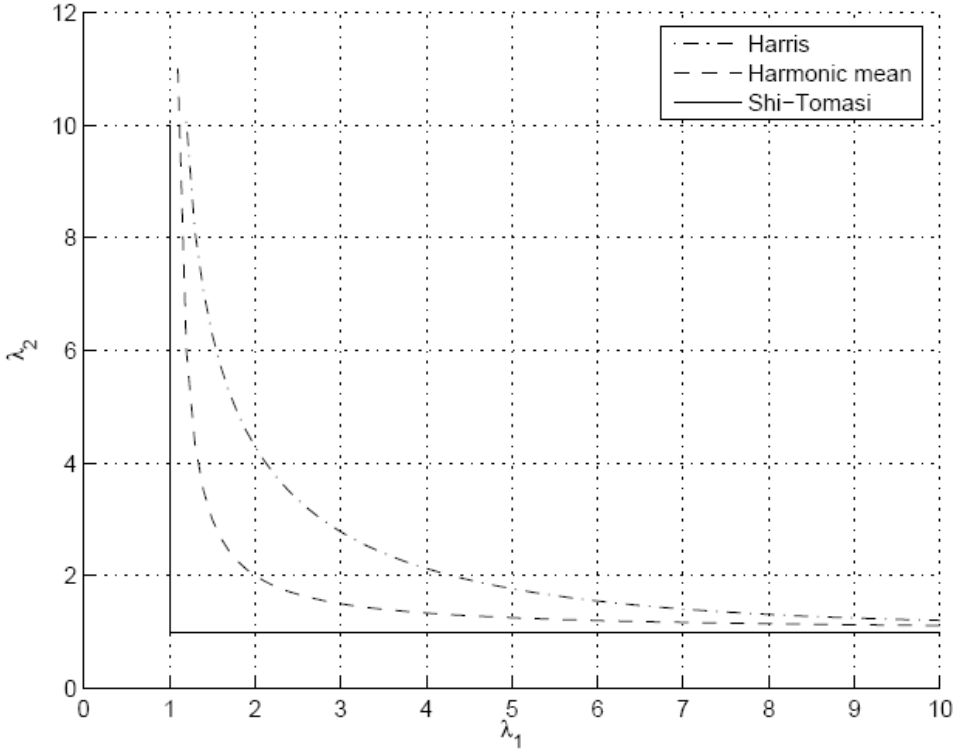
Pick local maxima of 3x3 and larger than 10

Keypoint detection function

$$\text{Harris} \quad f_H = \lambda_1\lambda_2 - 0.04(\lambda_1+\lambda_2)^2 = \det \mathbf{H} - 0.04(\operatorname{tr} \mathbf{H})^2$$

$$\text{Harmonic mean} \quad f_{HM} = \lambda_1\lambda_2/(\lambda_1+\lambda_2) = \det \mathbf{H}/\operatorname{tr} \mathbf{H}$$

$$\text{Shi-Tomasi} \quad f_{ST} = \min(\lambda_1, \lambda_2)$$

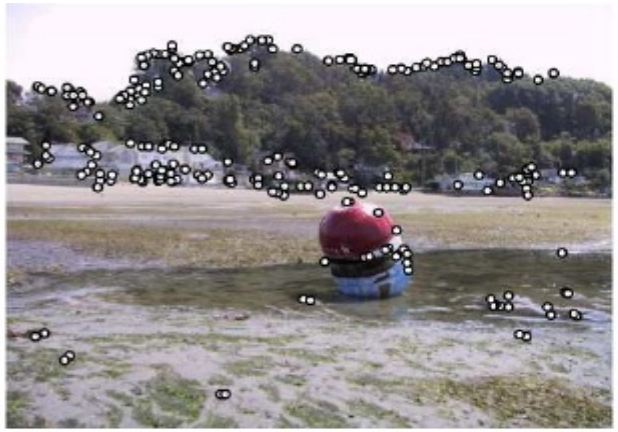


Experiments show roughly the same performance.

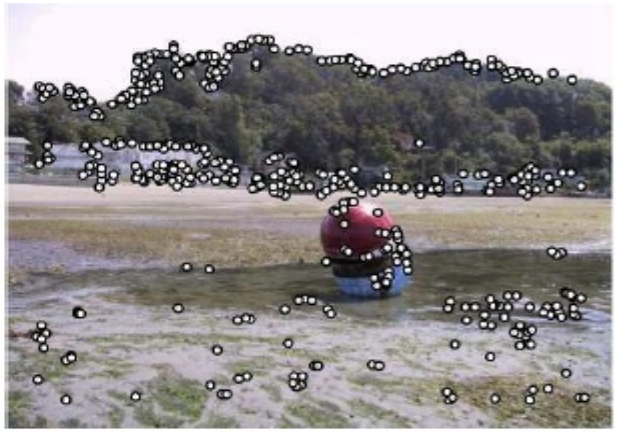
Non-maximal suppression

- Restrict the maximal number of interest points, but also want them spatially well distributed
- Only retain maximums in a neighborhood of radius *r*.
- Sort them by strength, decreasing *r* from infinity until the number of keypoints (500) is satisfied.

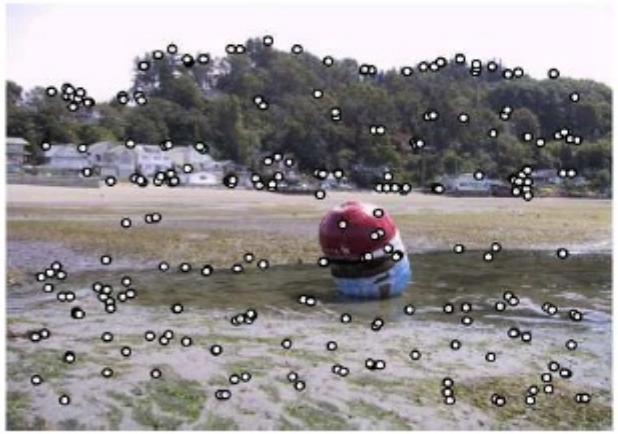
Non-maximal suppression



(a) Strongest 250



(b) Strongest 500



(c) ANMS 250, $r = 24$

(d) ANMS 500, $r = 16$

Sub-pixel refinement

$$f(\mathbf{x}) = f + \frac{\partial f}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 f}{\partial \mathbf{x}^2}\mathbf{x}$$

$$\mathbf{x}_m = -\frac{\partial^2 f}{\partial \mathbf{x}^2}^{-1} \frac{\partial f}{\partial \mathbf{x}}$$

$f_{-1,1}$	$f_{0,1}$	$f_{1,1}$
$f_{-1,0}$	$f_{0,0}$	$f_{1,0}$
$f_{-1,-1}$	$f_{0,-1}$	$f_{1,-1}$

$$\begin{aligned}
\frac{\partial f}{\partial x} &= (f_{1,0} - f_{-1,0})/2 \\
\frac{\partial f}{\partial y} &= (f_{0,1} - f_{0,-1})/2 \\
\frac{\partial^2 f}{\partial x^2} &= f_{1,0} - 2f_{0,0} + f_{-1,0} \\
\frac{\partial^2 f}{\partial y^2} &= f_{0,1} - 2f_{0,0} + f_{0,-1} \\
\frac{\partial^2 f}{\partial x \partial y} &= (f_{-1,-1} - f_{-1,1} - f_{1,-1} + f_{1,1})/4
\end{aligned}$$

Orientation assignment

- Orientation = blurred gradient

$$\mathbf{u}_l(x, y) = \nabla_{\sigma_o} P_l(x, y)$$

$$\sigma_o = 4.5$$

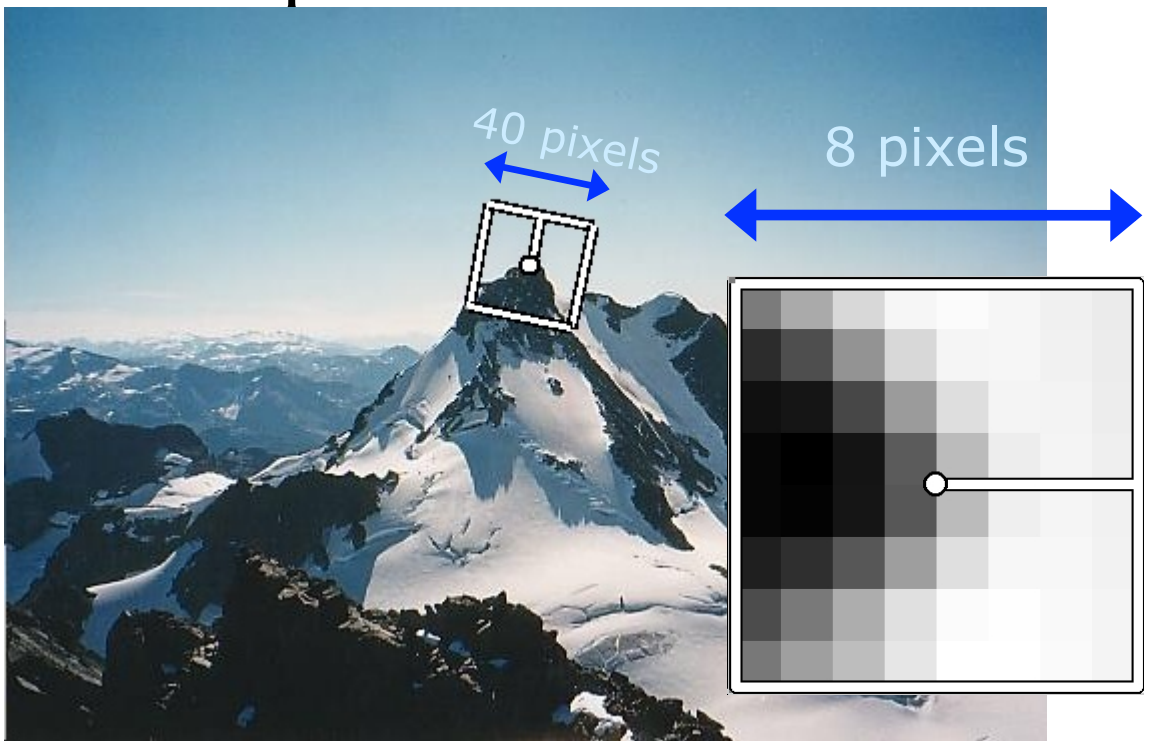
$$[\cos\theta, \sin\theta] = \mathbf{u}/|\mathbf{u}|$$

Descriptor Vector

- Rotation Invariant Frame

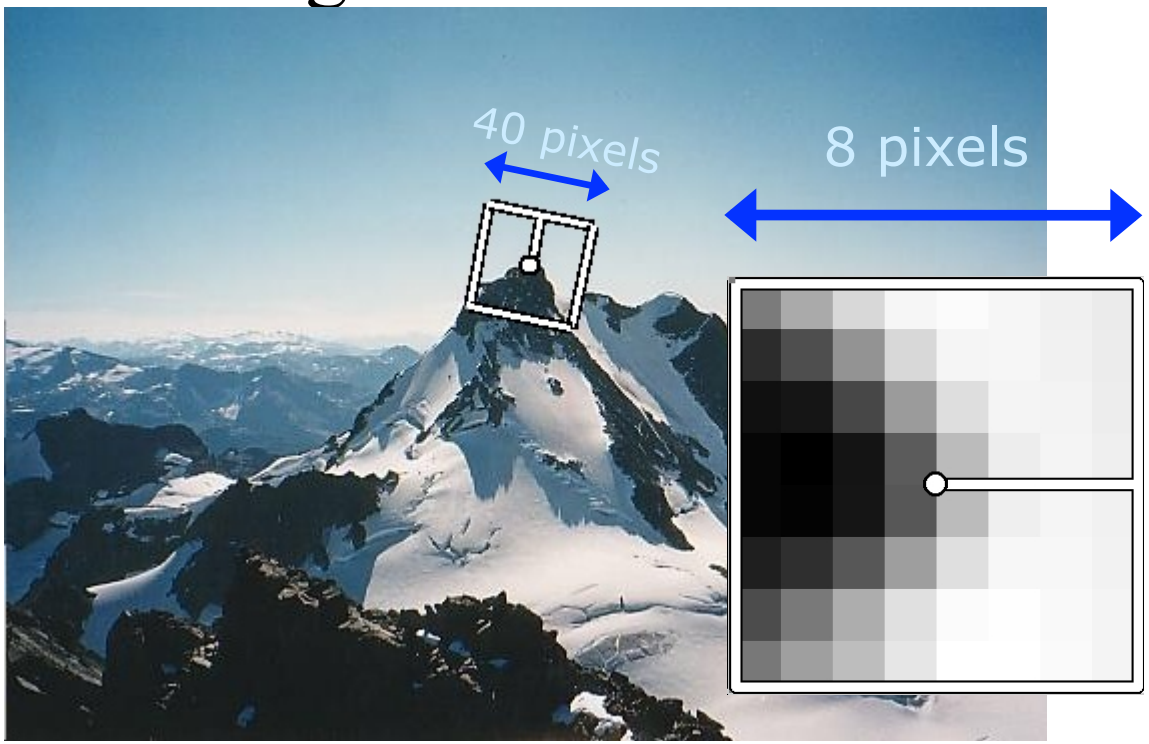
MSOP descriptor vector

- 8x8 oriented patch sampled at 5 x scale. See TR for detail $P_l(x, y) * g_{2\times\sigma_p}(x, y)$
- Sampled from with



MSOP descriptor vector

- 8x8 oriented patch sampled at 5 x scale. See TR for details.
- Bias/gain normalisation: $I' = (I - \mu)/\sigma$



Detections at multiple scales

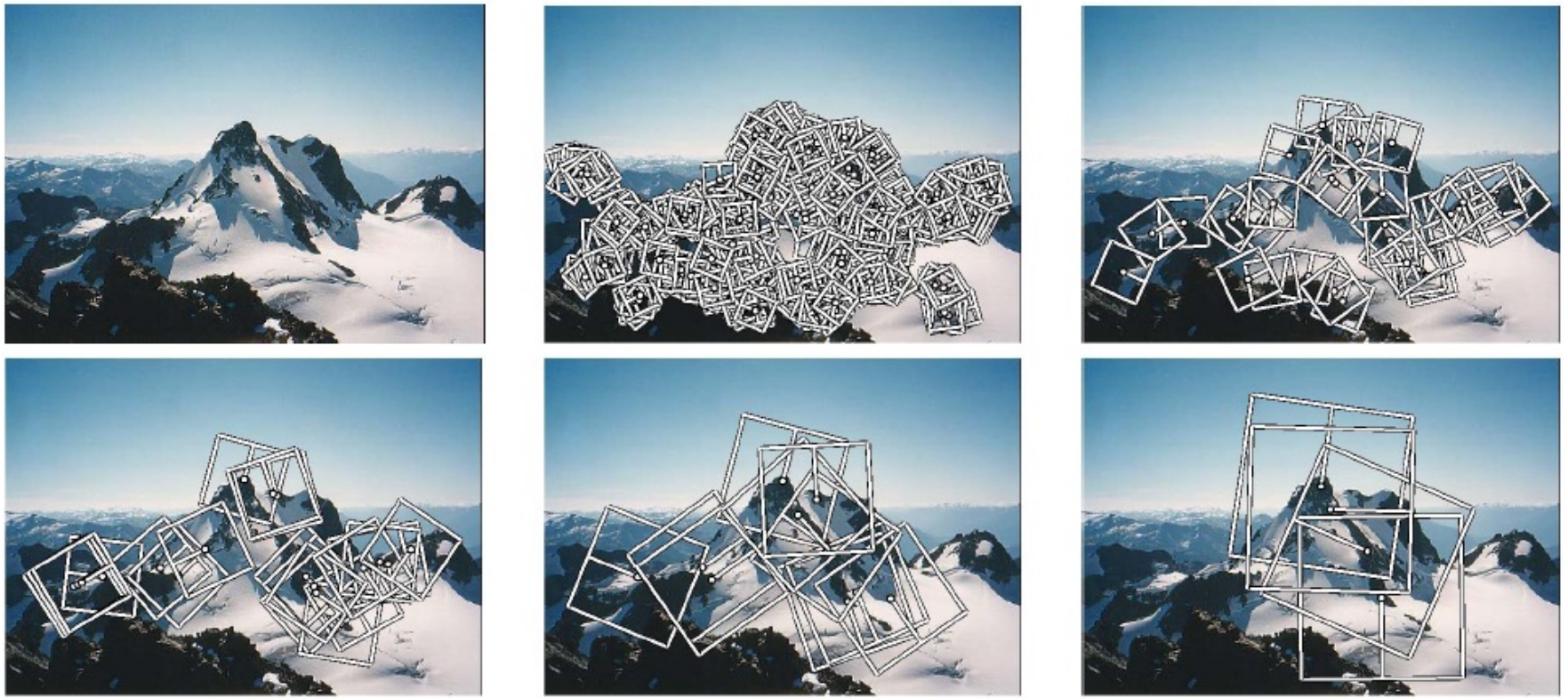


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

Summary

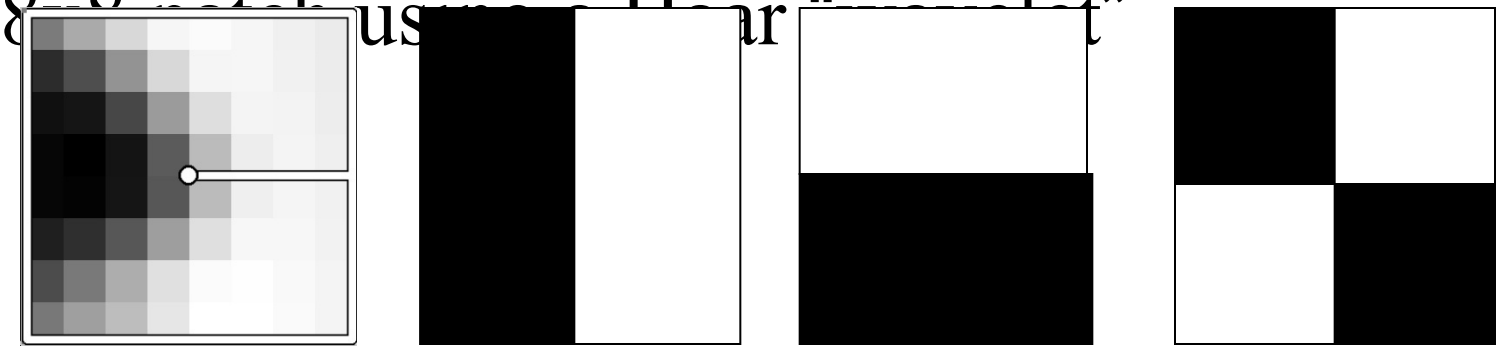
- Multi-scale Harris corner detector
- Sub-pixel refinement
- Orientation assignment by gradients
- Blurred intensity patch as descriptor

Feature matching

- Exhaustive search
  - for each feature in one image, look at *all* the other features in the other image(s)
- Hashing
  - compute a short descriptor from each feature vector, or hash longer descriptors (randomly)
- Nearest neighbor techniques
  - *k*-trees and their variants (Best Bin First)

Wavelet-based hashing

- Compute a short (3-vector) descriptor from an 8×8 patch using a Haar "wavelet"



- Quantize each value into 10 (overlapping) bins (10^3 total entries)
- [Brown, Szeliski, Winder, CVPR'2005]

Nearest neighbor techniques

- *k*-D tree and
- Best Bin First (BBF)

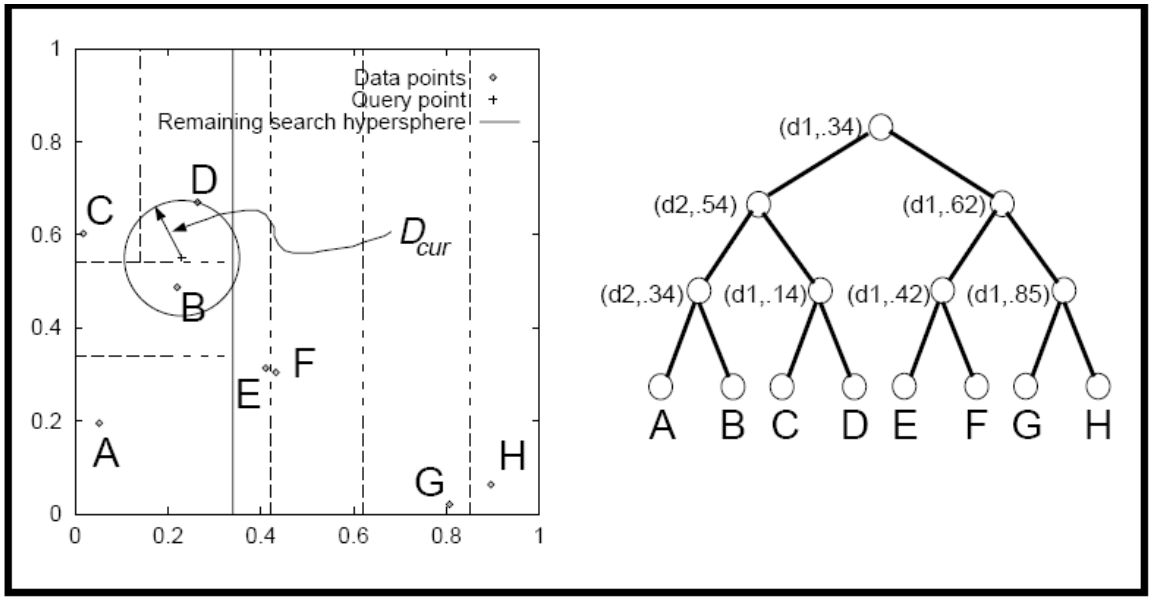
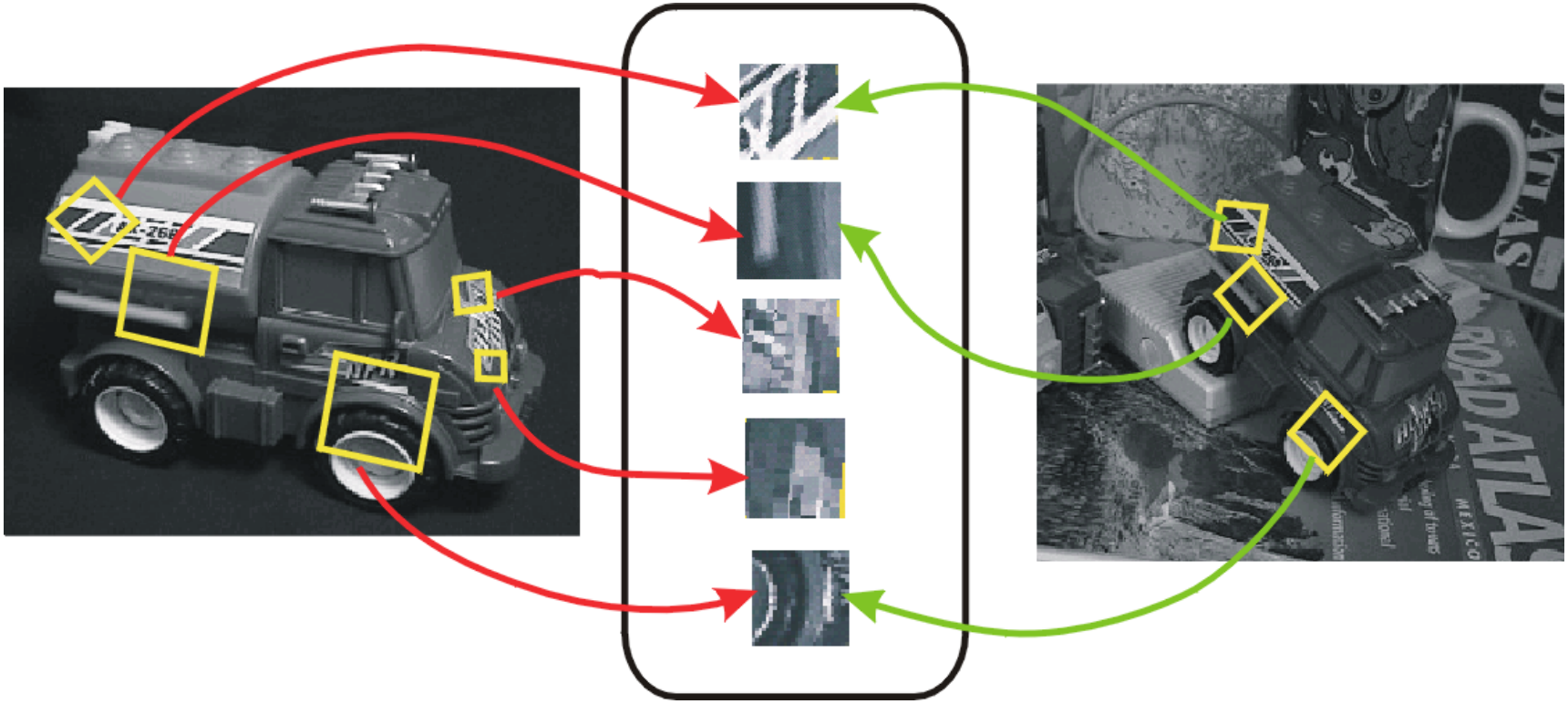


Figure 6: kd-tree with 8 data points labelled A-H, dimension of space k=2. On the right is the full tree, the leaf nodes containing the data points. Internal node information consists of the dimension of the cut plane and the value of the cut in that dimension. On the left is the 2D feature space carved into various sizes and shapes of bin, according to the distribution of the data points. The two representations are isomorphic. The situation shown on the left is after initial tree traversal to locate the bin for query point "+" (contains point D). In standard search, the closest nodes in the tree are examined first (starting at C). In BBF search, the closest bins to query point q are examined first (starting at B). The latter is more likely to maximize the overlap of (i) the hypersphere centered on q with radius D_{cur}, and (ii) the hyperrectangle of the bin to be searched. In this case, BBF search reduces the number of leaves to examine, since once point B is discovered, all other branches can be pruned.

Indexing Without Invariants in 3D Object Recognition, Beis and Lowe, PAMI'99

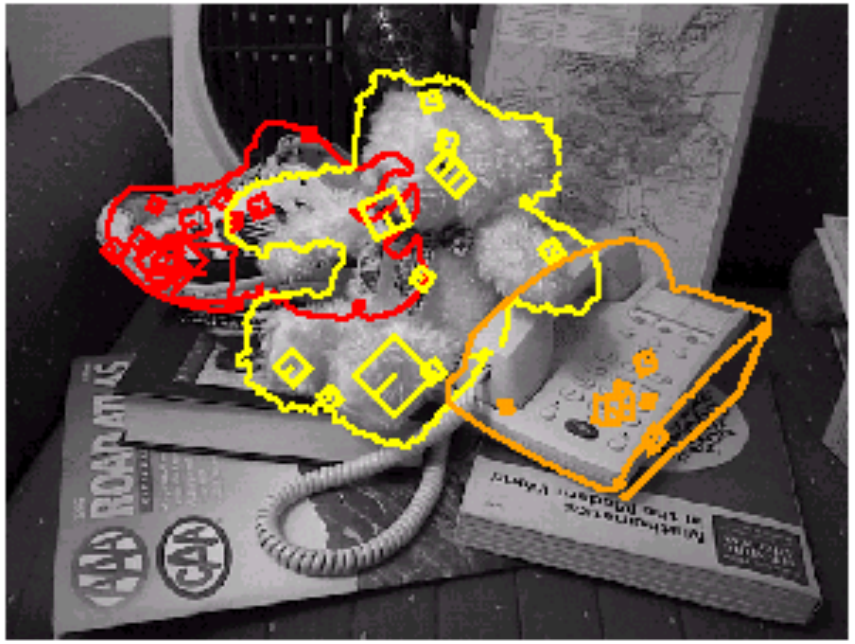
Applications

Recognition



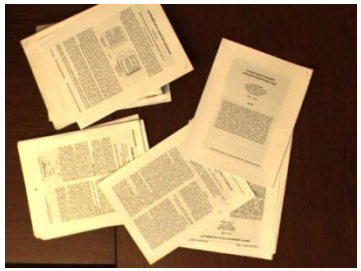
3D object recognition

3D object recognition



Office of the past

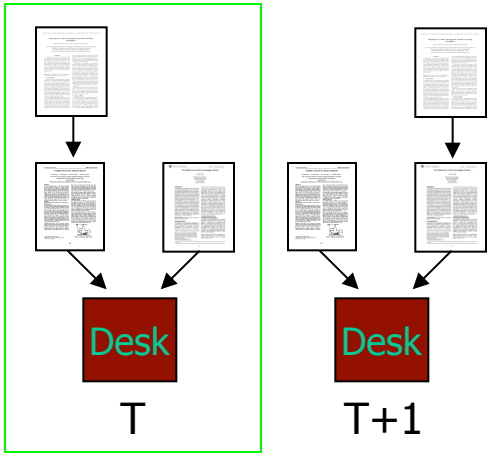
Video of desk



Track & recognize

Images from PDF

Internal representation



Scene Graph

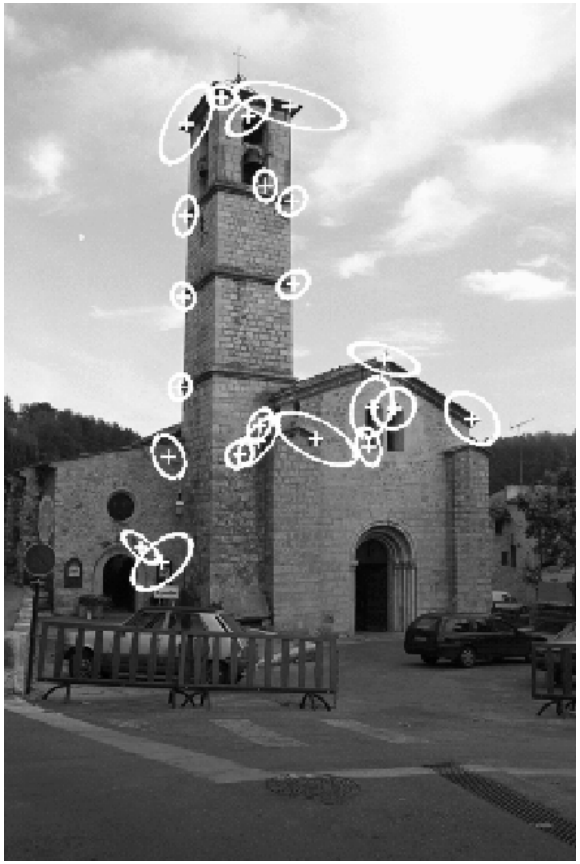
Image retrieval

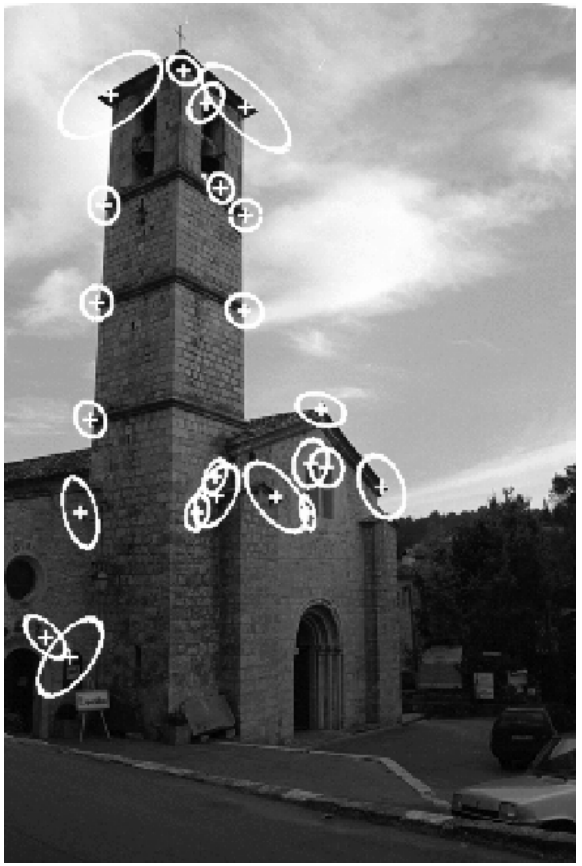
change in viewing angle

• • •

> 5000 images

Image retrieval





22 correct matches

Image retrieval

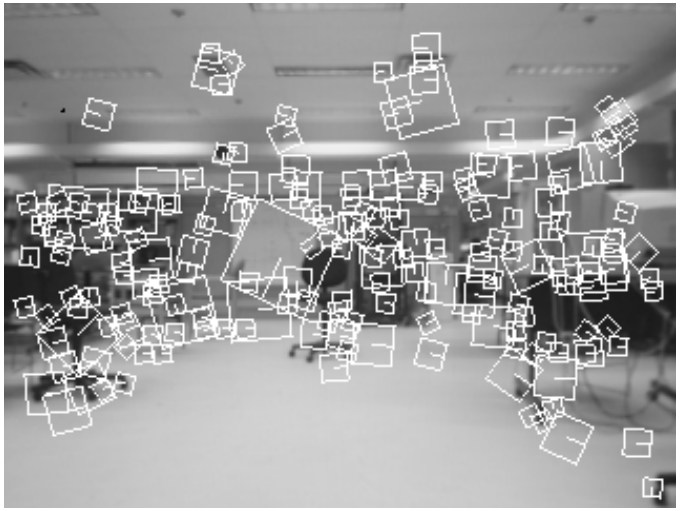
change in viewing angle

+ scale change

• • •

> 5000 images

Robot location



Robotics: Sony Aibo

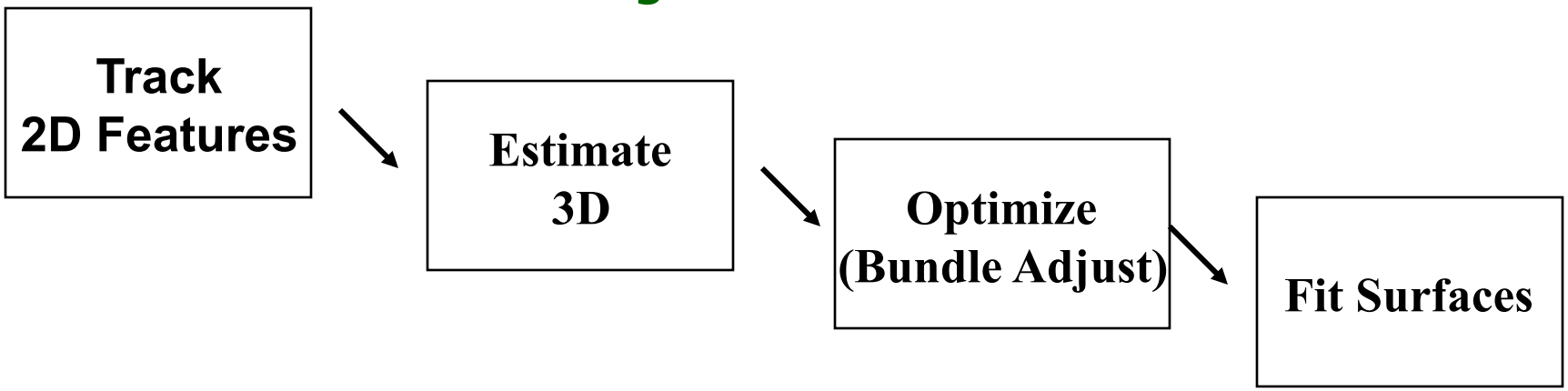
SIFT is used for

- Recognizing charging station
- Communicating with visual cards
- Teaching object recognition

- soccer

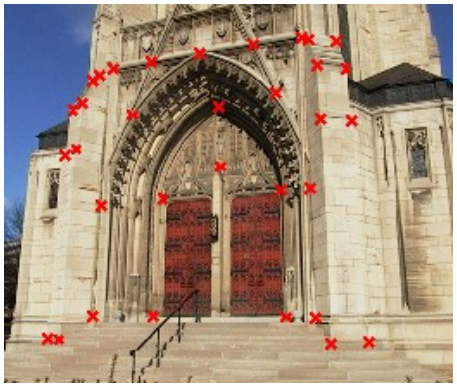
Structure from Motion

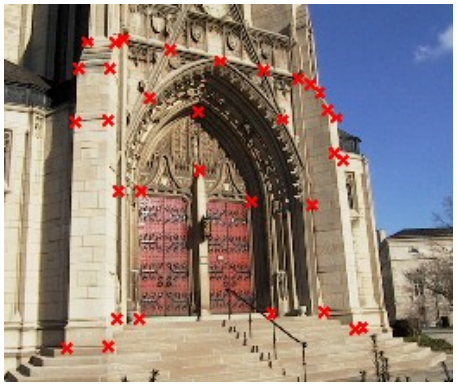
- The SFM Problem
  - Reconstruct scene geometry and camera motion from two or more images

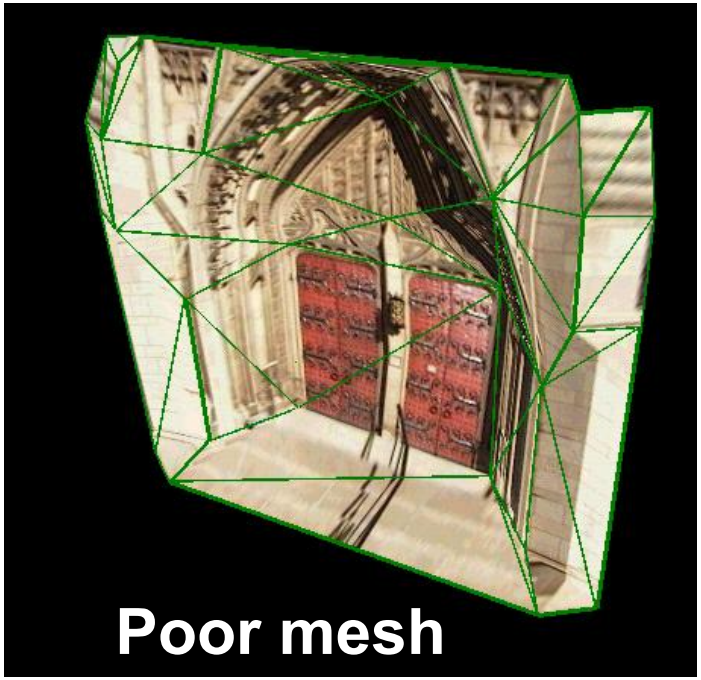


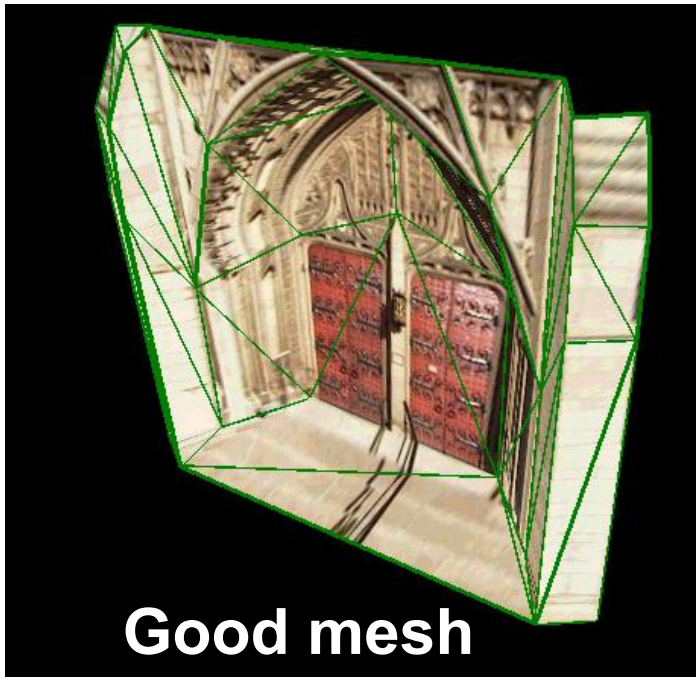
SFM Pipeline

Structure from Motion



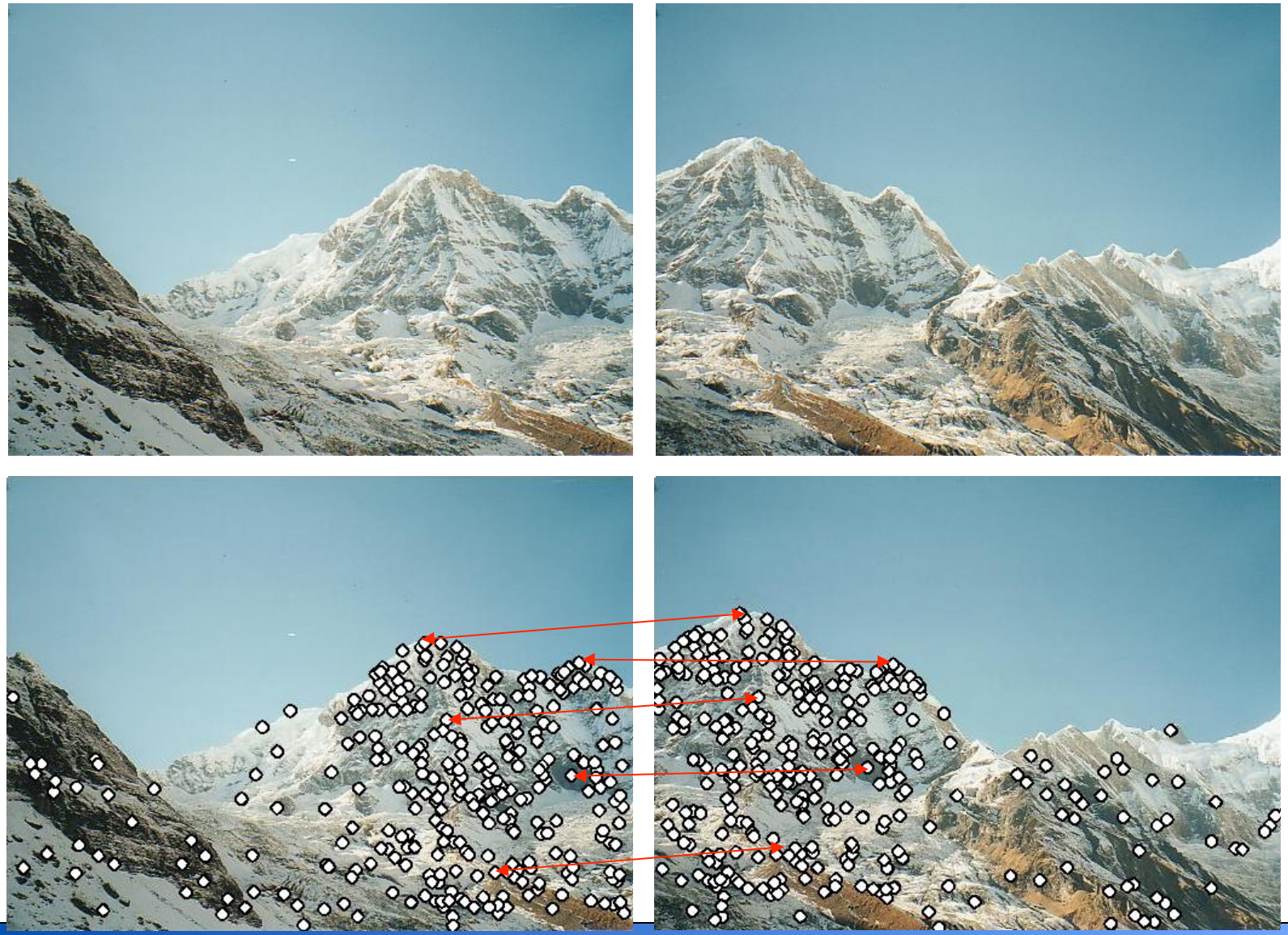




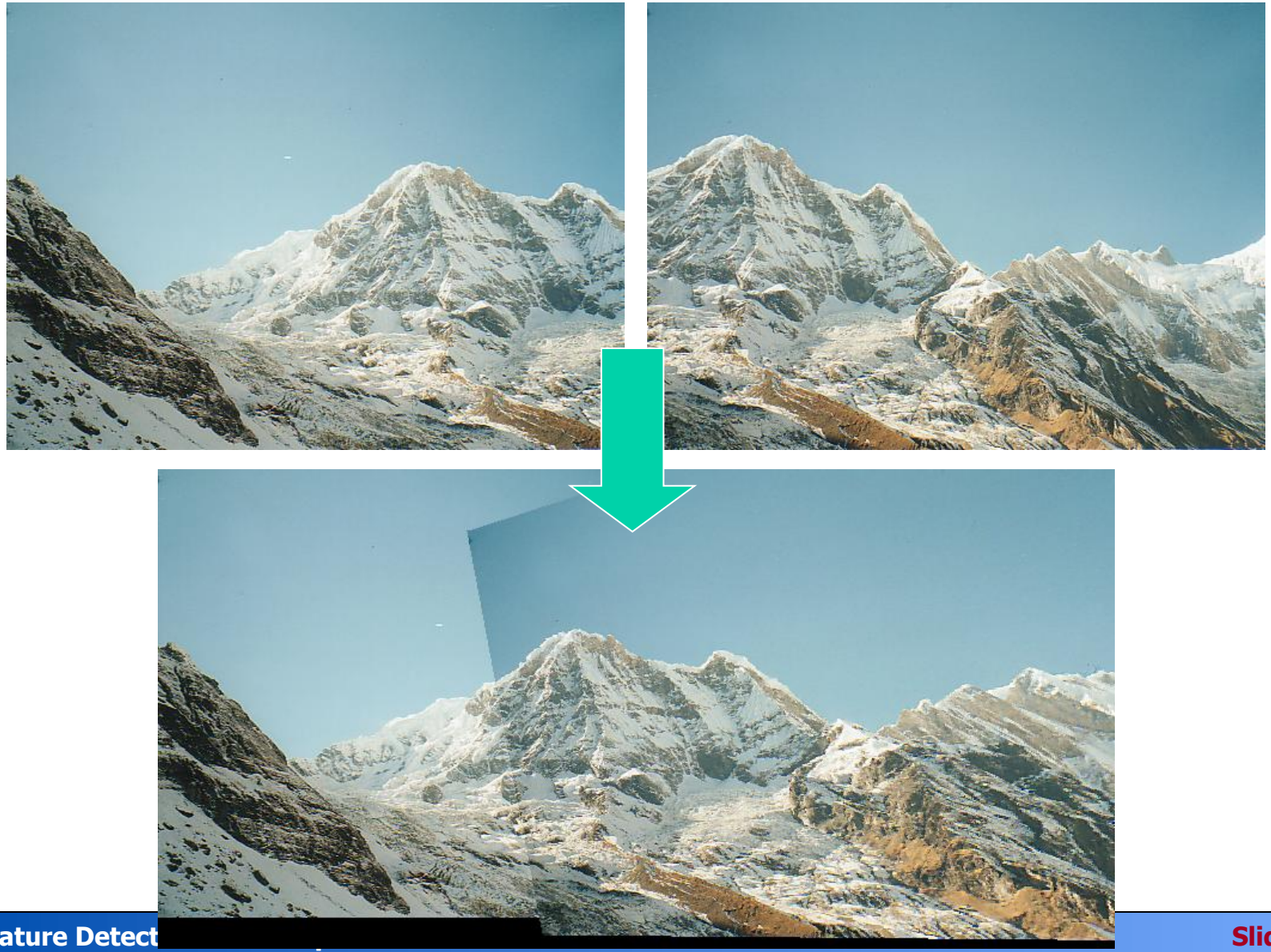


Augmented reality

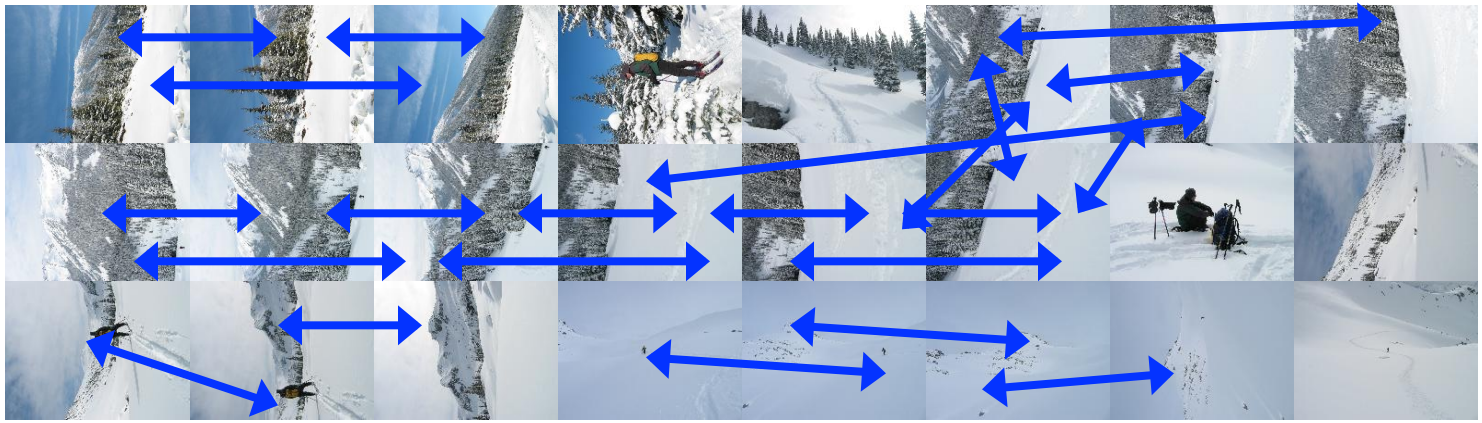
Automatic image stitching



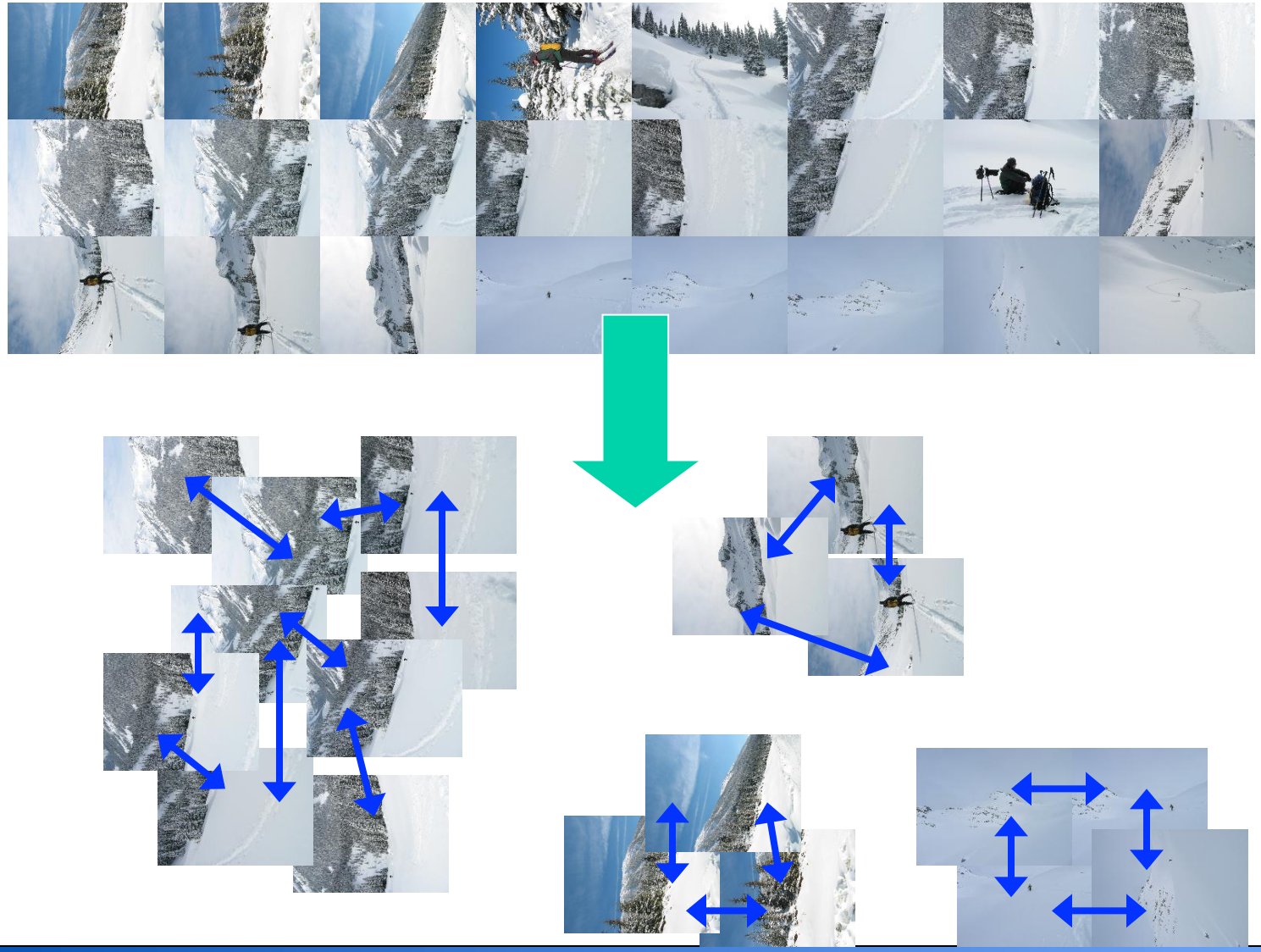
Automatic image stitching



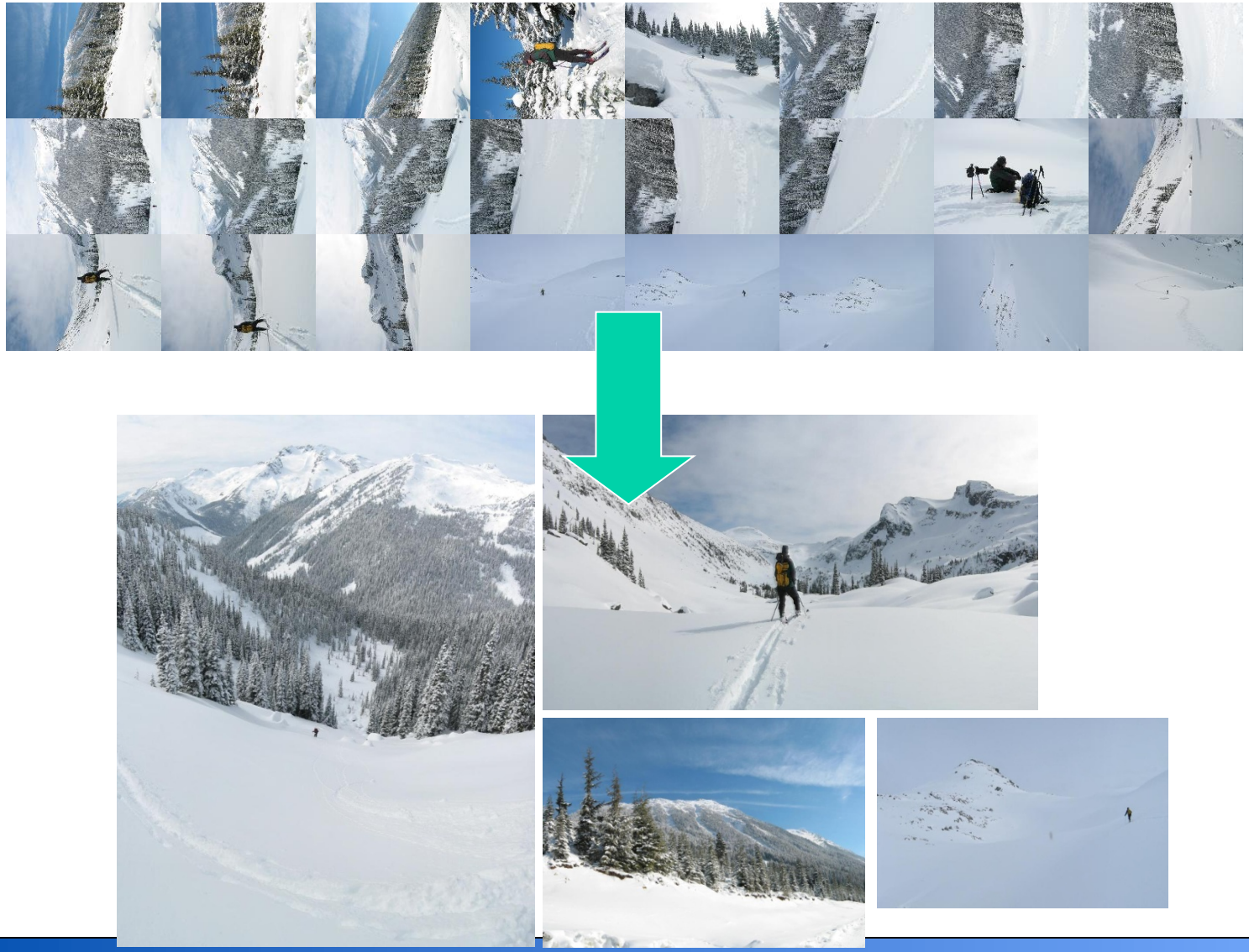
Automatic image stitching



Automatic image stitching



Automatic image stitching



Reference

- Chris Harris, Mike Stephens, A Combined Corner and Edge Detector, 4th Alvey Vision Conference, 1988, pp147-151.
- David G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, 60(2), 2004, pp91-110.
- Yan Ke, Rahul Sukthankar, PCA-SIFT: A More Distinctive Representation for Local Image Descriptors, CVPR 2004.
- Krystian Mikolajczyk, Cordelia Schmid, A performance evaluation of local descriptors, Submitted to PAMI, 2004.
- SIFT Keypoint Detector, David Lowe.