Advanced Image Processing

Interest points and Corners

SYDE 671

Filtering → Edges → Corners

Feature points

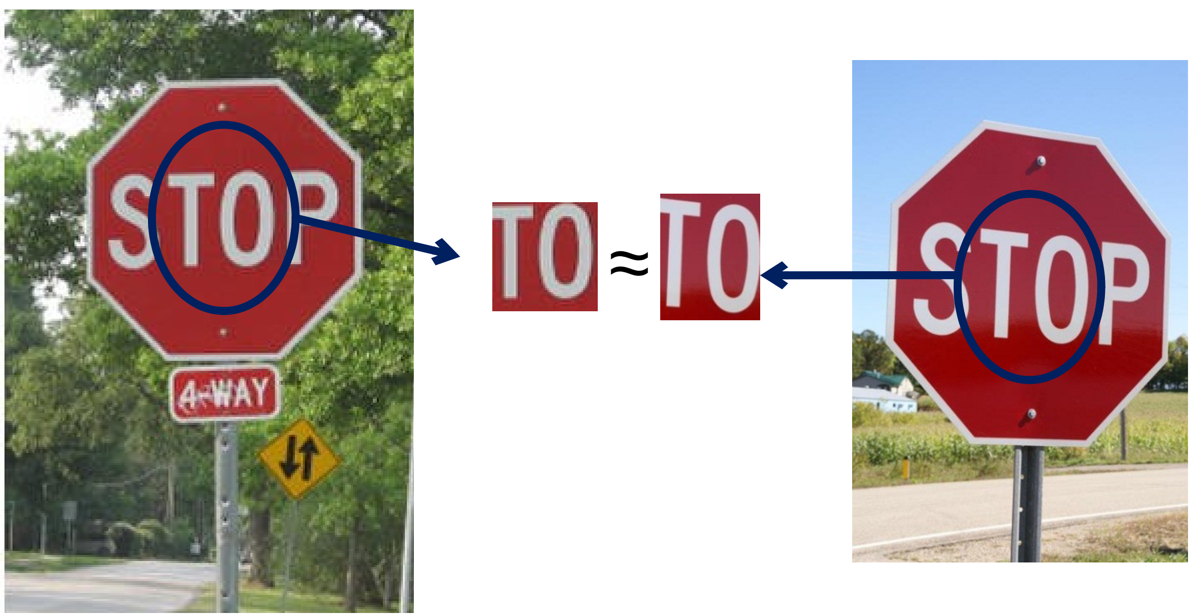
Also called interest points, key points, etc.
Often described as ‘local’ features.

Szeliski 4.1

Correspondence across views

- Correspondence: matching points, patches, edges, or regions across images.

Feature Matching



Hays

Fundamental to Applications

- Feature points are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking (robots, drones,
  - Indexing and database retrieval
  - Object recognition
  - …

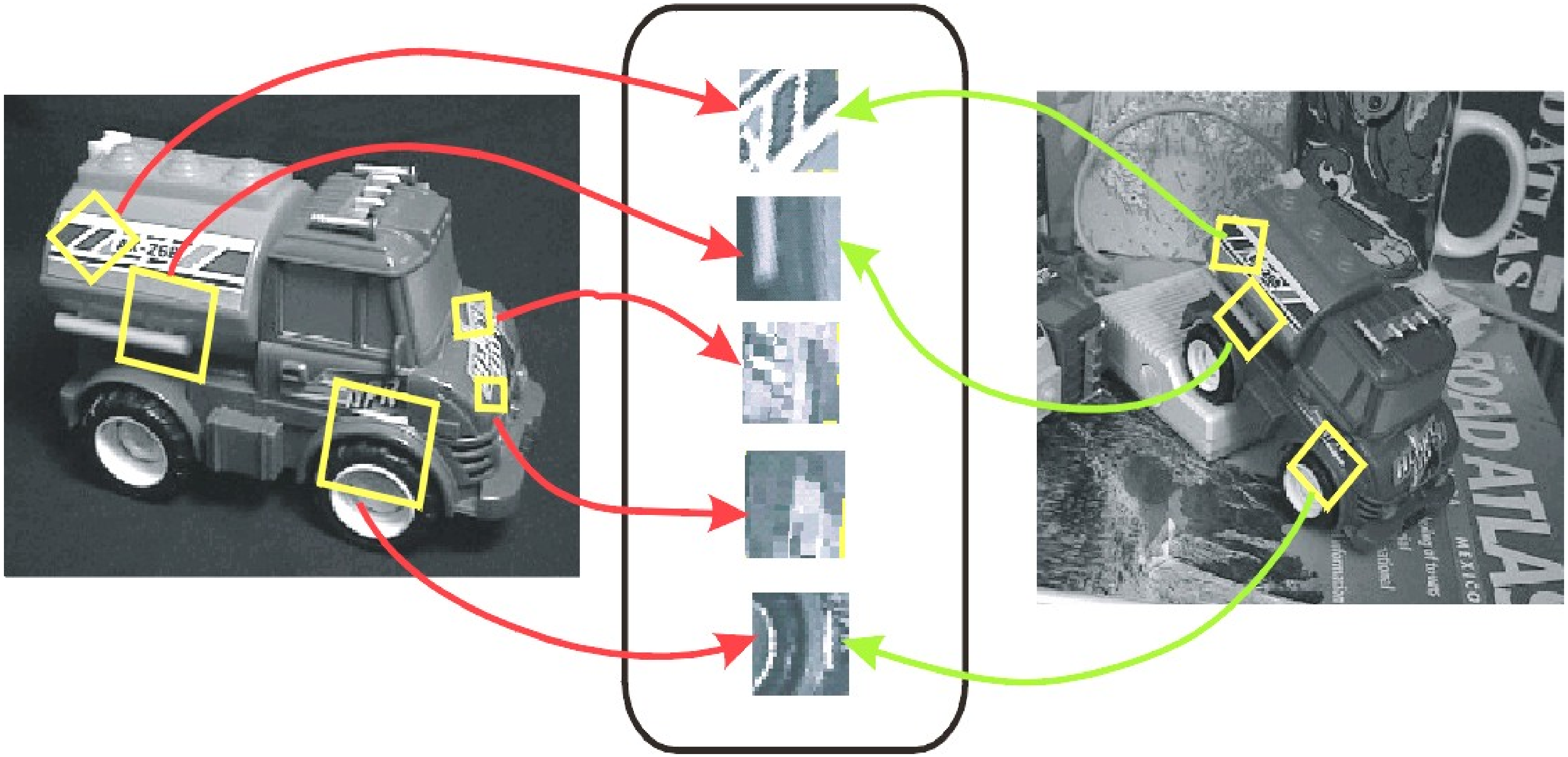
Hays

Example: Invariant Local Features

Detect points that are *repeatable* and *distinctive.*

I.E., invariant to image transformations:

- appearance variation (brightness, illumination)
- geometric variation (translation, rotation, scale).



Keypoint Descriptors

James Hays

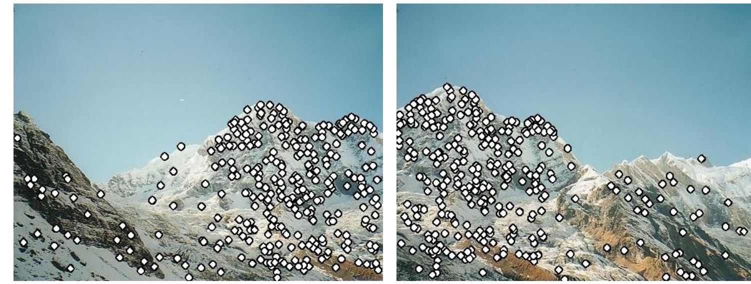
Example application

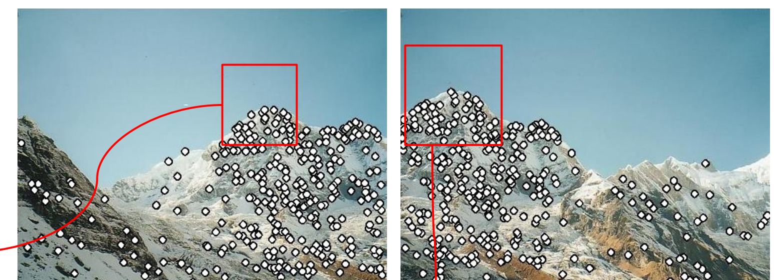
Panorama stitching

We have two images – how do we combine them?

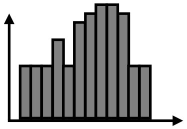

Local features: main components

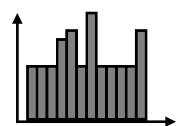
1) **Detection:**
Find a set of distinctive key points.





2) **Description:**
Extract feature descriptor around each interest point as vector.

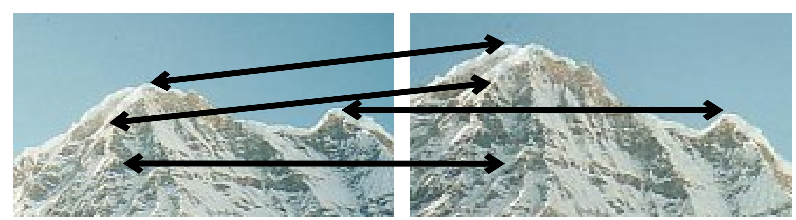
x_1  $\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$

x_2  $\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$

3) **Matching:**

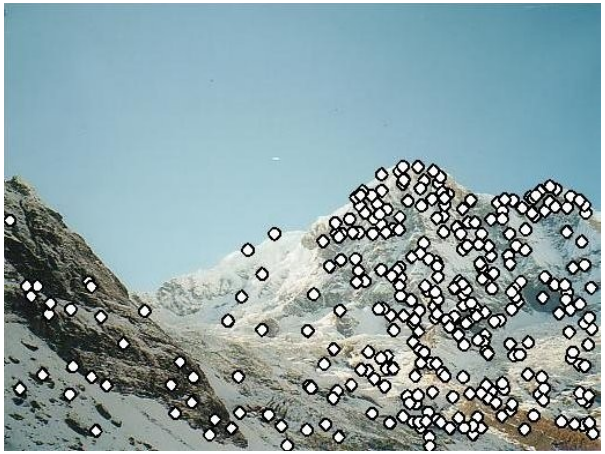
$$d(x_1, x_2) < T$$

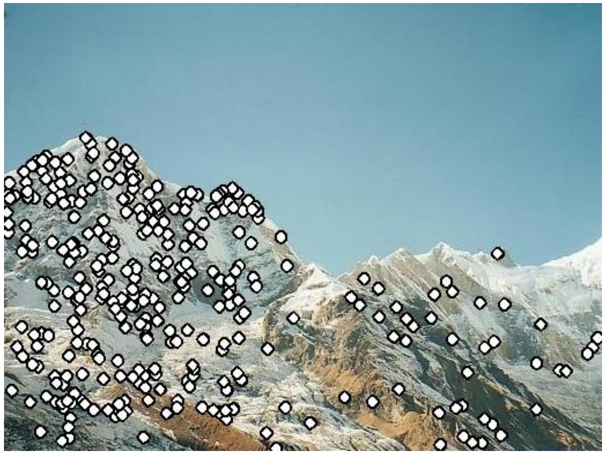
Compute distance between feature vectors to find correspondence.



K. Grauman, B. Leibe

Characteristics of good features





Repeatability

- The same feature can be found in several images despite geometric and photometric transformations

Saliency

- Each feature is distinctive

Compactness and efficiency

- Many fewer features than image pixels

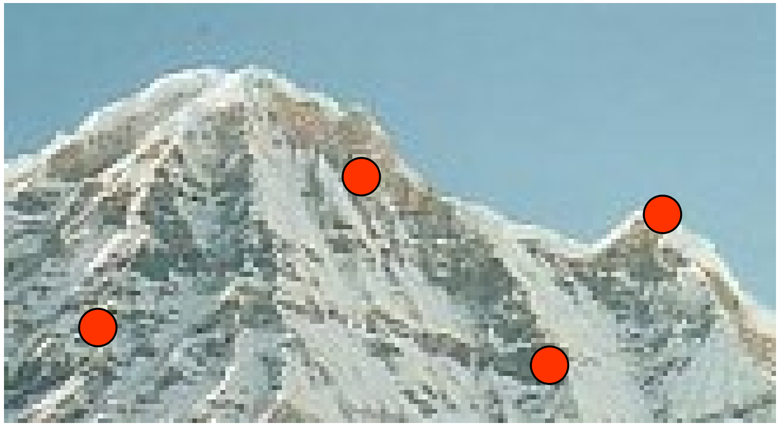
Locality

- A feature occupies a relatively small area of the image; robust to clutter and occlusion

Kristen Grauman

Goal: interest operator repeatability

We want to detect (at least some of) the same points in both images.



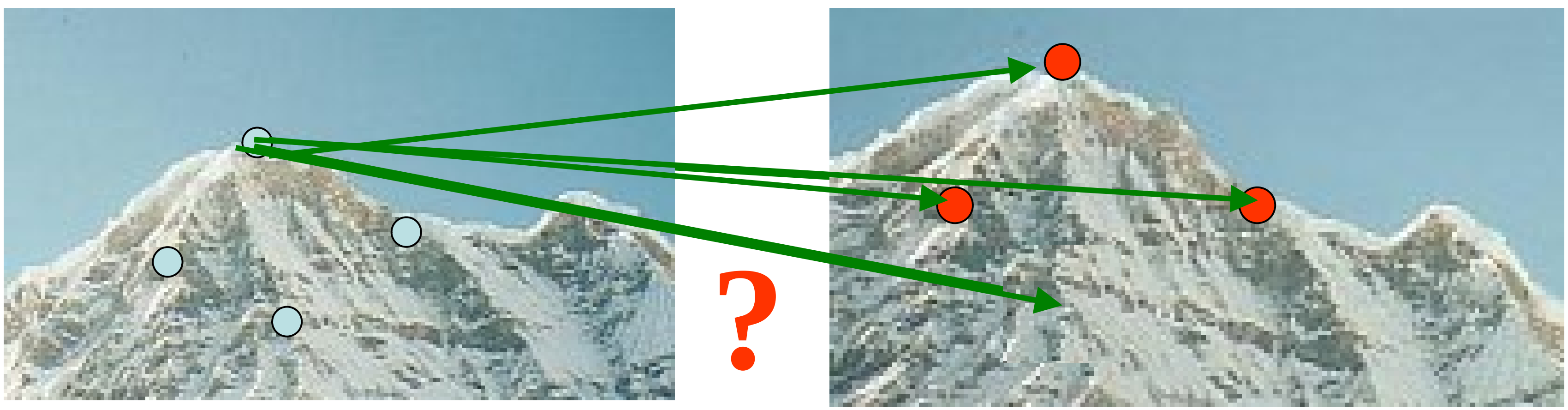
With these points, there's no chance to find true matches!

Yet we have to be able to run the detection procedure *independently* per image.

Kristen Grauman

Goal: descriptor distinctiveness

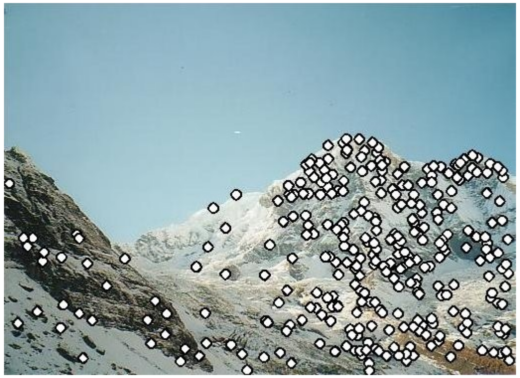
We want to be able to reliably determine which point goes with which.



Must provide some invariance to geometric and photometric differences between the two views.

Local features: main components

1) Detection:
   Find a set of distinctive key points.



2) Description:
   Extract feature descriptor around each interest point as vector.

3) Matching:
   Compute distance between feature vectors to find correspondence.

Detection: Basic Idea

We do not know which other image locations the feature will end up being matched against.

But we can compute how stable a location is in appearance with respect to small variations in position u.

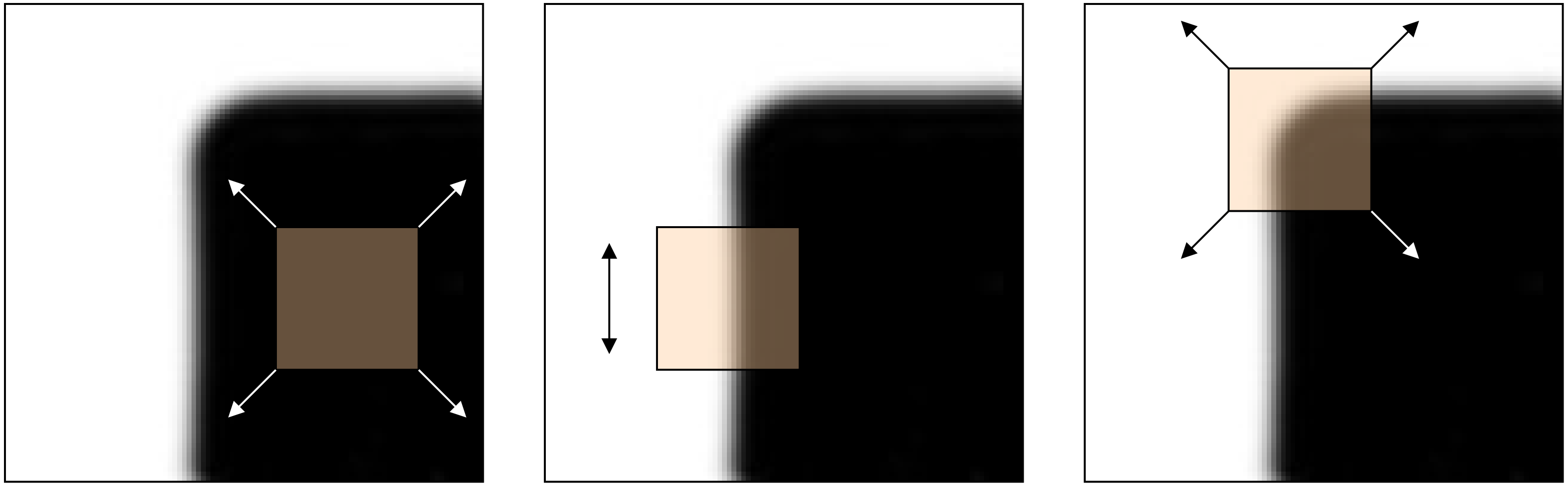
Compare image patch against local neighbors.

A. Efros

Corner Detection: Basic Idea

We might recognize the point by looking through a small window.

We want a window shift in *any direction* to give *a large change* in intensity.



"Flat" region: no change in all directions

"Edge": no change along the edge direction

"Corner": significant change in all directions

A. Efros

Corner Detection by Auto-correlation

Change in appearance of window $w(x,y)$ for shift $[u,v]$:

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

Intensity of shifted image

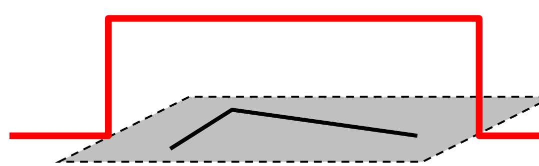
Intensity of original image.

Window function

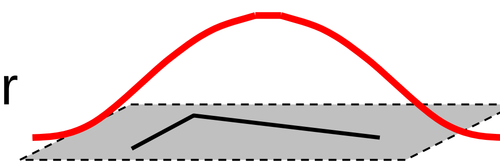
Shifted intensity

Intensity

Window function $w(x,y) =$



or



1 in window, 0 outside

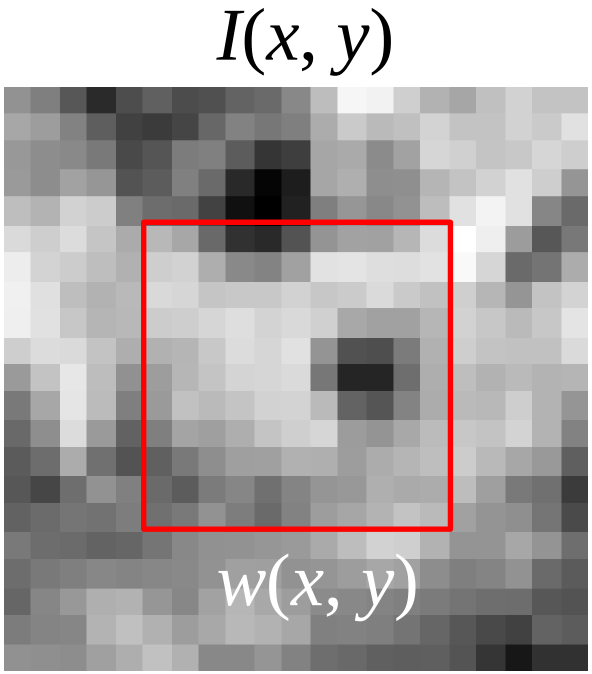
Gaussian

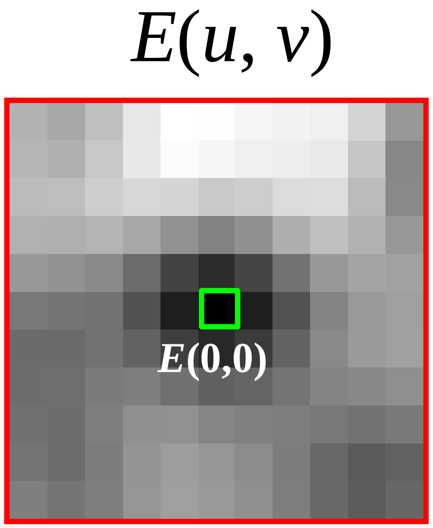
Source: R. Szeliski

Corner Detection by Auto-correlation

Change in appearance of window *w*(*x*,*y*) for shift [*u,v*]:

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



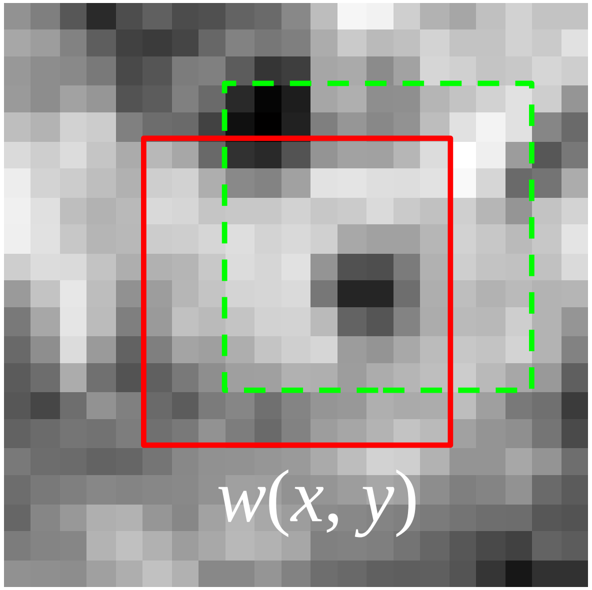


Corner Detection by Auto-correlation

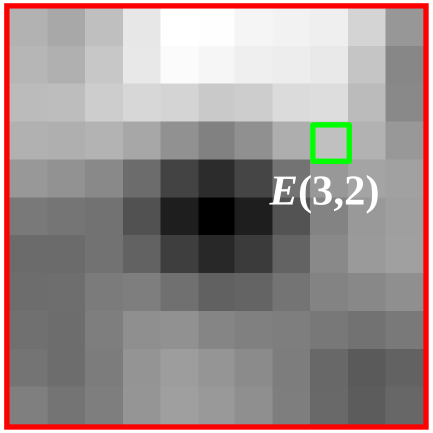
Change in appearance of window $w(x,y)$ for shift $[u,v]$:

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

$I(x, y)$



$E(u, v)$

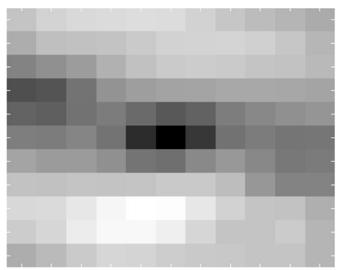


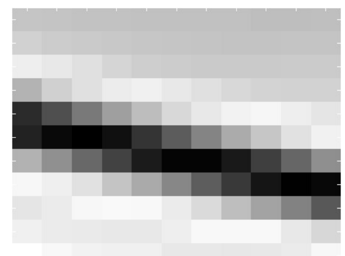
$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u, y+v) - I(x,y) \right]^2$$

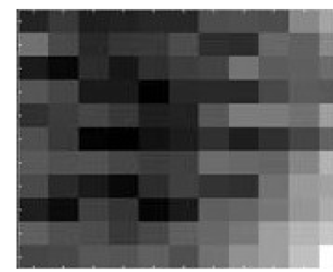
Think-Pair-Share:

Correspond the three red crosses to (b,c,d).

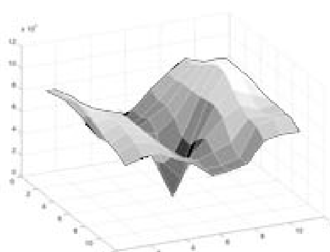
$E(u,v)$

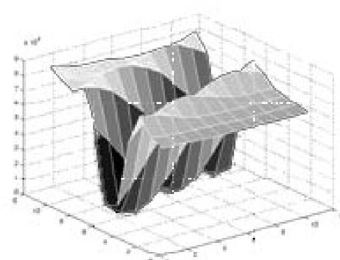


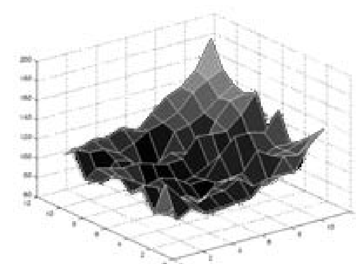




$E(u,v)$







As a surface

Corner Detection by Auto-correlation

Change in appearance of window *w*(*x*,*y*) for shift [*u*,*v*]:

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

We want to discover how E behaves for small shifts

But this is very slow to compute naively.
O(window_width^2 * shift_range^2 * image_width^2)

O(11^2 * 11^2 * 600^2) = 5.2 billion of these
14.6 thousand per pixel in your image

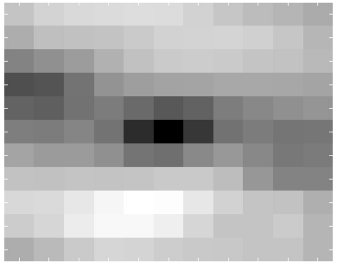
Corner Detection by Auto-correlation

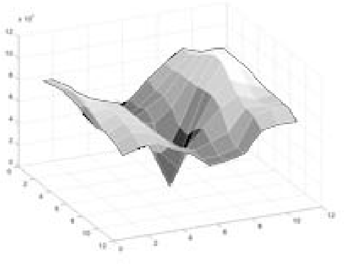
Change in appearance of window *w*(*x*,*y*) for shift [*u*,*v*]:

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

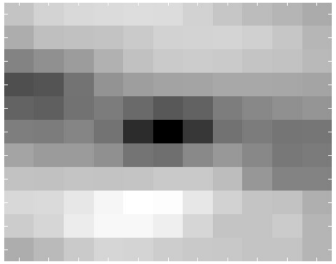
We want to discover how E behaves for small shifts

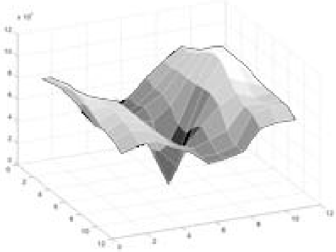
But we know the response in *E* that we are looking for – strong peak.



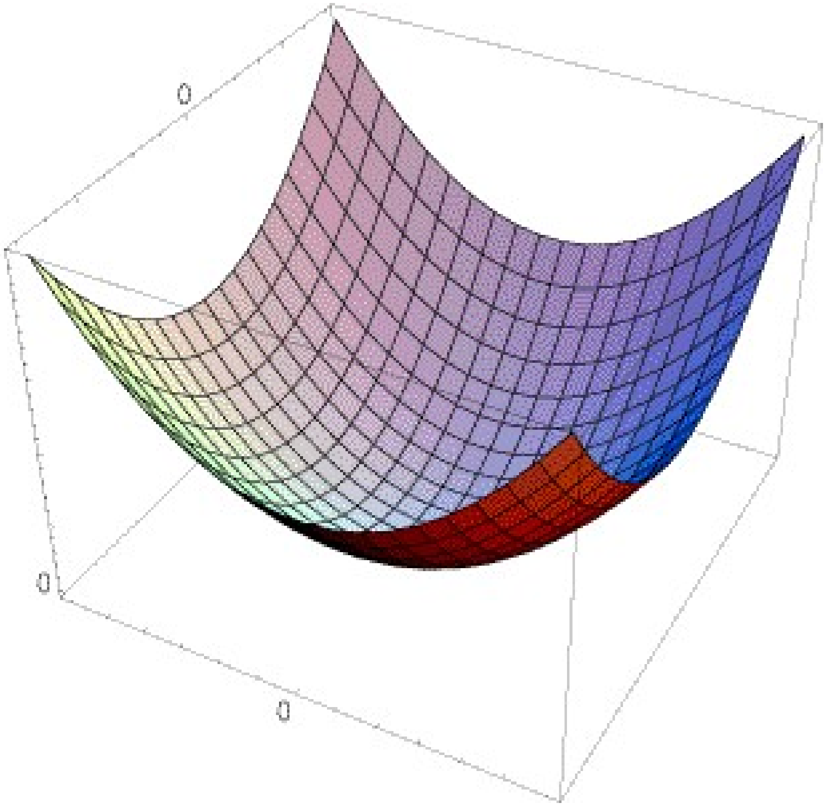


Can we just approximate $E(u,v)$ locally by a quadratic surface?





$\approx$

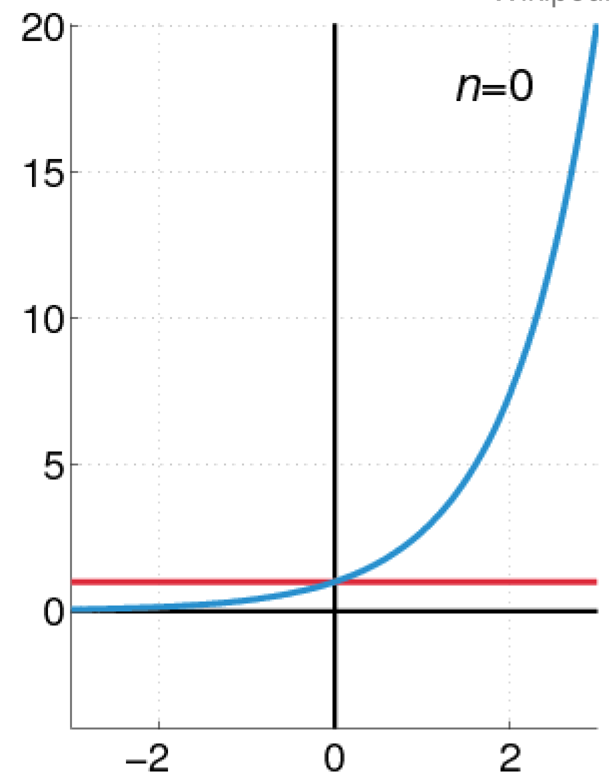


Recall: Taylor series expansion

A function f can be represented by an infinite series of its derivatives at a single point *a*:

$$f(a)+\frac{f'(a)}{1!}(x-a)+\frac{f''(a)}{2!}(x-a)^2+\frac{f'''(a)}{3!}(x-a)^3+\cdots.$$

Wikipedia



Approximation of $f(x) = e^x$ centered at f(0)

Taylor series approximation (2D)

Robert Collins
CSE486, Penn State

$$f(x+u,y+v) = f(x,y) + uf_x(x,y) + vf_y(x,y) +$$

First partial derivatives

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

Second partial derivatives

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

Third partial derivatives

$$+ \ \ldots \text{ (Higher order terms)}$$

First order approx

$$f(x+u,y+v) \approx f(x,y) + uf_x(x,y) + vf_y(x,y)$$

Corner Detection: Mathematics

Robert Collins
CSE486, Penn State

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

$$\approx \sum [I(x,y) + uI_x + vI_y - I(x,y)]^2$$ First order approx

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

$$= \sum \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$ Rewrite as matrix equation

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

Corner Detection: Mathematics

The quadratic approximation simplifies to

$$E(u,v) \approx [u \ \ v] \ M \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a *second moment matrix* computed from image derivatives:

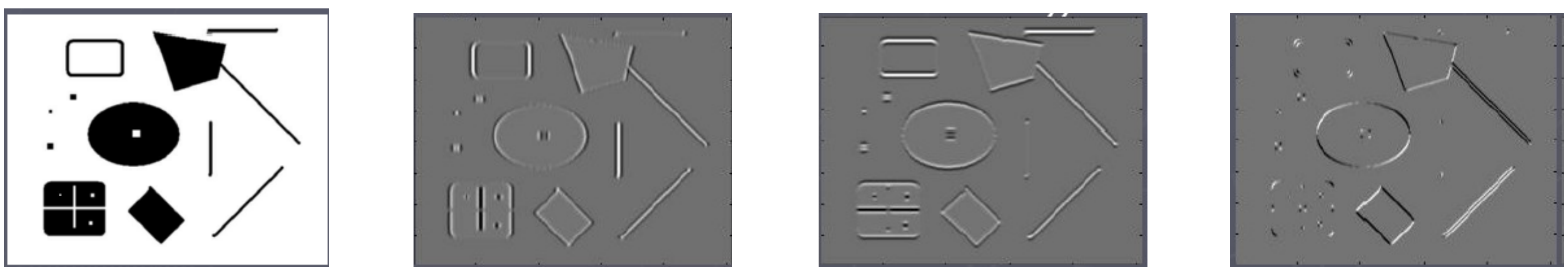
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

$$M = \left[\begin{array}{cc} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{array} \right] = \sum \left[\begin{array}{c} I_x \\ I_y \end{array} \right] [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

Corners as distinctive interest points

$$M = \sum w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

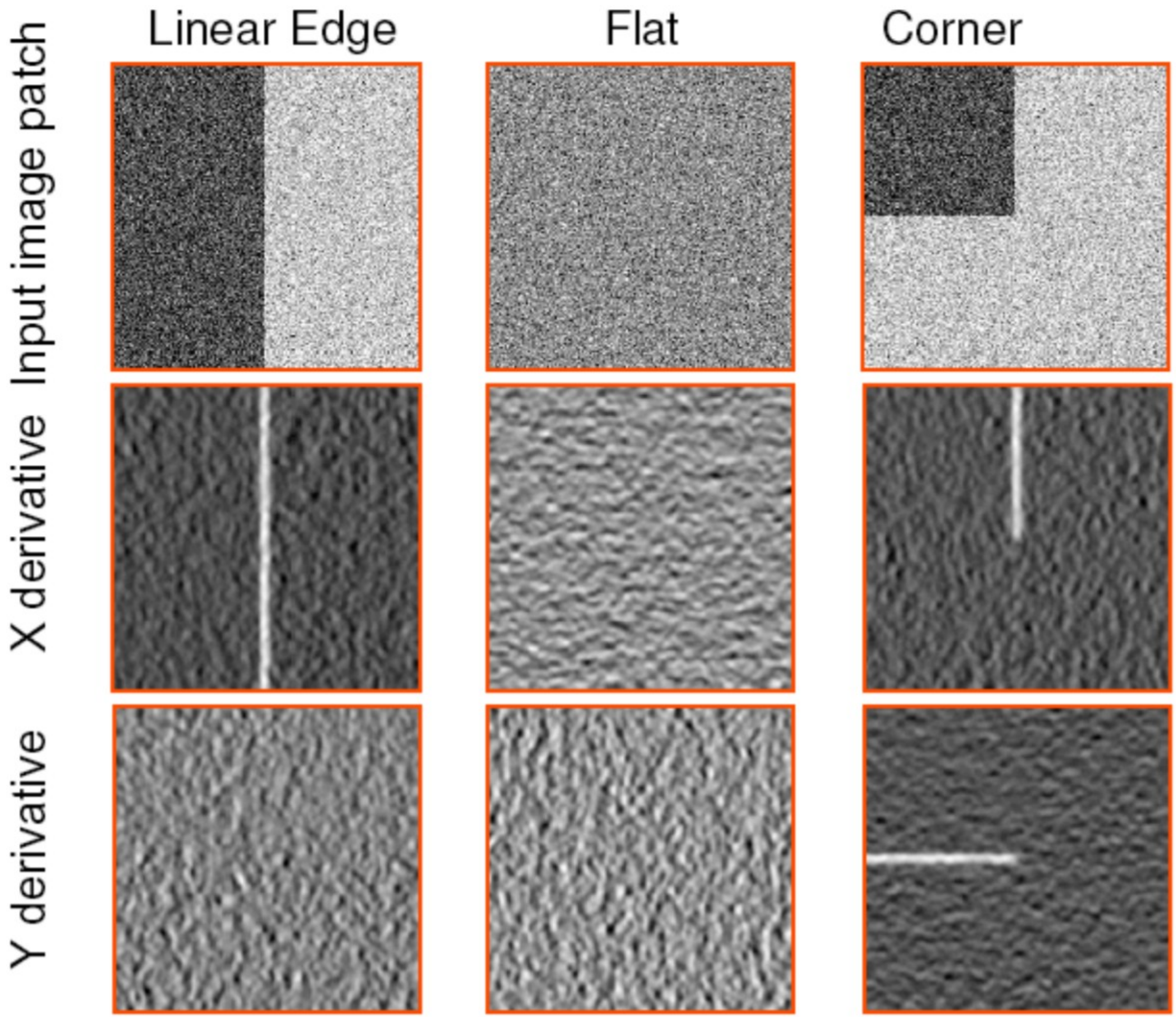
2 x 2 matrix of image derivatives
(averaged in neighborhood of a point)



Notation: $I_x \Leftrightarrow \frac{\partial I}{\partial x}$ $\quad I_y \Leftrightarrow \frac{\partial I}{\partial y}$ $\quad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$

James Hays

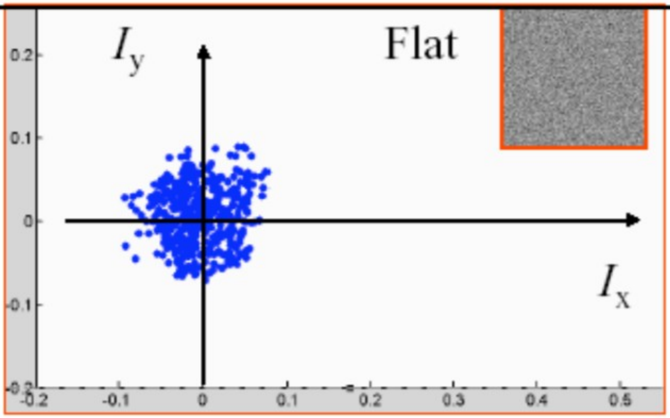
Example: 2D Gradients

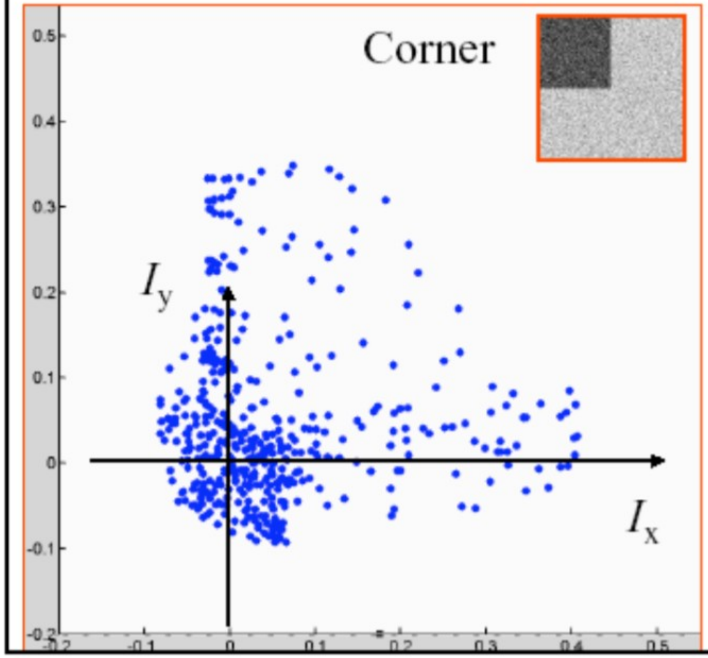


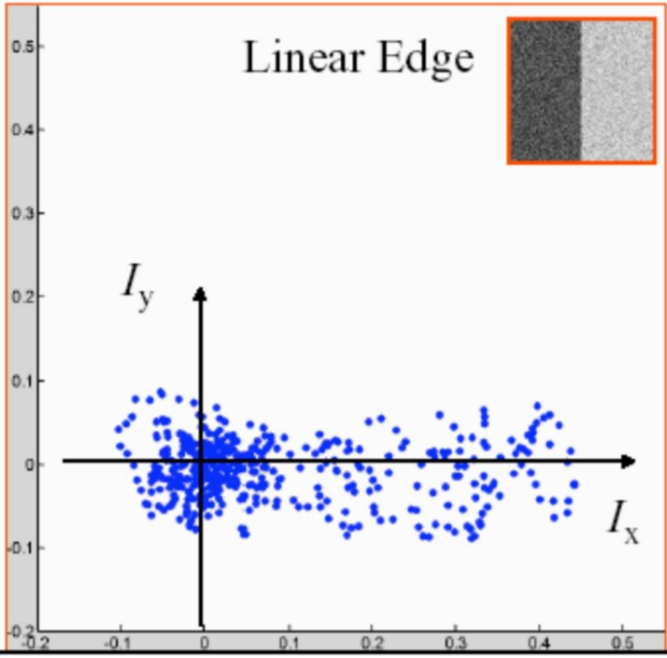
Robert Collins
CSE486, Penn State

Example: 2D Gradients - Distribution

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



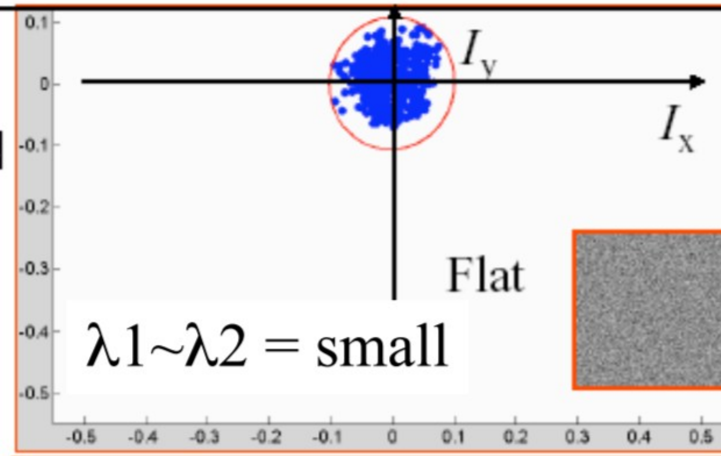


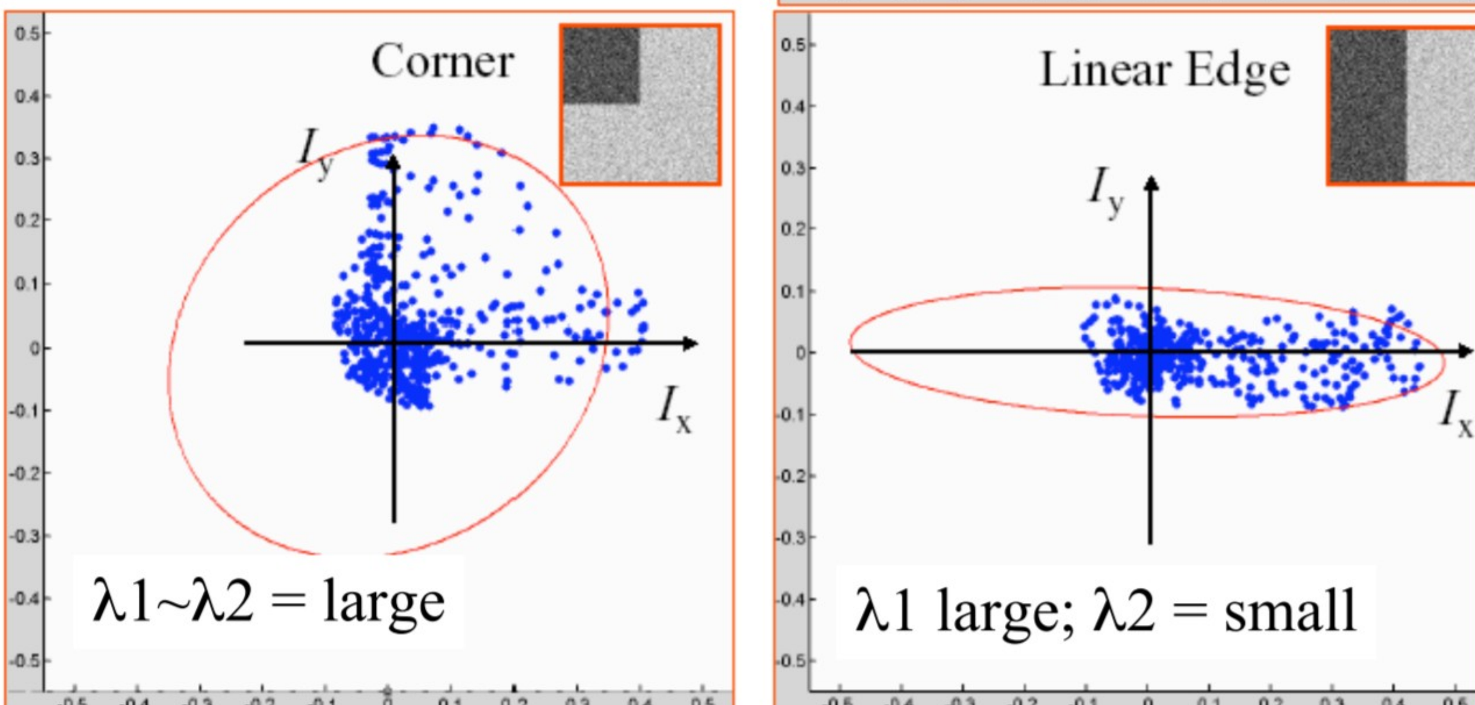


Robert Collins
CSE486, Penn State

Example: 2D Gradients - Distribution

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





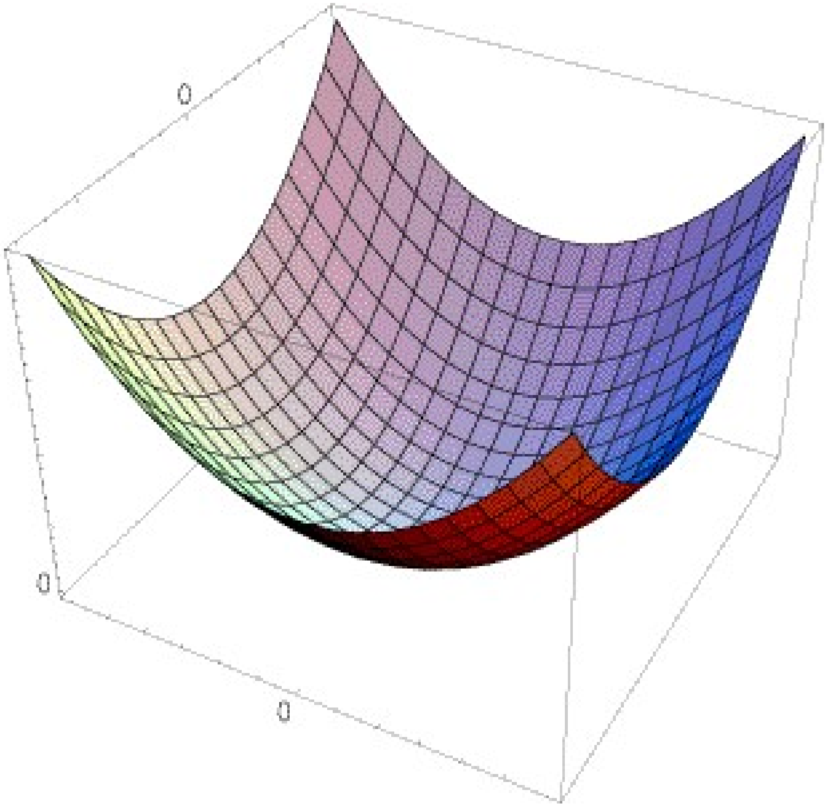
Robert Collins
CSE486, Penn State

Interpreting the second moment matrix

The surface *E*(*u*,*v*) is locally approximated by a quadratic form. Let's try to understand its shape.

$$E(u,v) \approx [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

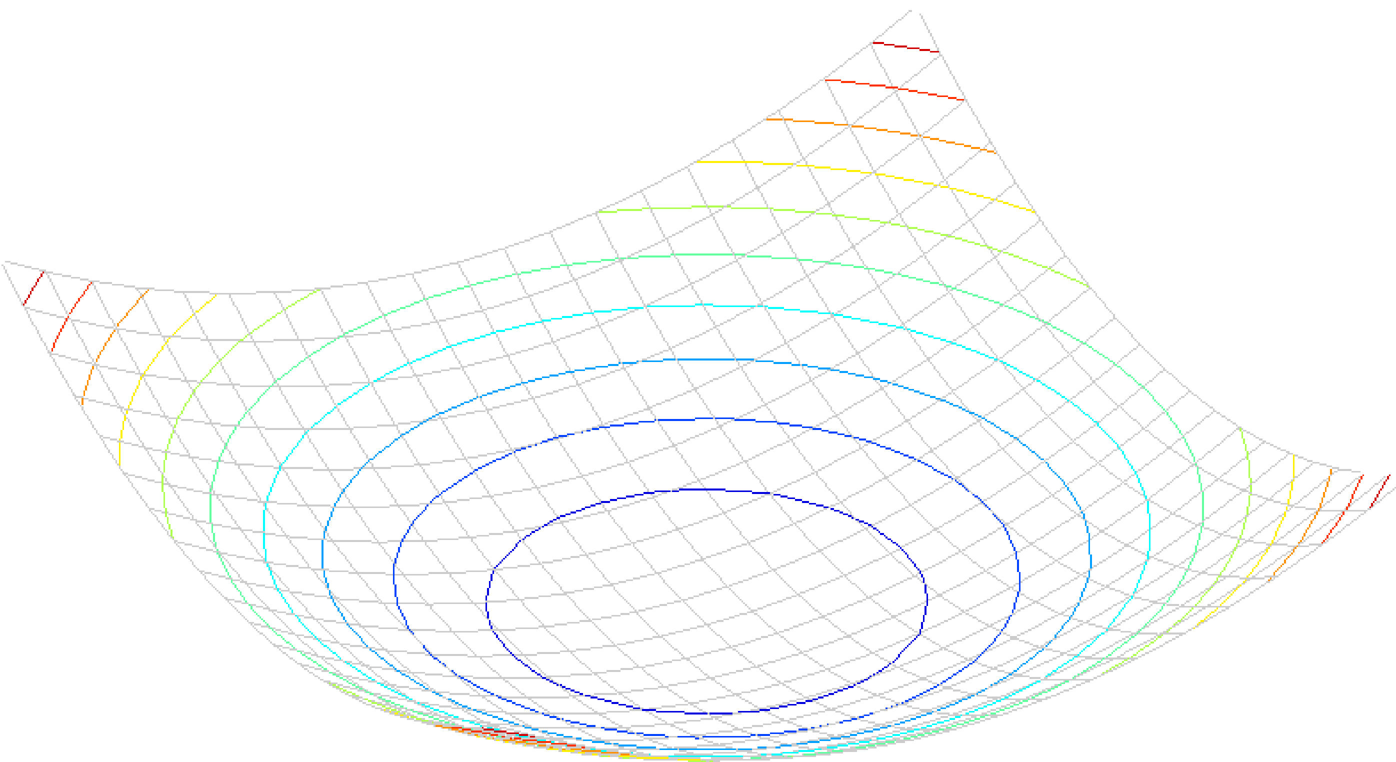
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



James Hays

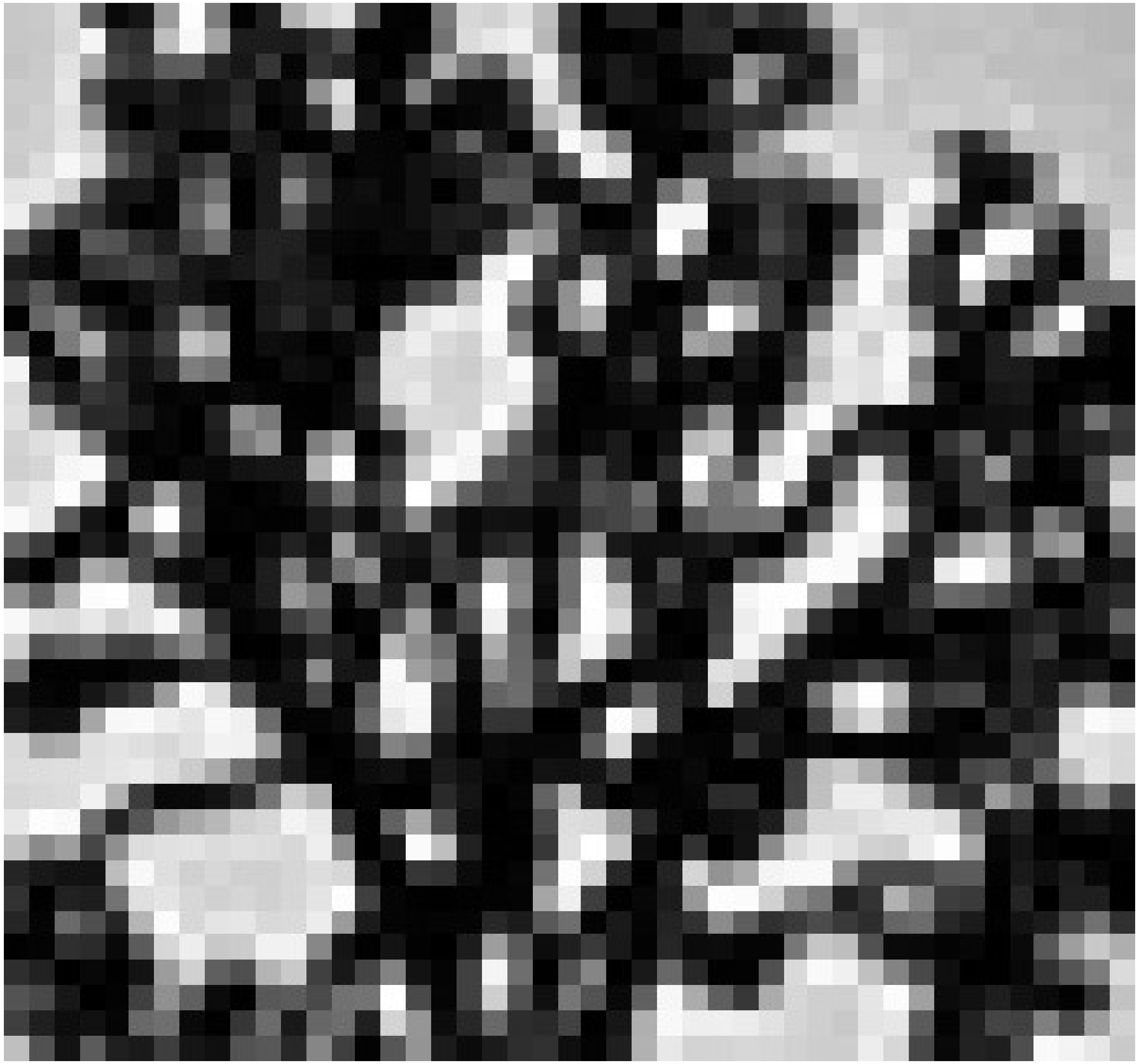
Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $\quad [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$



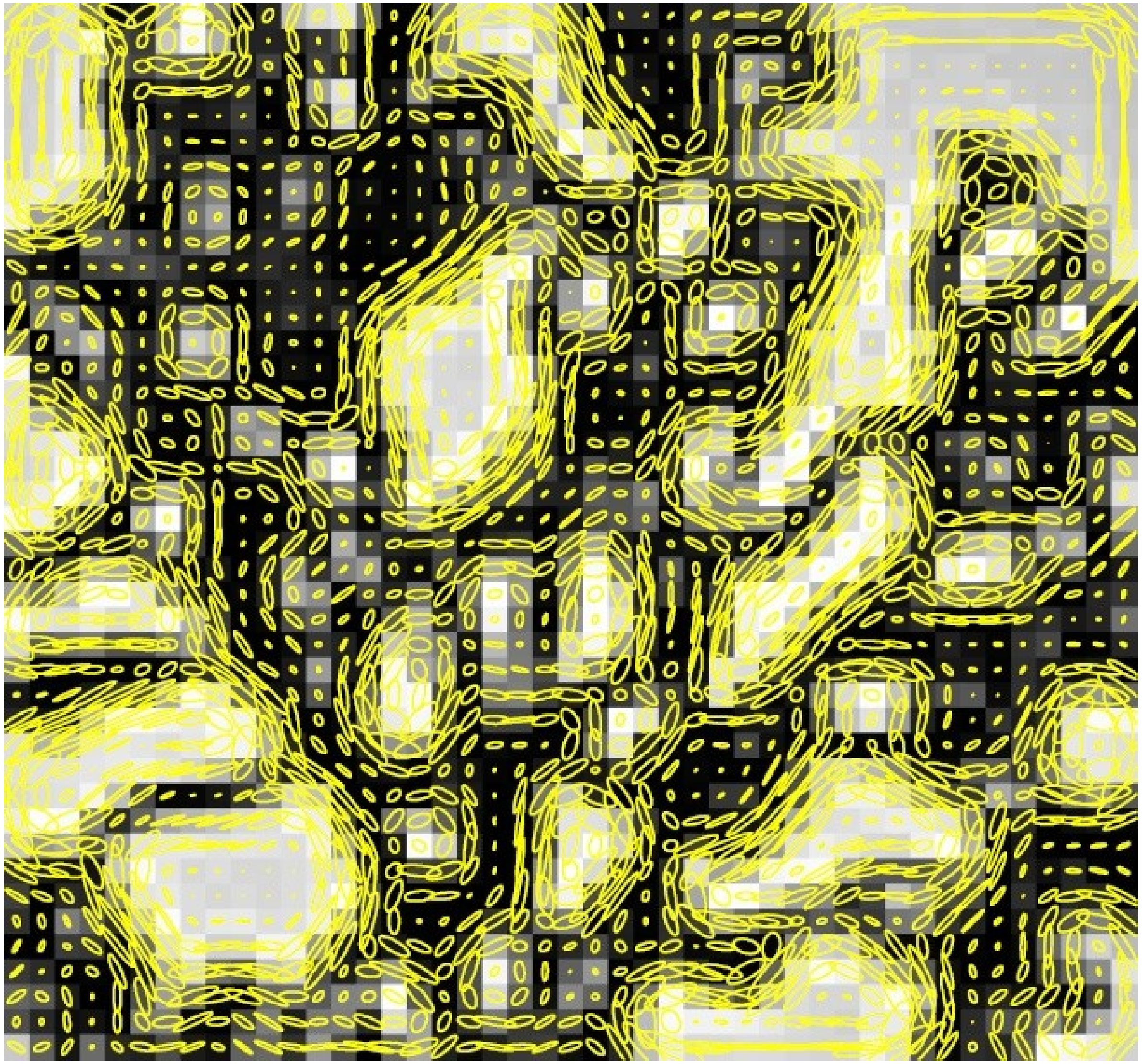
James Hays

Visualization of second moment matrices



James Hays

Visualization of second moment matrices



James Hays

Interpreting the second moment matrix

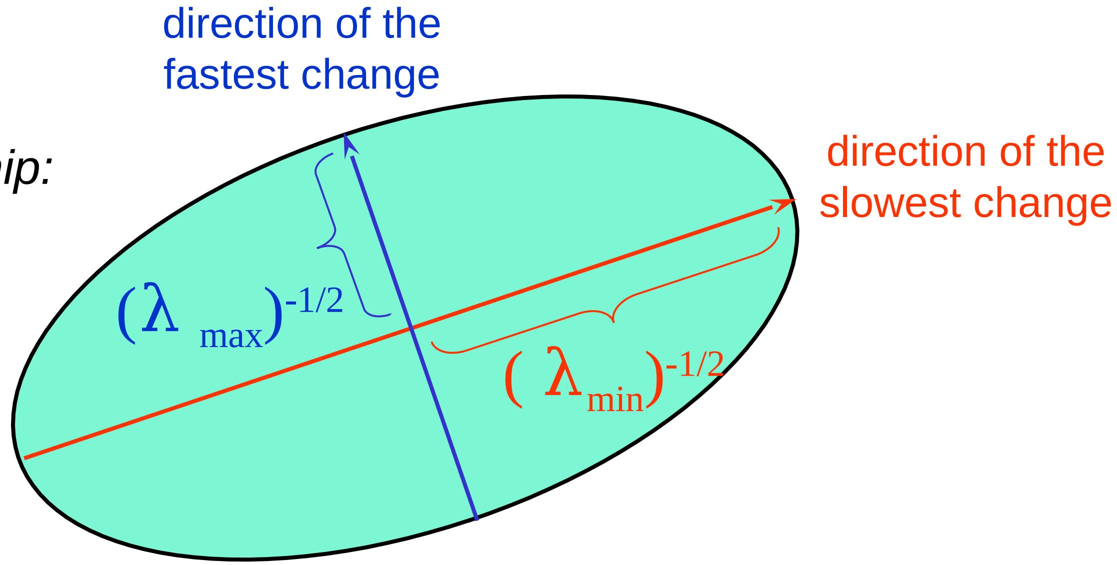
Consider a horizontal "slice" of $E(u, v)$: $[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

Diagonalization of M: $M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$

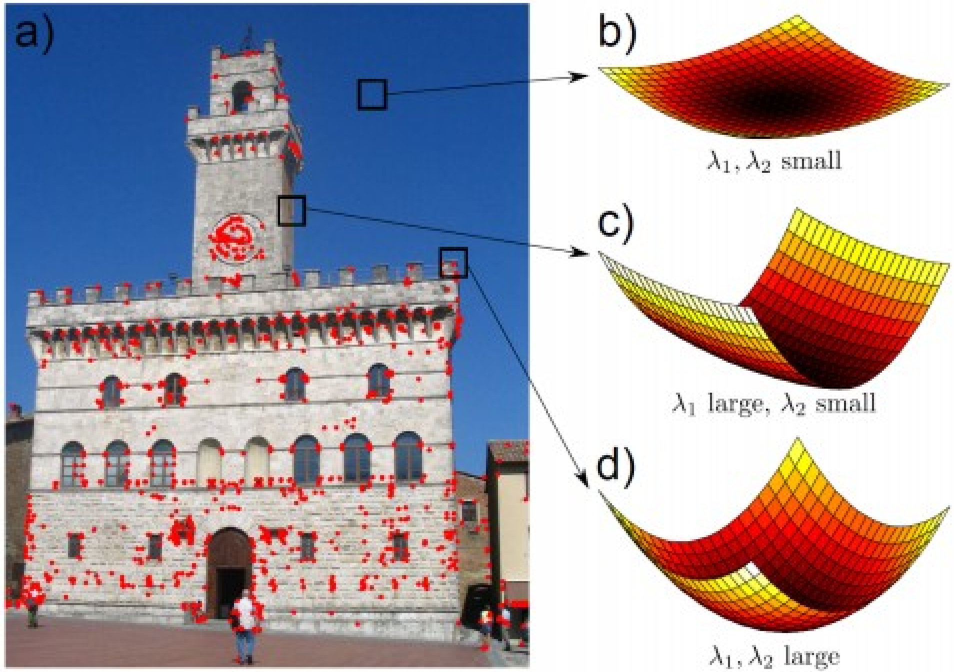
The axis lengths of the ellipse are determined by the eigenvalues, and the orientation is determined by a rotation matrix .

Note inverse relationship: larger eigenvalue = steeper slope; smaller ellipse in visualization



James Hays

Interpreting the second moment matrix

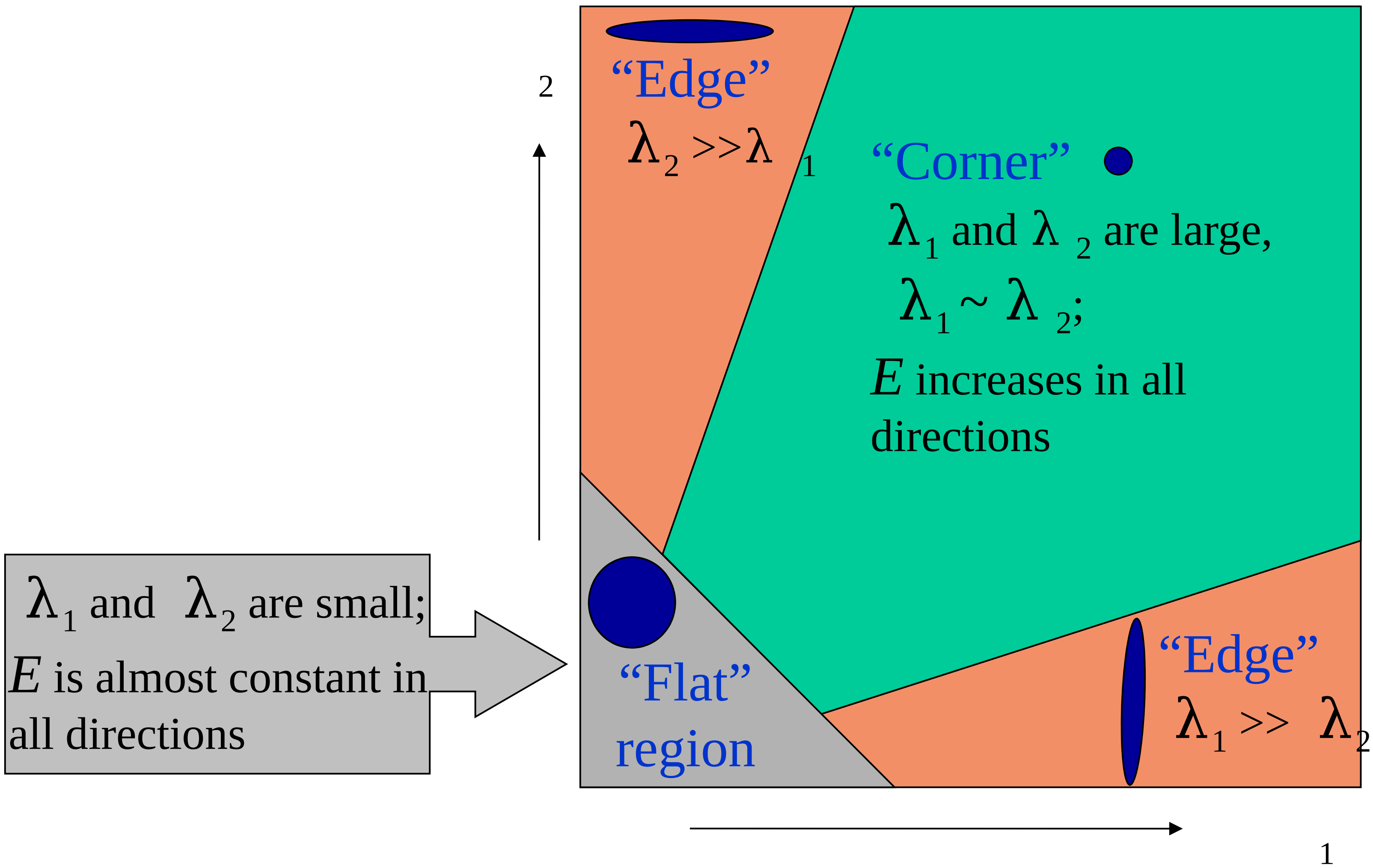


Make decision based on image structure tensor

$$\mathbf{S}_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{i+D} w_{mn} \begin{bmatrix} h_{ij}^2 & h_{ij}v_{ij} \\ h_{ij}v_{ij} & v_{ij}^2 \end{bmatrix}$$

James Hays

Classification of image points using eigenvalues of *M*

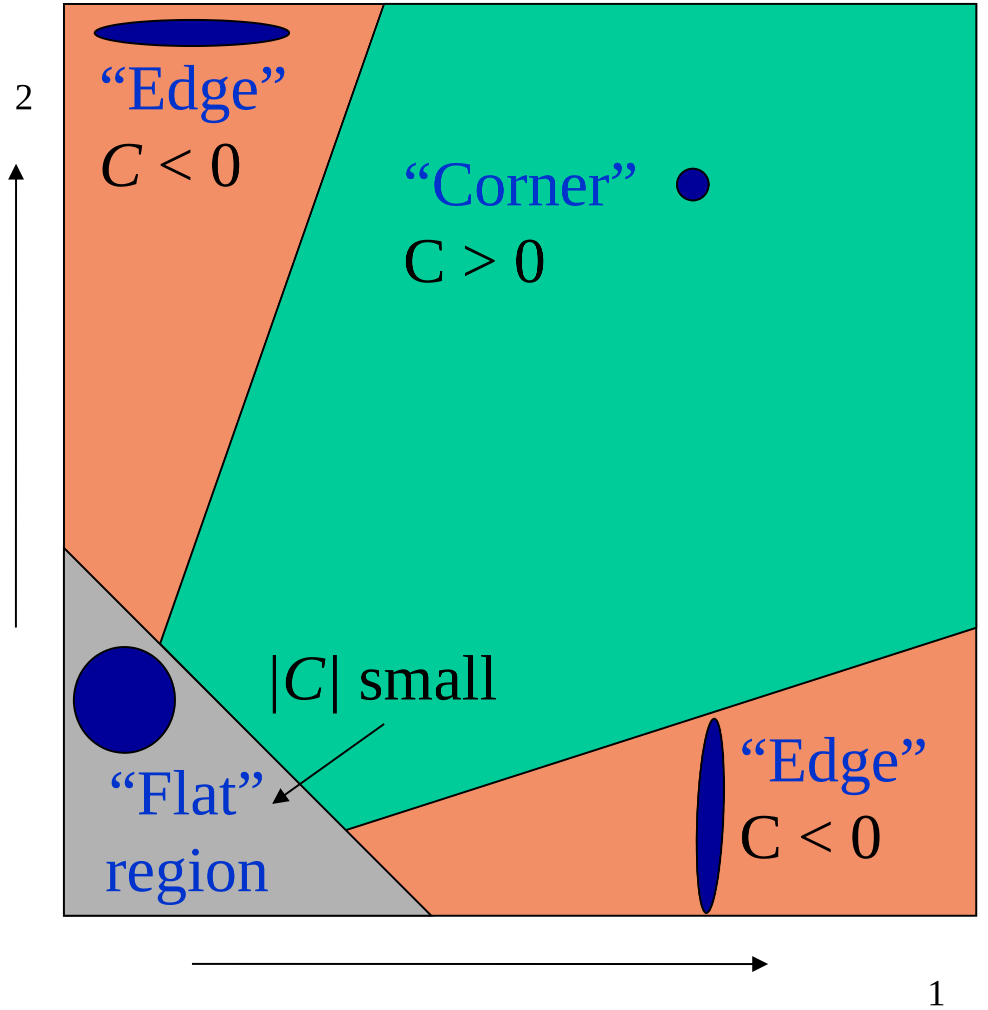


Classification of image points using eigenvalues of *M*

Cornerness

$$C = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)



Classification of image points using eigenvalues of *M*

Cornerness

$$C = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

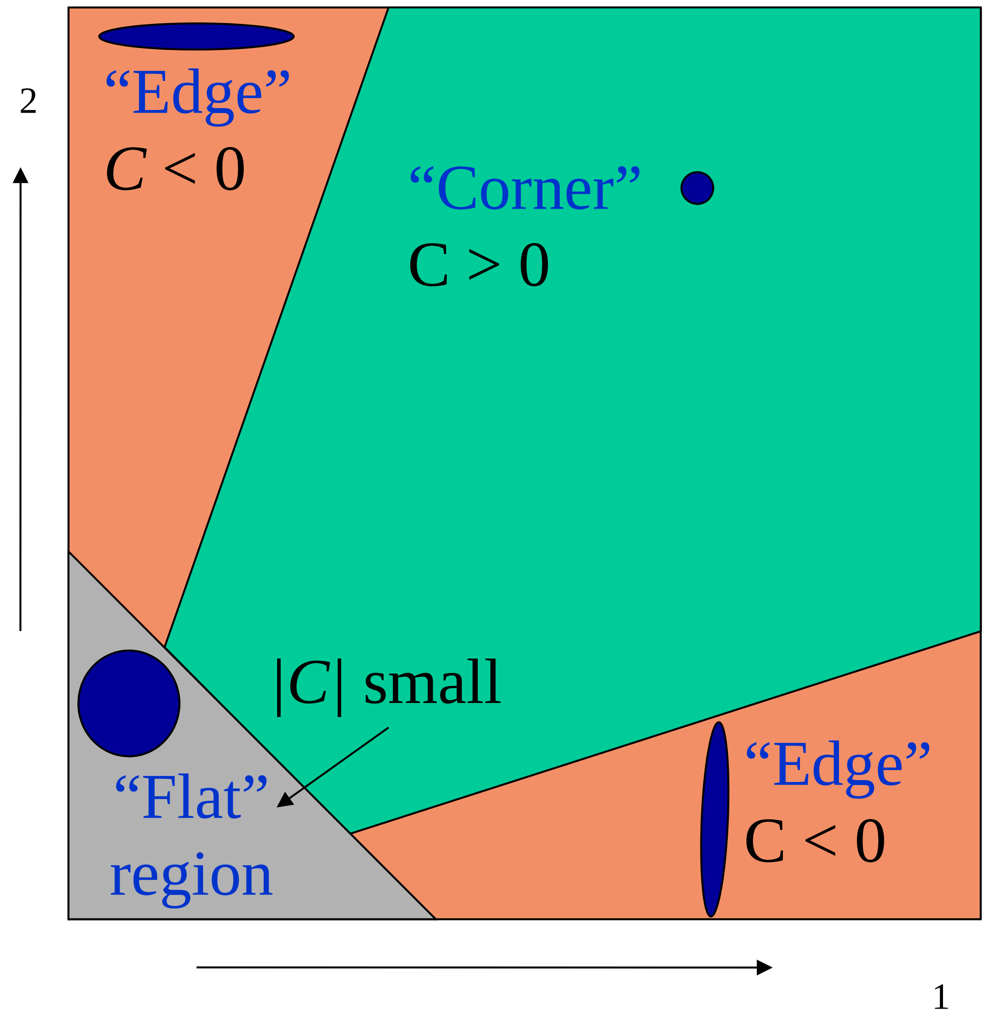
α: constant (0.04 to 0.06)

Remember your linear algebra:

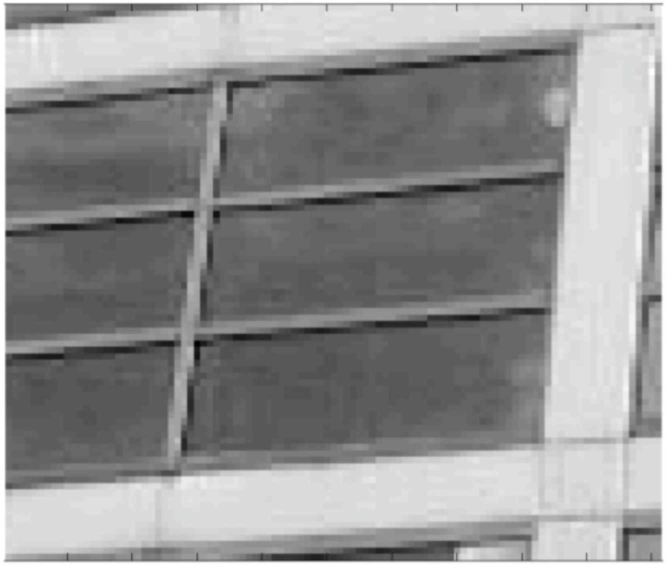
Determinant: $\det(A) = \prod_{i=1}^{n} \lambda_i = \lambda_1 \lambda_2 \cdots \lambda_n .$

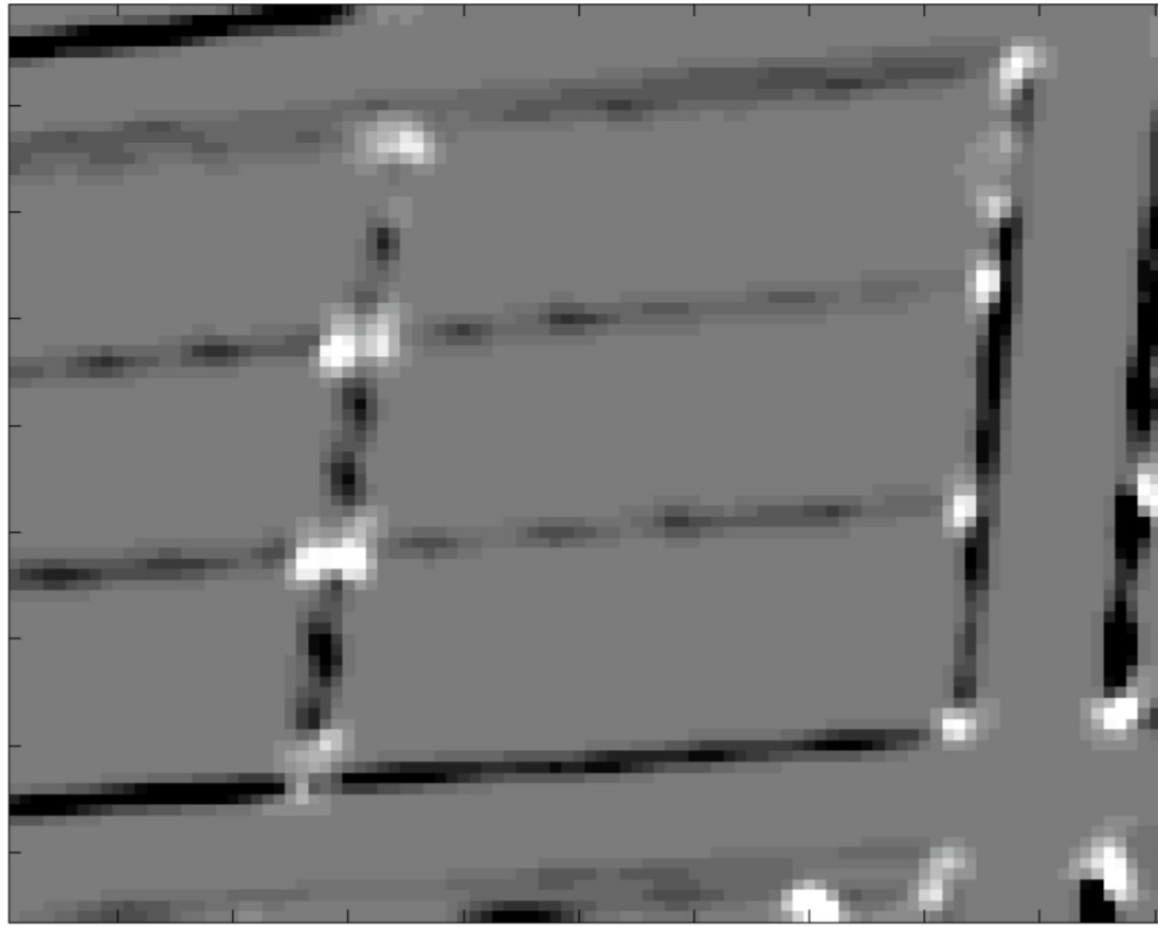
Trace: $\mathrm{tr}(A) = \sum_i \lambda_i .$

$$C = \det(M) - \alpha \, \mathrm{trace}(M)^2$$



Cornerness Response Example



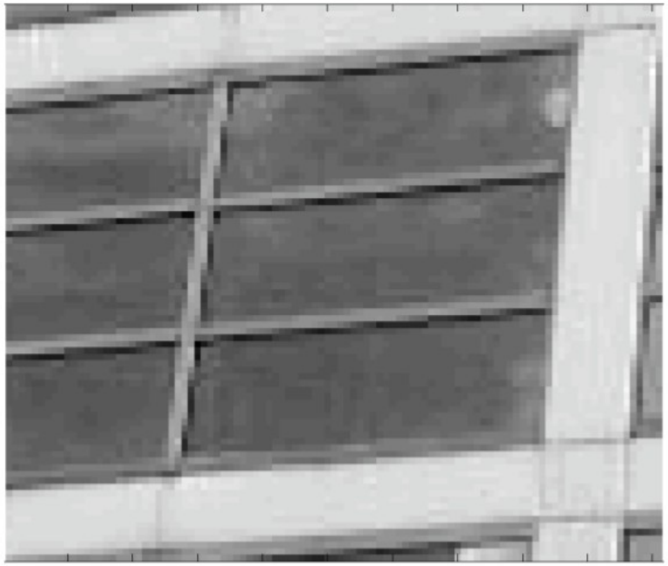


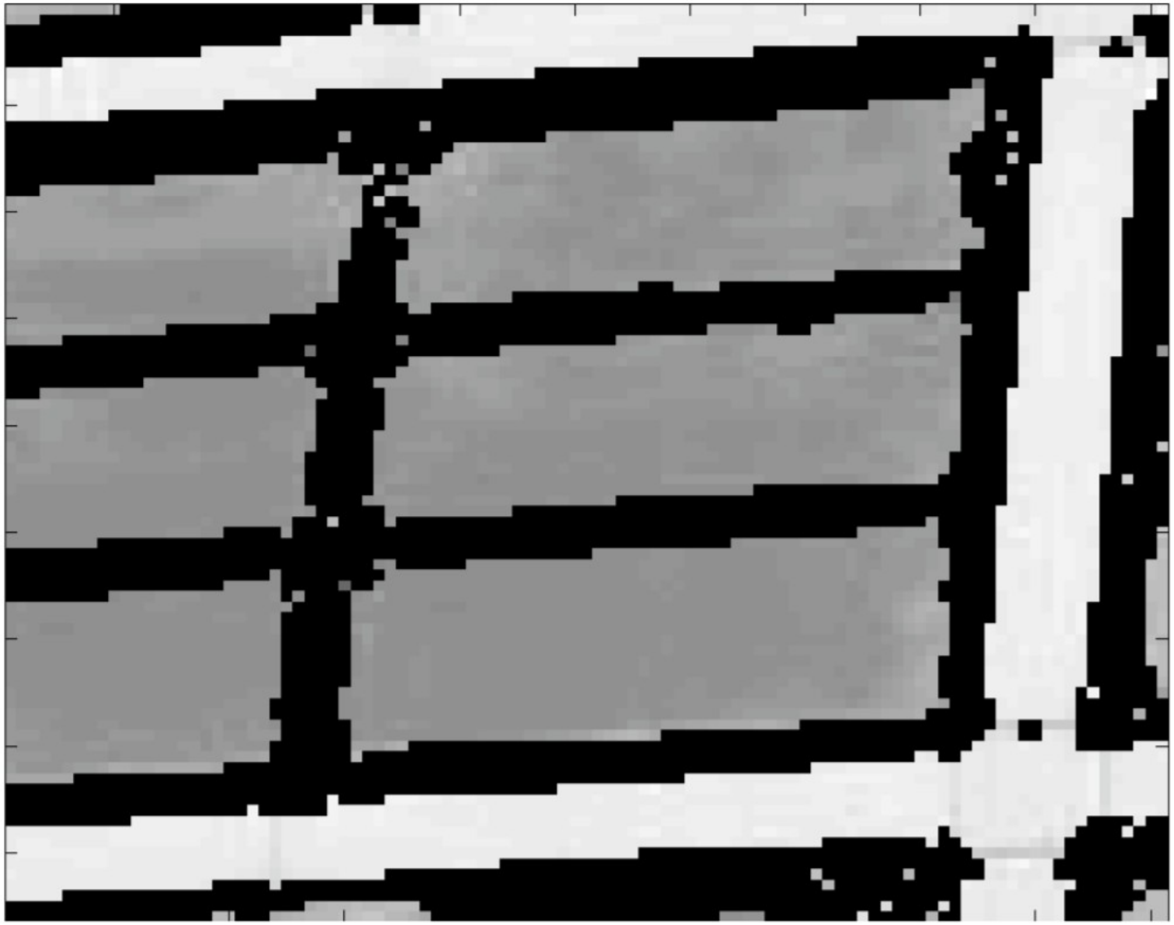
Harris c score.

Ix, Iy computed using Sobel operator

Windowing function w = Gaussian, sigma=1

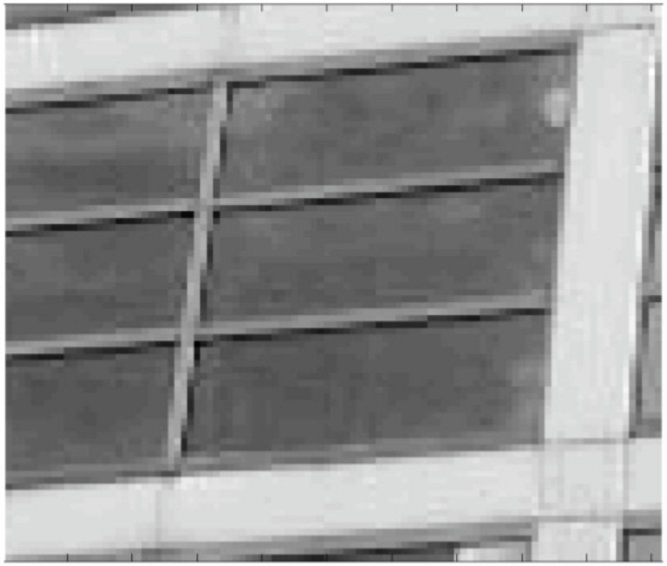
Cornerness Response Example

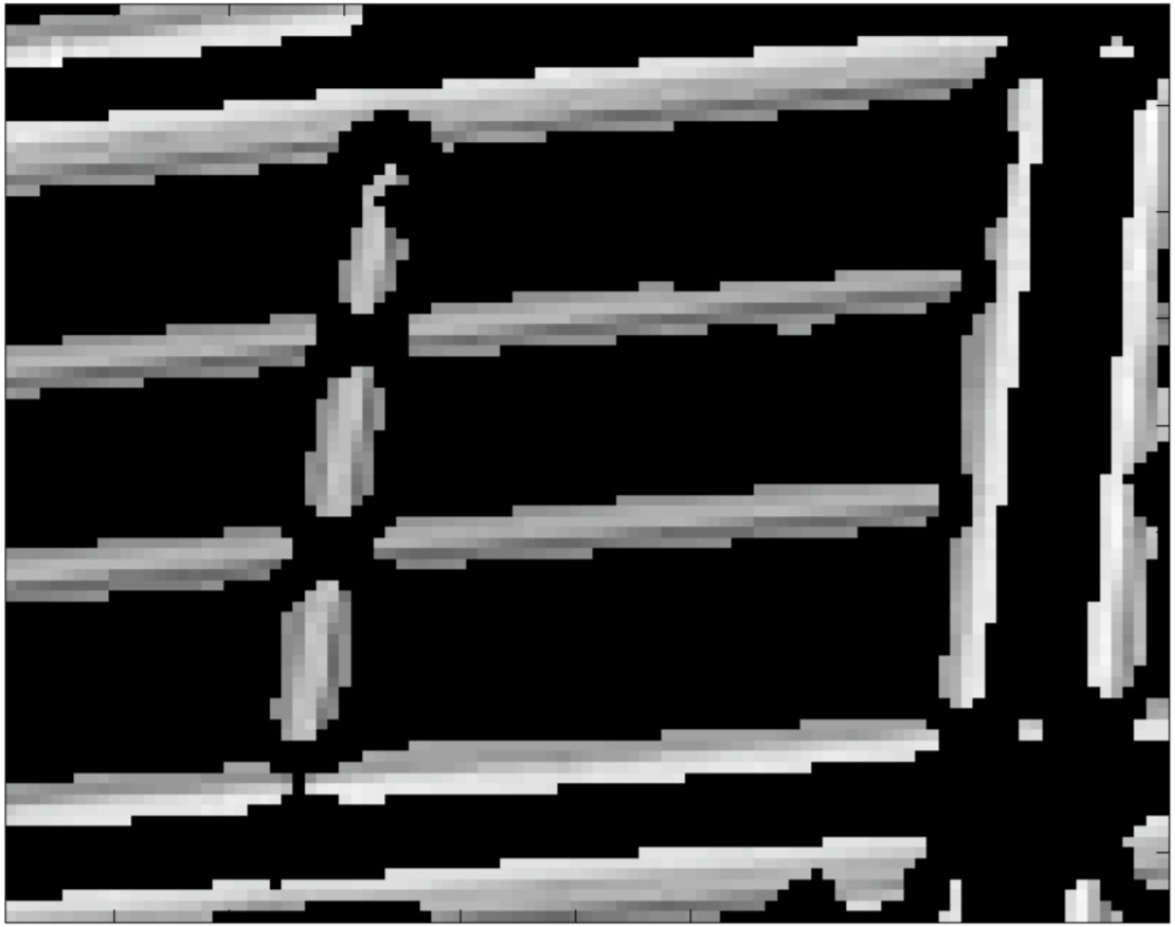




Threshold: $-10000 < c < 10000$
(neither edges nor corners)

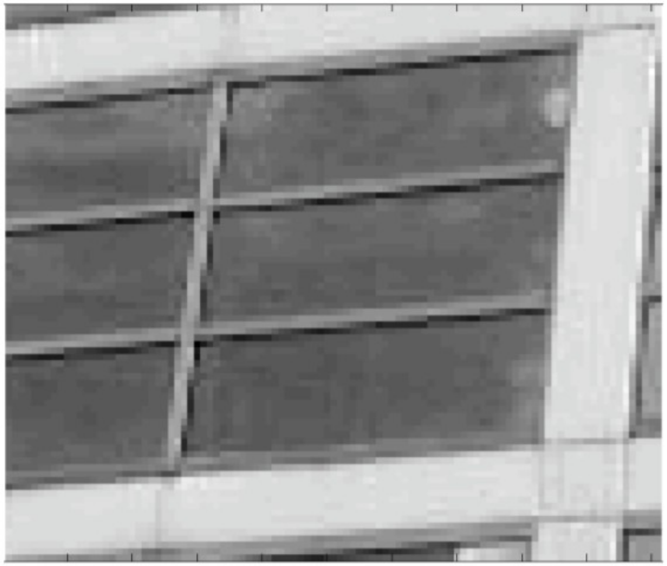
Cornerness Response Example

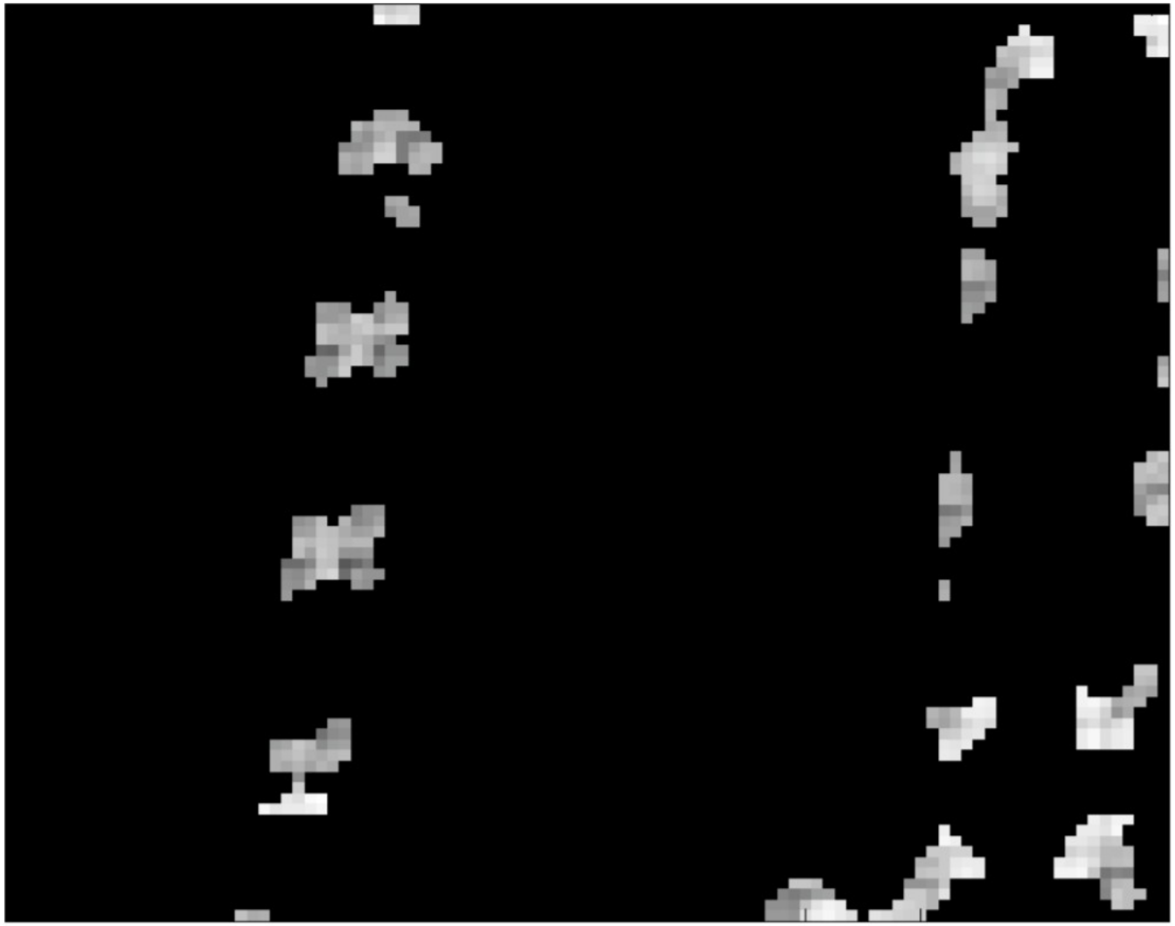




Threshold: c < -10000
(edges)

Cornerness Response Example





Threshold: > 10000
(corners)

Harris corner detector

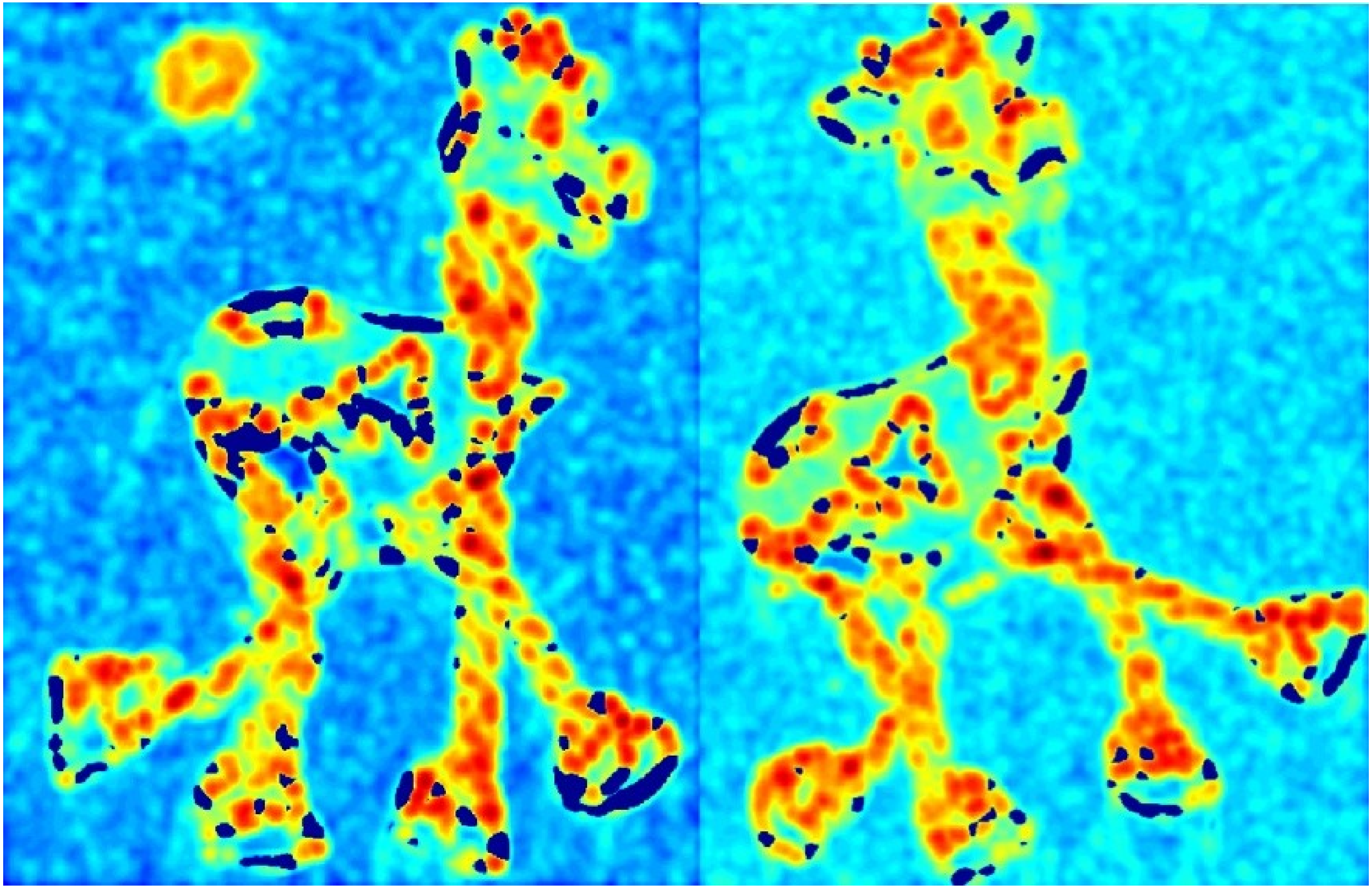
1) Compute *M* matrix for each window to recover a *cornerness* score .
   - Note: We can find *M* purely from the per-pixel image derivatives!
2) Threshold to find pixels which give large corner response (> threshold).
3) Find the local maxima pixels, i.e., suppress non-maxima.

C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

Harris Detector: Steps

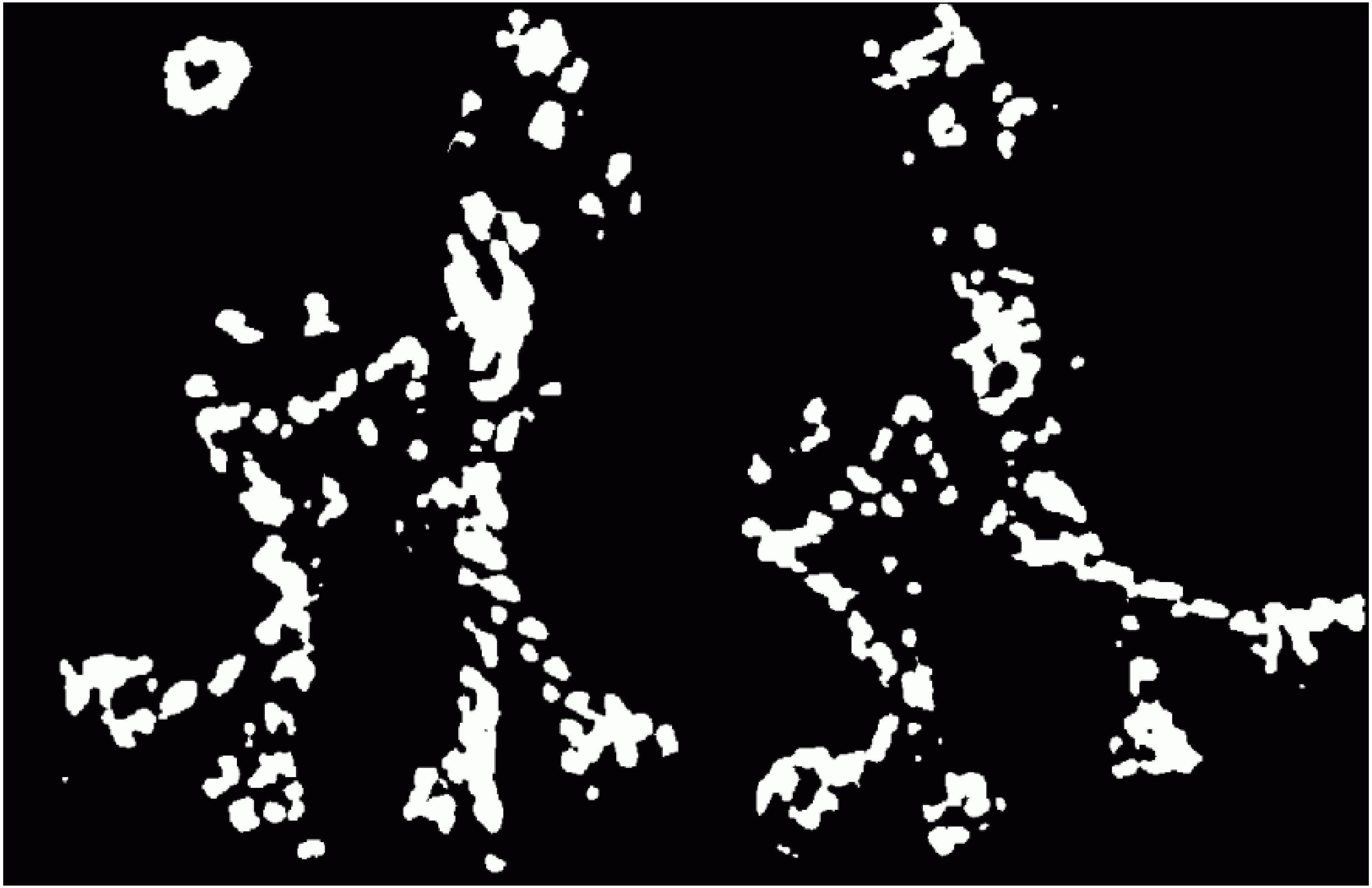
Harris Detector: Steps

Compute corner response



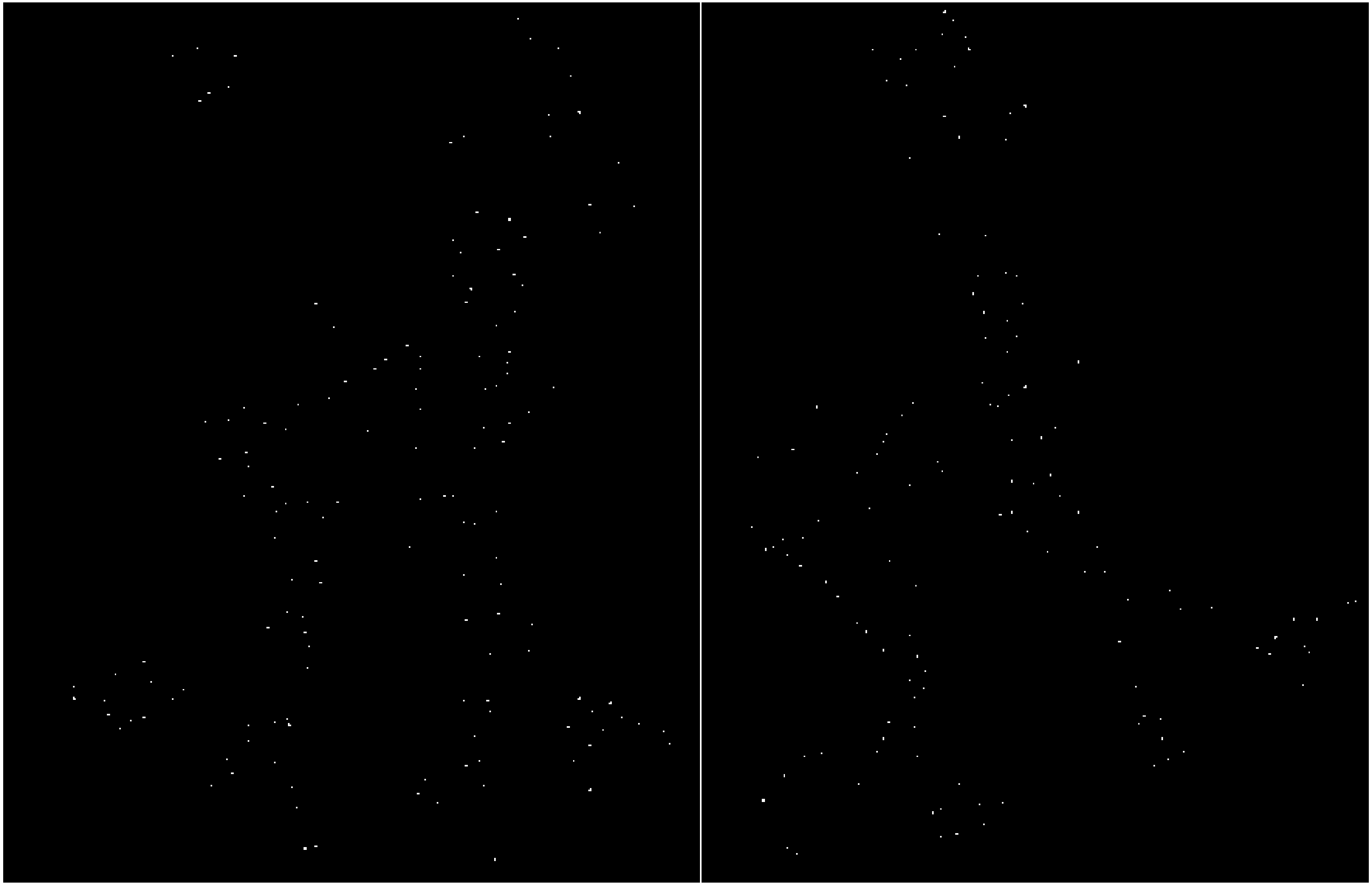
Harris Detector: Steps

Find points with large corner response: > threshold

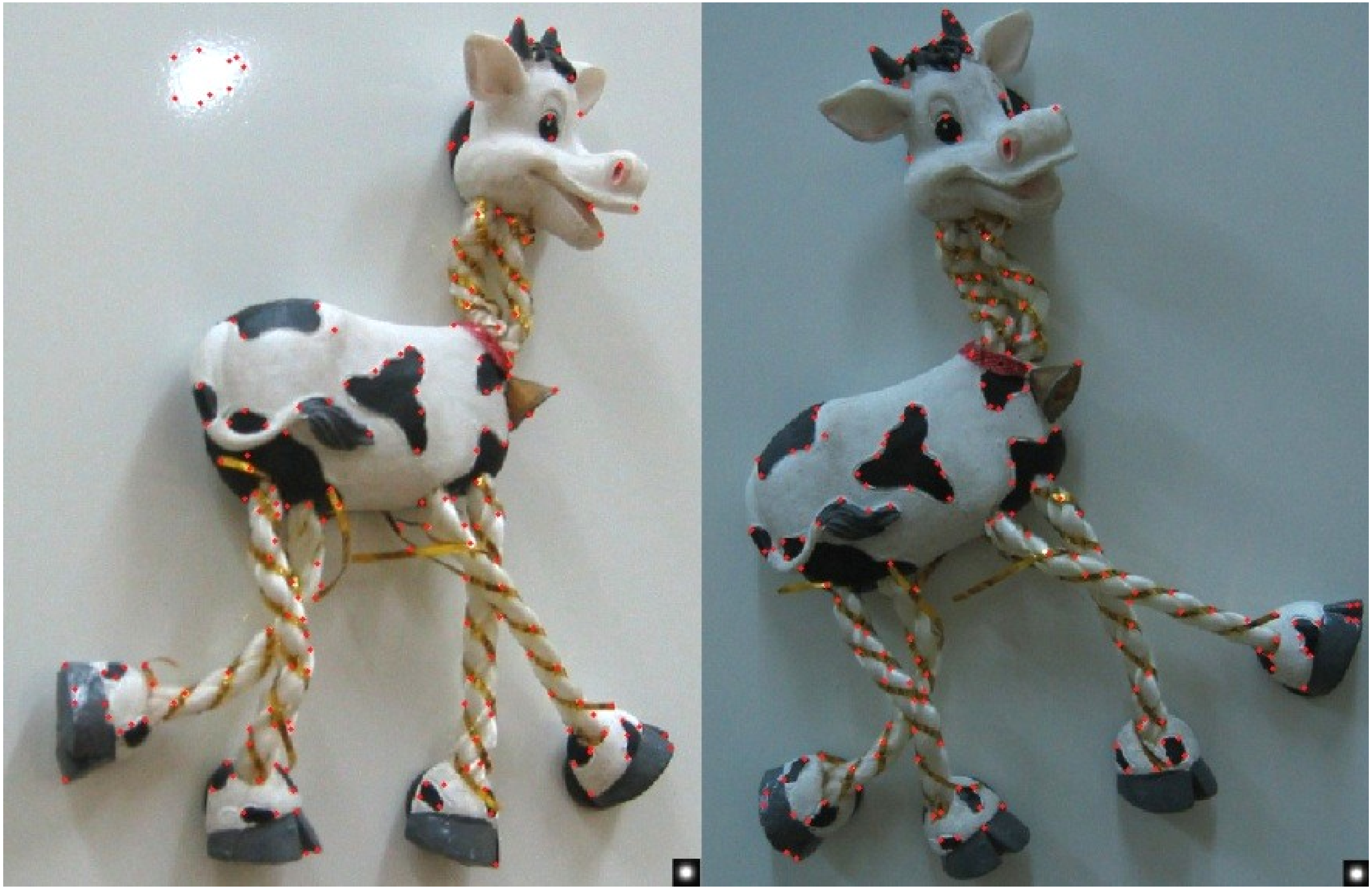


Harris Detector: Steps

Take only the points of local maxima of



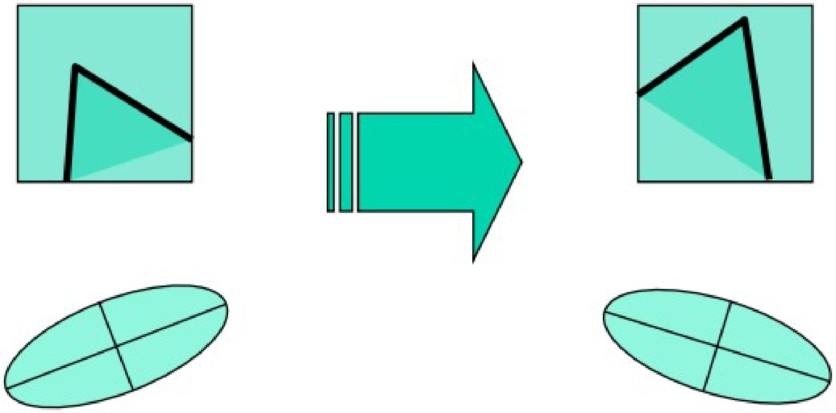
Harris Detector: Steps



Invariance and covariance

Invariance (does not change in different transformed images): e.g. the cornerness response value does not change when the image is transformed(rotated – translated)

Covariance (changes homogeneously such that we can still identify it using some process): if we have two transformed versions of the same image, the location and orientation of the feature are said to be covariant



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

Corner response R is invariant w.r.t. rotation and *corner location and orientation* are covariant