Local features

370: Intro to Computer Vision

Subhransu Maji

March 6 & 11, 2025

College of
INFORMATION AND
COMPUTER SCIENCES

UMASS
AMHERST

Topics

Why extract features?

Corner detector

Scale-invariant feature detector (or blob detector)

Why extract features?

Motivation: panorama stitching

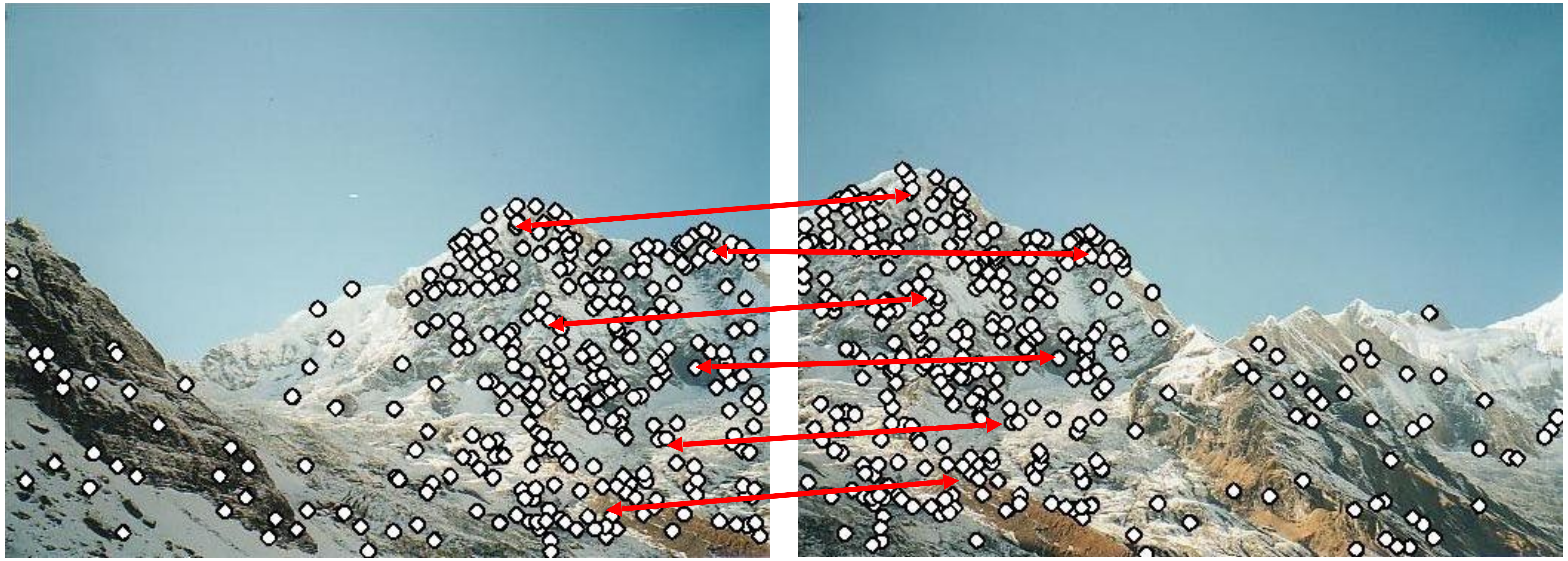
- We have two images – how do we combine them?

Slide credit: L. Lazebnik

Why extract features?

Motivation: panorama stitching

- We have two images – how do we combine them?



Step 1: extract features

Step 2: match features

Slide credit: L. Lazebnik

Why extract features?

Motivation: panorama stitching

- We have two images – how do we combine them?

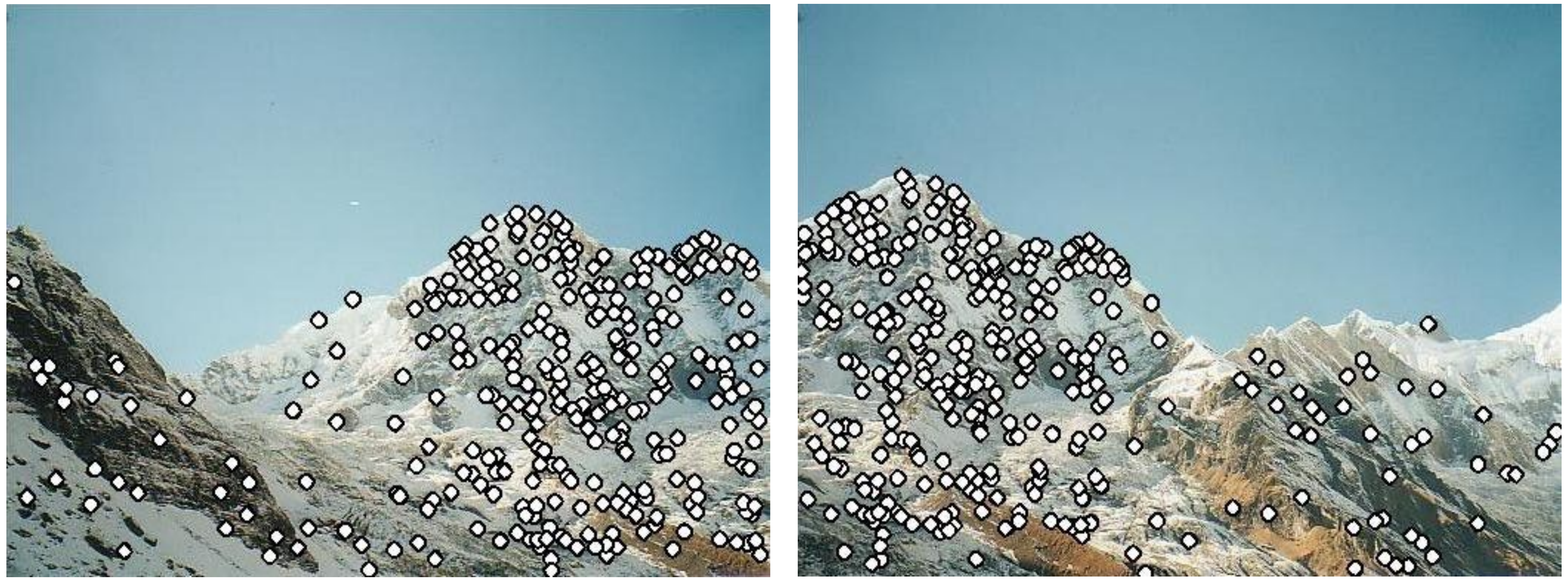
Step 1: extract features

Step 2: match features

Step 3: align images

Slide credit: L. Lazebnik

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

Slide credit: L. Lazebnik

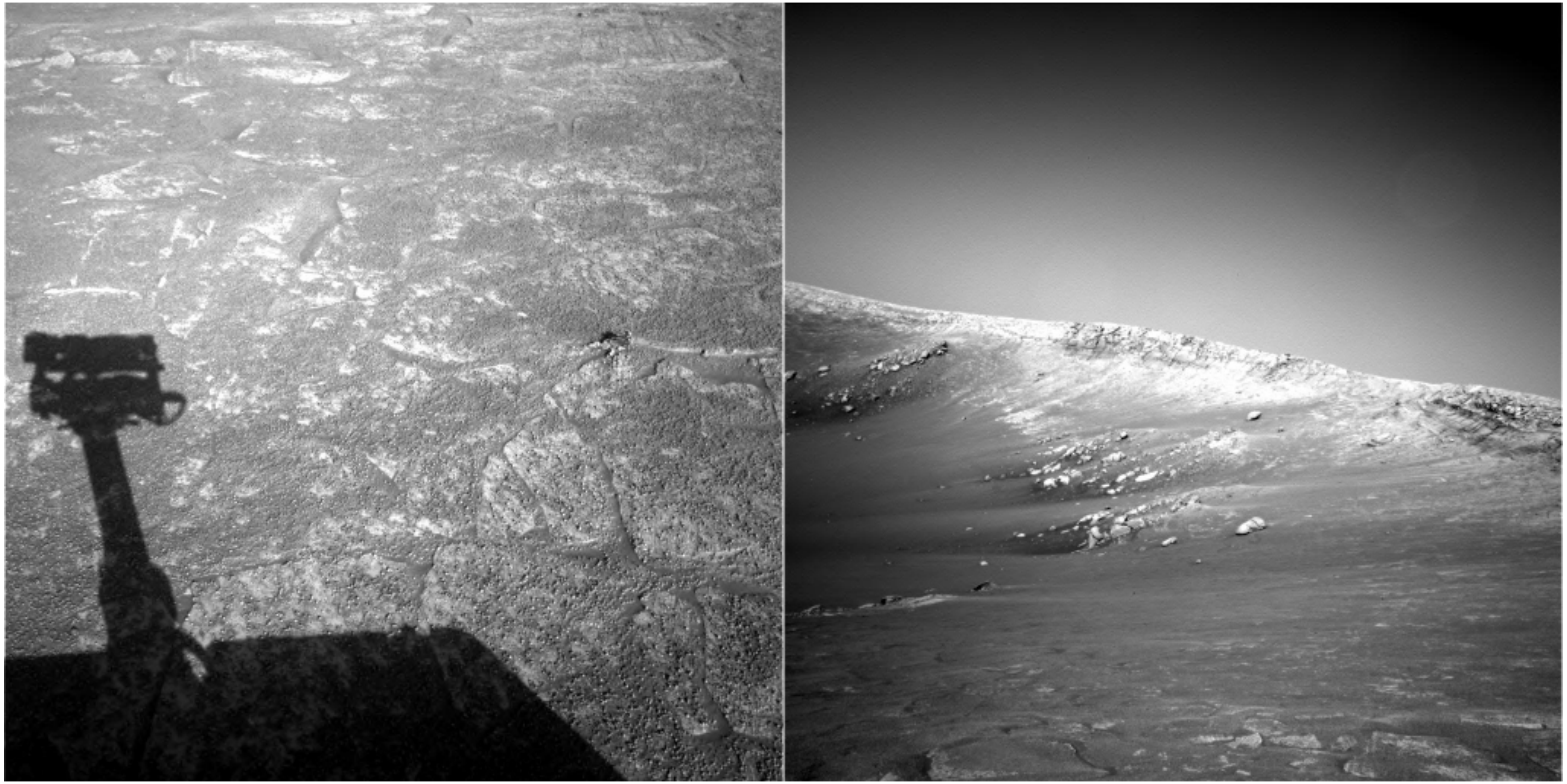
Applications

Feature points are used for:

- Image alignment
- 3D reconstruction
- Motion tracking
- Robot navigation
- Indexing and database retrieval
- Object recognition

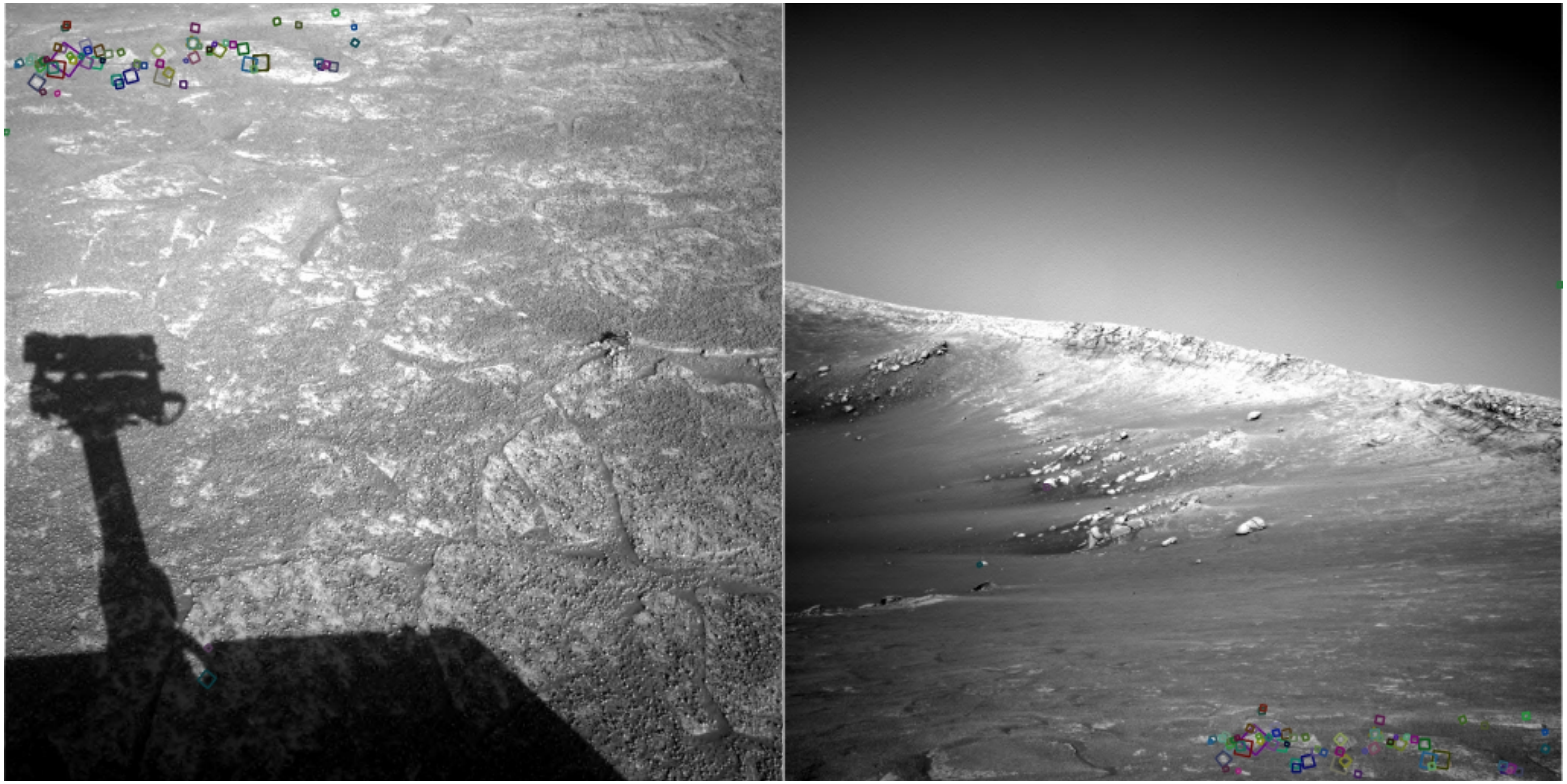
Slide credit: L. Lazebnik

A hard feature matching problem



NASA Mars Rover images

Answer below (look for tiny colored squares…)



NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

Feature extraction: Corners

Slide credit: L. Lazebnik

Corner detection: Attempt one

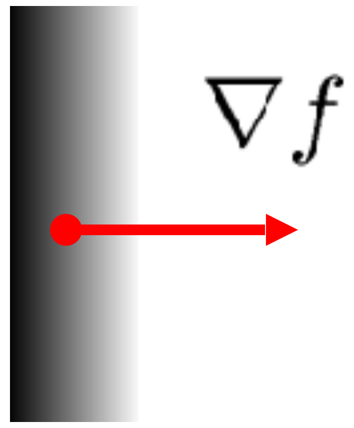
A corner is the intersection of two edges

We know how to detect edges

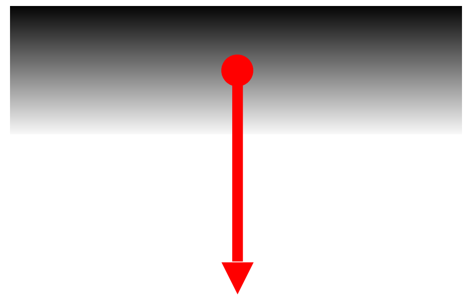
Corner detector (attempt #1)

- Detect edges in images (G_x and G_y)
- Find places where both G_x and G_y are high

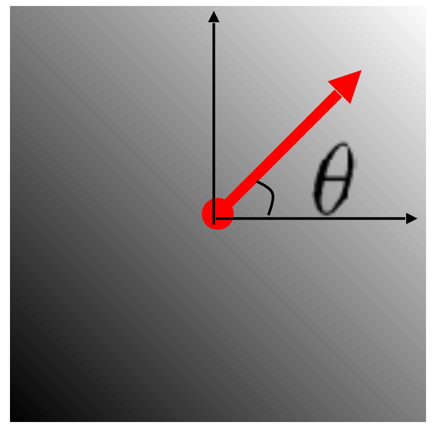
Problem: also finds slanted edges!



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

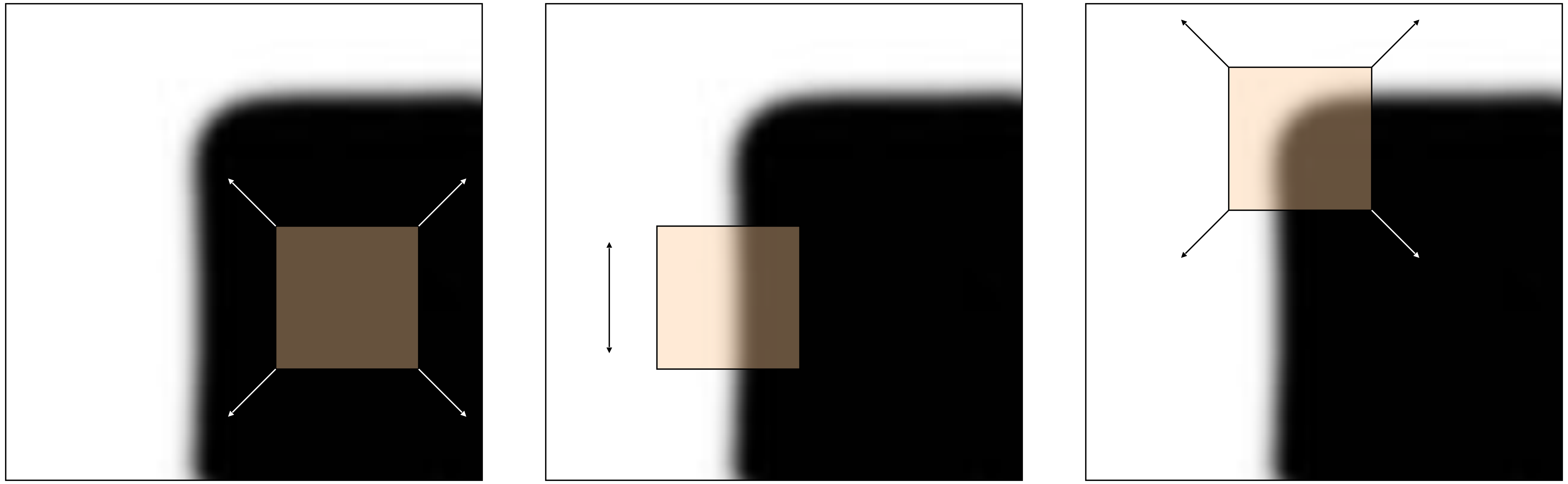


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Corner detection: Attempt two

We should easily recognize the corners by looking through a small window

Shifting a window in *any direction* should give *a large change* in intensity at a corner



"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

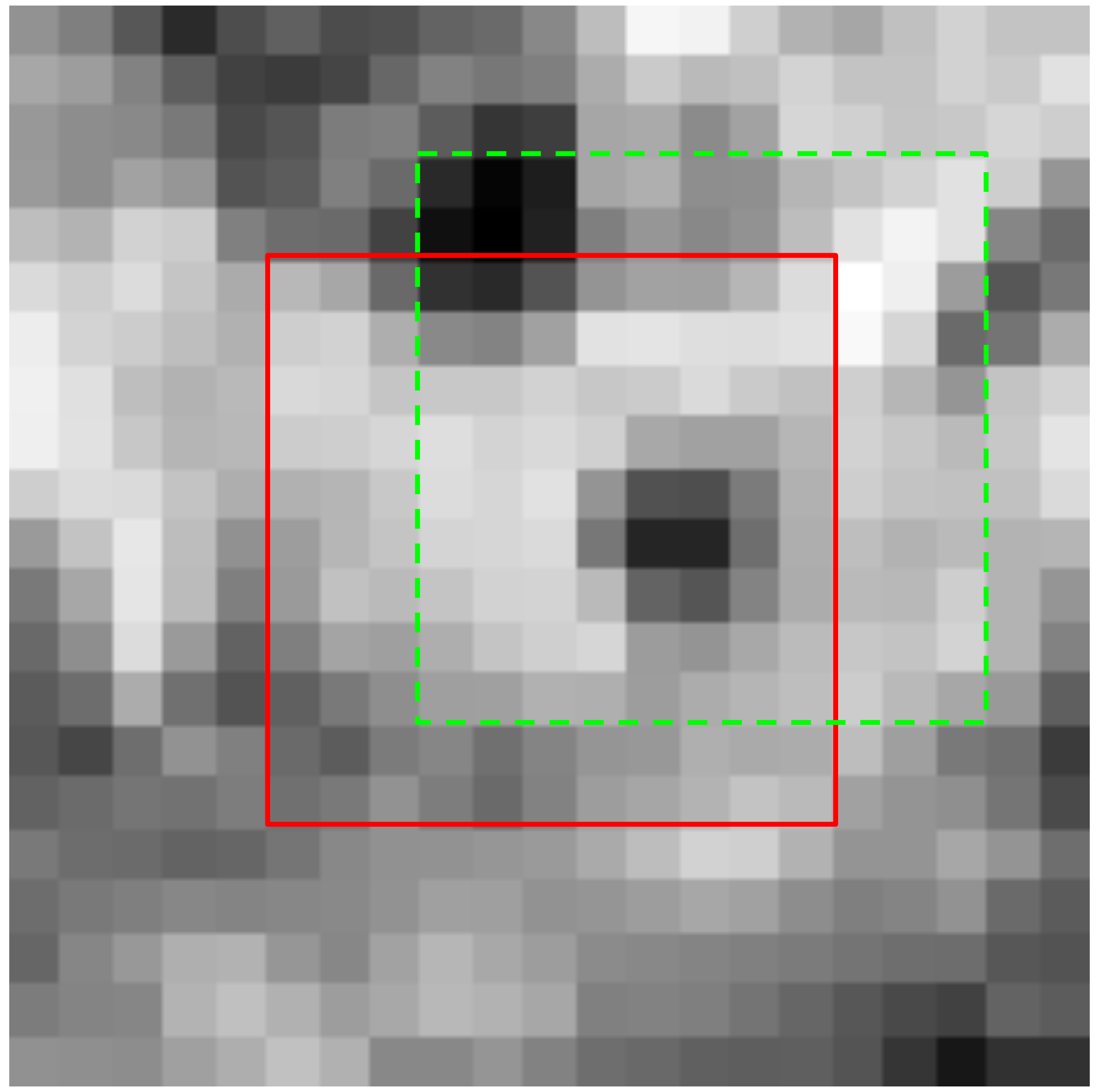
"corner":
significant change
in all directions

Corner detection: Mathematics

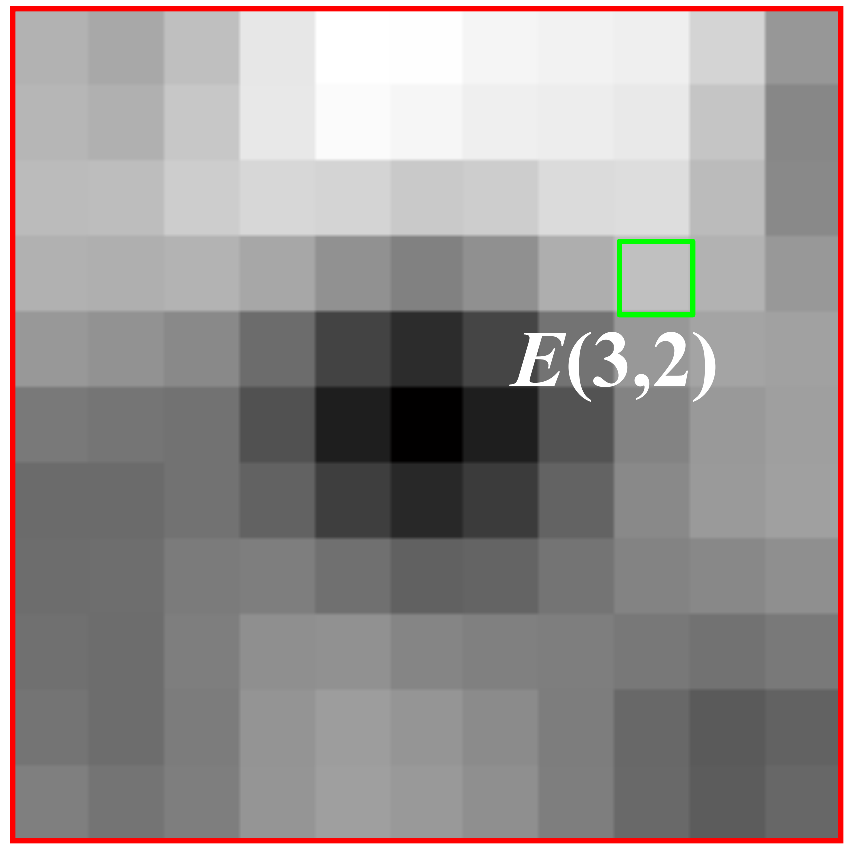
The change in appearance of window *W* for the shift [*u,v*]:

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

$I(x,y)$



$E(u,v)$

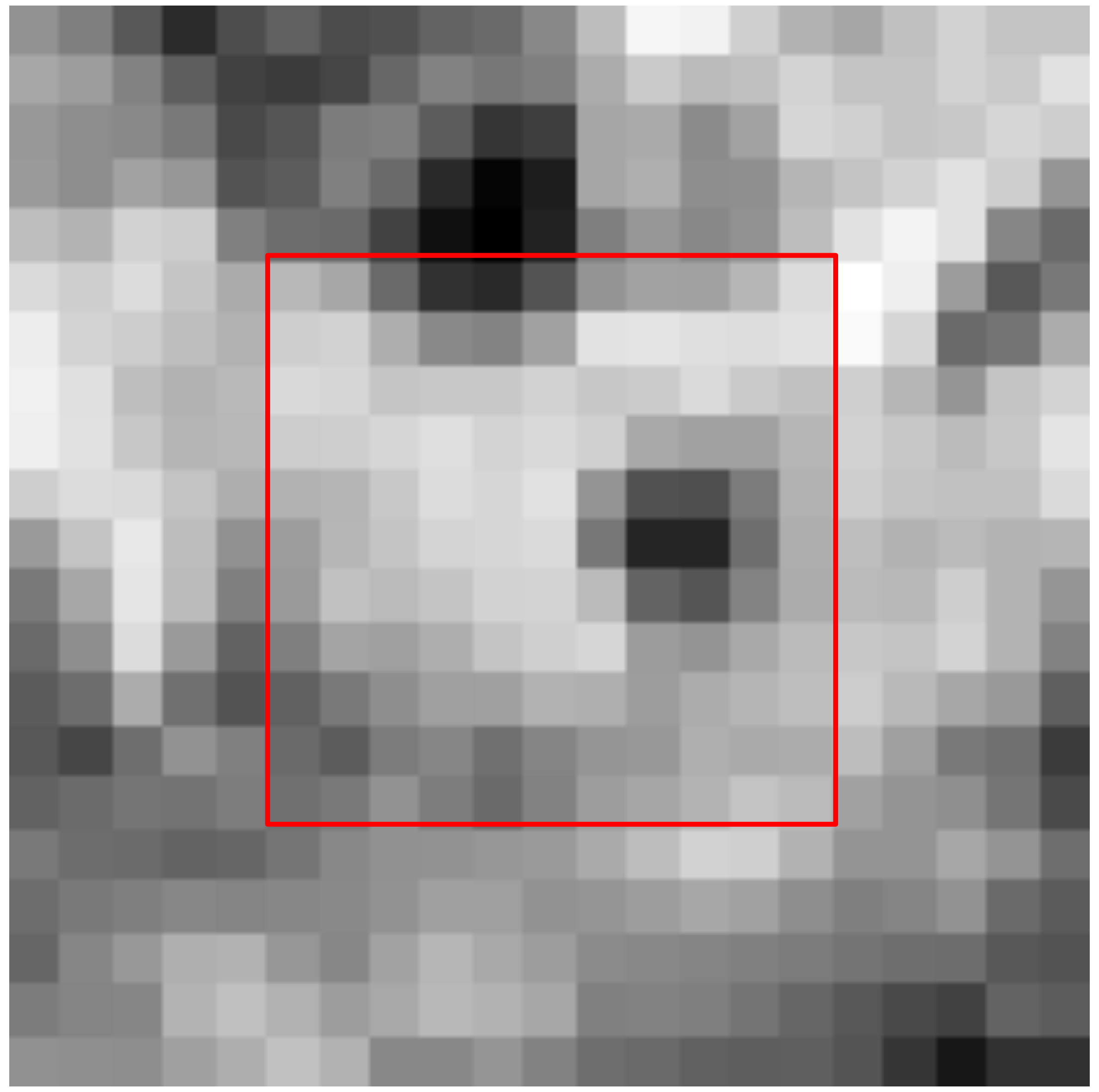


Corner detection: Mathematics

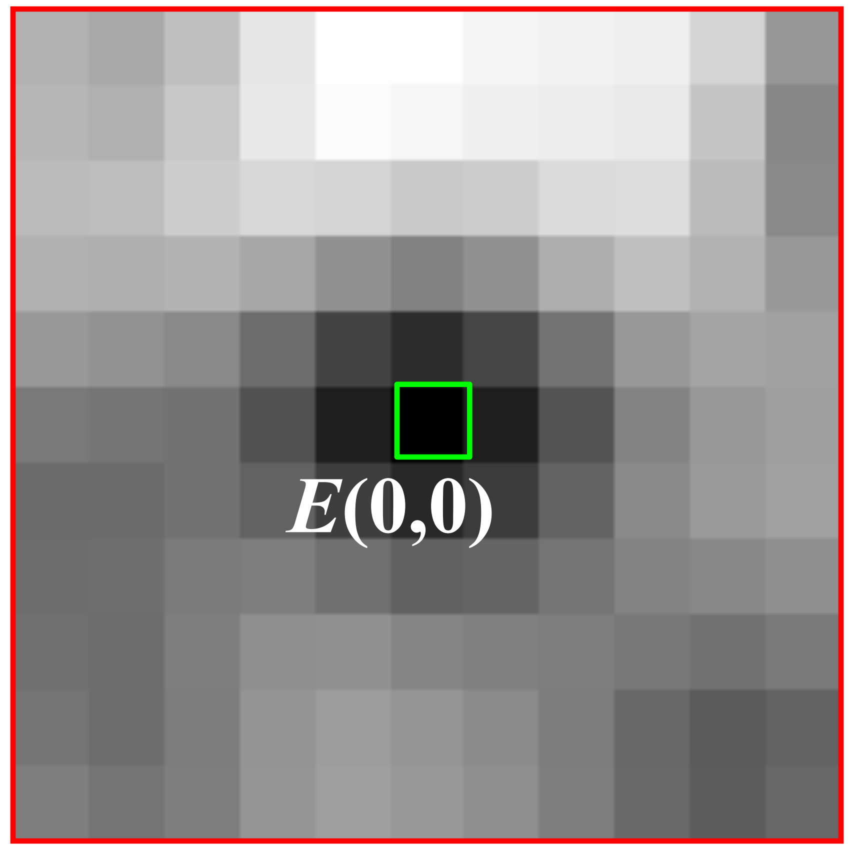
The change in appearance of window *W* for the shift [*u,v*]:

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

$I(x,y)$



$E(u,v)$



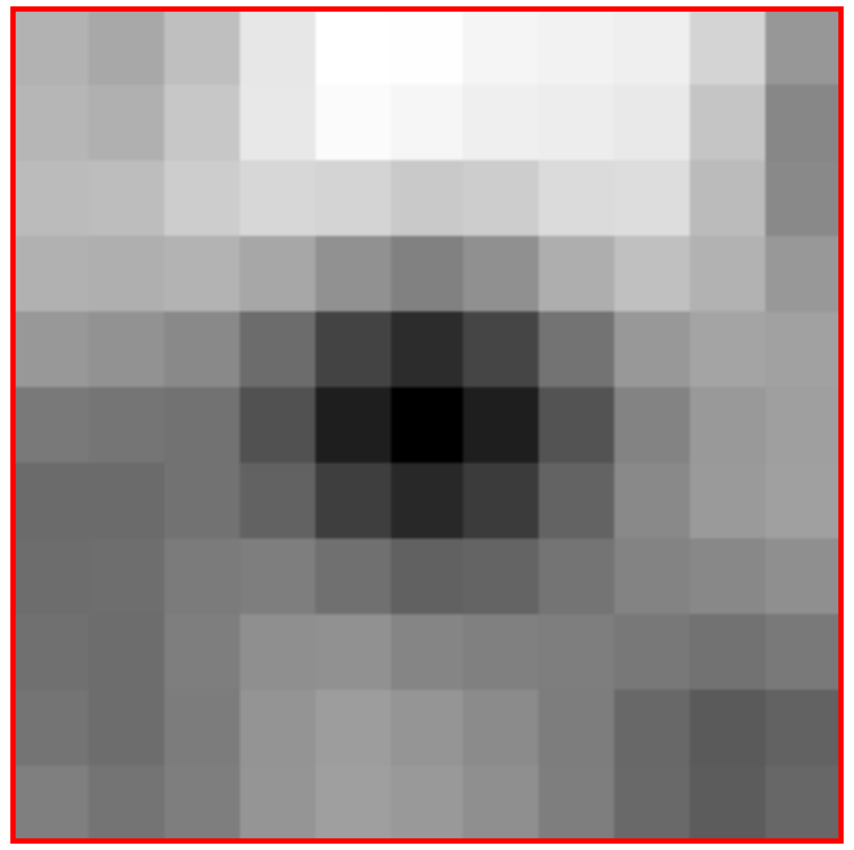
Corner detection: Mathematics

The change in appearance of window *W* for the shift [*u,v*]:

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

We want to find out how this function behaves for small shifts

$$E(u,v)$$



Corner detection: Mathematics

First-order Taylor approximation for small motions [*u*, *v*]:

$$I(x+u, y+v) = I(x,y) + I_x u + I_y v$$

Let's plug this into *E*(*u*,*v*)

$$\begin{aligned}
E(u,v) &= \sum_{(x,y)\in W} \left[I(x+u, y+v) - I(x,y)\right]^2 \\
&\simeq \sum_{(x,y)\in W} \left[I(x,y) + I_x u + I_y v - I(x,y)\right]^2 \\
&= \sum_{(x,y)\in W} \left[I_x u + I_y v\right]^2 \\
&= \sum_{(x,y)\in W} \left[I_x^2 u^2 + I_x I_y uv + I_y I_x uv + I_y^2 v^2\right]
\end{aligned}$$

Corner Detection: Mathematics

The quadratic approximation can be written as

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

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

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

(the sums are over all the pixels in the window *W*)

Interpreting the second moment matrix

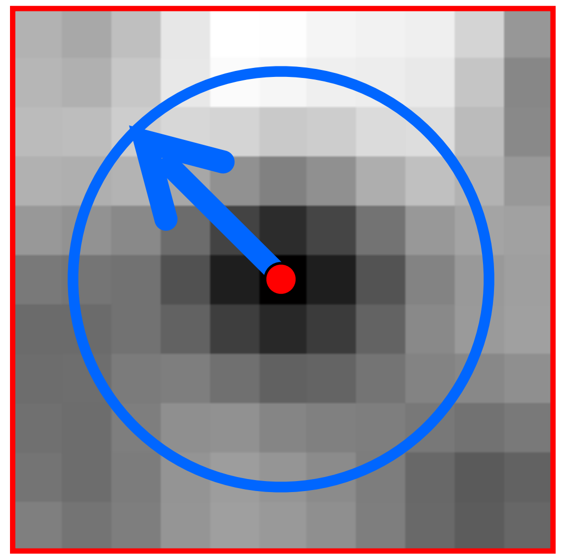
The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

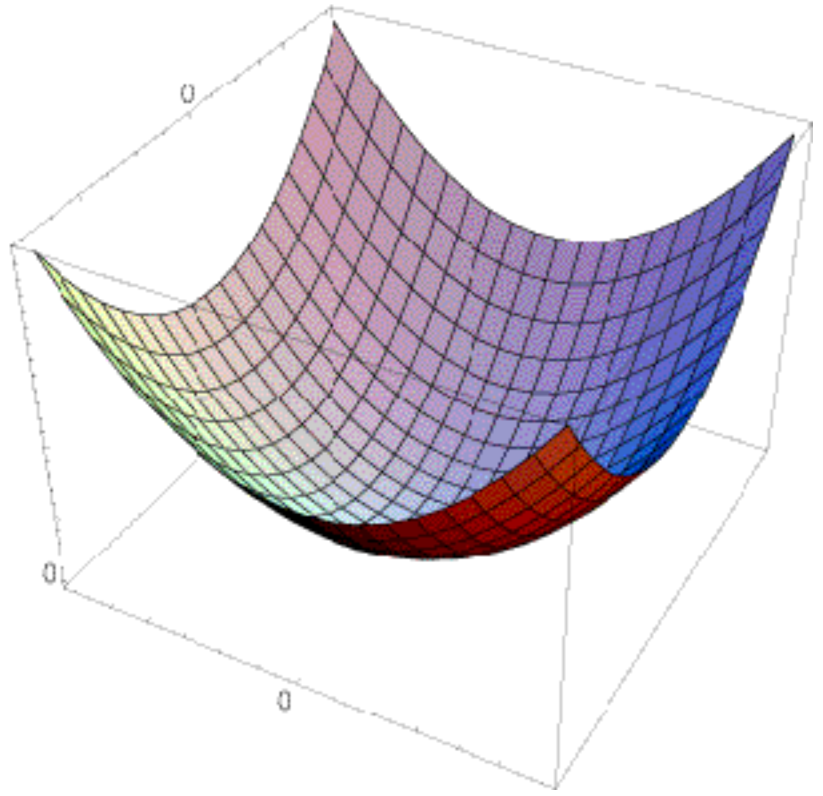
Specifically, in which directions does it have the smallest/greatest change?

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

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

$E(u, v)$





Interpreting the second moment matrix

First, consider the axis-aligned case
(gradients are either horizontal or vertical)

$$M = \begin{bmatrix} \sum_{x,y} I_x^2 & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y^2 \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

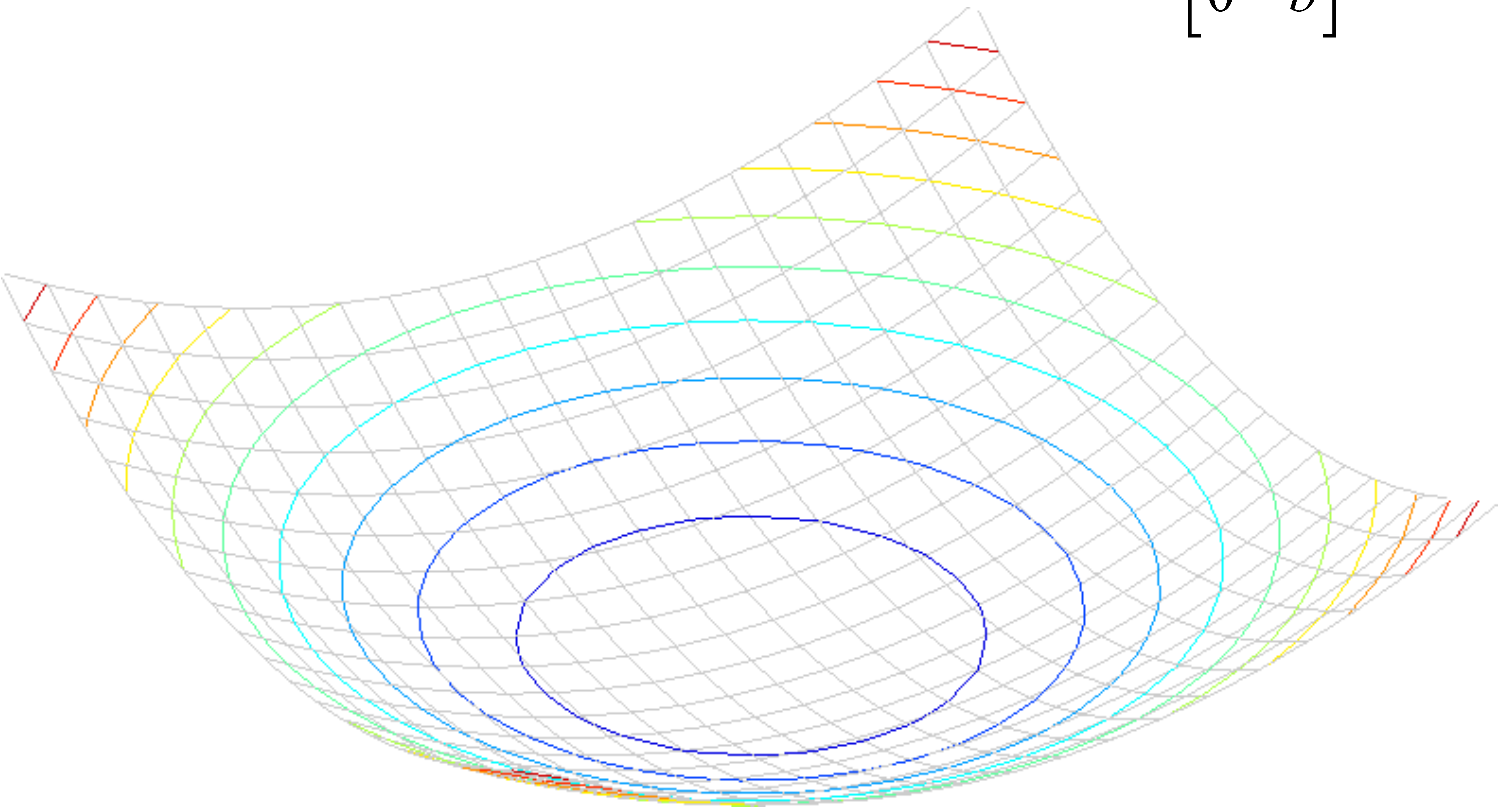
If either *a* or *b* is close to 0, then this is **not** a corner, so look for locations where both are large.

Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$:

$$[u \ \ v] \ \underset{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}{\underset{\downarrow}{M}} \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

This is the equation of an ellipse.



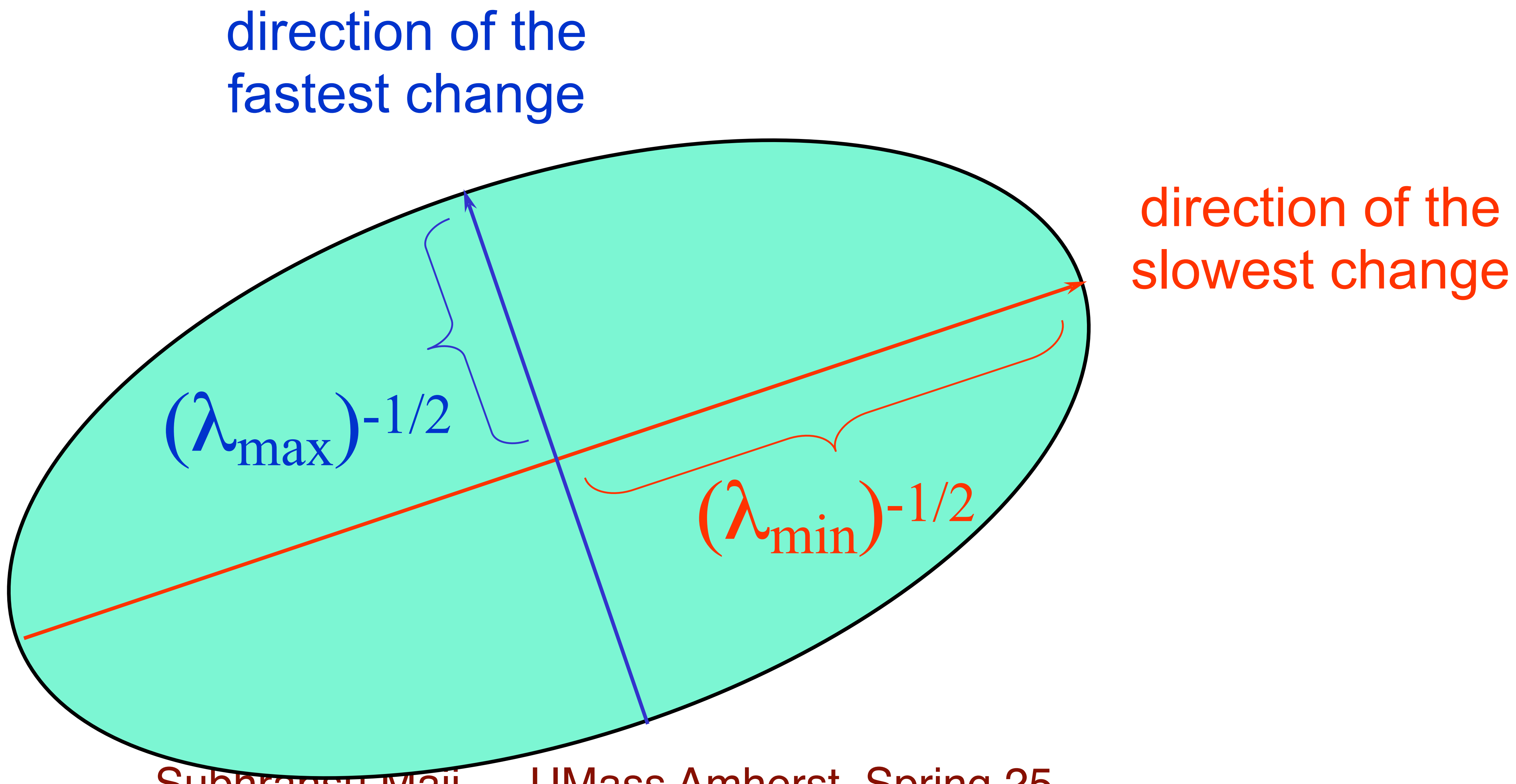
Interpreting the second moment matrix

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.

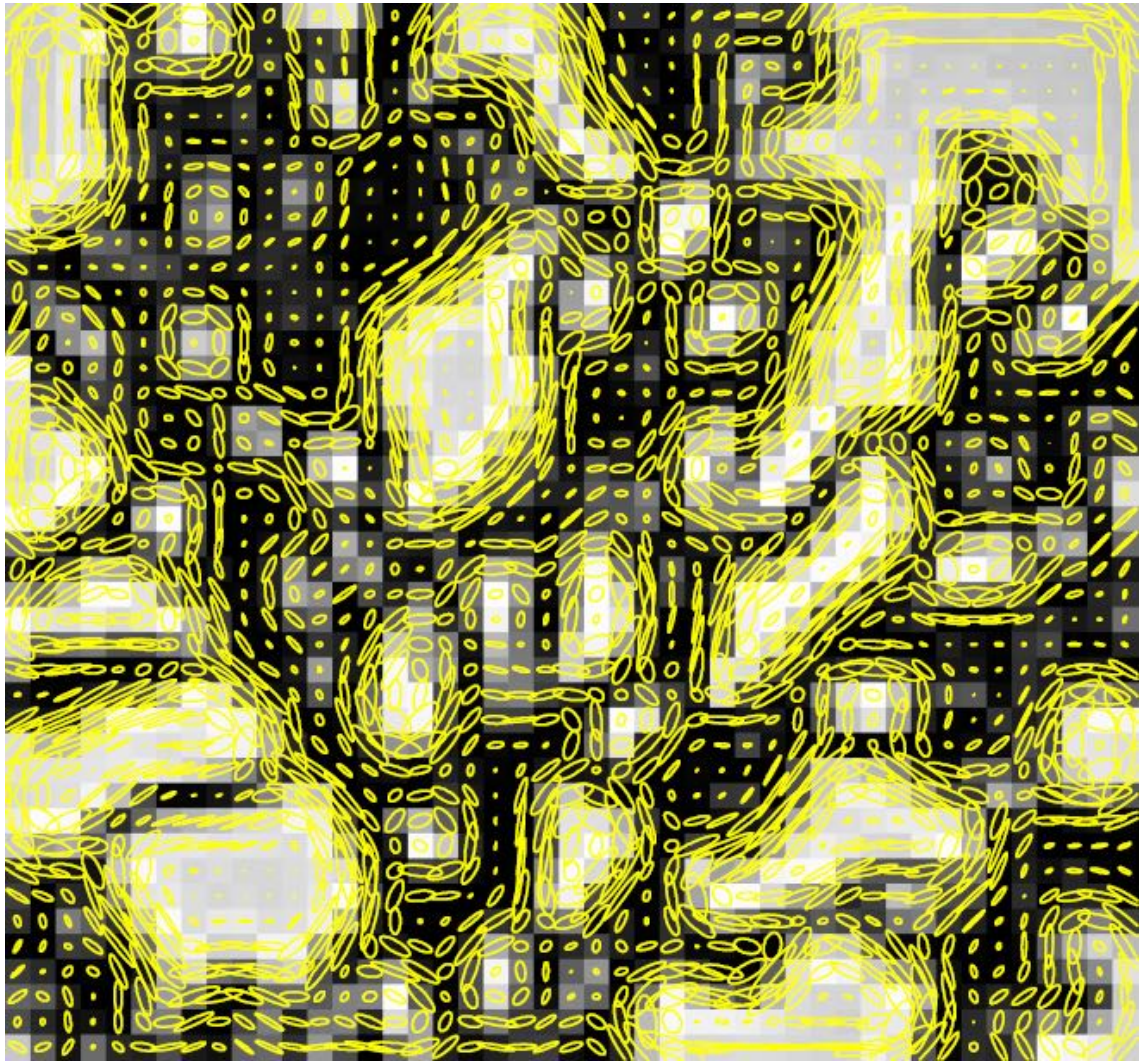
Diagonalization of M: $$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by *R*



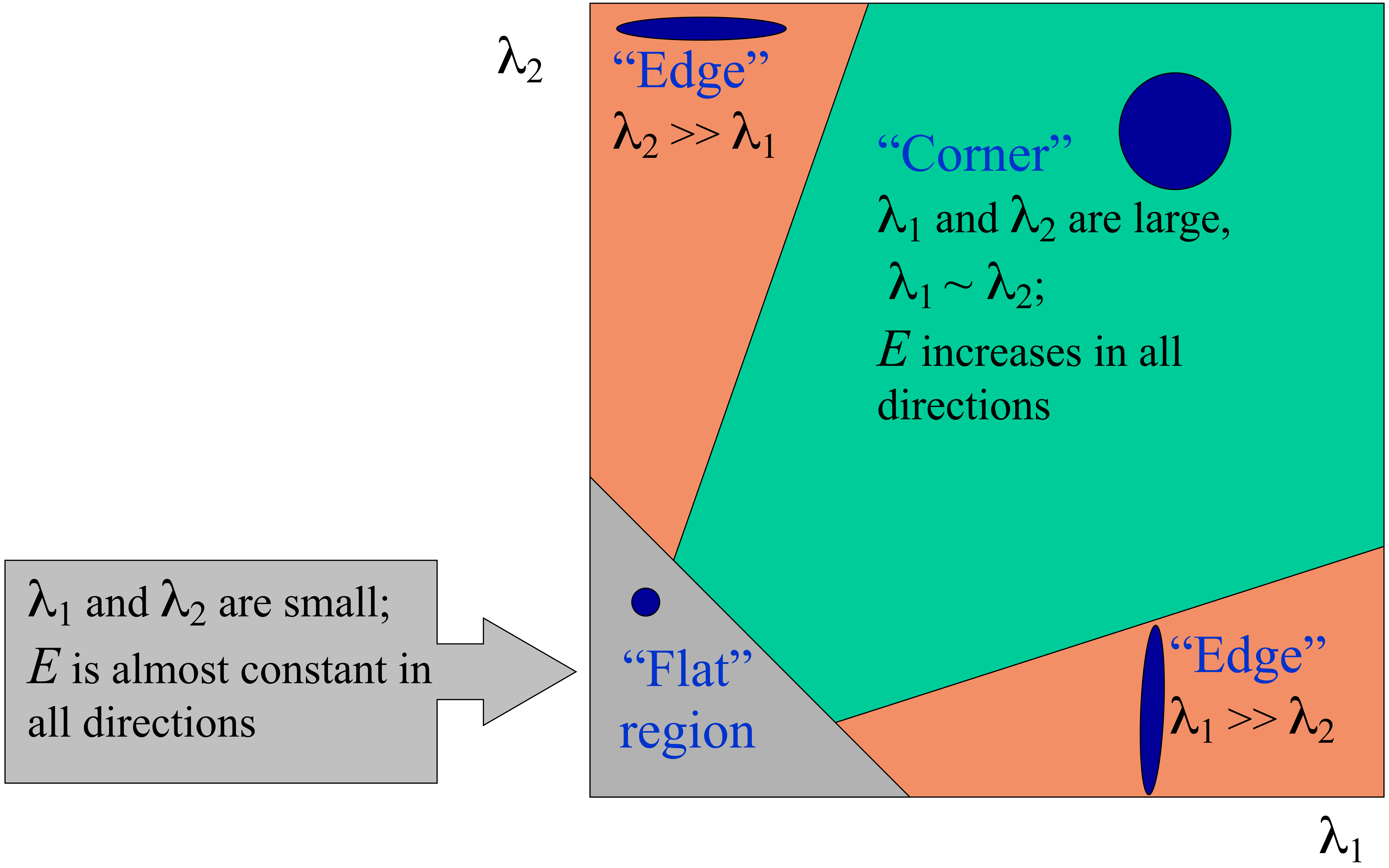
Visualization of second moment matrices

Visualization of second moment matrices



Interpreting the eigenvalues

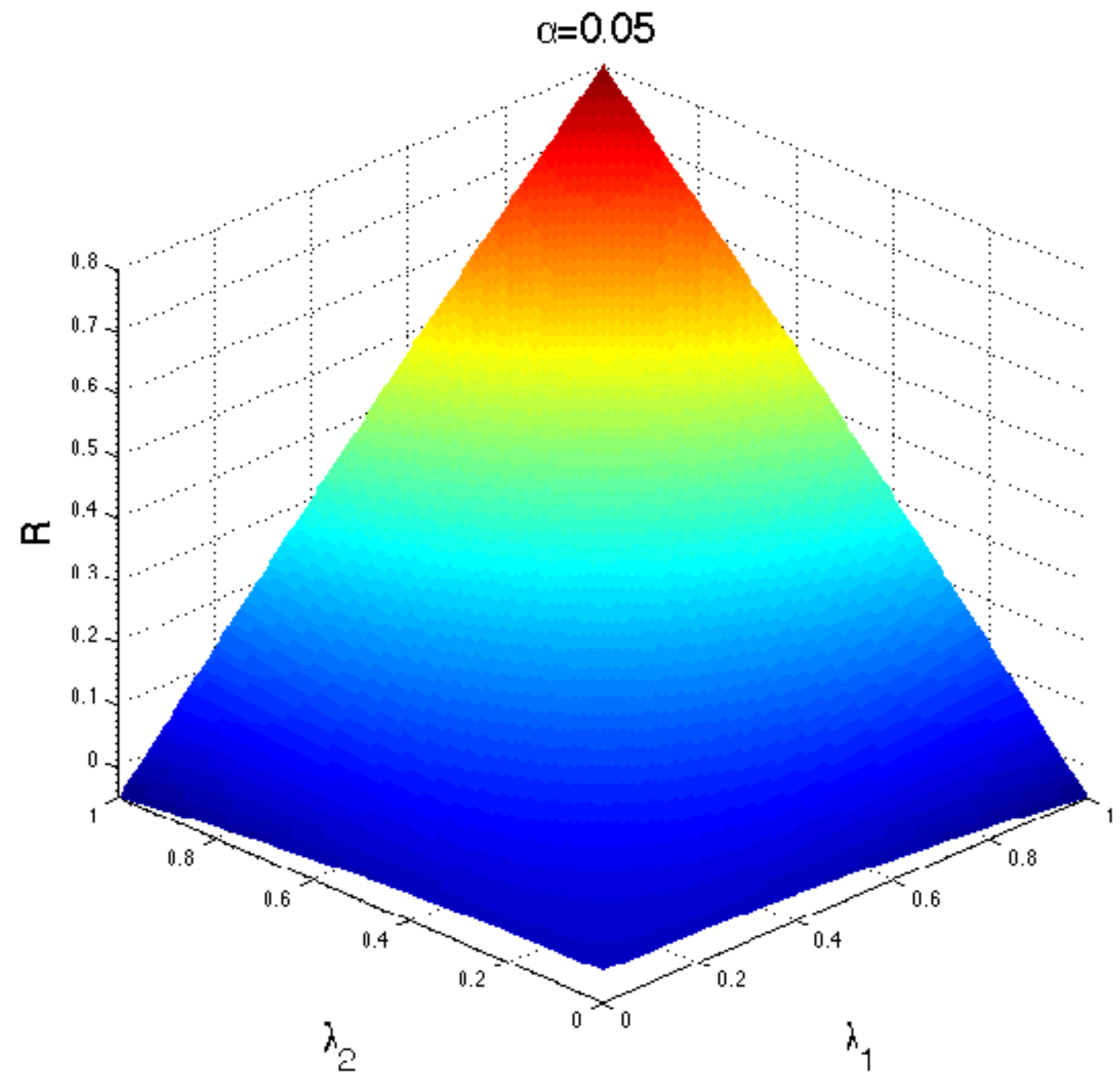
Classification of image points using eigenvalues of *M*:

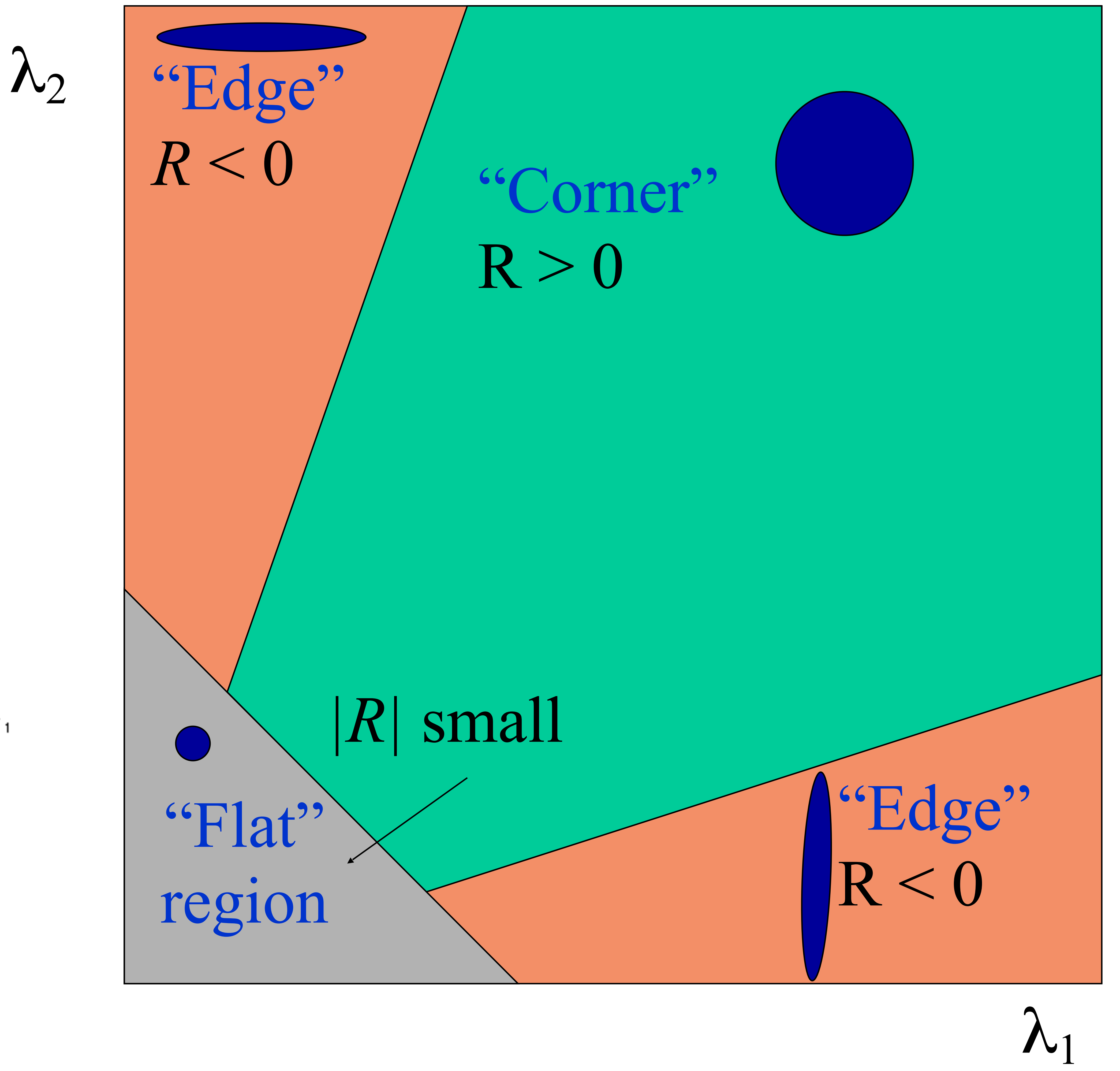


Corner response function

$$R = \det(M) - \alpha\ \mathrm{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)





The Harris corner detector

1. Compute partial derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel:

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

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

The Harris corner detector

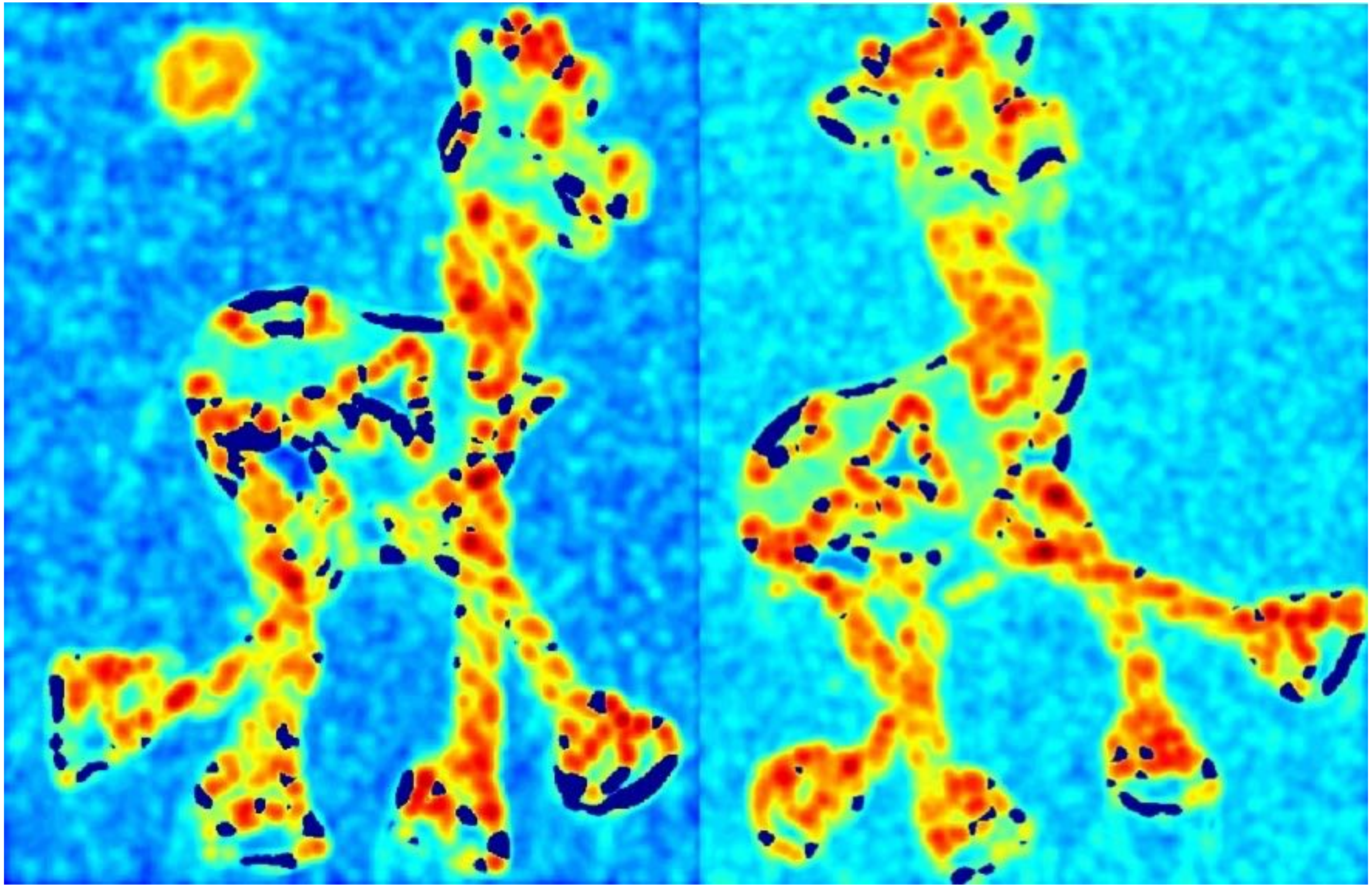
1. Compute partial derivatives at each pixel
2. Compute second moment matrix M in a Gaussian window around each pixel
3. Compute corner response function R

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 R



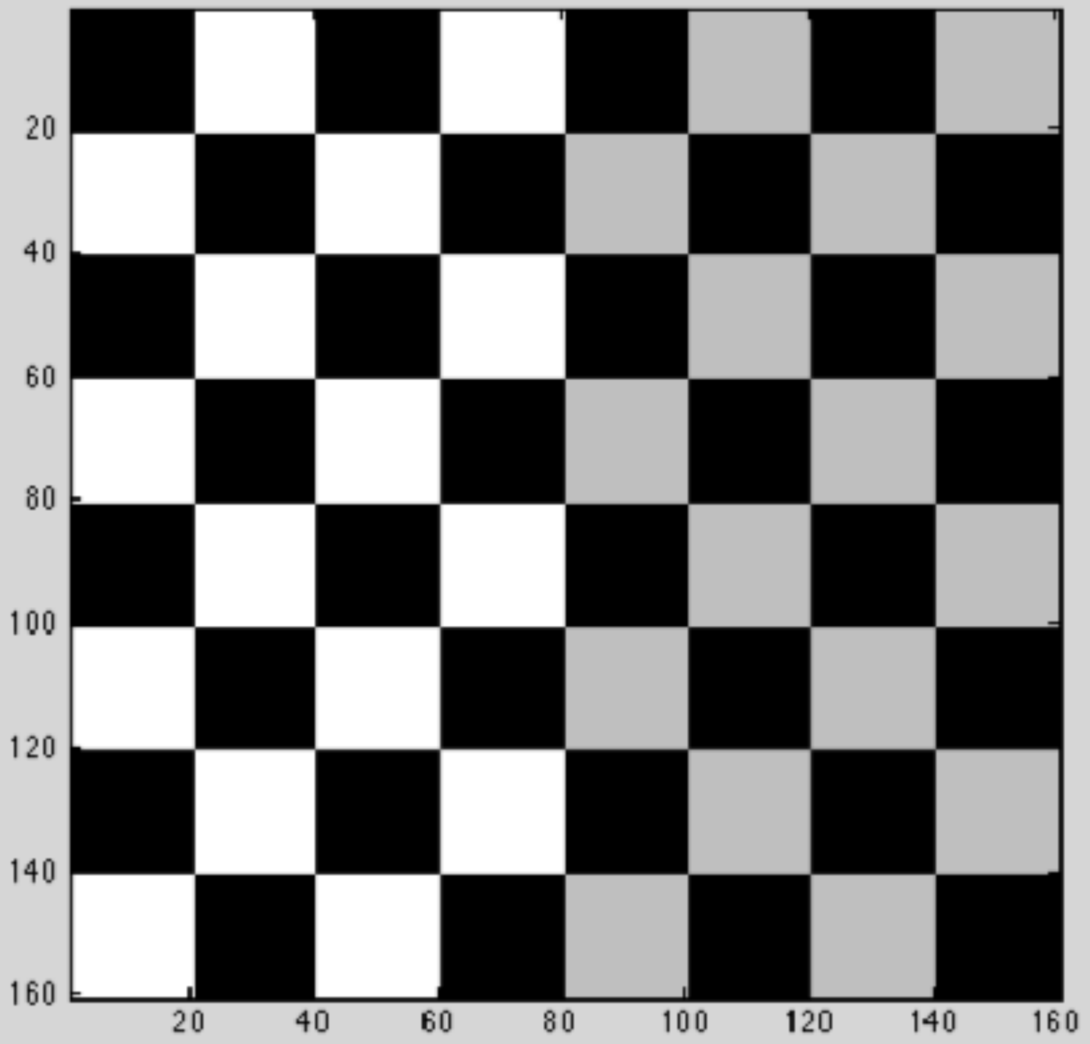
The Harris corner detector

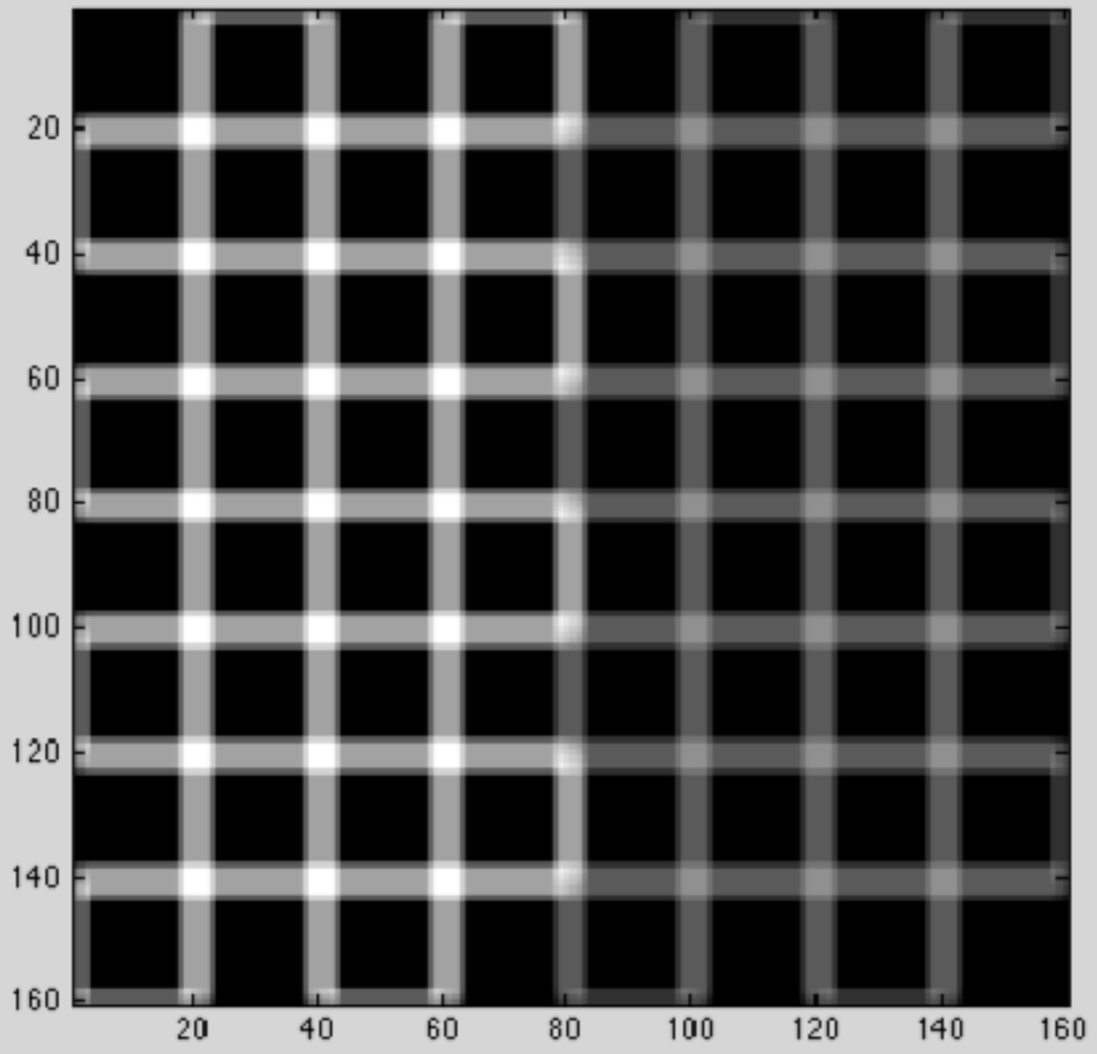
1. Compute partial derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (non-maximum suppression)

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

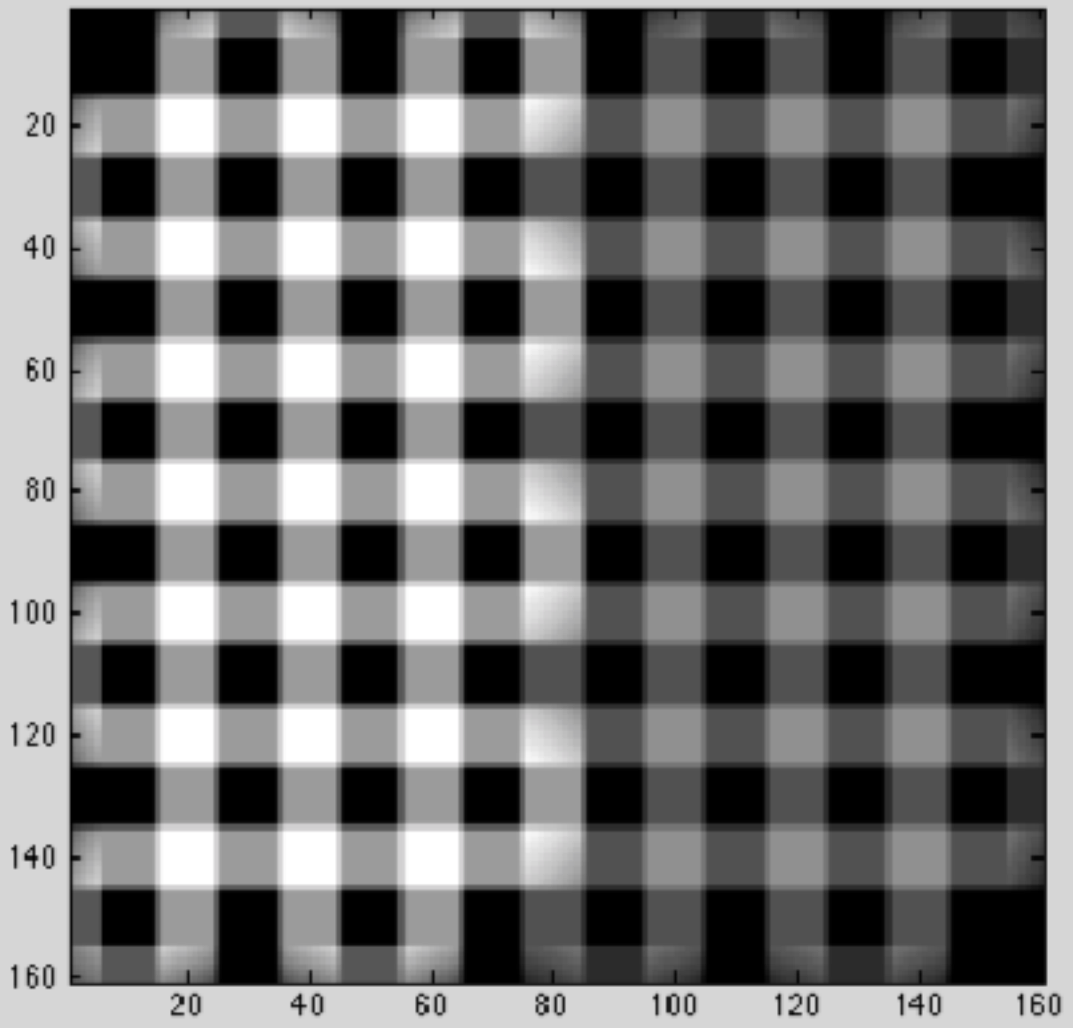
Corner score example

image



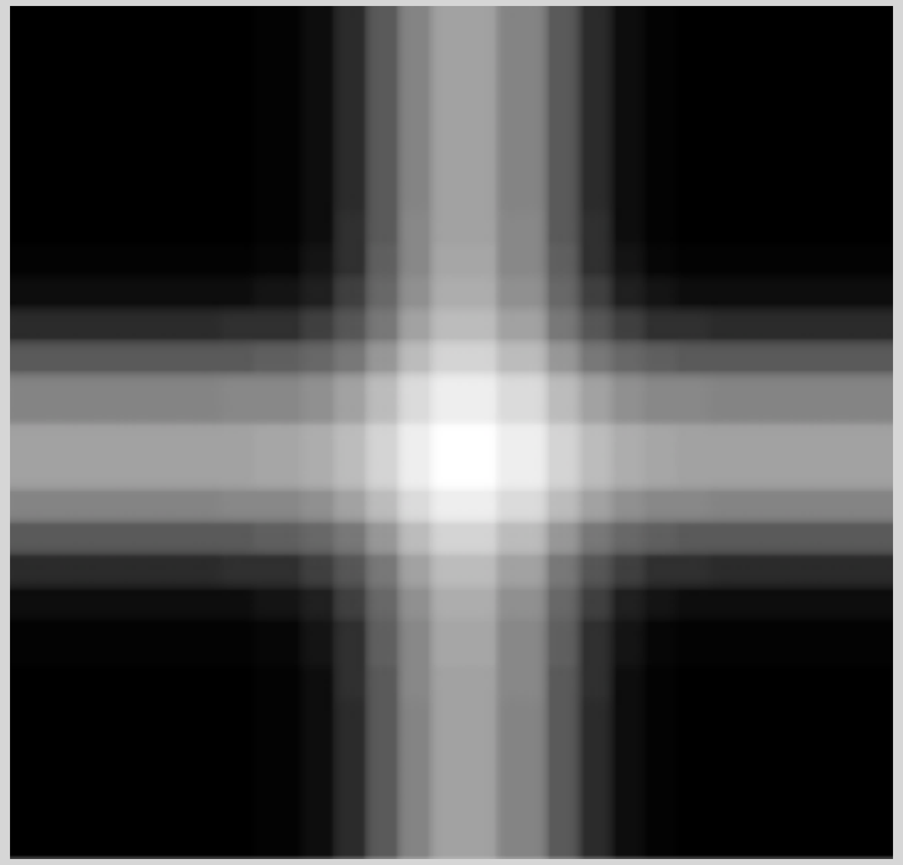


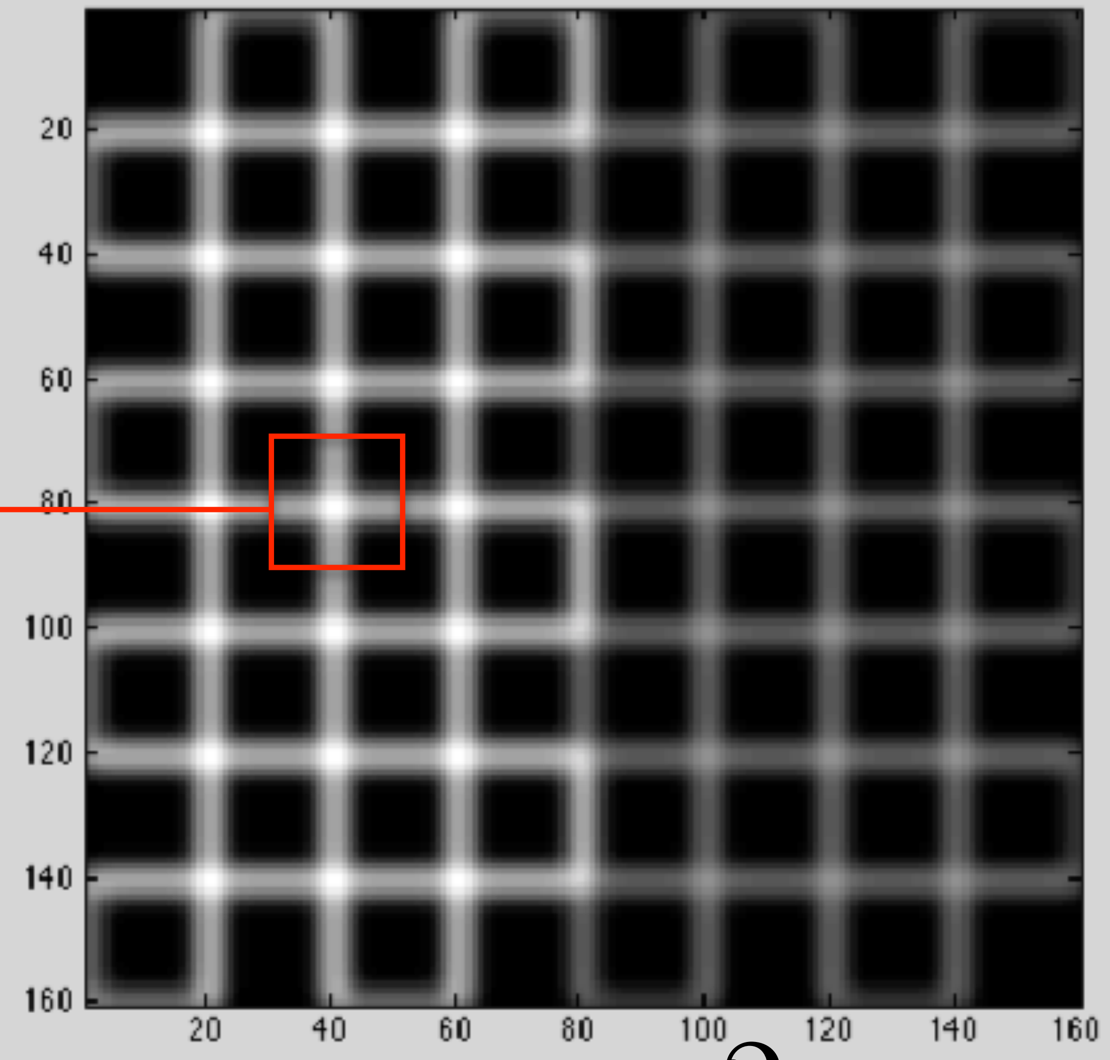
$W = 5$



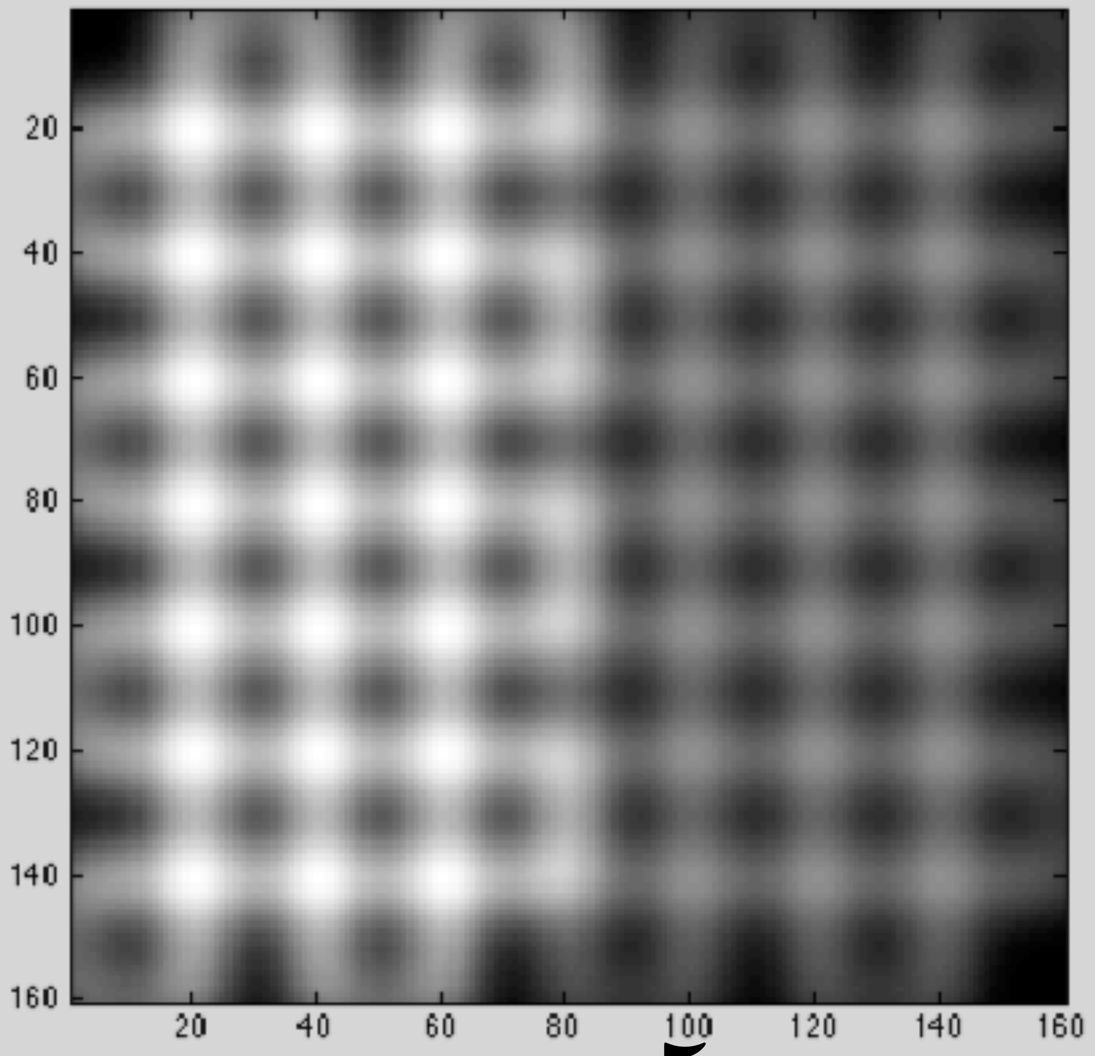
$W = 10$

Box filter





$\sigma = 2$

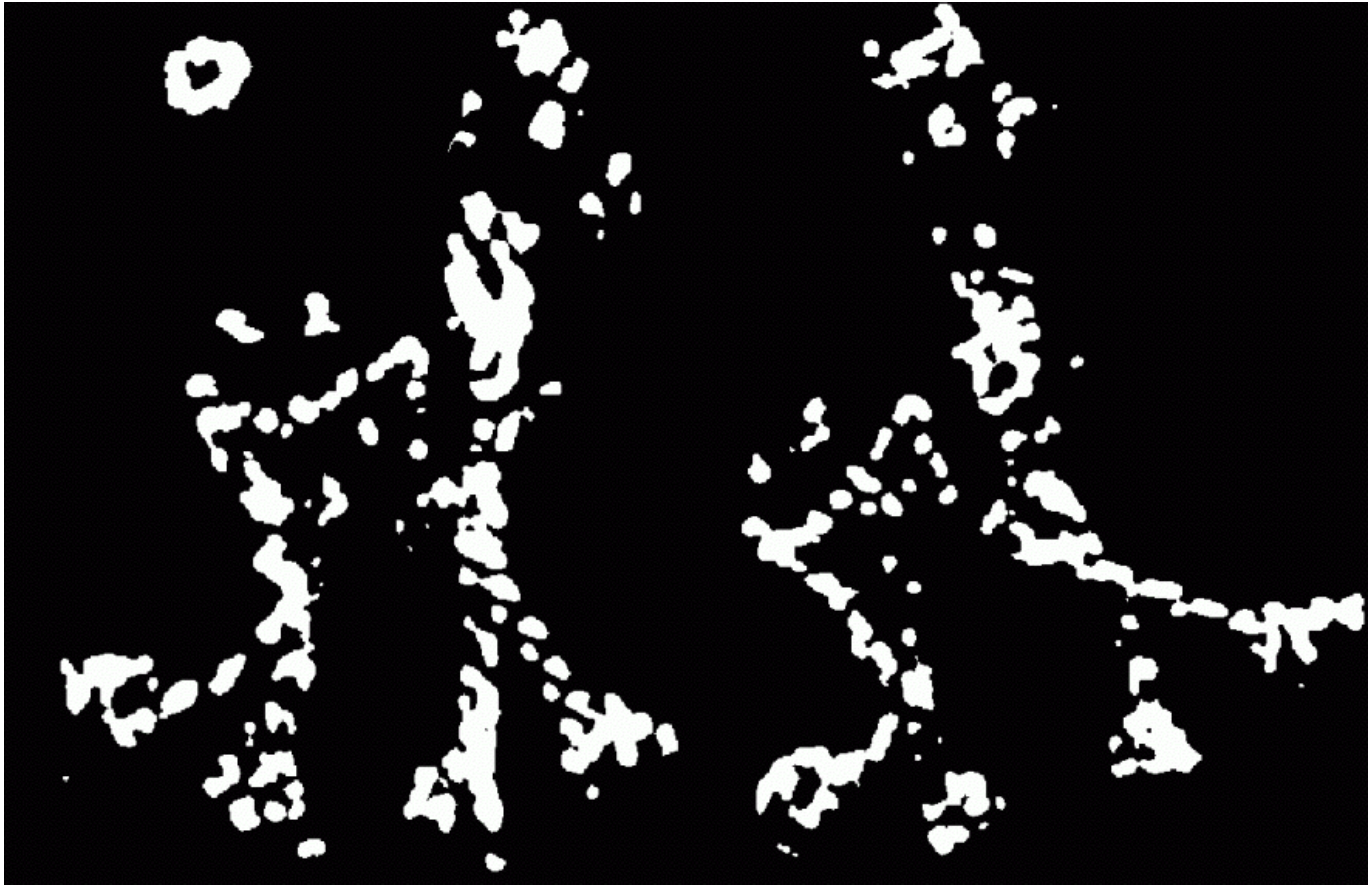


$\sigma = 5$

Gaussian filter

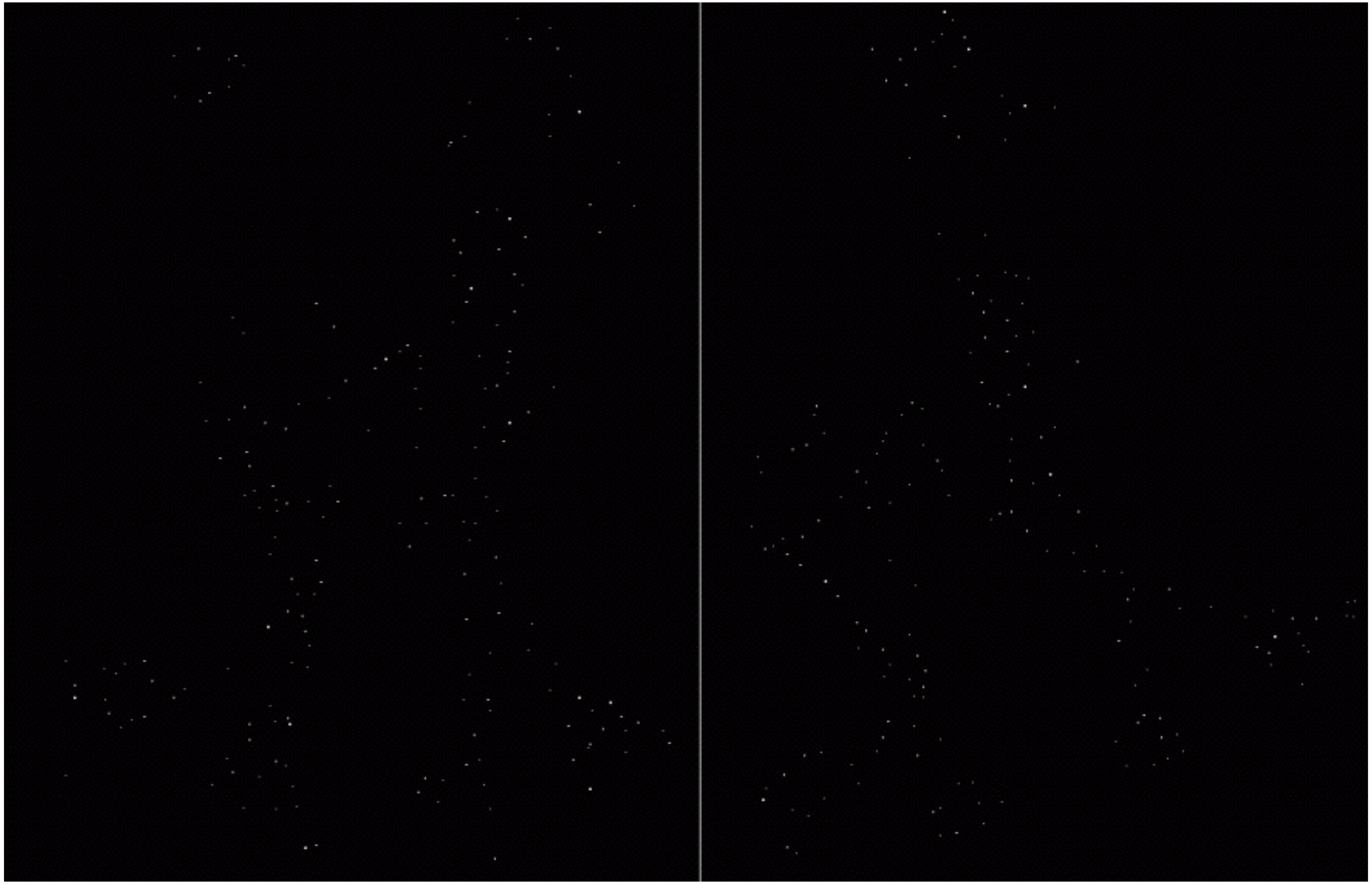
Harris Detector: Steps

Find points with large corner response: R > threshold



Harris Detector: Steps

Take only the points of local maxima of R



Harris Detector: Steps

Further thoughts and readings…

Original corner detector paper

- C.Harris and M.Stephens, **“A Combined Corner and Edge Detector.”** Proceedings of the 4th Alvey Vision Conference,1988

Other corner functions

- Can you think of other $f(\lambda_1, \lambda_2)$ that work for finding corners?

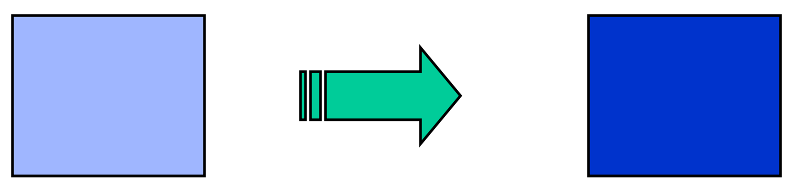
Invariance and covariance

Invariance: transformations *do not change* the corner locations

Covariance or Equivariance: transformations change corner locations *in a predictable way*

We want corners to be *invariant* to photometric transformations and *covariant* to geometric transformations

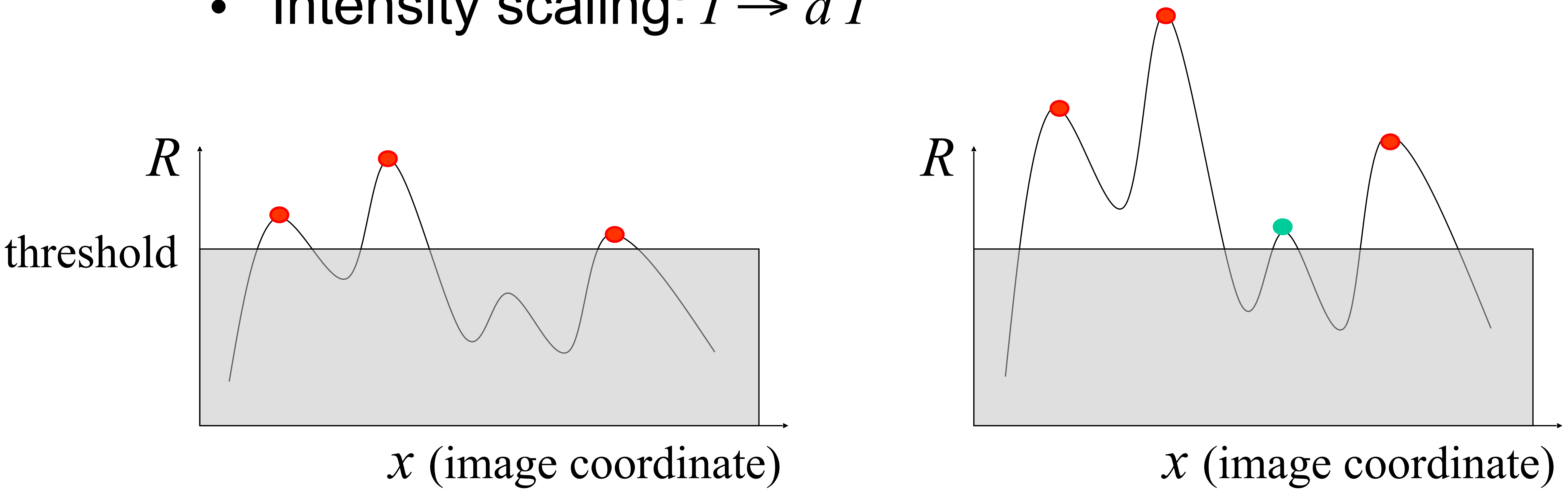
Affine intensity change



$$I \rightarrow a\,I + b$$

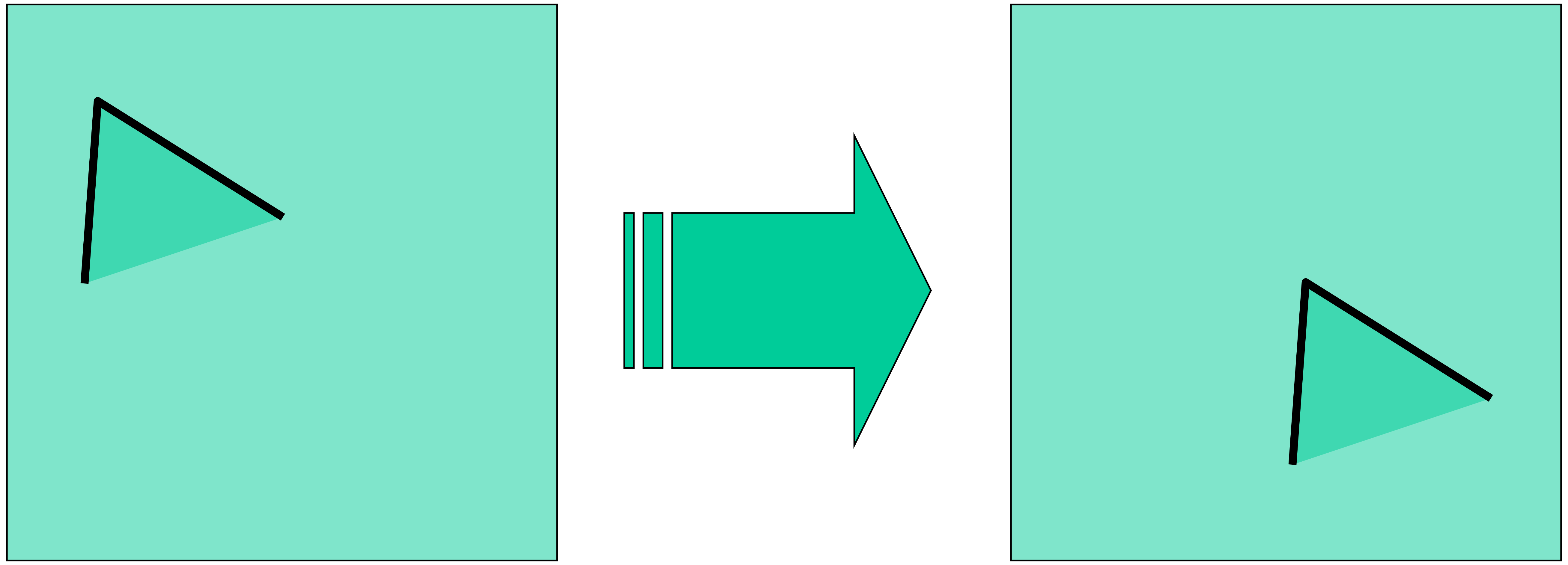
- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

- Intensity scaling: $I \rightarrow a\,I$



Corner location is partially invariant to affine intensity change

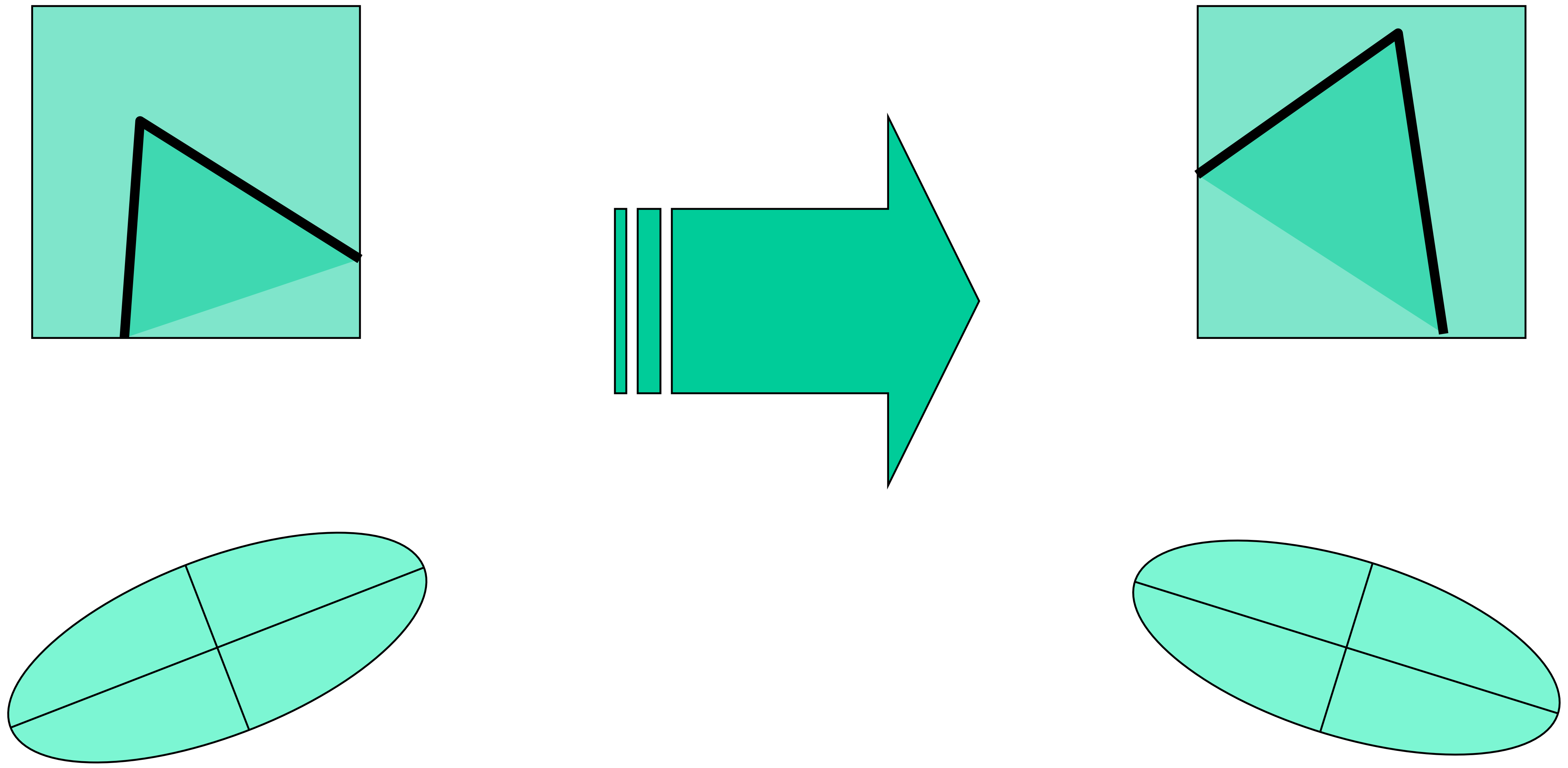
Translation



- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation

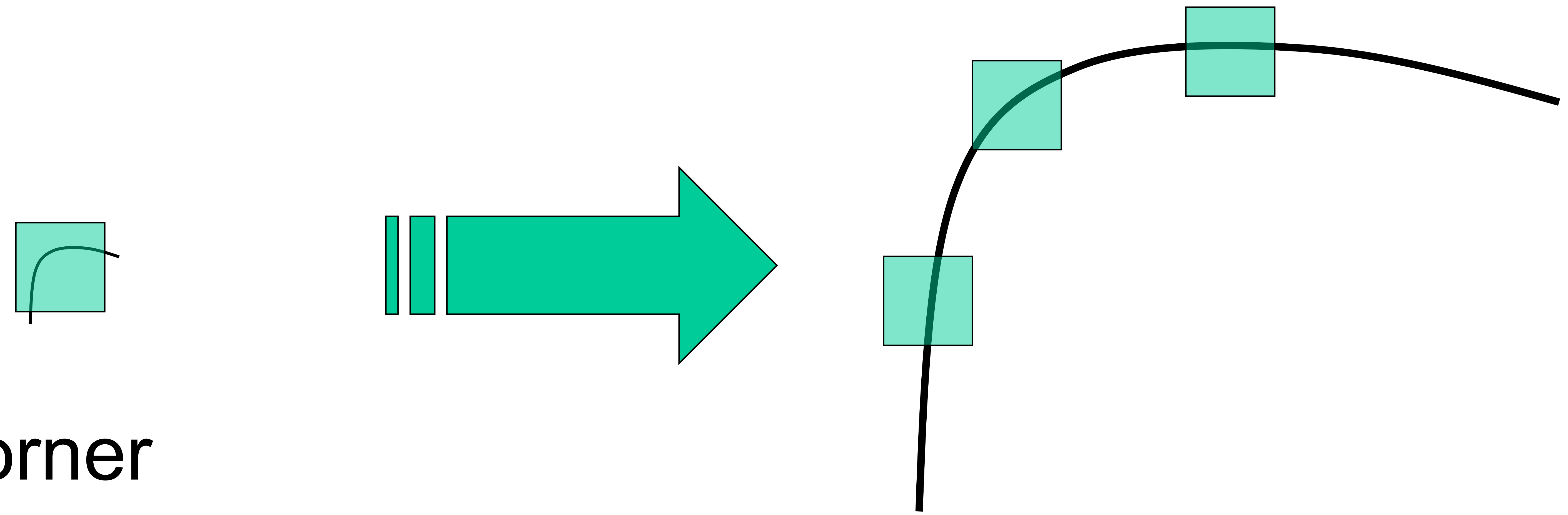
Rotation



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

Corner location is covariant w.r.t. rotation

Scaling



Corner

All points will be classified as edges

Corner detection is sensitive to the image scale!

Source: L. Lazebnik