Computer vision: edge and corner detectors

Building an (Canny) edge detector

original image

final output

Start from greyscale

1. Reduce noise

Use, e.g., a 5x5 Gaussian Filter

2. Calculate the gradient (e.g., Sobel operator)

norm of the gradient

3. Threshold the gradient intensity

Thresholded norm of the gradient

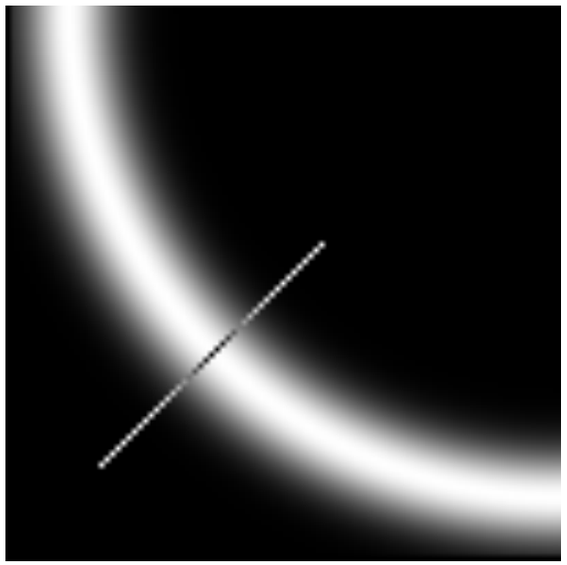
Problem 1: Thick trails

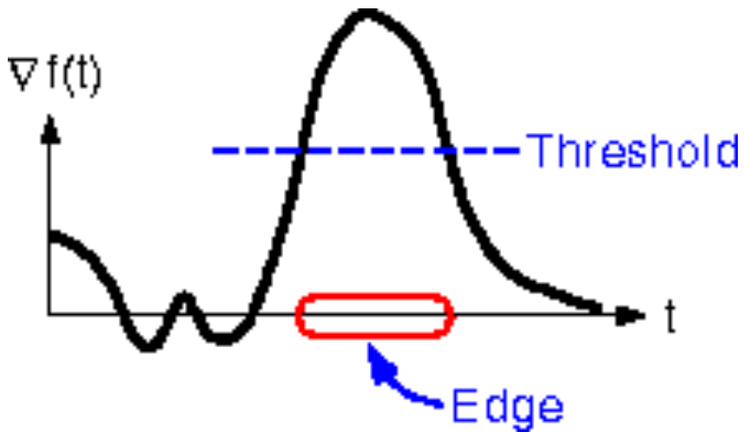
How do we turn these thick regions into edges?

Thresholded norm of the gradient

4. Non-maximum suppression

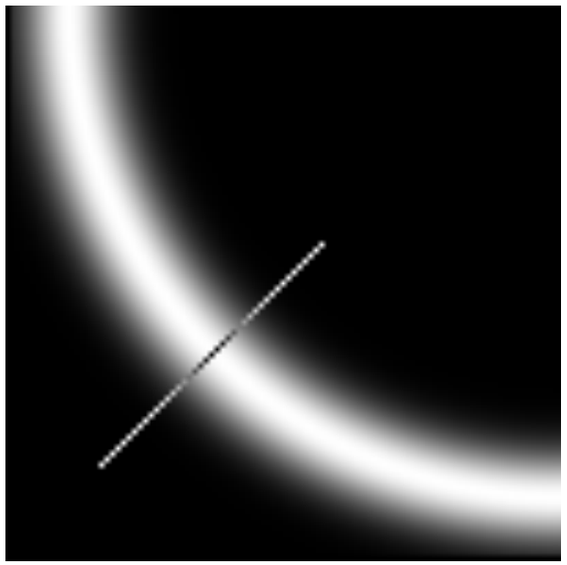
- Look for local maximum along gradient direction

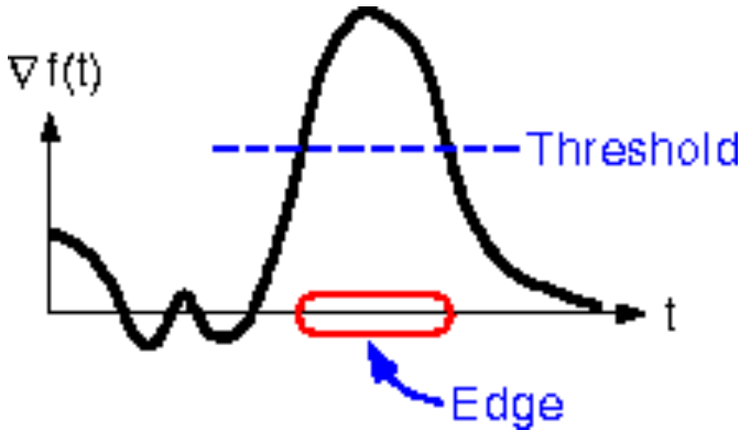


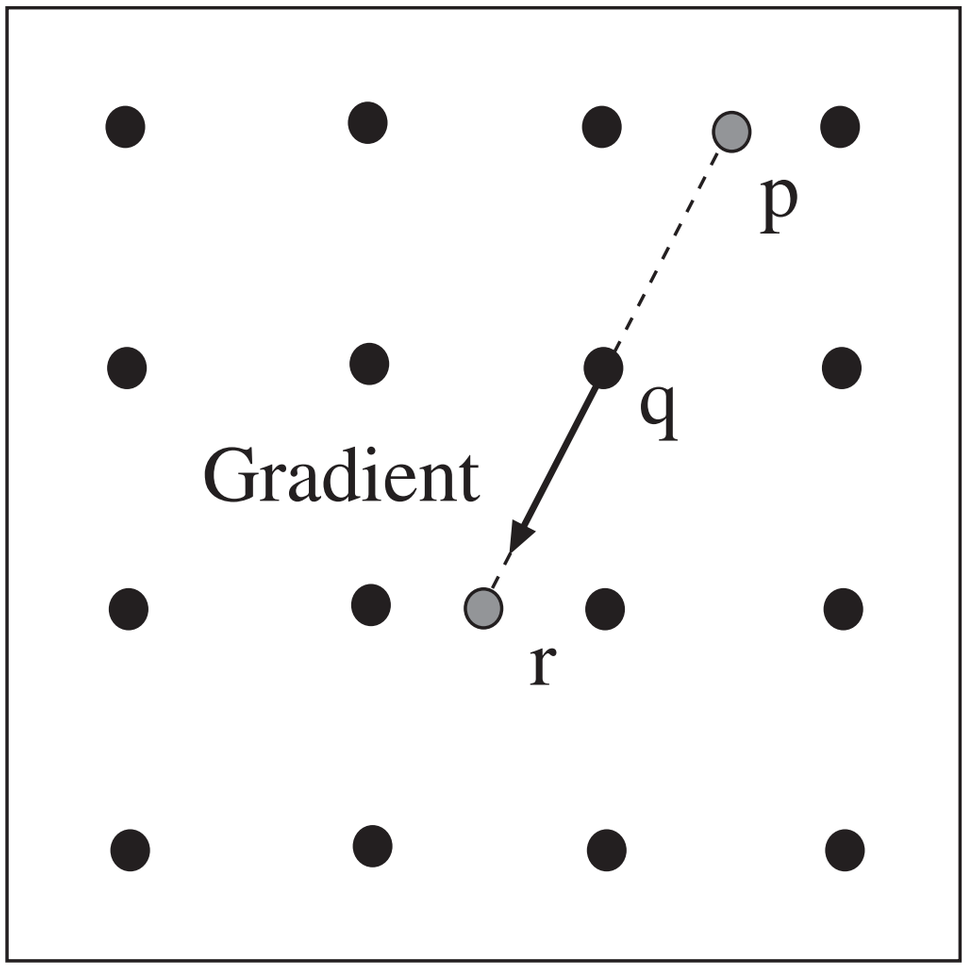


4. Non-maximum suppression

- Look for local maximum along gradient direction
  - Requires checking interpolated pixels p and r

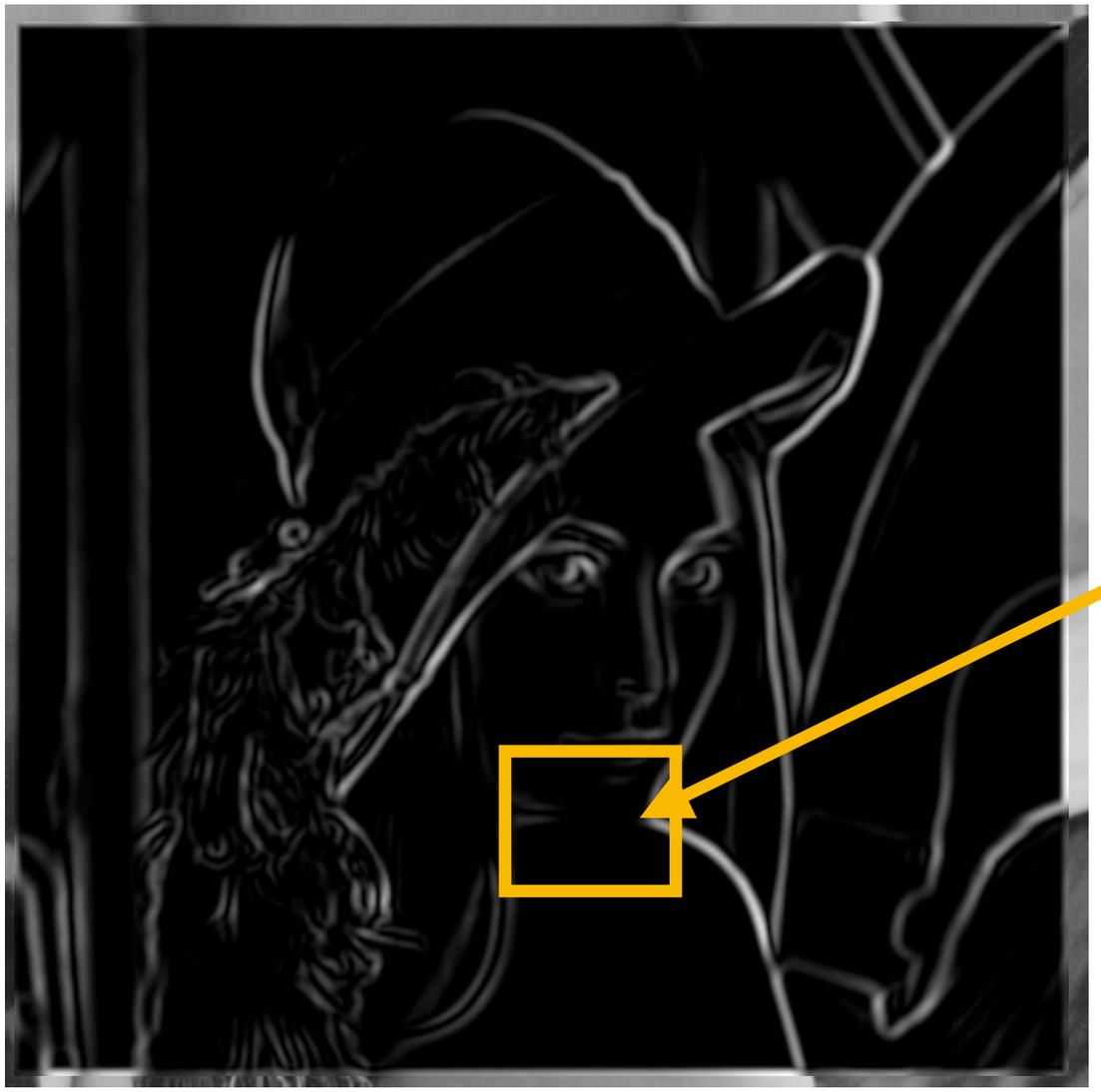






If q is a local maximum in the direction of the gradient, keep. If not, set to 0.

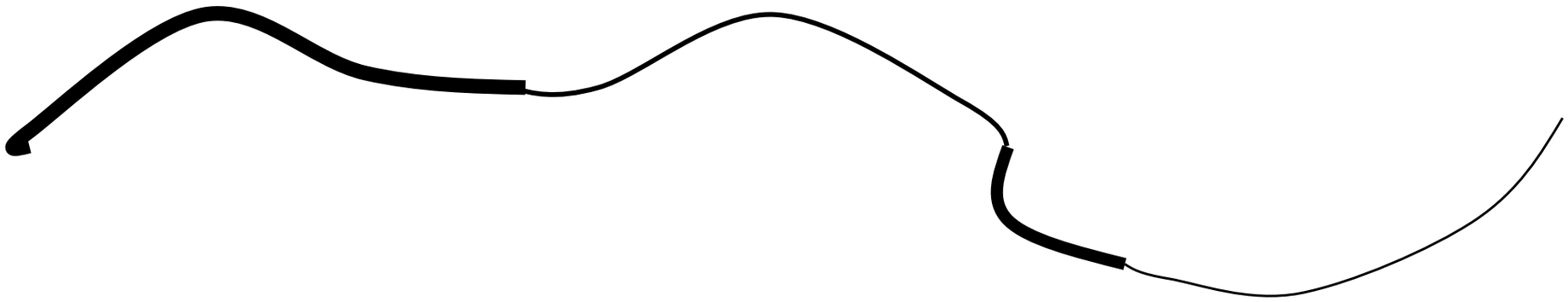
Problem 2: continuity of edges



Another problem: pixels along this edge didn't survive thresholding

Hysteresis (or double) thresholding

Use a high threshold to start edge curves, and a low threshold to continue them.



Hysteresis (or double) thresholding

original image

high threshold
(strong edges)

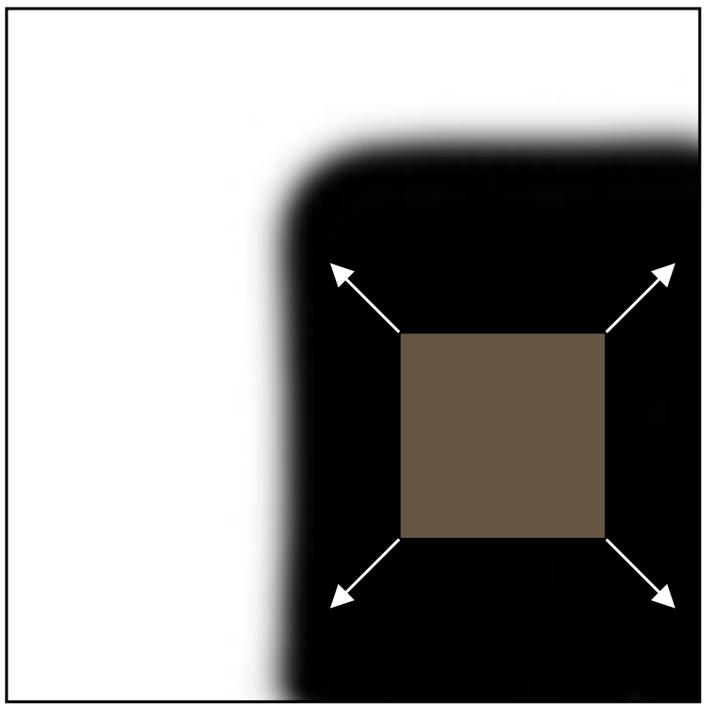
low threshold
(weak edges)

hysteresis threshold

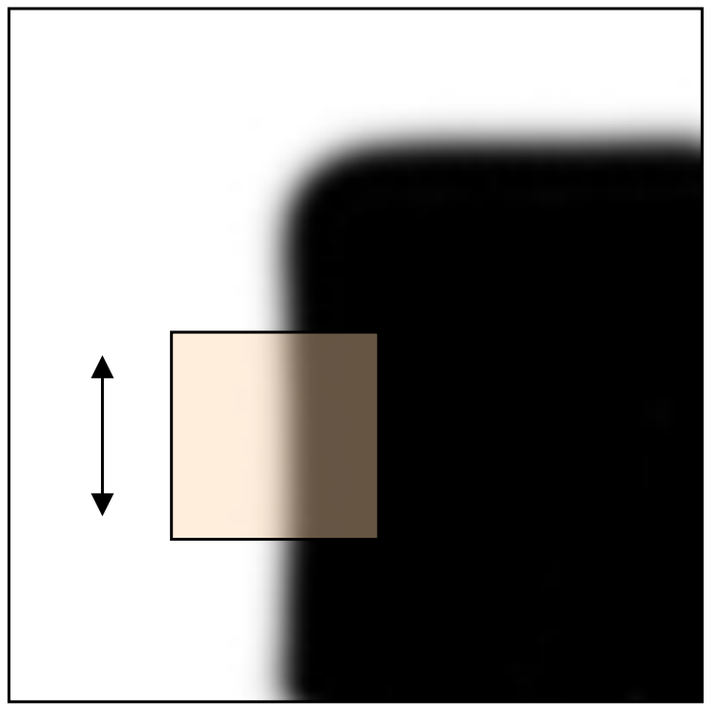
Corner detection

Corner detection: Basic idea

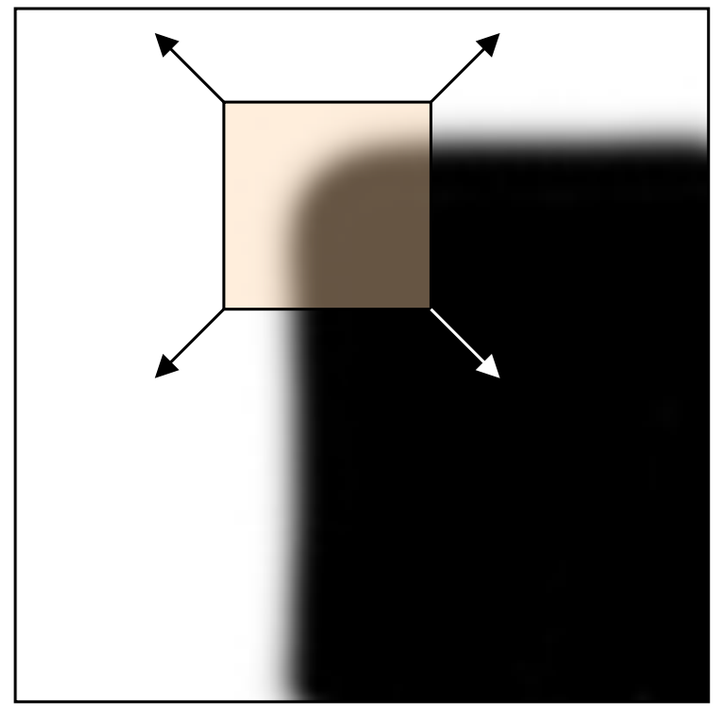
- The interest point should be recognizable using a local window
- Shifting the window in *any direction* should result in a *large change* in intensity



"flat" region: no change in all directions



"edge": no change along edge direction



"corner": significant changes in all directions

Corner detection: Basic idea

- The interest point should be recognizable using a local window
- Shifting the window in *any direction* should result in a *large change* in intensity

isotropic aperture:
motion downwards to the right

elongated aperture:
motion appears to be vertical

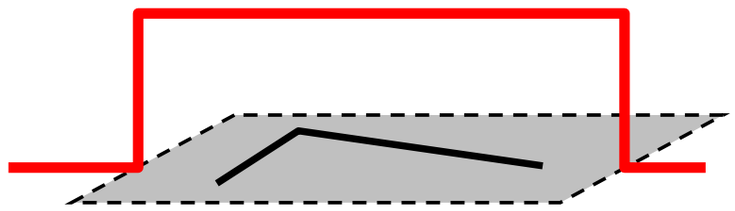
Corner detection: Mathematics

- Window-averaged squared change in intensity due to shift by $[u, v]$

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

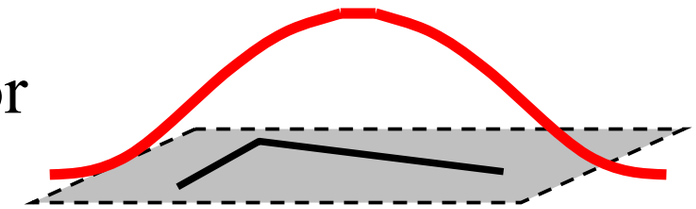
window function (can be modeled as convolution) ***shifted intensity*** ***intensity***

Window function $w(x,y)$ =



1 in window, 0 outside

or



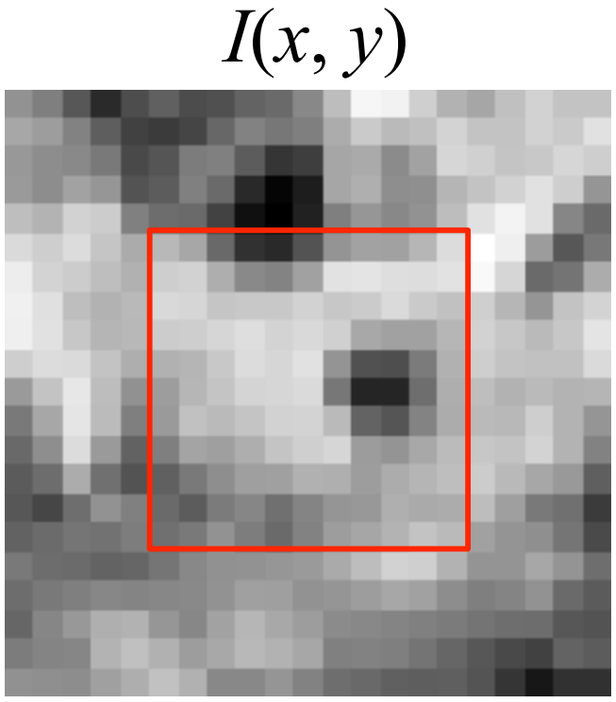
Gaussian

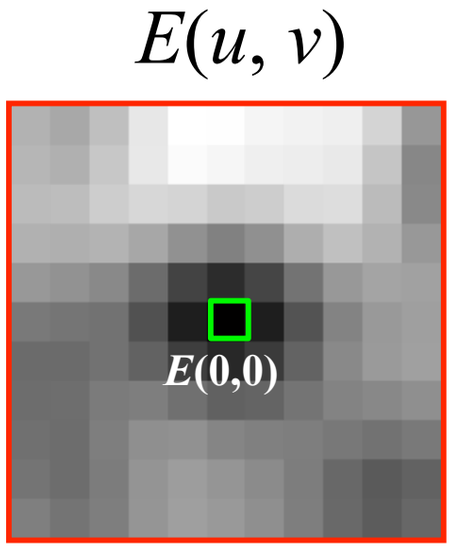
Corner detection: Mathematics

- Window-averaged squared change in intensity due to shift by $[u, v]$

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

window function (can be modeled as convolution) — ***shifted intensity*** — ***intensity***





Corner detection: Mathematics

- How does this function behave with small image shifts?
- Consider a first-order Taylor series approximation

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

Corner detection: Mathematics

- How does this function behave with small image shifts?
- Consider a first-order Taylor series approximation

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

- This gives a *quadratic* approximation to the error function

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

Corner detection: Mathematics

- How does this function behave with small image shifts?
- Consider a first-order Taylor series approximation

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

- This gives a *quadratic* approximation to the error function

$$\begin{aligned} E(u,v) &= \sum_{x,y} w(x,y)\,[I(x+u,y+v) - I(x,y)]^2 \\ &\approx \sum_{x,y} w(x,y)\,[(I(x,y) + I_x u + I_y v) - I(x,y)]^2 \end{aligned}$$

Corner detection: Mathematics

- How does this function behave with small image shifts?
- Consider a first-order Taylor series approximation

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

- This gives a *quadratic* approximation to the error function

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

Corner detection: Mathematics

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

- We can express this in matrix form

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \left(\sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}$$

$$= \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

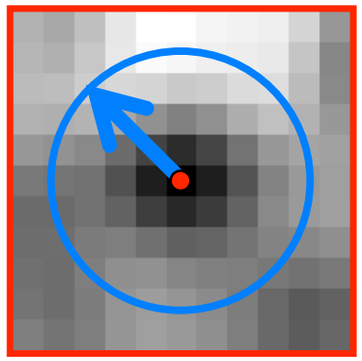
where M is a *second moment matrix (structure tensor)* computed from image derivatives

Corner detection: Second moment matrix

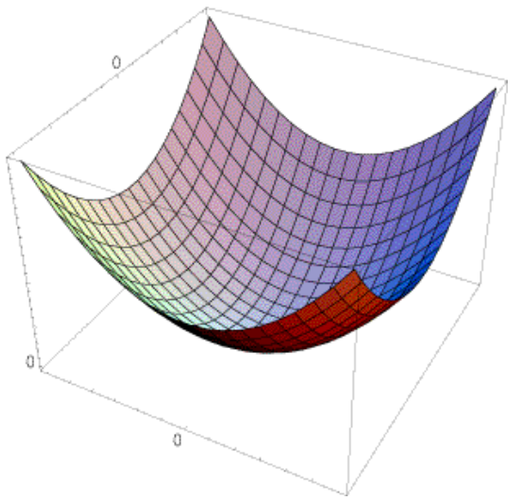
- The surface $E(u, v)$ is locally approximated by a quadratic form
- In which directions does it experience the greatest/least change?

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

$E(u, v)$



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



Corner detection: Second moment matrix

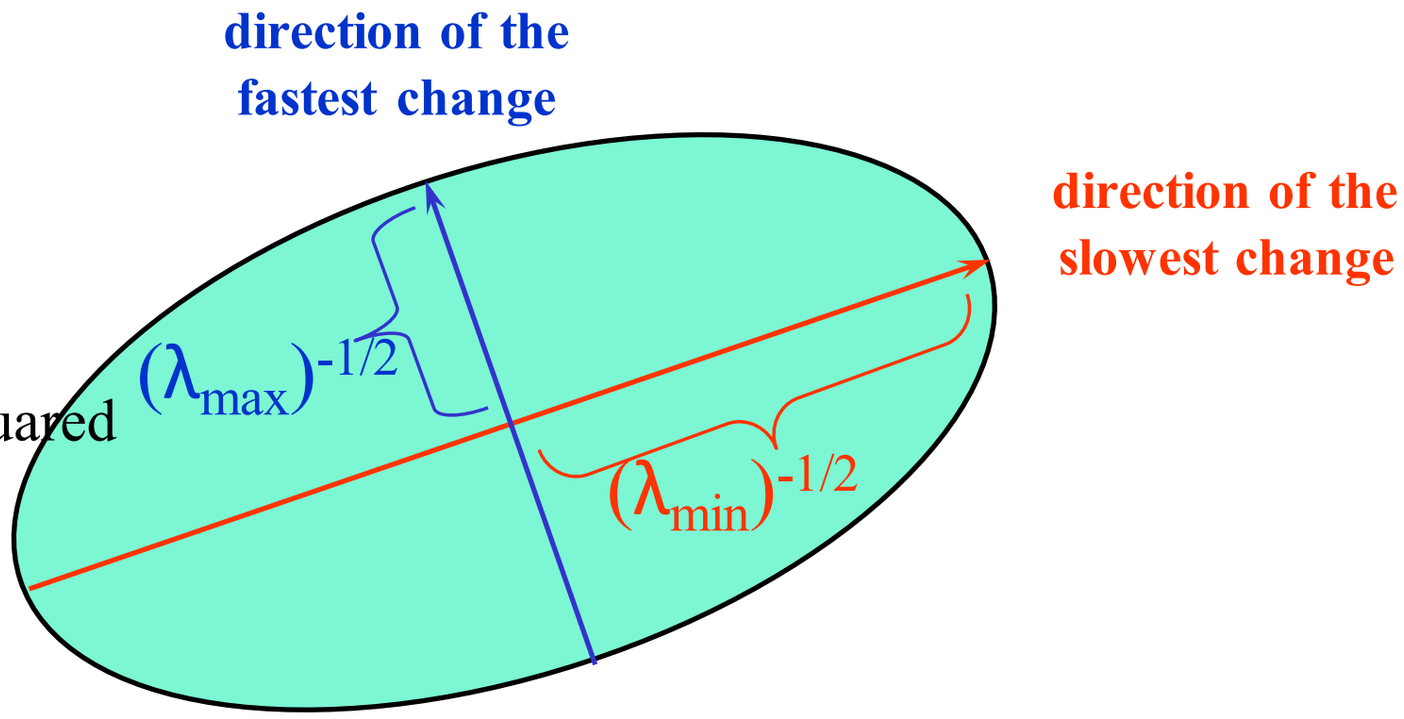
- The surface $E(u, v)$ is locally approximated by a quadratic form
- In which directions does it experience the greatest/least change?

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \qquad M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

λ_1, λ_2 **- eigenvalues of** M

- $E(u, v) = constant$ is an ellipse

Iso-contour of the squared error, *E(u,v)*



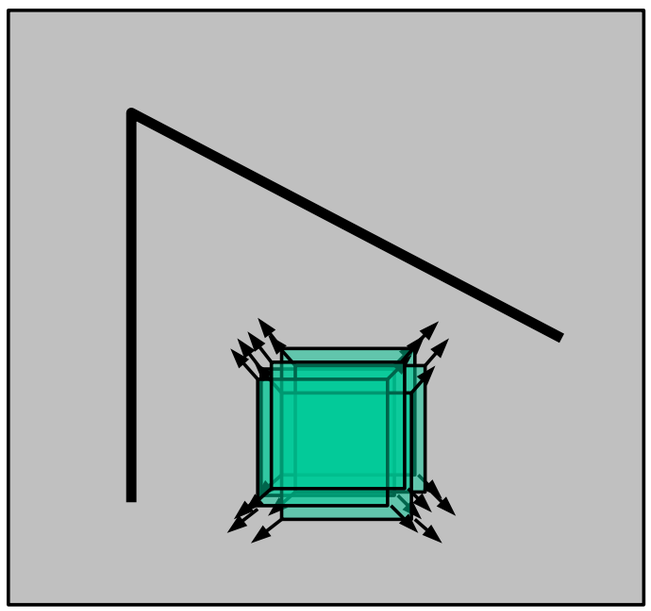
Corner detection: Second moment matrix

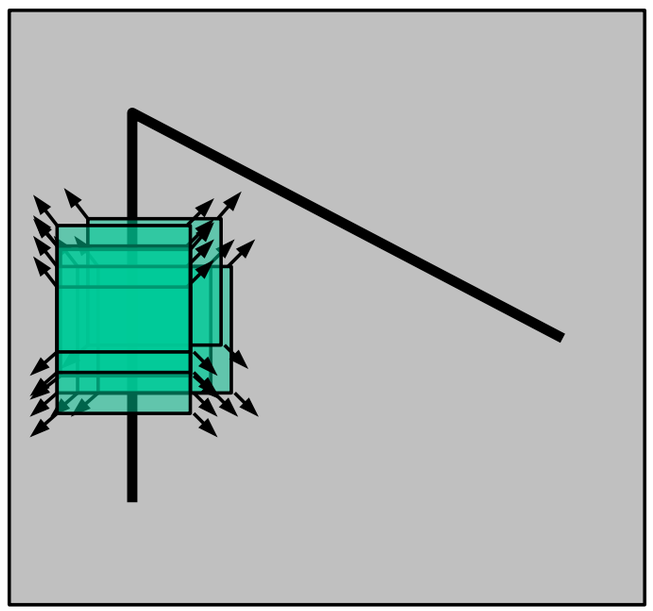
- Consider axis-aligned case (horizontal or vertical gradients) (that's what R does)

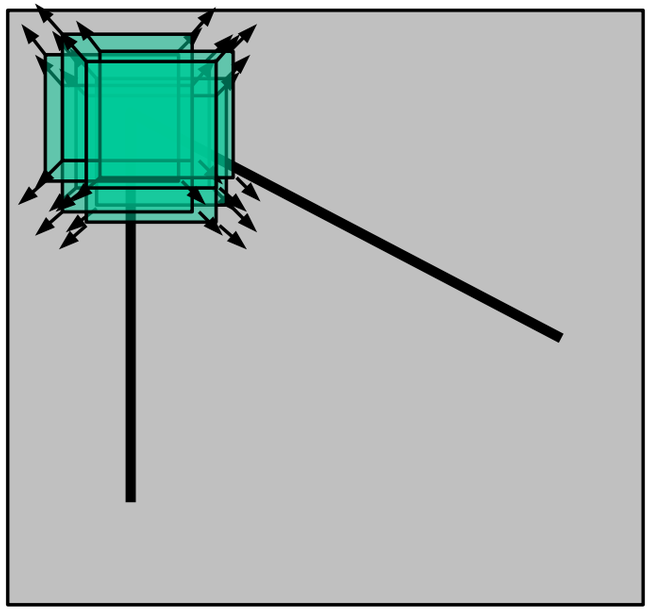
$$M = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

- If either a or b are close to 0, then it is not a corner

- Look for locations where both components are large

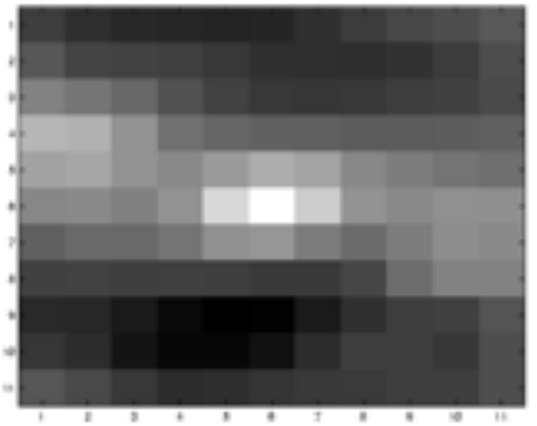






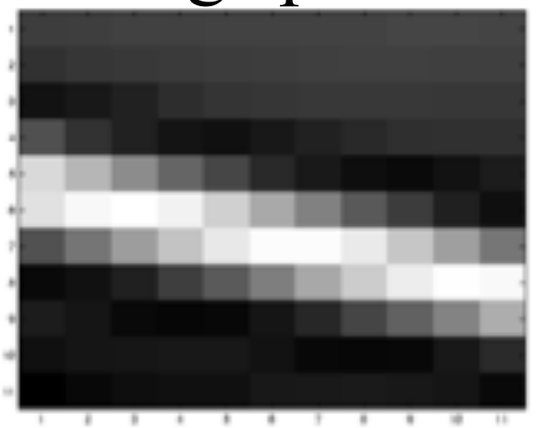
Selecting good features

Image patch



Selecting good features

Image patch



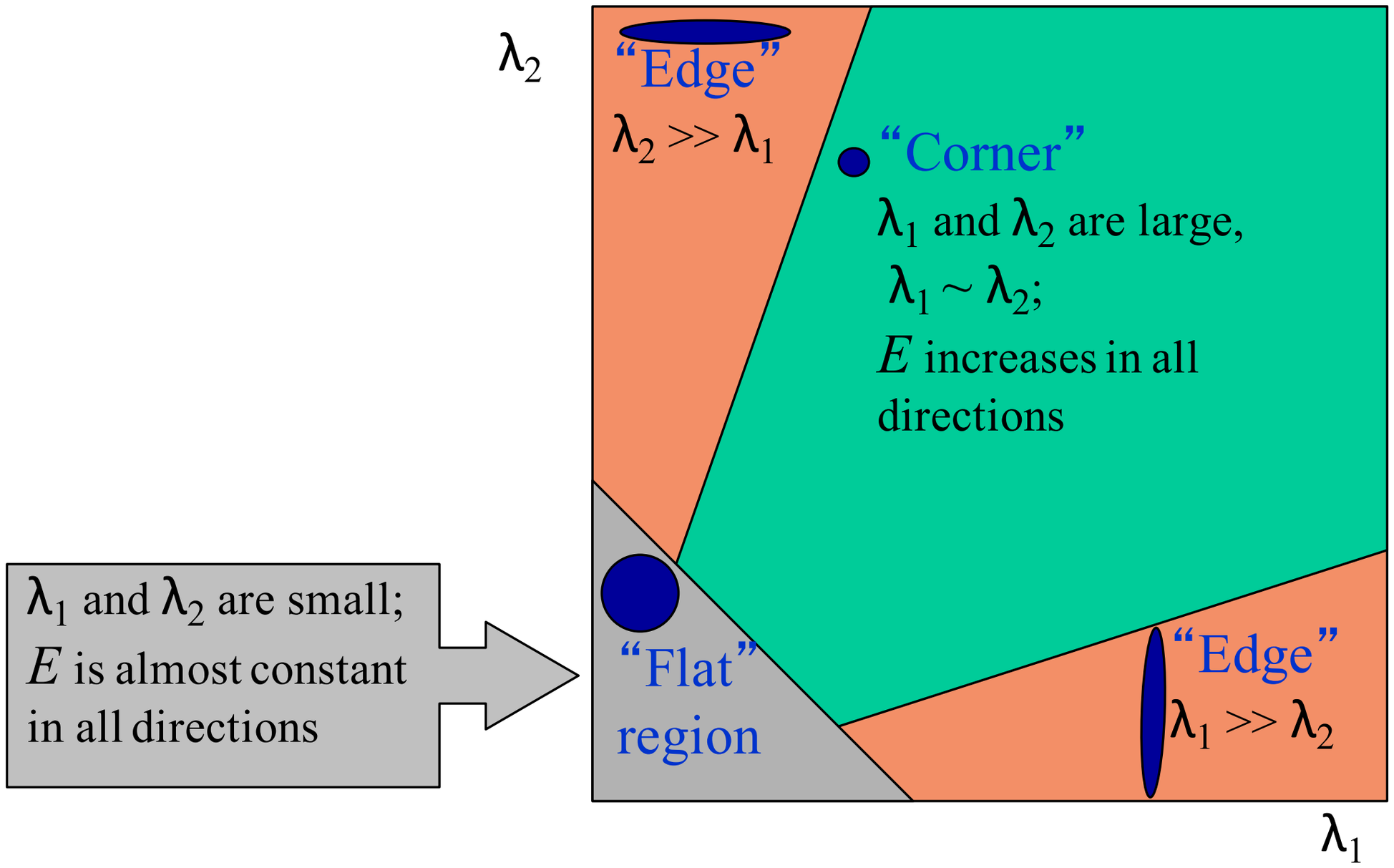
Selecting good features

Image patch

(contrast auto-scaled)

Interpreting eigenvalues

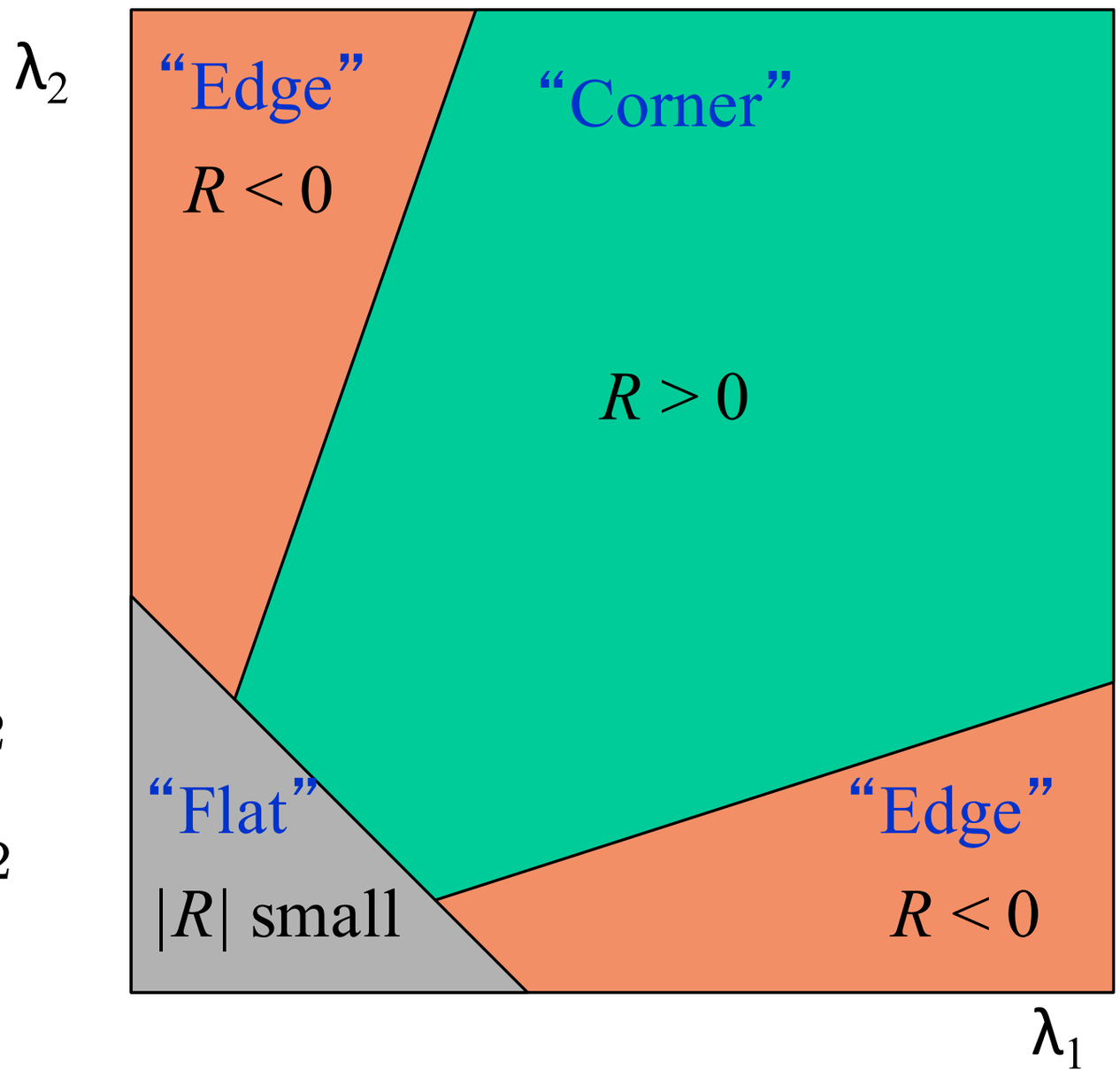
- Classification of image points using eigenvalues of M



Harris corner detector

- Proposed by Harris and Stephens (1988)
- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region

$$R = \det(M) - \alpha \operatorname{tr}(M)^2$$
$$= \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$
$$\alpha \in (0.04, 0.06)$$



Harris corner detector: Algorithm

1. Compute partial derivatives at each pixel

2. Compute second moment matrix M in a Gaussian window around each pixel

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

3. Compute corner response function $R = \det(M) - \alpha \mathrm{tr}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$

4. Find points with large corner response function $R >$ threshold

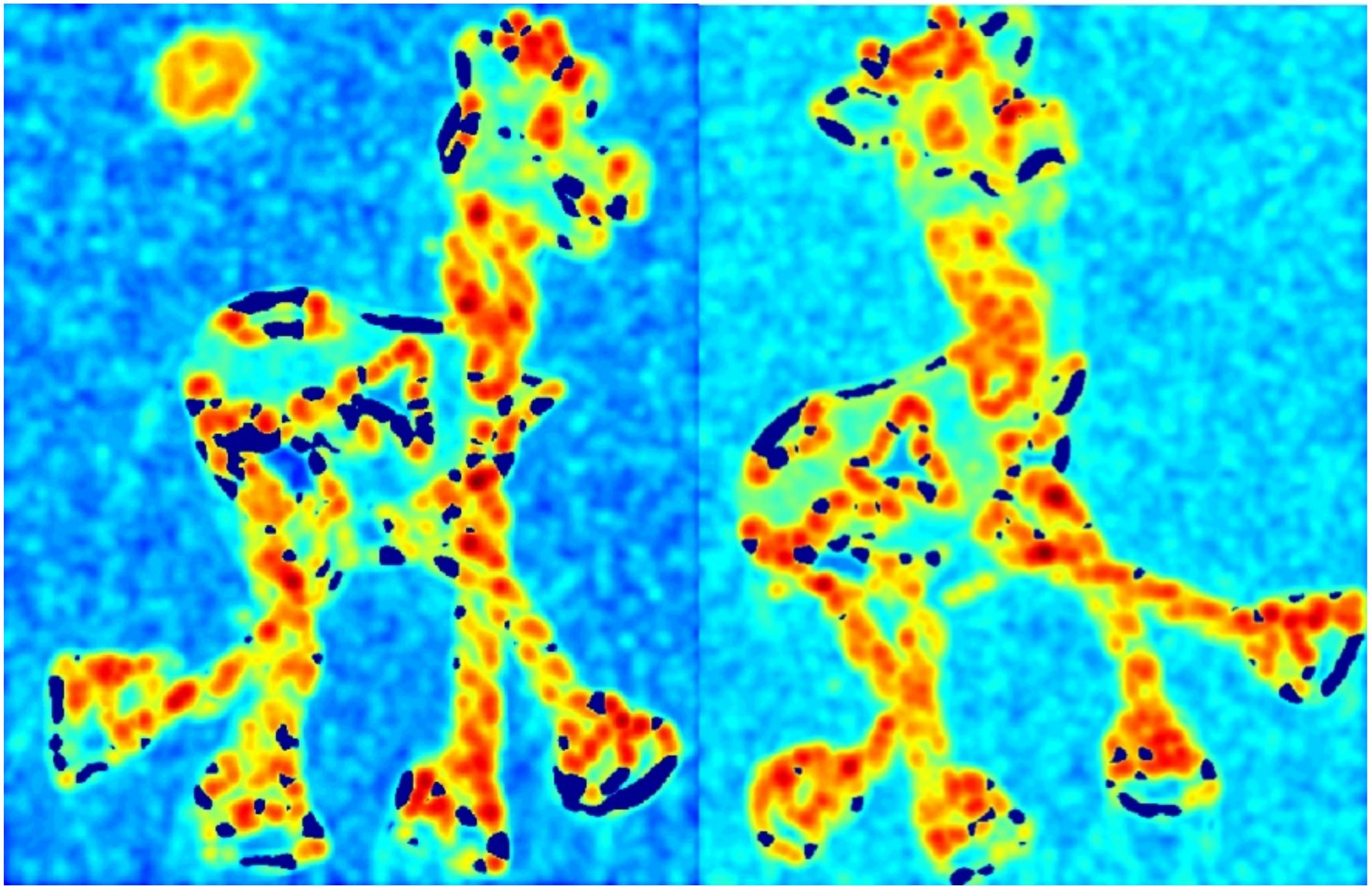
5. Keep points that are *local maxima* in R (e.g., points for which R is bigger than that of 4 or 8 neighbors)

Harris corner detector: Workflow

Compute corner response R

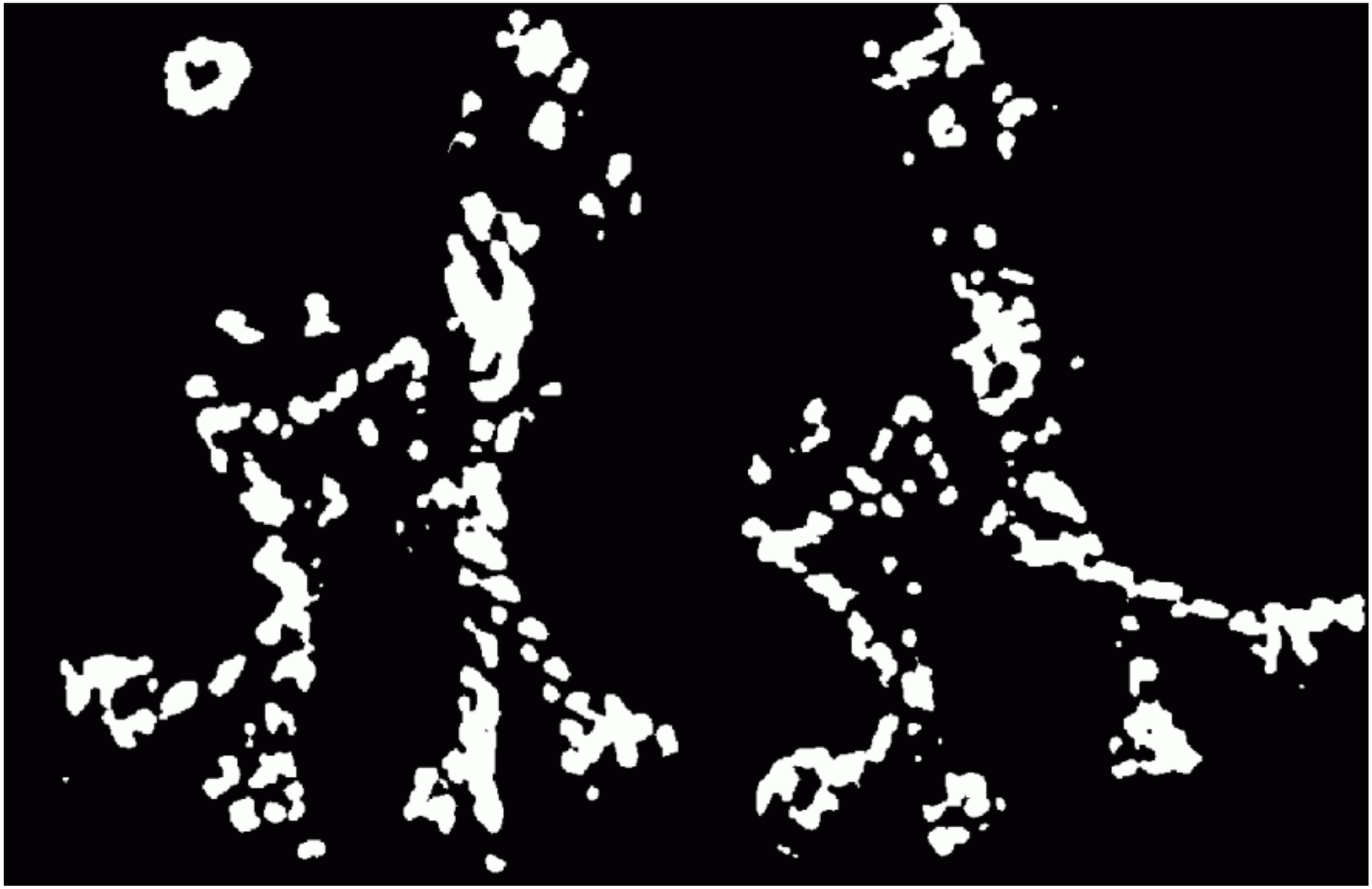
Harris corner detector: Workflow

Compute corner response R



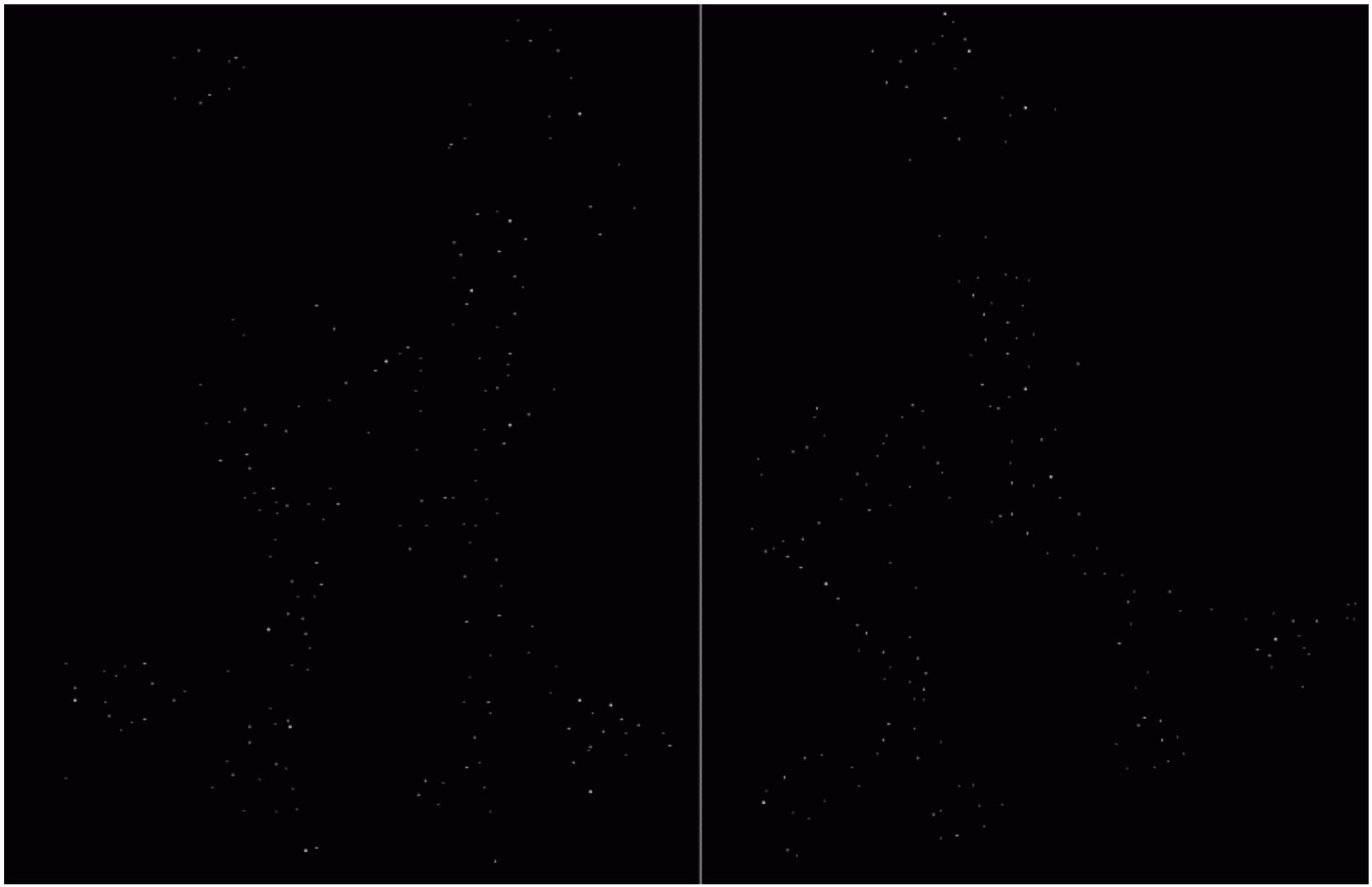
Harris corner detector: Workflow

Find points with large corner response $R >$ threshold

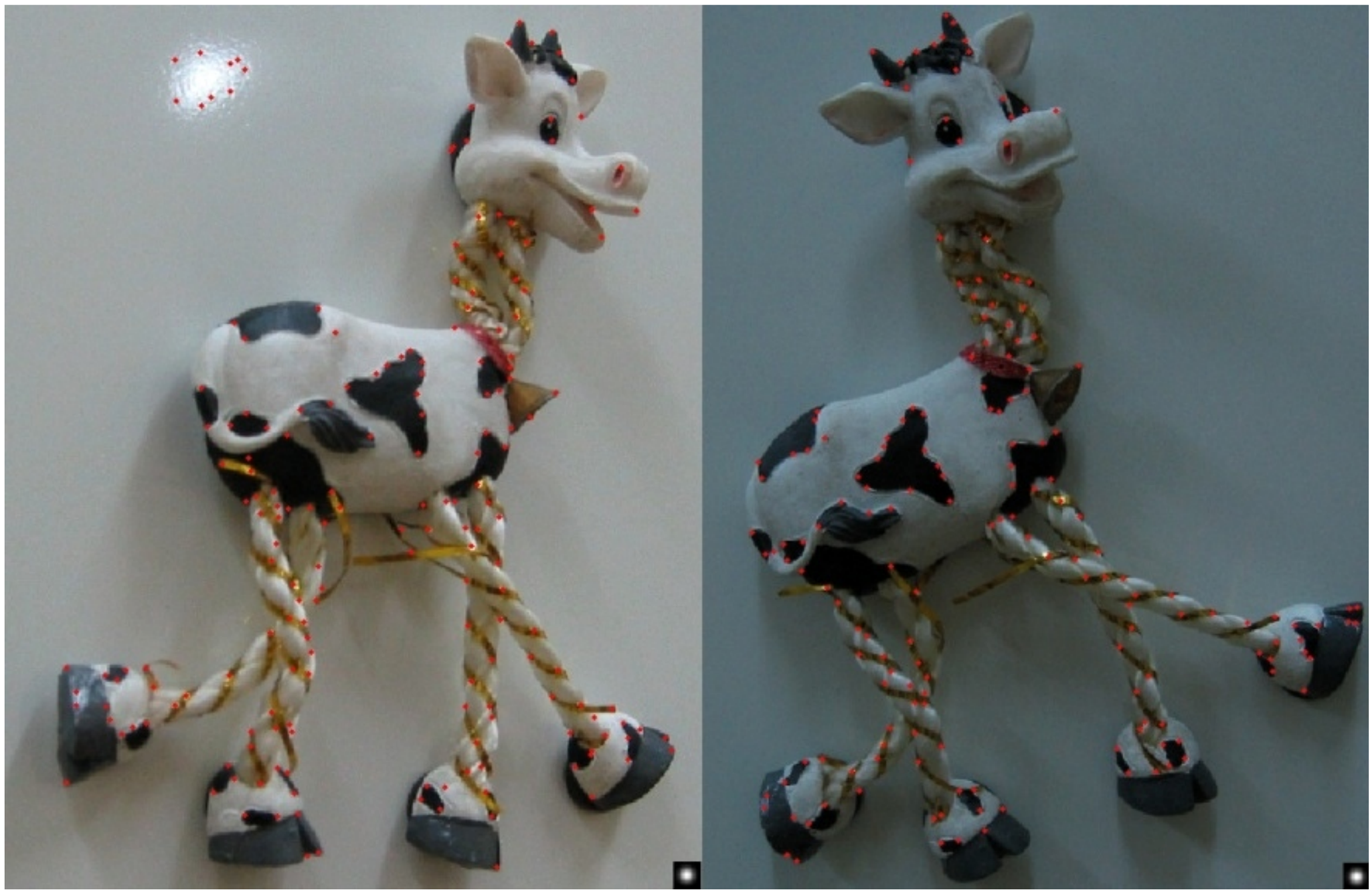


Harris corner detector: Workflow

Take local maxima in R



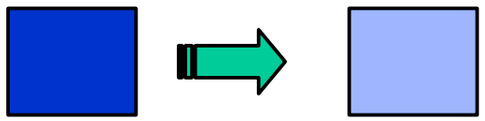
Harris corner detector: Workflow



Invariance of corner features

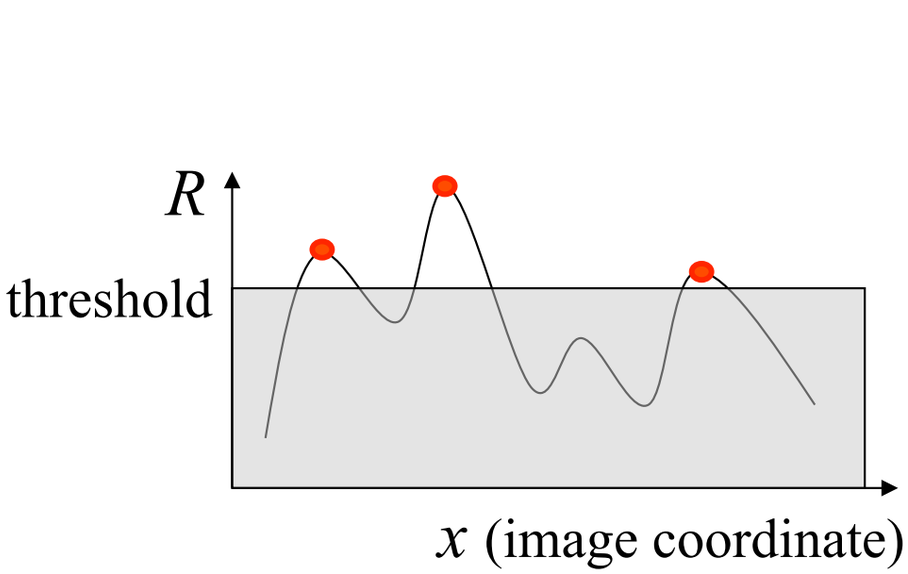
- What happens when the image undergoes geometric (rotation, scale) or photometric changes

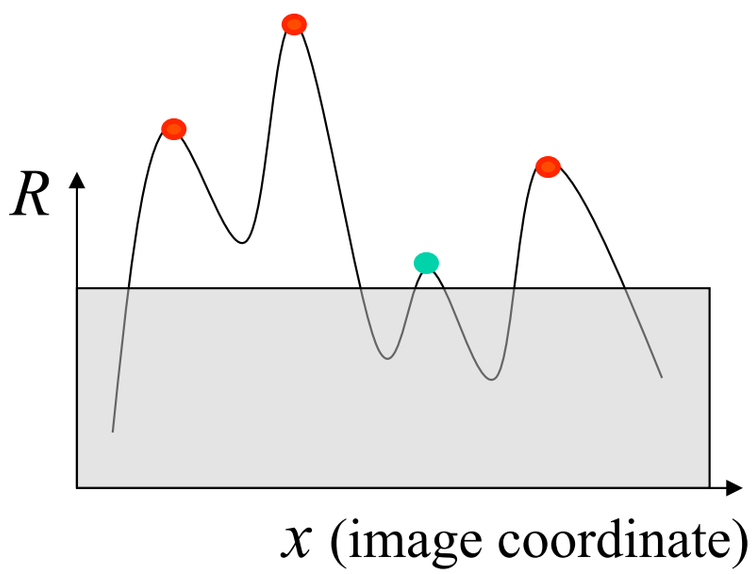
Invariance of corner features: Affine intensity change



$$I \rightarrow aI + b$$

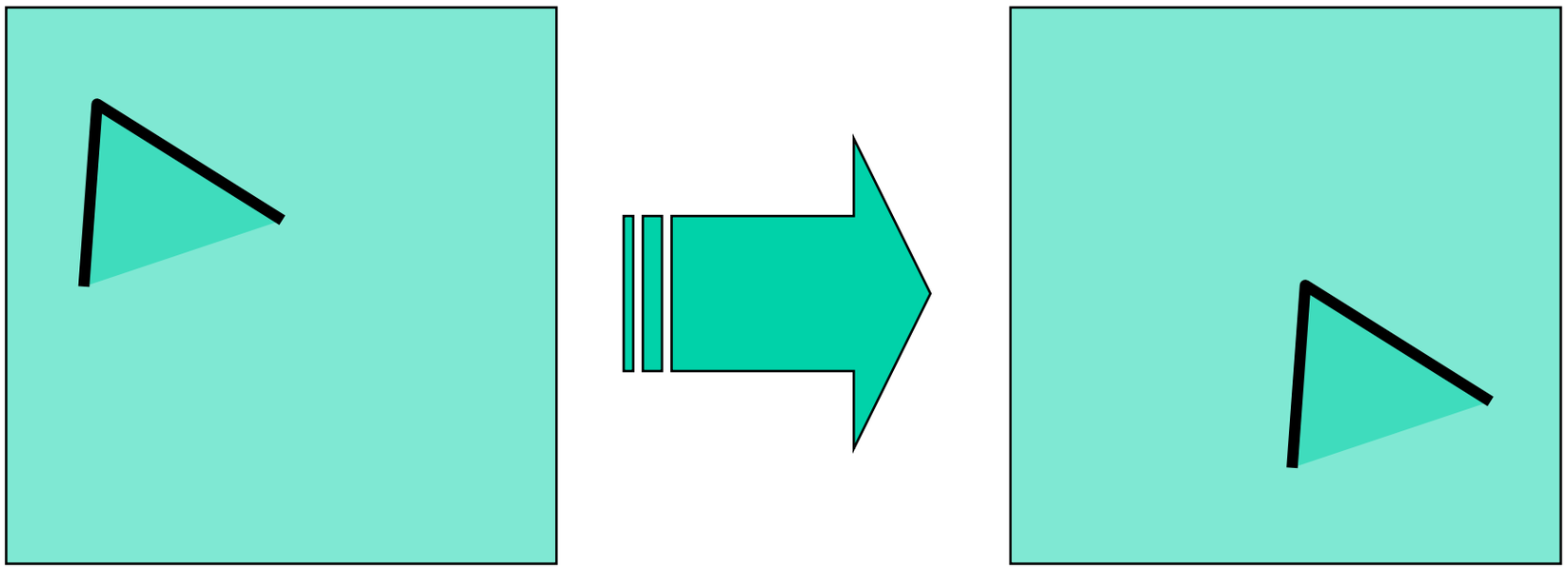
- Only derivatives are used —> invariance to intensity shift: $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow aI$





Partially invariant to affine intensity change

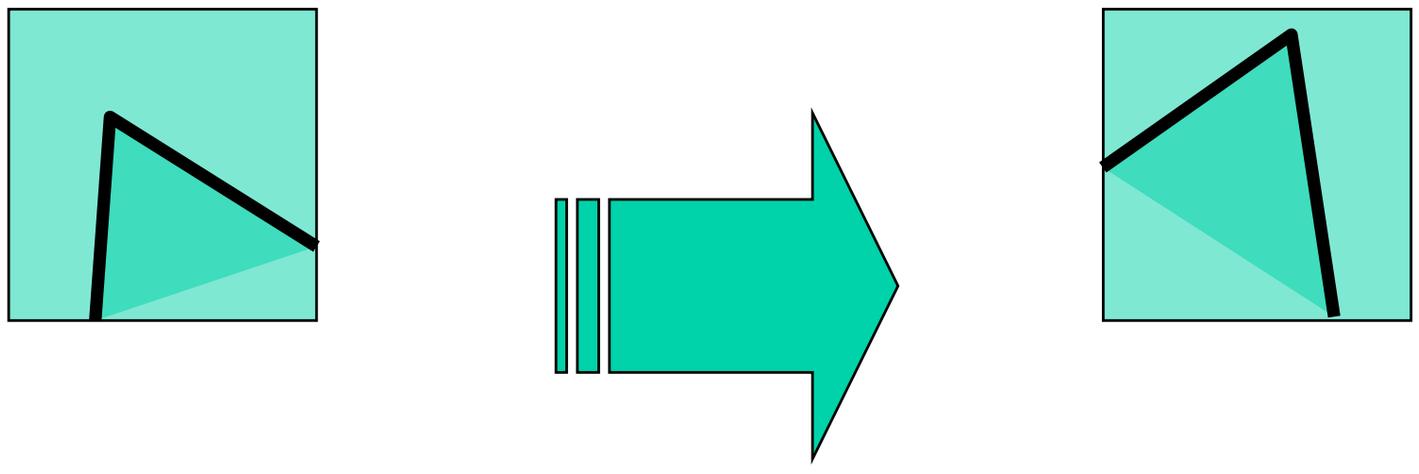
Invariance of corner features: Image translation



- Derivatives and window function are shift-invariant

Corner location is *covariant* w.r.t. translation

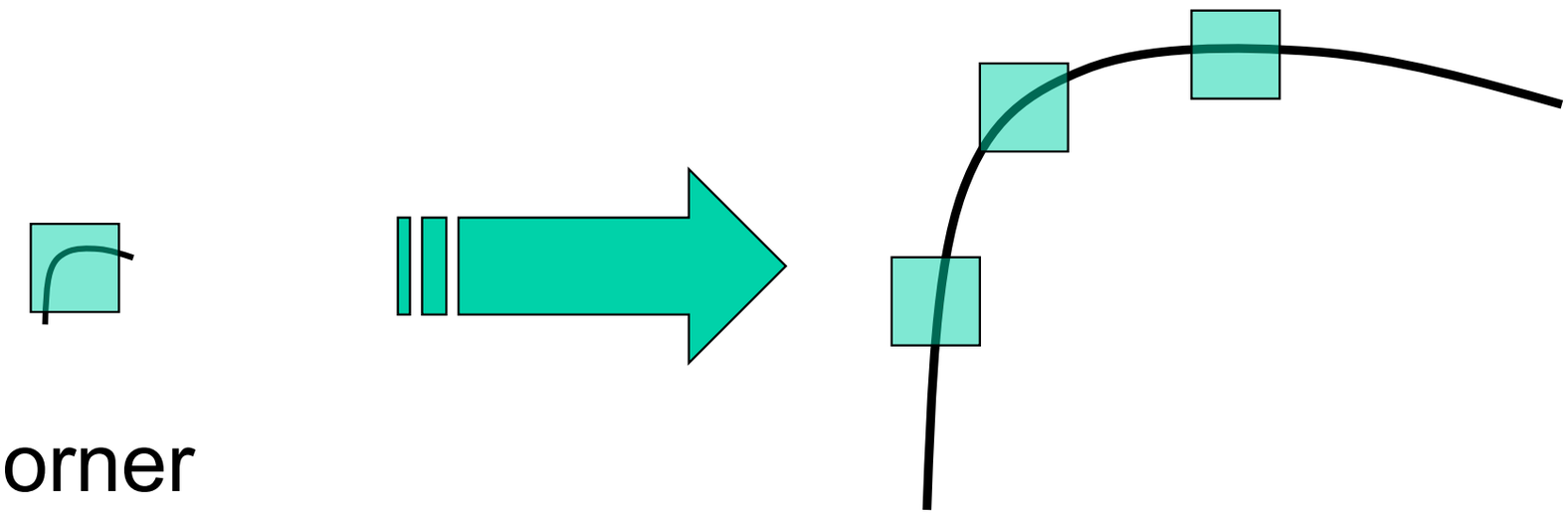
Invariance of corner features: Image rotation



- **Second moment ellipse rotates, but shape (i.e., eigenvalues) remain the same**

Corner location is covariant w.r.t. rotation

Invariance of corner features: Scaling



Corner

All points will be classified as edges

Corner location is not covariant to scaling!

Invariance of corner features: Scaling

image

zoomed image