计算机视觉

邬向前

计算学部

多模态智能及应用研究中心

电子邮箱: xqwu@hit.edu.cn

Corners, Blobs & Descriptors

Motivation: Build a Panorama

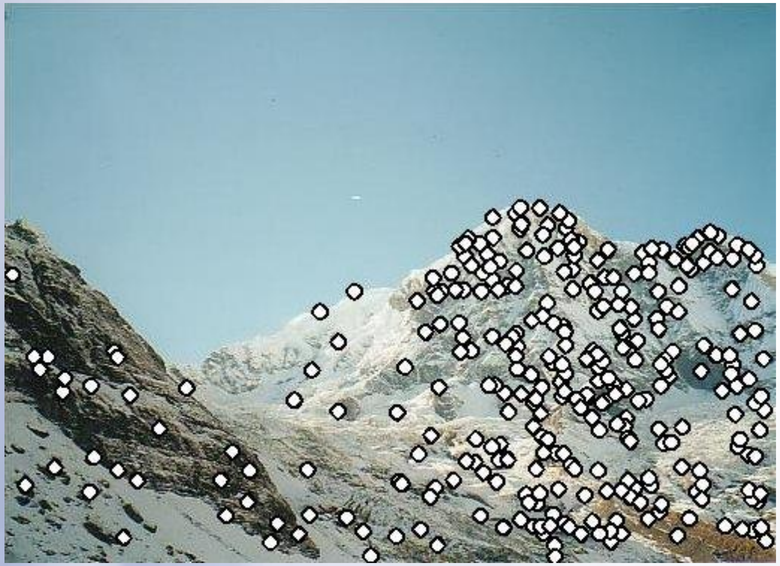
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

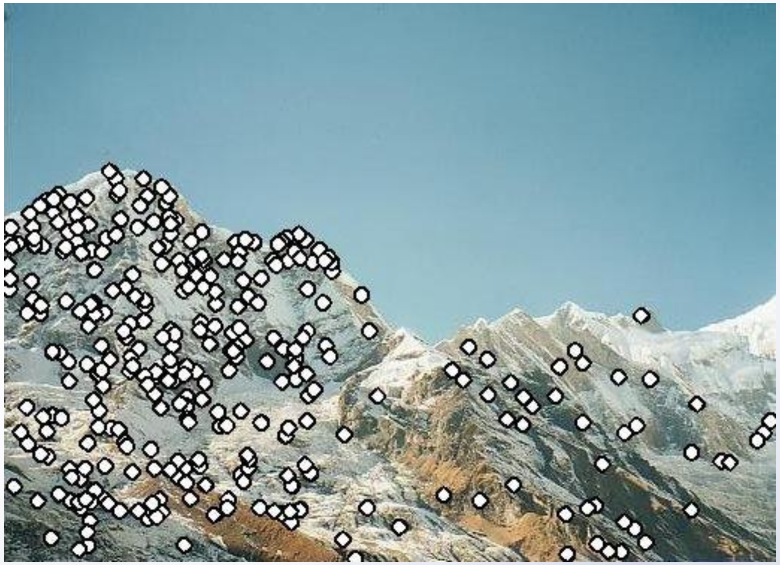
How do we build panorama?

- We need to match (align) images

Matching with Features

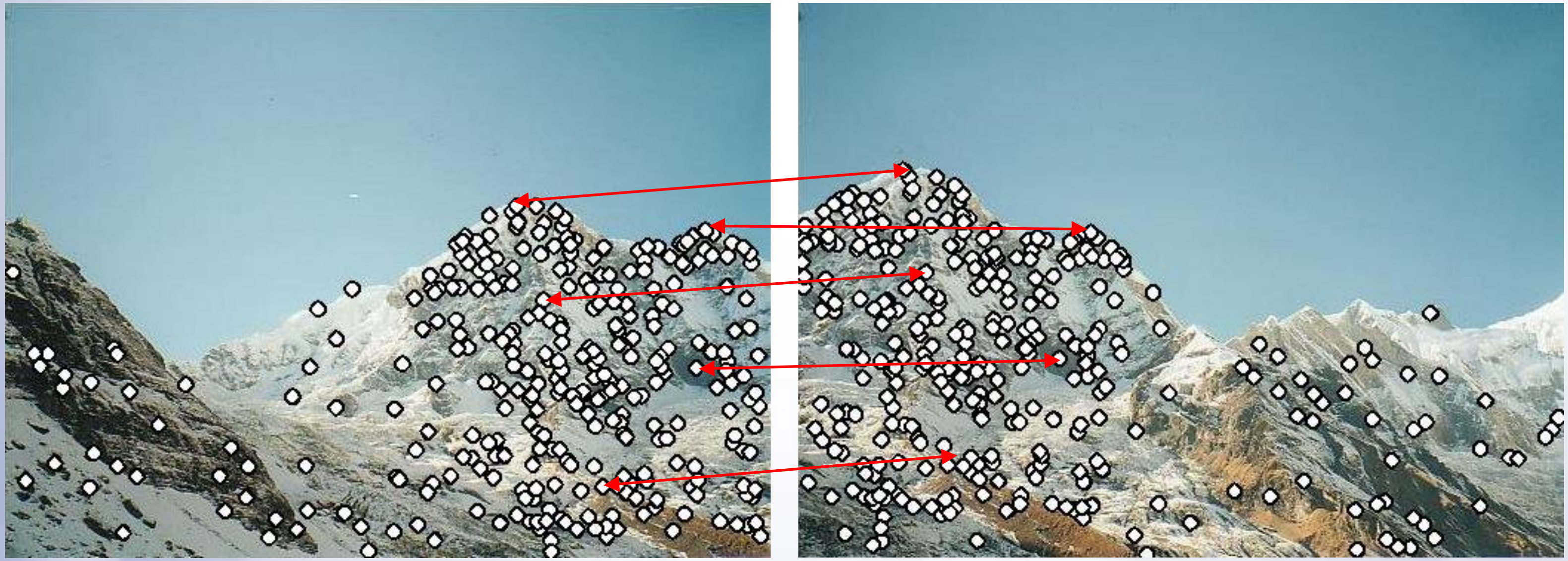
- Detect feature points in both images





Matching with Features

- Detect feature points in both images
- Find corresponding pairs



Matching with Features

- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Matching with Features

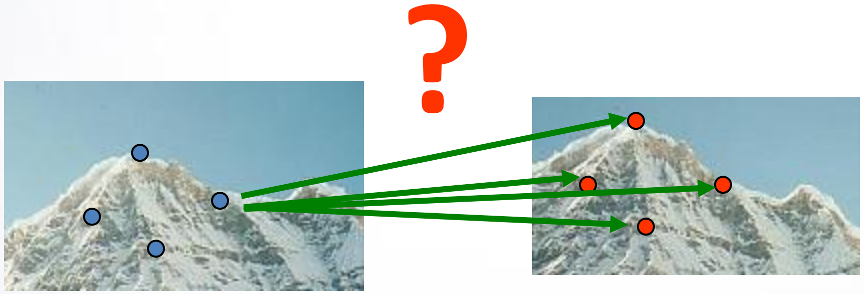
- Problem 1:
  - Detect the *same* point *independently* in both images

no chance to match!

We need a repeatable detector

Matching with Features

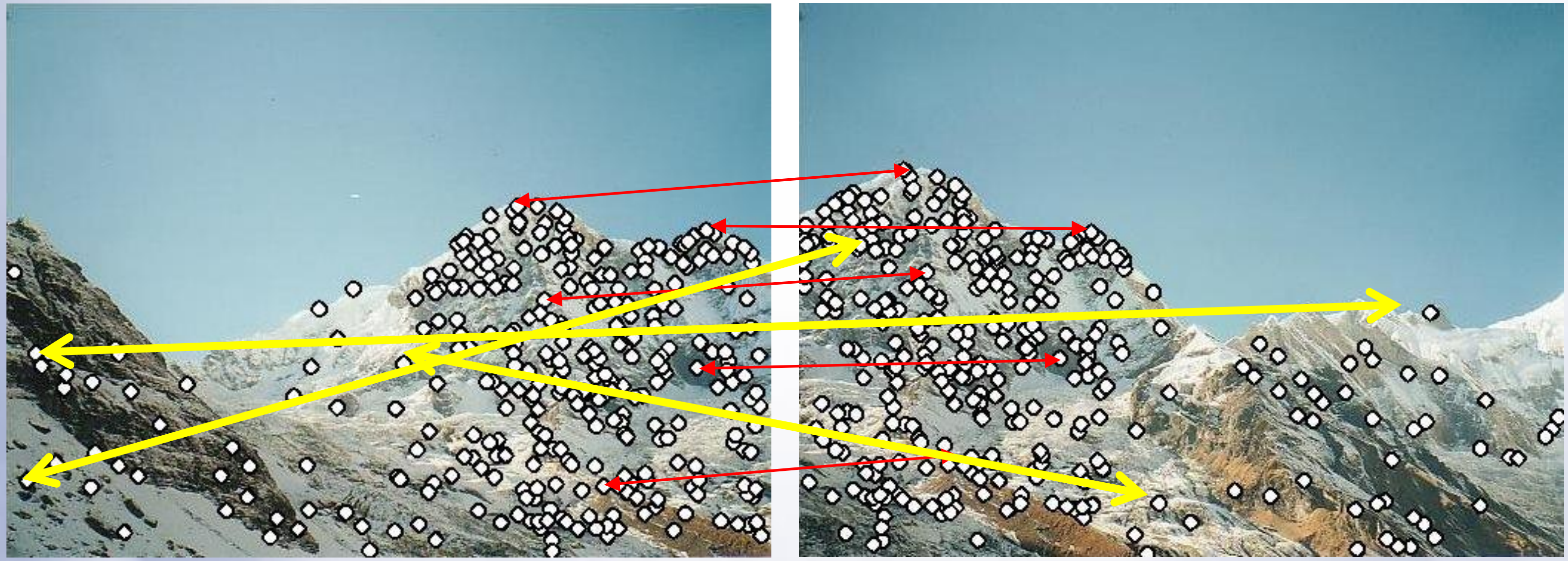
- Problem 2:
  - For each point correctly recognize the corresponding one



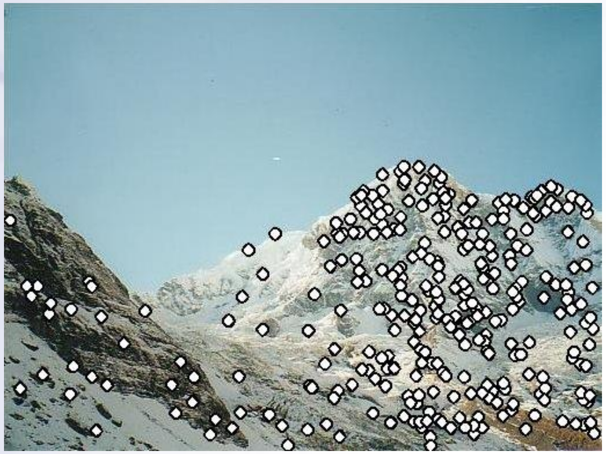
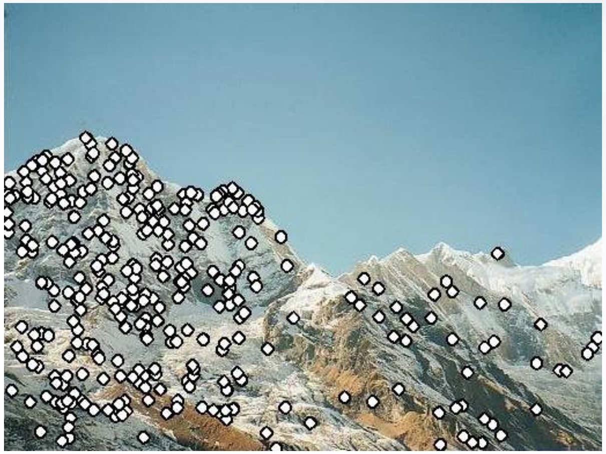
We need a reliable and distinctive descriptor

Matching with Features

- Problem 3:
  - Need to estimate transformation between images, despite erroneous correspondences.



Characteristics of good features

- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- 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

Applications

- Feature points are used for:
  - Motion tracking
  - Image alignment
  - 3D reconstruction
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation

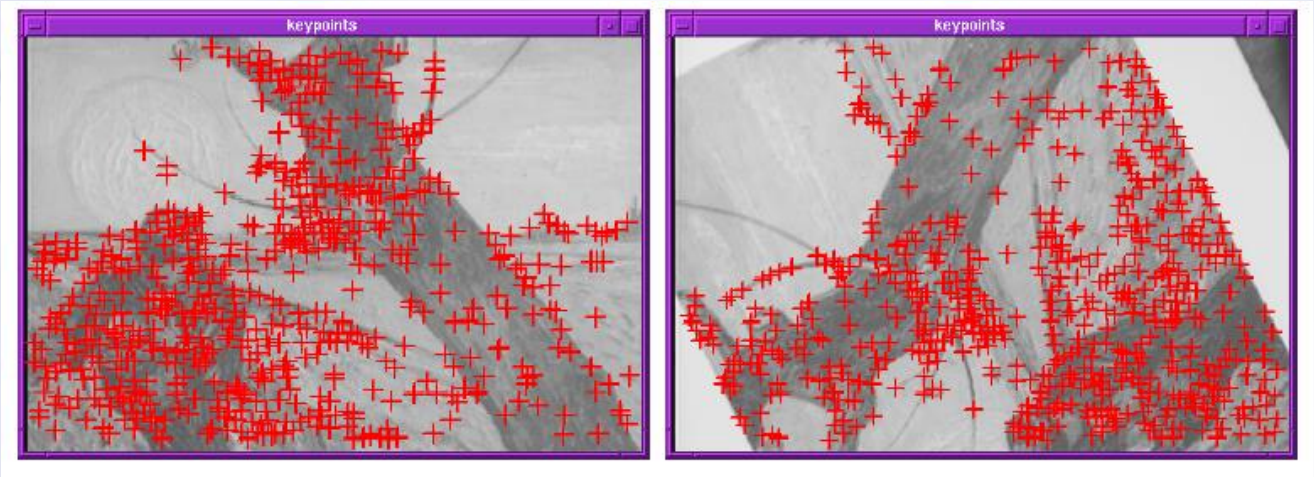
Overview

- Corners (Harris Detector)
- Blobs
- Descriptors

Overview

- Corners (Harris Detector)
- Blobs
- Descriptors

Finding Corners

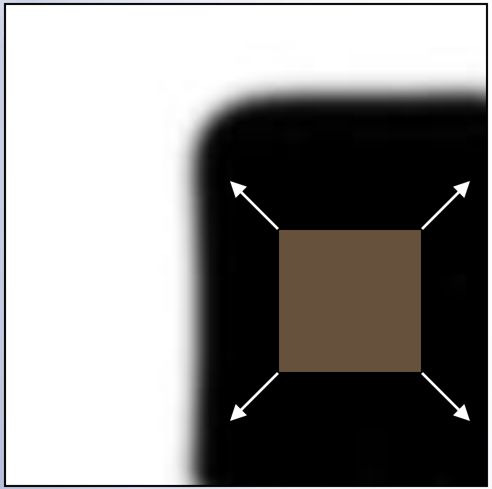


- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

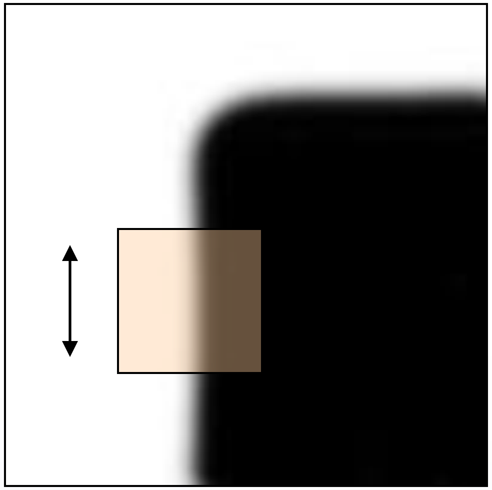
C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147--151.

Corner Detection: Basic Idea

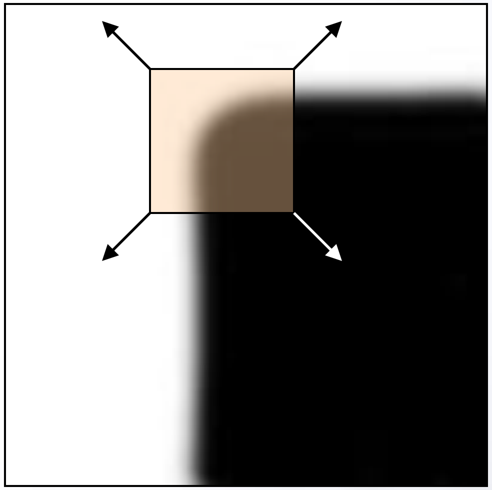
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in
all directions



"edge":
no change
along the edge
direction



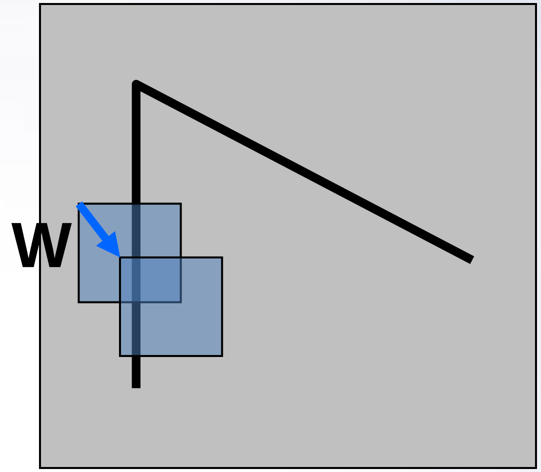
"corner":
significant
change in all
directions

Source: A. Efros

Feature detection: the math

Consider shifting the window **W** by (u,v)

- how do the pixels in **W** change?
- compare each pixel before and after by summing up the squared differences (SSD)
- this defines an SSD "error" of *E(u,v)*:



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

Source: S. Seitz

Small motion assumption

Taylor Series expansion of I:

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

If the motion (u,v) is small, then first order approx. is good

$$I(x+u, y+v) \approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x, y) + [I_x \;\; I_y] \begin{bmatrix} u \\ v \end{bmatrix}$$

shorthand: $I_x = \frac{\partial I}{\partial x}$

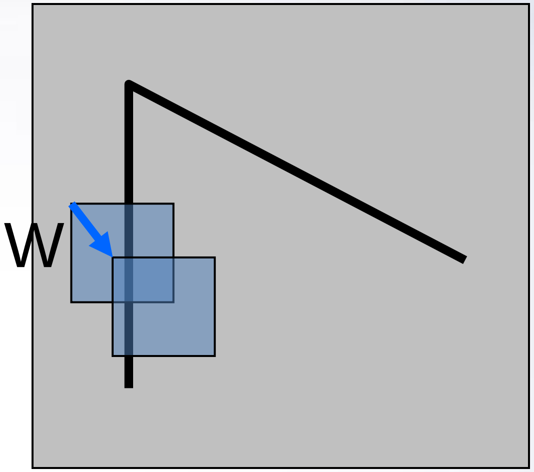
Plugging this into the formula on the previous slide…

Source: S. Seitz

Feature detection: the math

Consider shifting the window W by (u,v)

- how do the pixels in W change?
- compare each pixel before and after by summing up the squared differences
- this defines an "error" of E(u,v):



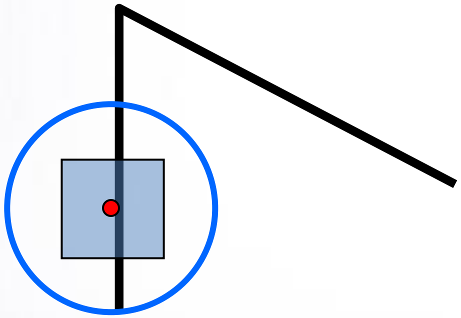
$$
\begin{aligned}
E(u,v) &= \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2 \\
&\approx \sum_{(x,y)\in W} \left[I(x,y) + [I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)\right]^2 \\
&\approx \sum_{(x,y)\in W} \left[[I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix}\right]^2
\end{aligned}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

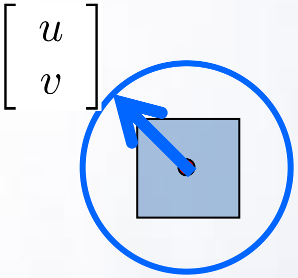
Source: S. Seitz

Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u \ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





For the example above

- You can move the center of the green window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?
- We can find these directions by looking at the eigenvectors of ***H***

Source: S. Seitz

$$E(u,v) = e^T He, e^T e = 1, e = \begin{bmatrix} u \\ v \end{bmatrix}$$

拉格朗日乘数法，求解约束优化问题

$$\begin{cases} E_1(e,\lambda) = e^T He - \lambda(e^T e - 1) \\ e^T e - 1 = 0 \end{cases}$$

求极值，就是令一阶导等于0

$$\frac{dE_1(e,\lambda)}{de} = He - \lambda e = 0 \rightarrow He = \lambda e$$

带入原函数

$$E(u,v)_{max} = e^T \lambda e = \lambda$$

Quick eigenvalue/eigenvector review

The **eigenvectors** of a matrix **A** are the vectors **x** that satisfy:

$$Ax = \lambda x$$

The scalar λ is the **eigenvalue** corresponding to **x**

- The eigenvalues are found by solving:

$$det(A - \lambda I) = 0$$

- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have

$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

- The solution:

$$\lambda_{\pm} = \frac{1}{2}\left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2}\right]$$

Once you know λ, you find **x** by solving

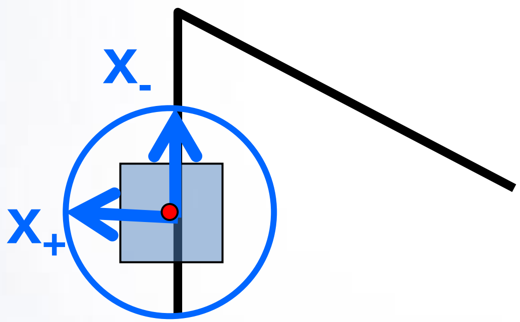
$$\begin{aligned} Hx_+ &= \lambda_+ x_+ \\ Hx_- &= \lambda_- x_- \end{aligned}$$

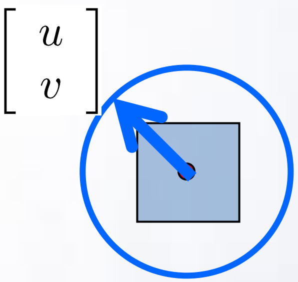
Source: S. Seitz

Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u \;\; v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





Eigenvalues and eigenvectors of H

- Define shifts with the smallest and largest change (E value)
- x_+ = direction of **largest** increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of **smallest** increase in E.
- λ- = amount of increase in direction x_+

$$\begin{aligned} Hx_+ &= \lambda_+ x_+ \\ Hx_- &= \lambda_- x_- \end{aligned}$$

Source: S. Seitz

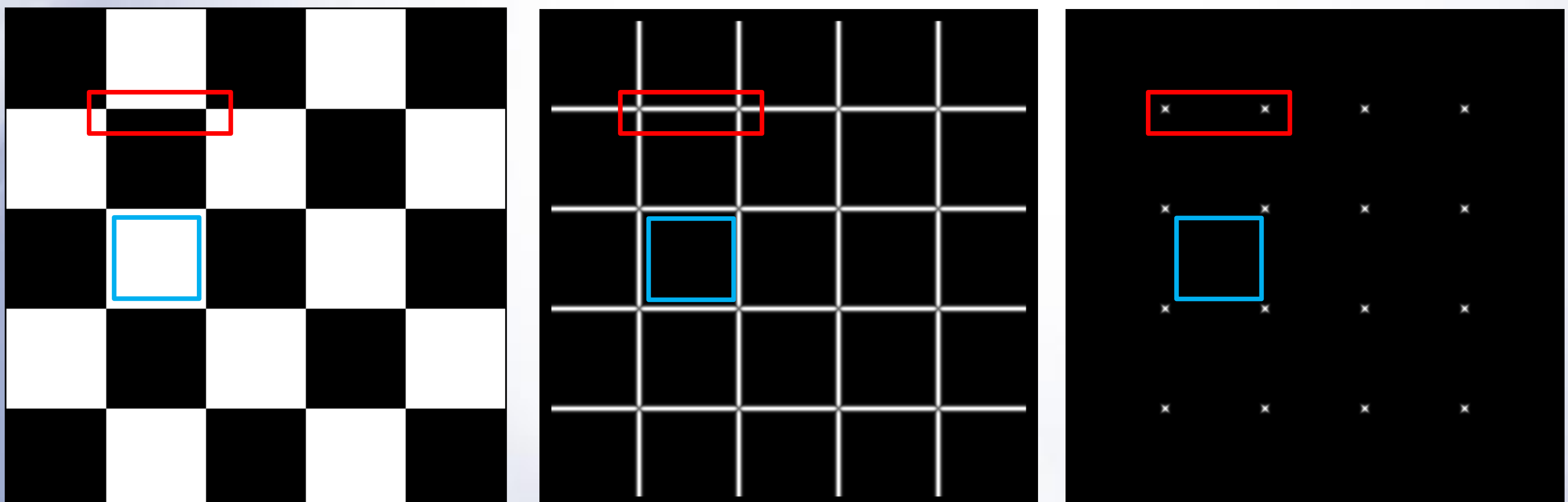
Feature detection: the math

How are λ_+, $\mathbf{x}_+$, λ_-, and $\mathbf{x}_+$ relevant for feature detection?

- What's our feature scoring function?

Want *E(u,v)* to be ***large*** for small shifts in ***all*** directions

- the *minimum* of *E(u,v)* should be large, over all unit vectors [u v]
- this minimum is given by the smaller eigenvalue (λ_-) of ***H***



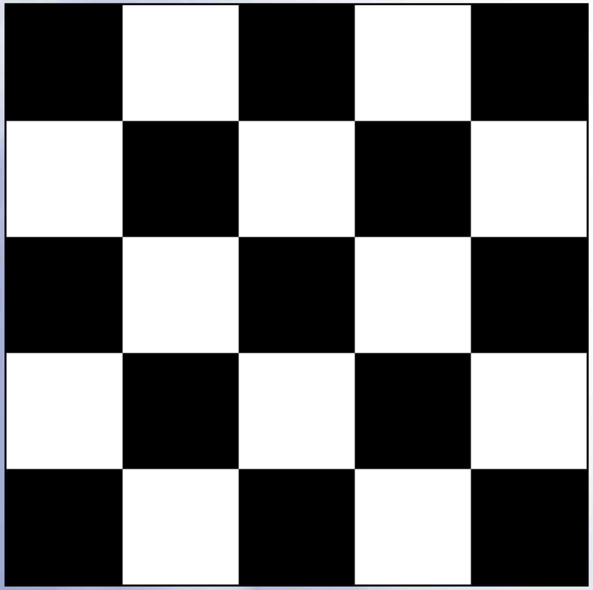
I λ_+ λ_-

Source: S. Seitz

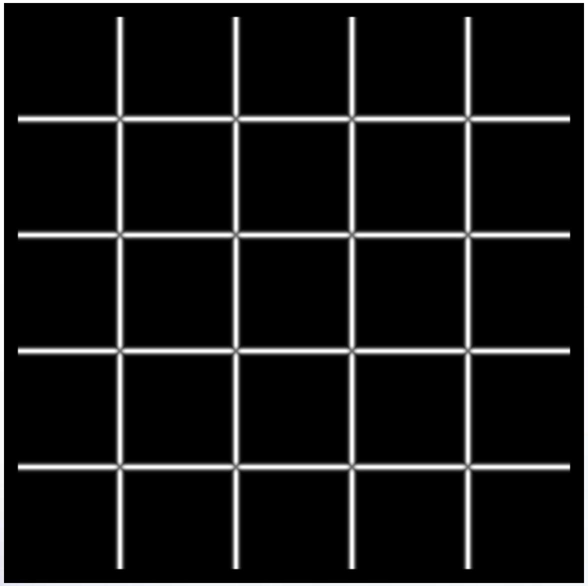
Feature detection summary

Here's what you do

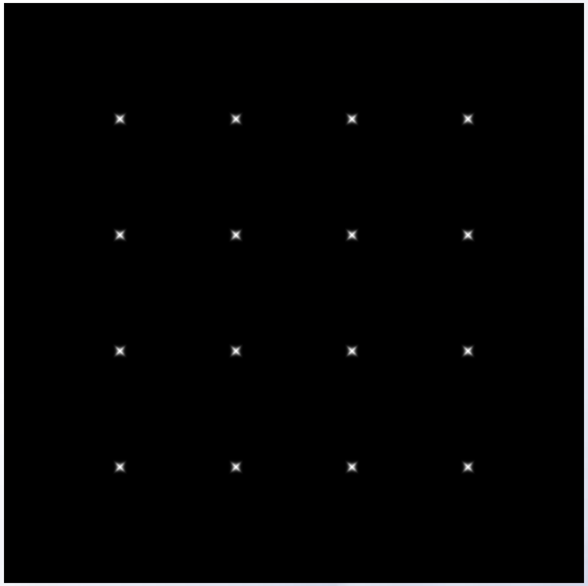
- Compute the gradient at each point in the image
- Create the ***H*** matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features



I



λ_+



λ_-

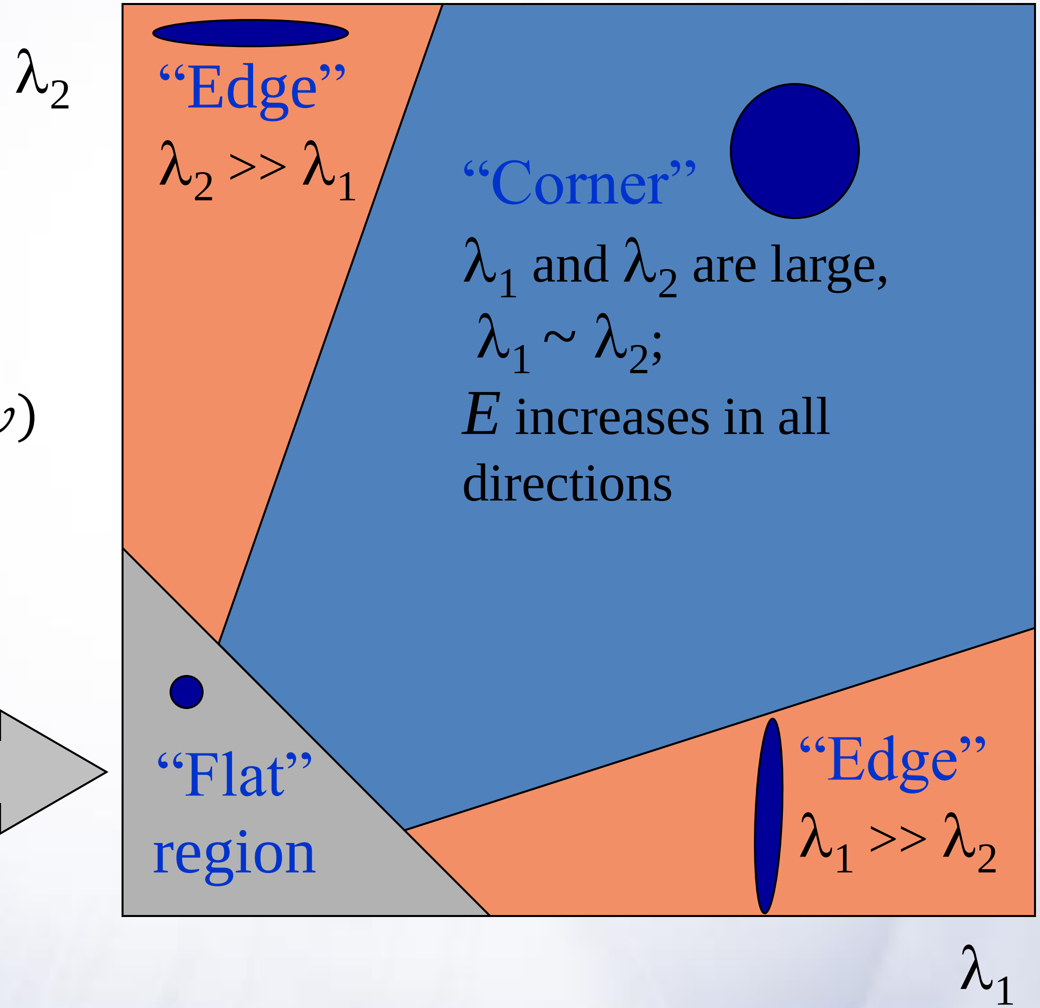
Source: S. Seitz

Interpreting the eigenvalues

Classification of image points using eigenvalues of *H*:

$$\lambda_+ u^2 + \lambda_- v^2 = E(u,v)$$

$$\left(\frac{u}{1/\sqrt{\lambda_+}}\right)^2 + \left(\frac{v}{1/\sqrt{\lambda_-}}\right)^2 = E(u,v)$$

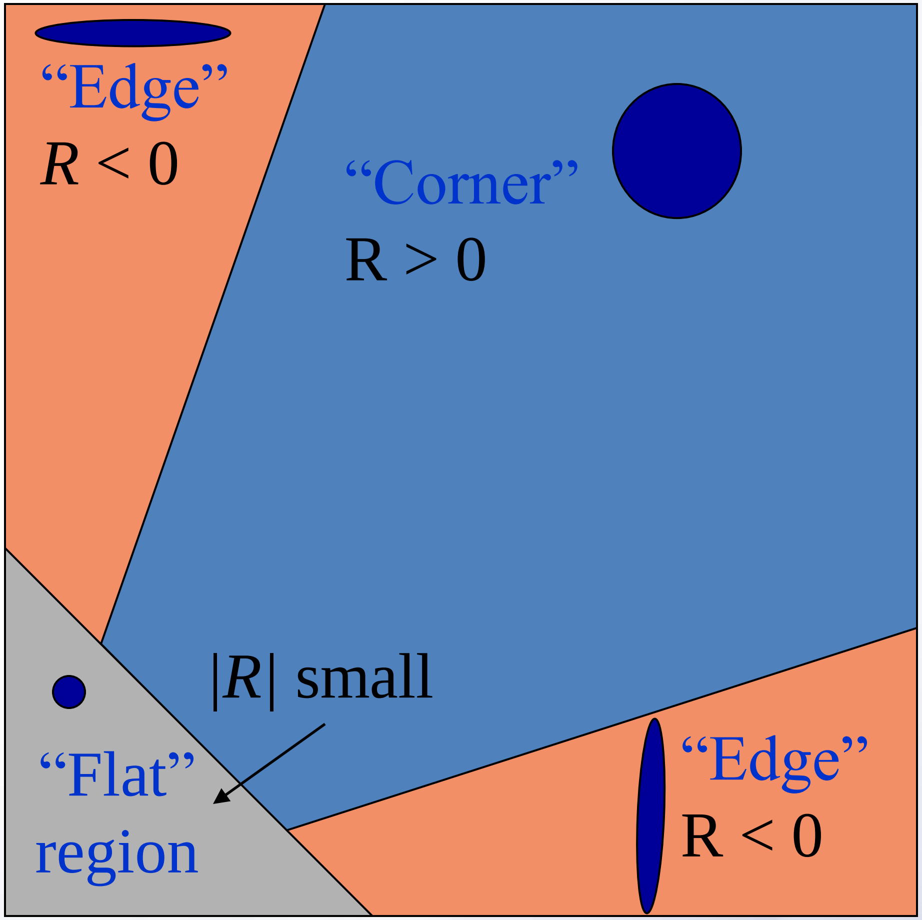


λ_1 and λ_2 are small;
E is almost constant in all directions

Corner response function

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

α: constant (0.04 to 0.06)



Harris detector: Steps

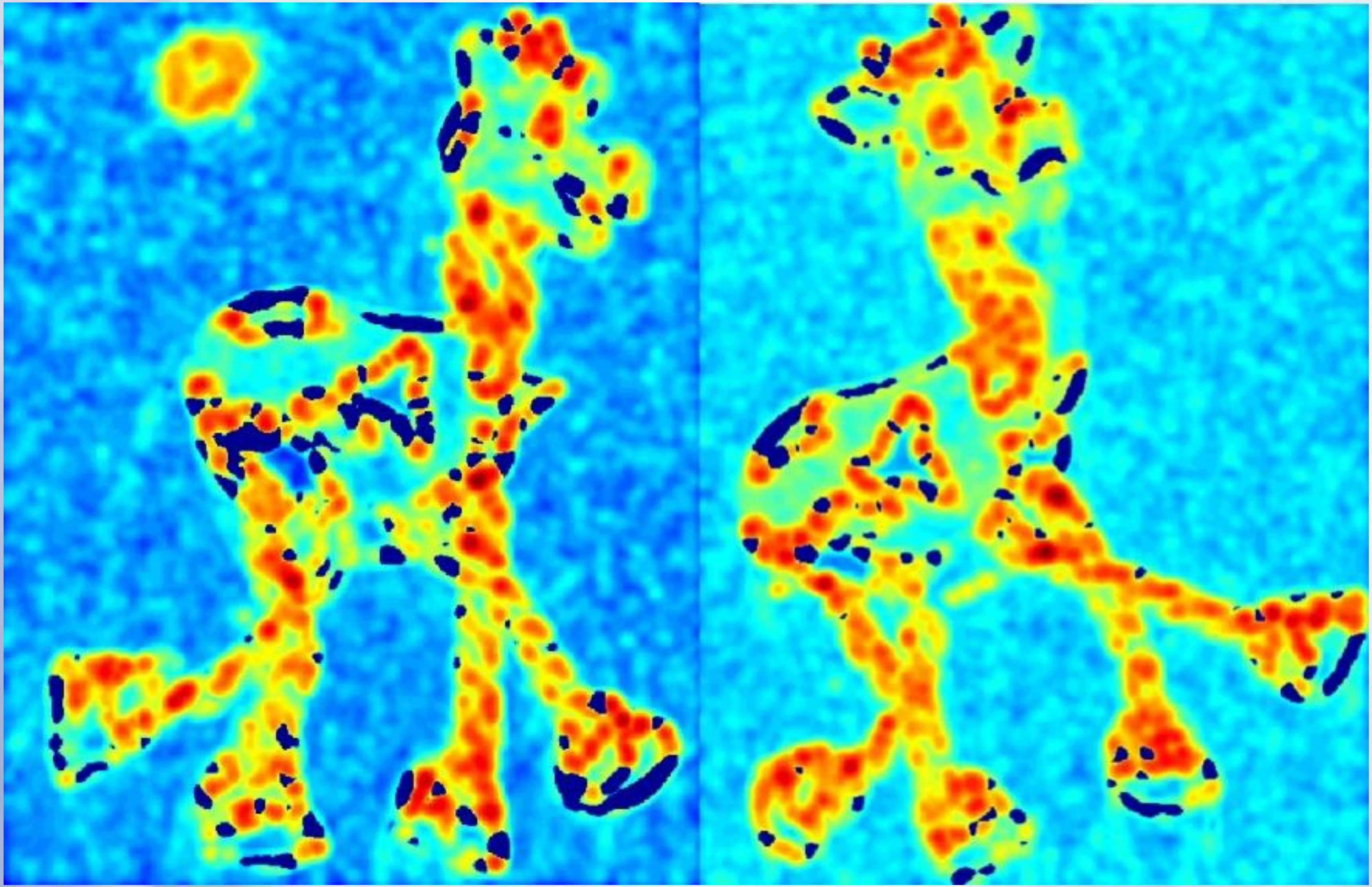
1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix H in a Gaussian window around each pixel
3. Compute corner response function R
4. Threshold R
5. Find local maxima of response function (nonmaximum suppression)

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

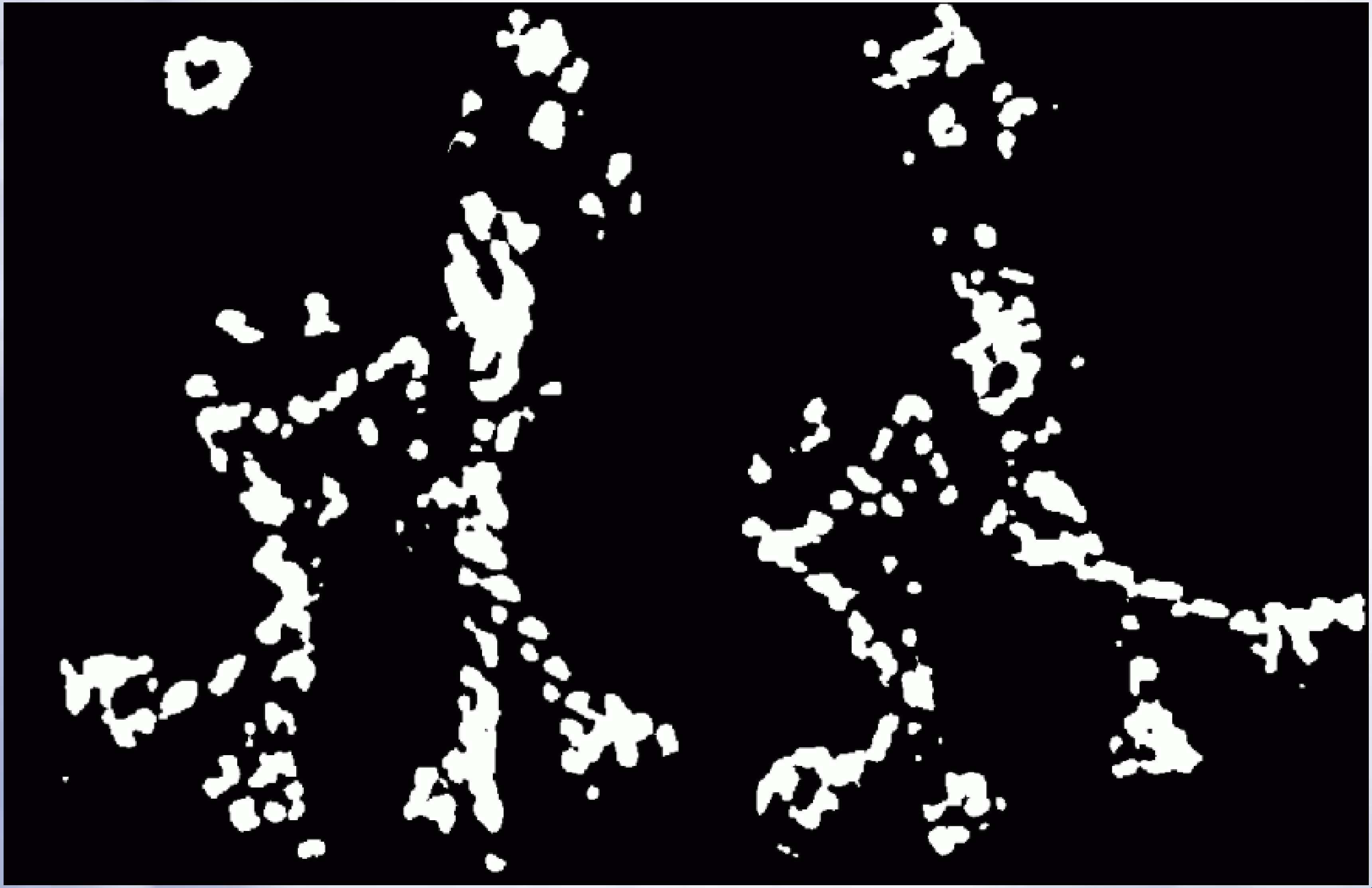
Harris Detector: Steps

Compute corner response R



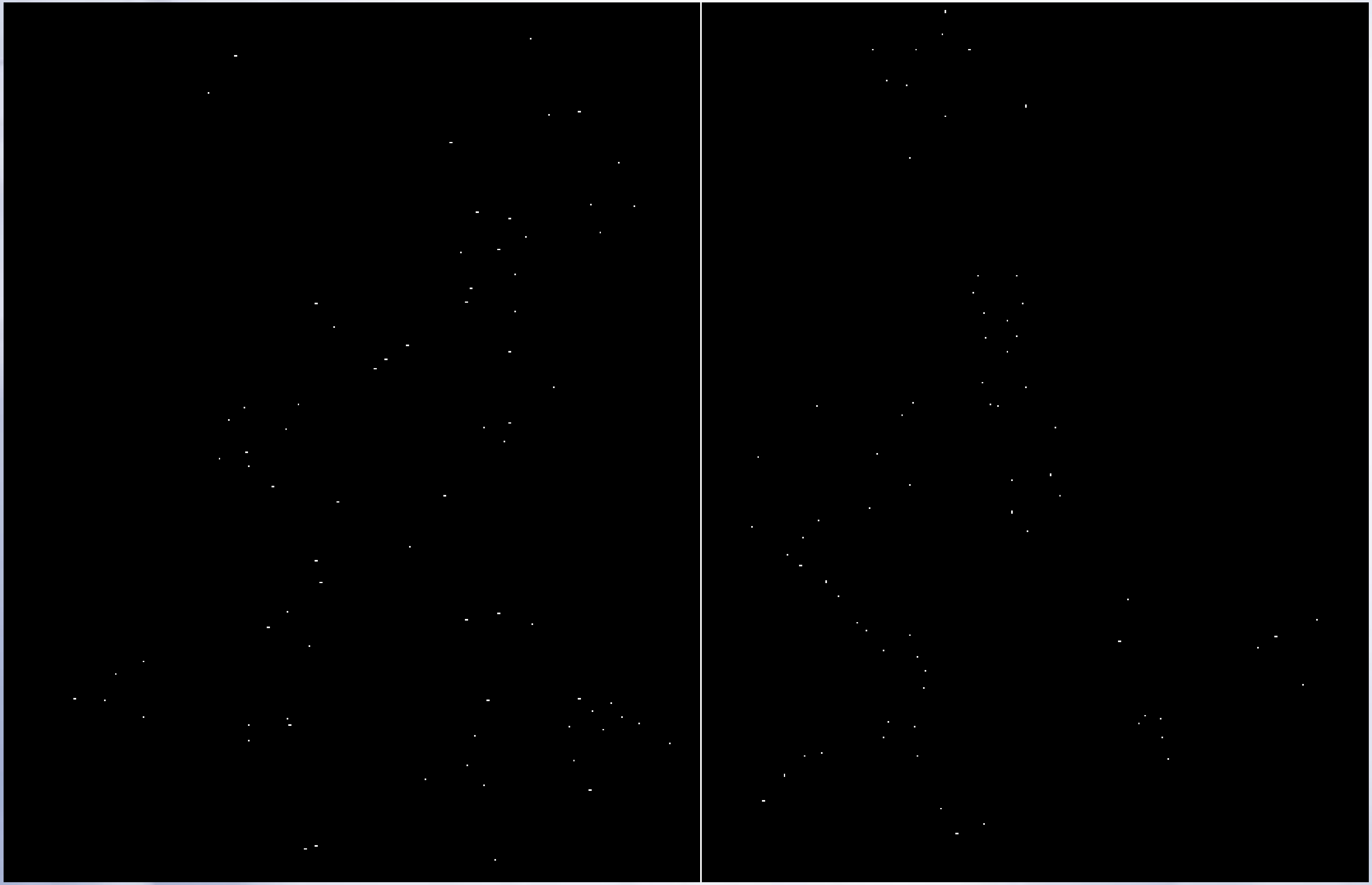
Harris Detector: Steps

Find points with large corner response: R>threshold



Harris Detector: Steps

Take only the points of local maxima of R

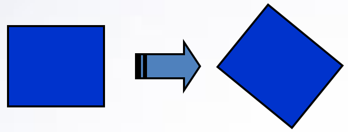


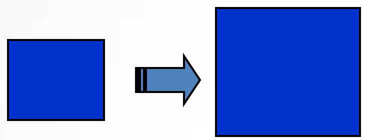
Harris Detector: Steps

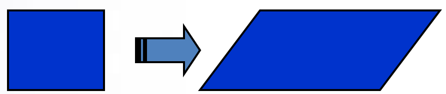
Invariance and covariance

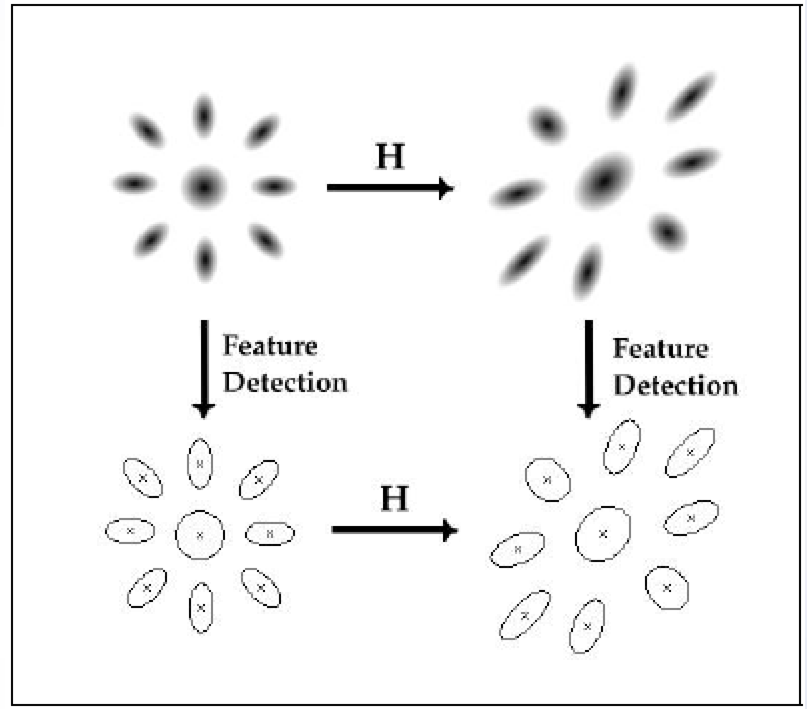
- We want features to be *invariant* to photometric transformations and *covariant* to geometric transformations
  - **Invariance:** image is transformed and features do not change
  - **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations

Transformations

- Geometric
  - **Rotation** 

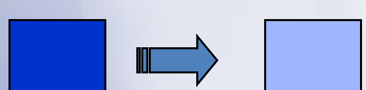
    Scale 

  - **Affine** 
    valid for:
    orthographic camera,
    locally planar object



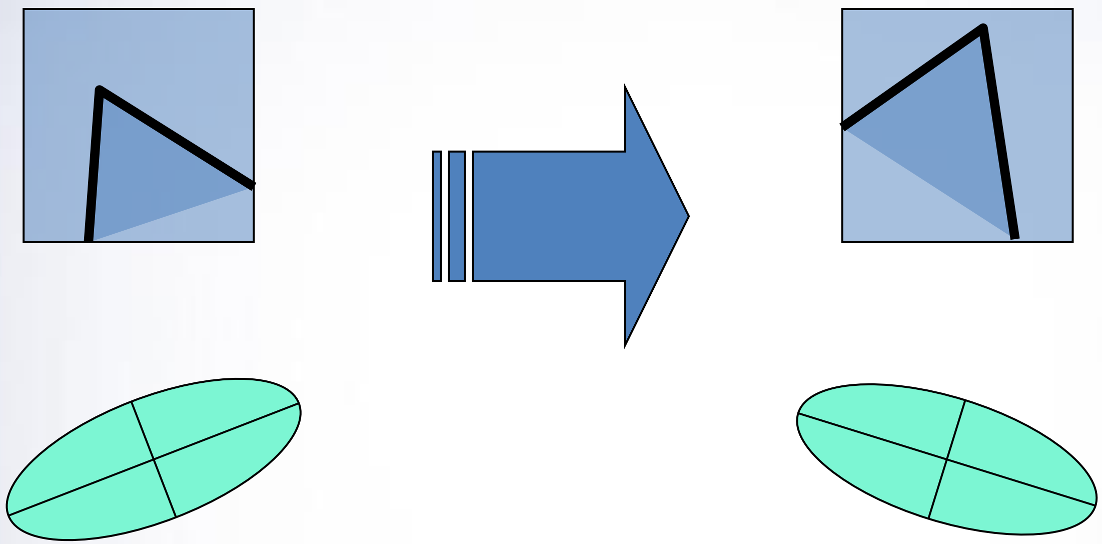
T. Kadir, A. Zisserman and M. Brady, An Affine invariant salient region detector, ECCV 2004

- Photometric
  - **Affine intensity change** $(I \rightarrow a\, I + b)$



Slide credit: S. Lazebnik

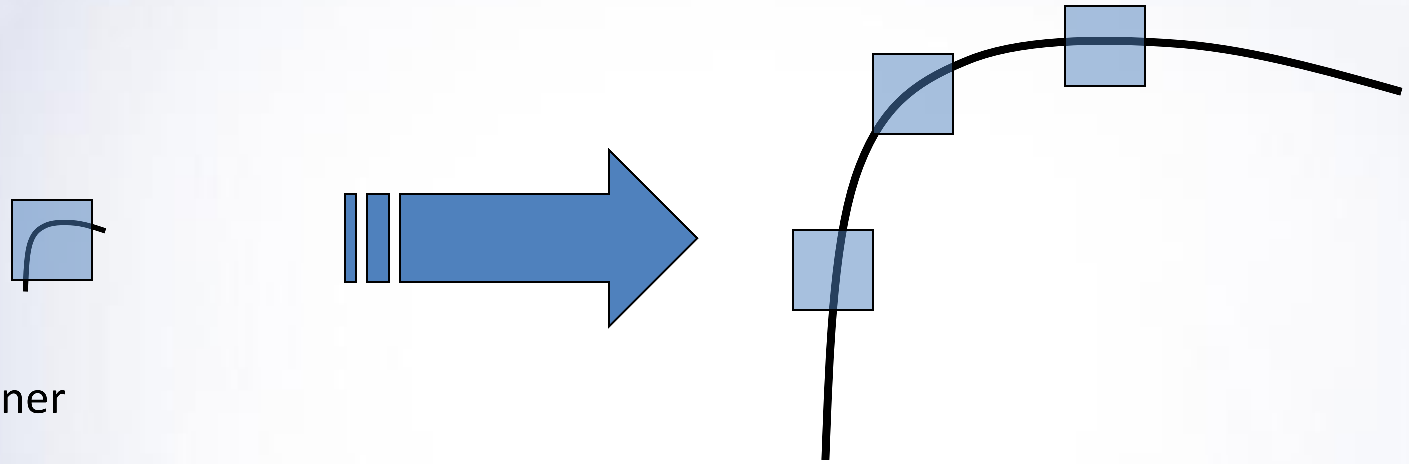
Image rotation



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

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

Scaling



Corner

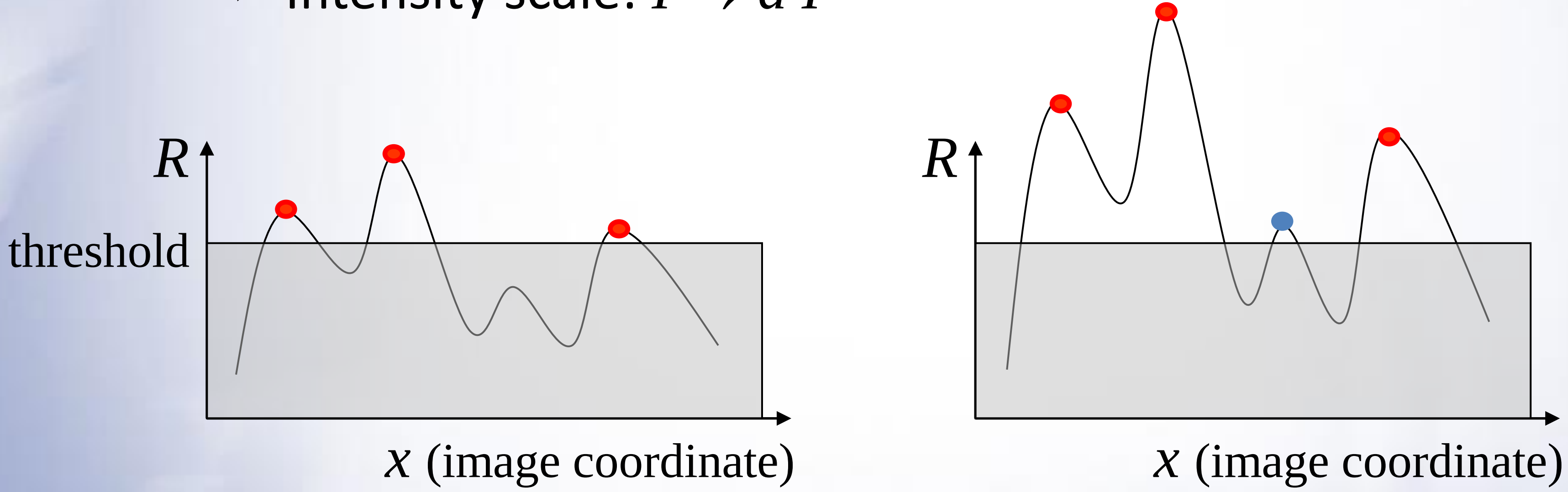
All points will be
classified as edges

Not invariant to scaling

Affine intensity change

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

✓ Intensity scale: $I \to a\, I$



Partially invariant to affine intensity change

作业：

1. 编程实现Harris角点检测算法；
2. 编程实现斑点检测算法。

So much for today!

Thank you !!!