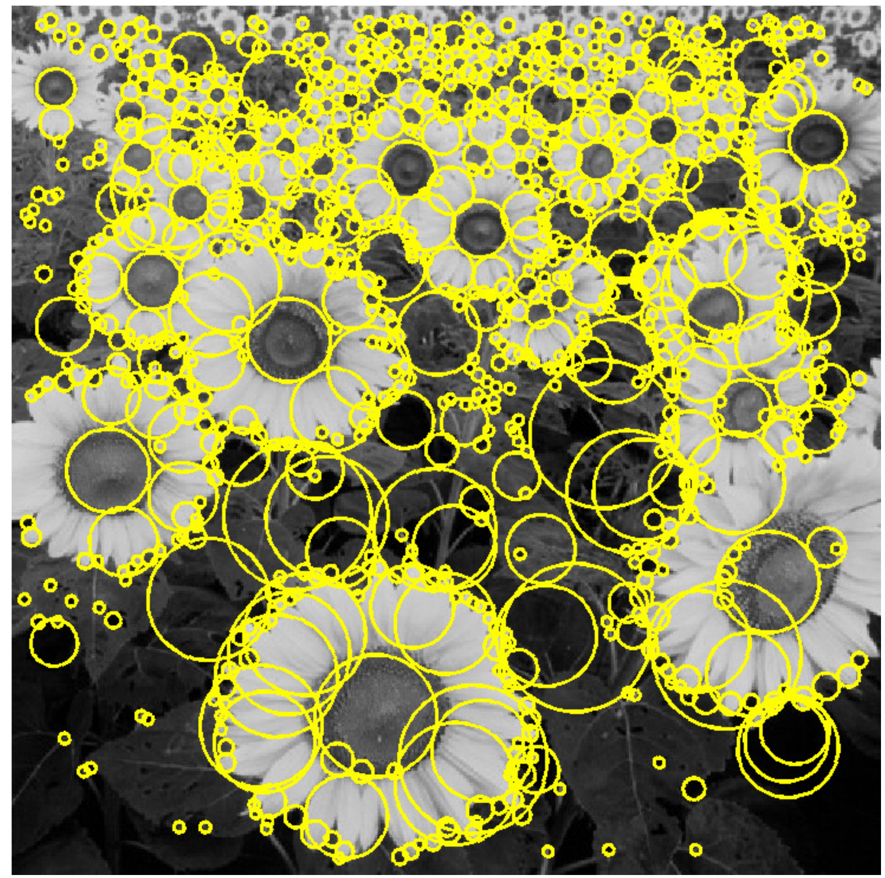
Harris corner detection

Feature extraction: Corners and blobs

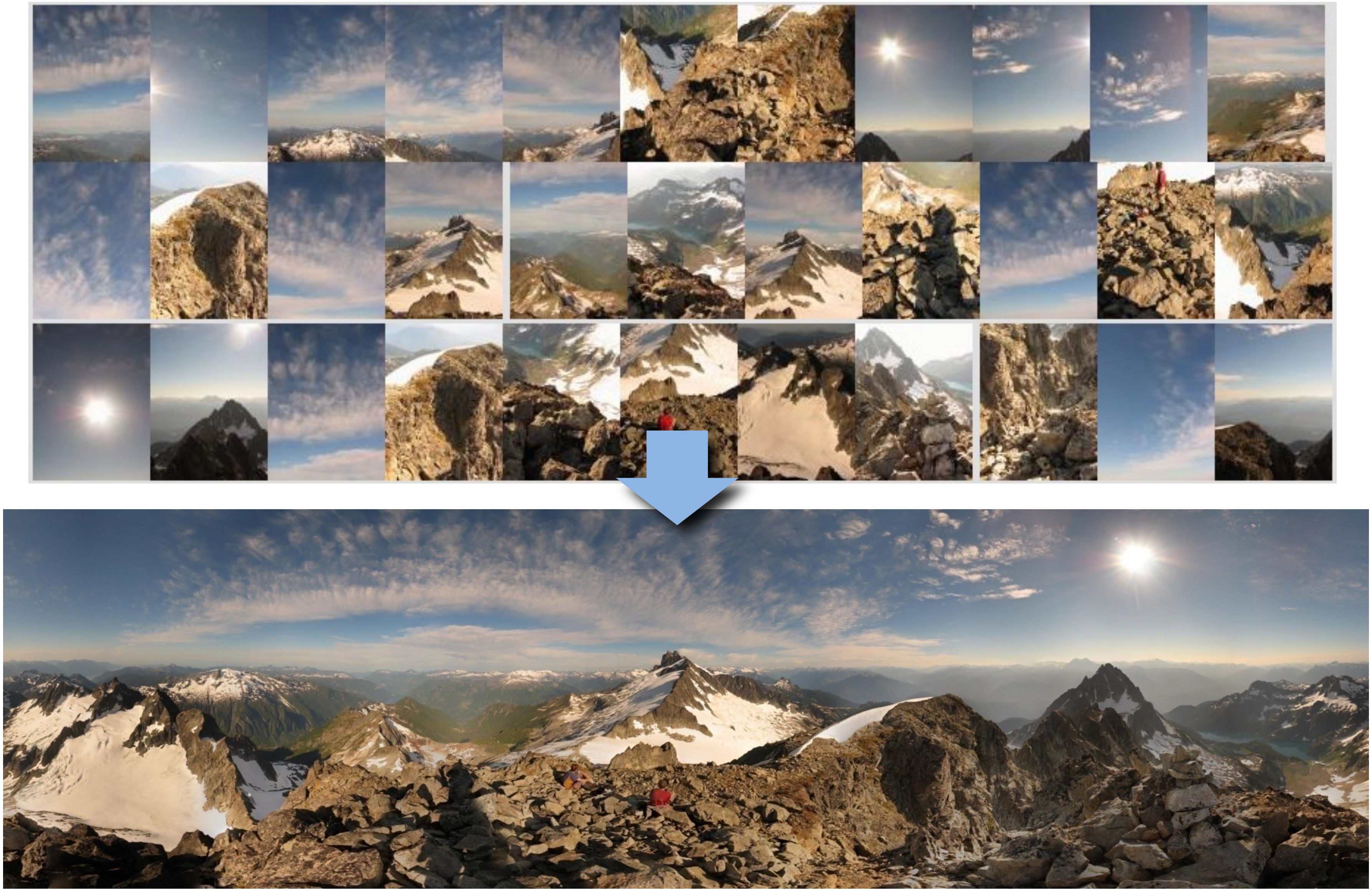


Feature extraction: Corners and blobs

- Interest points:
  - A corner can be defined as the intersection of two edges.
  - blob detection methods are aiming at detecting regions in the image that differ in properties, such as brightness or color, compared to surrounding regions

(credit: Wikipedia)

Motivation: Automatic panoramas



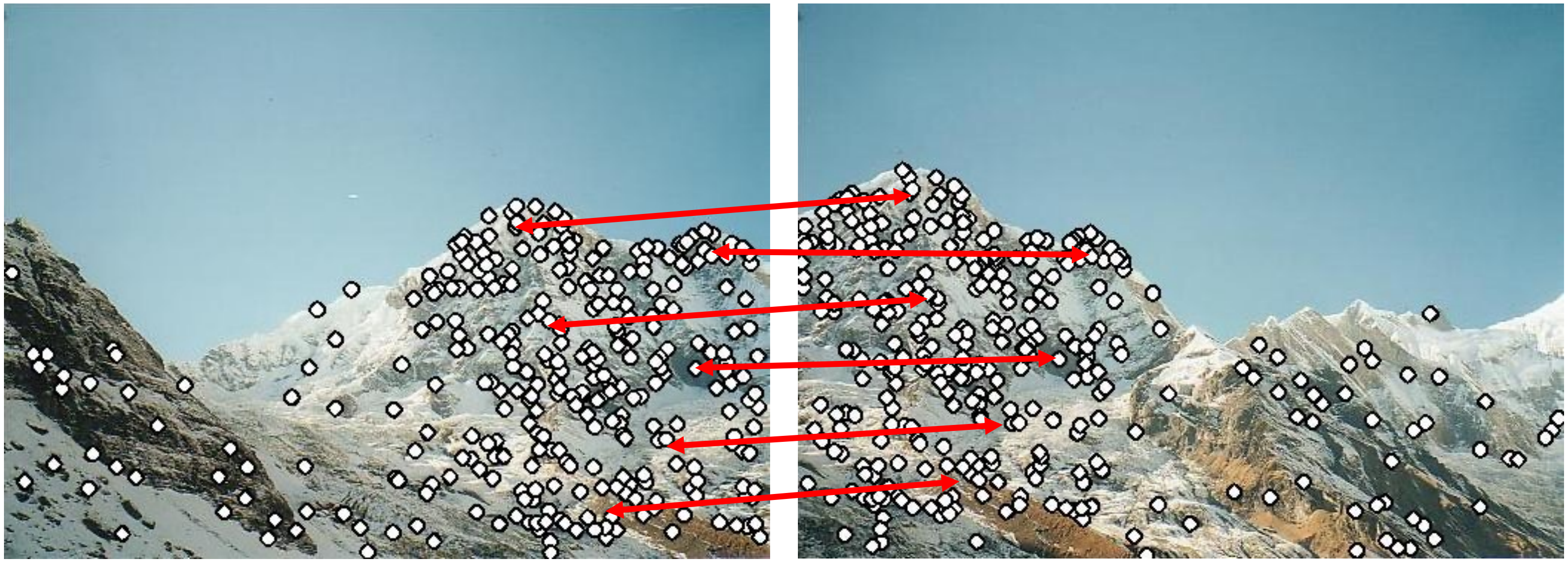
Credit: Matt Brown

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



Step 0: interest point detection
Step 1: extract features
Step 2: match features

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 0: interest point detection
Step 1: extract features
Step 2: match features
Step 3: align images

Image matching

by [Diva Sian]

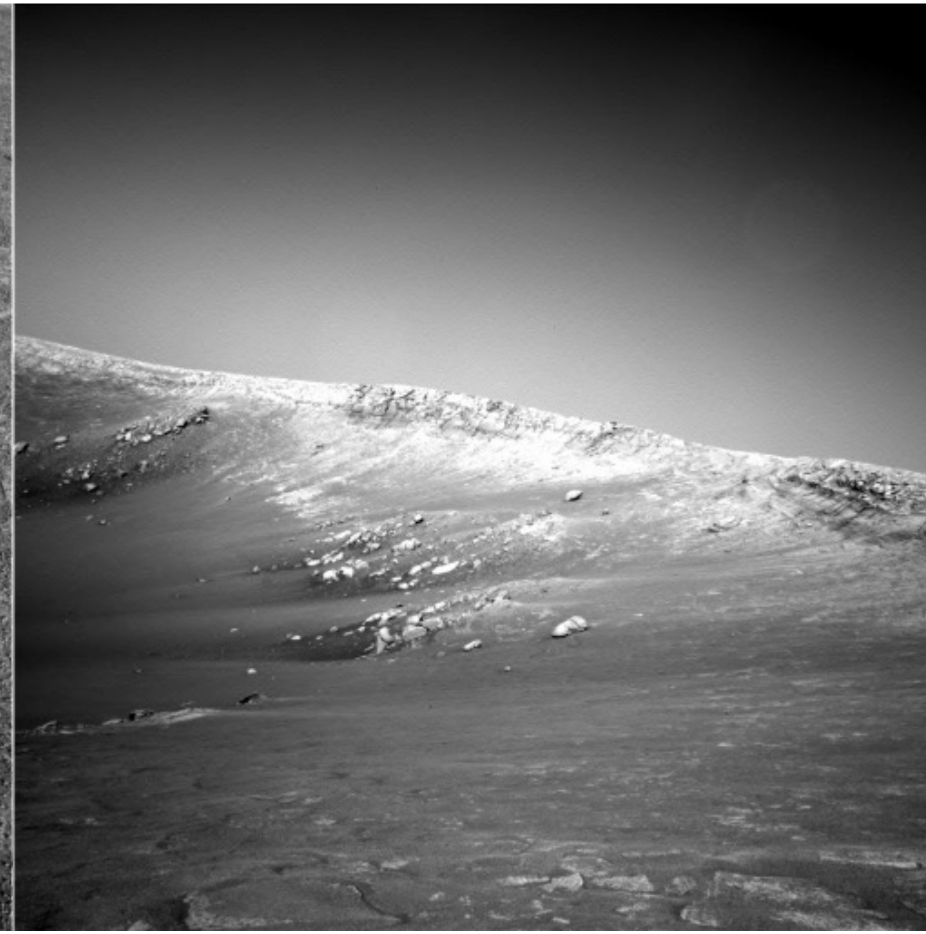
by [swashford]

Harder case

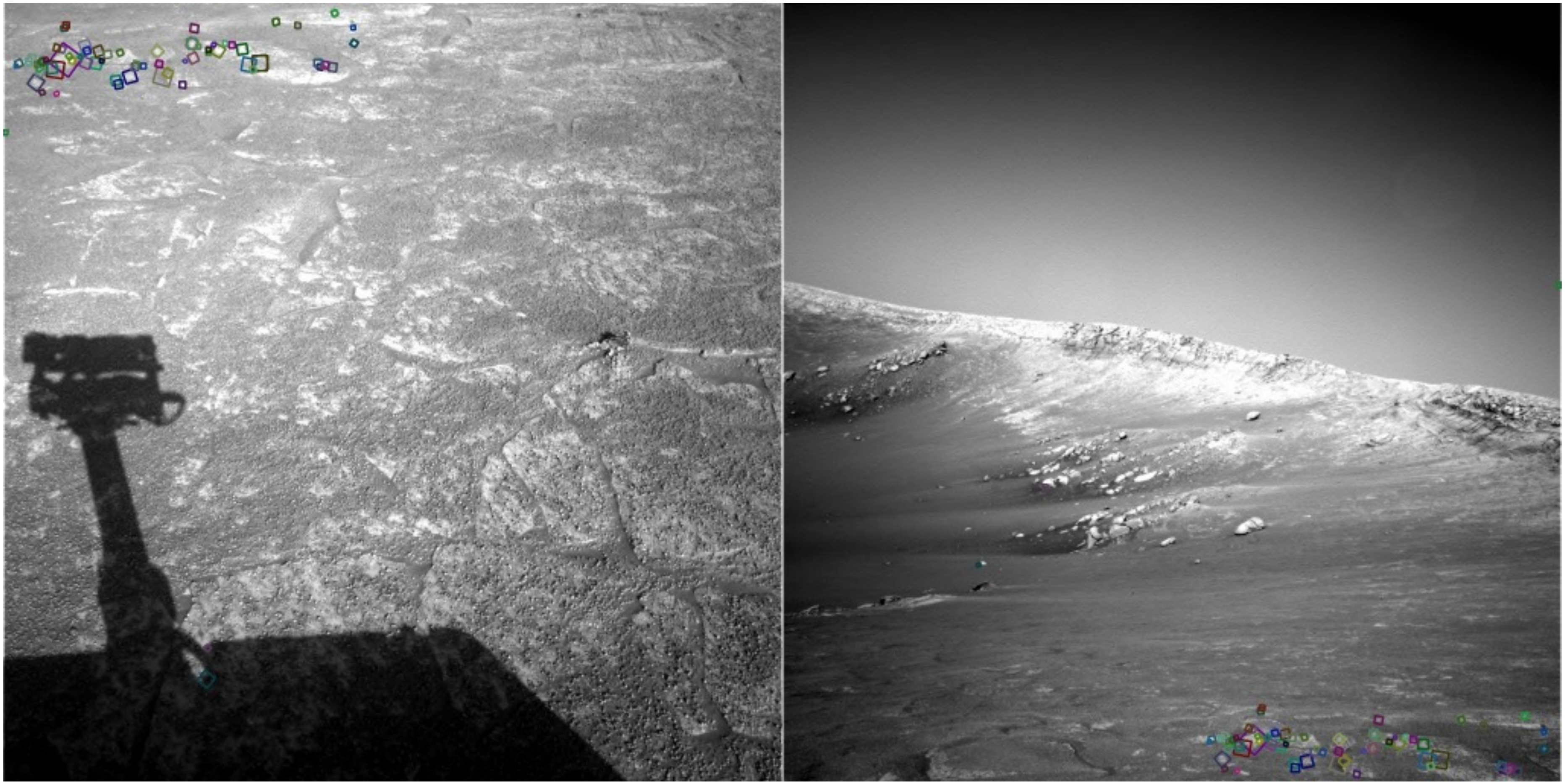
by Diva Sian

by scgbt

Harder still?

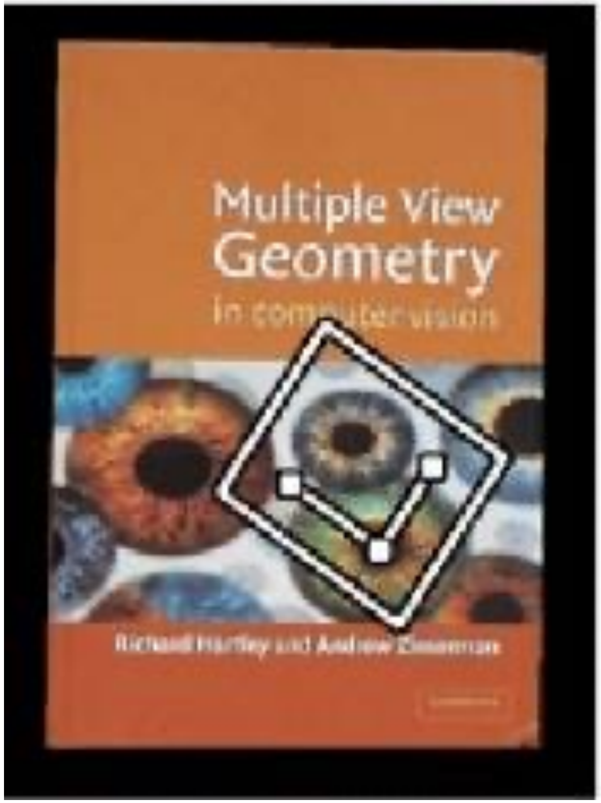



Answer below (look for tiny colored squares…)



NASA Mars Rover images
with SIFT feature matches

Feature Matching

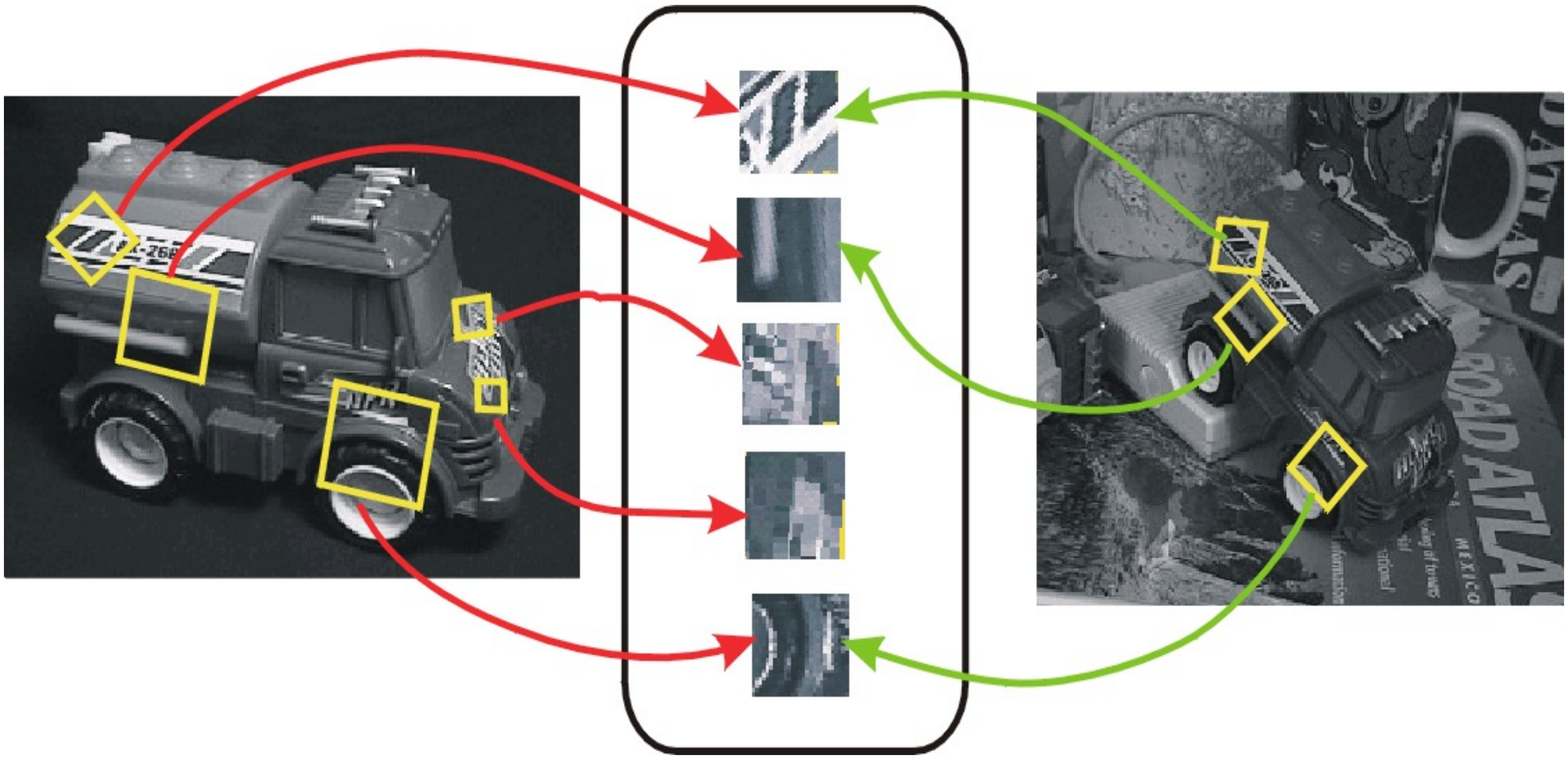


Feature Matching

Invariant local features

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, …



Feature Descriptors

Advantages of local features

Locality

– features are local, so robust to occlusion and clutter

Quantity

– hundreds or thousands in a single image

Distinctiveness:

– can differentiate a large database of objects

Efficiency

– real-time performance achievable

More motivation…

Feature points are used for:

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

Approach

Feature detection: find it

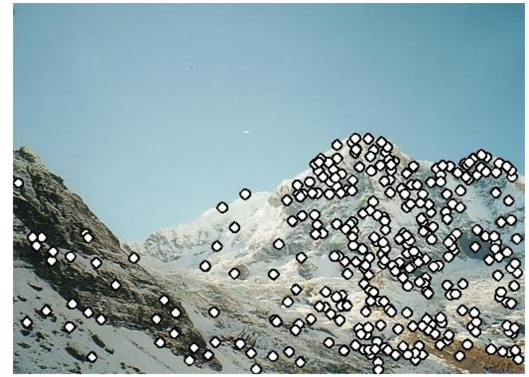
Feature descriptor: represent it

Feature matching: match it

Feature tracking: track it, when motion

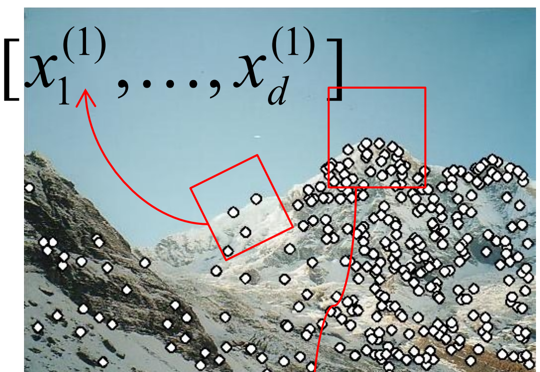
Local features: main components

1) **Detection:** Identify the interest points



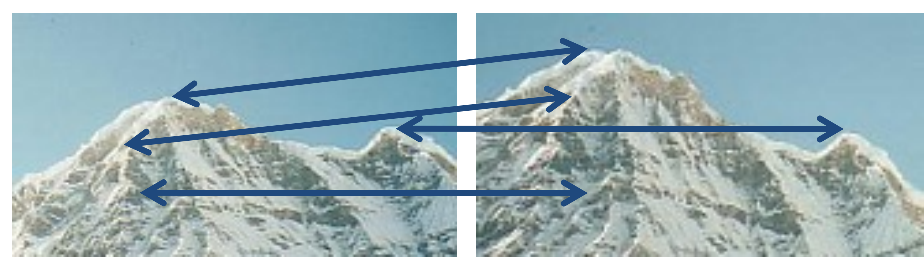
2) **Description:** Extract vector feature descriptor surrounding each interest point.

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



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

3) **Matching:** Determine correspondence between descriptors in two views



Kristen Grauman

What makes a good feature?

Want uniqueness

Look for image regions that are unusual

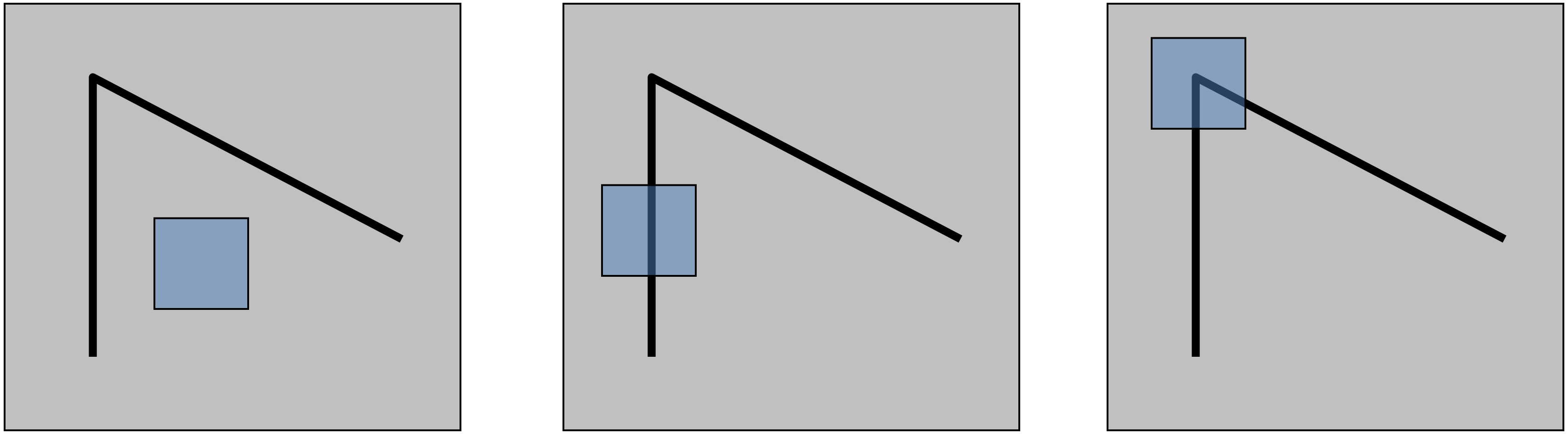
- Lead to distinct matches in other images
- The particular object in only a few number of images

How to define "unusual"?

Local measures of uniqueness

Suppose we only consider a small window of pixels

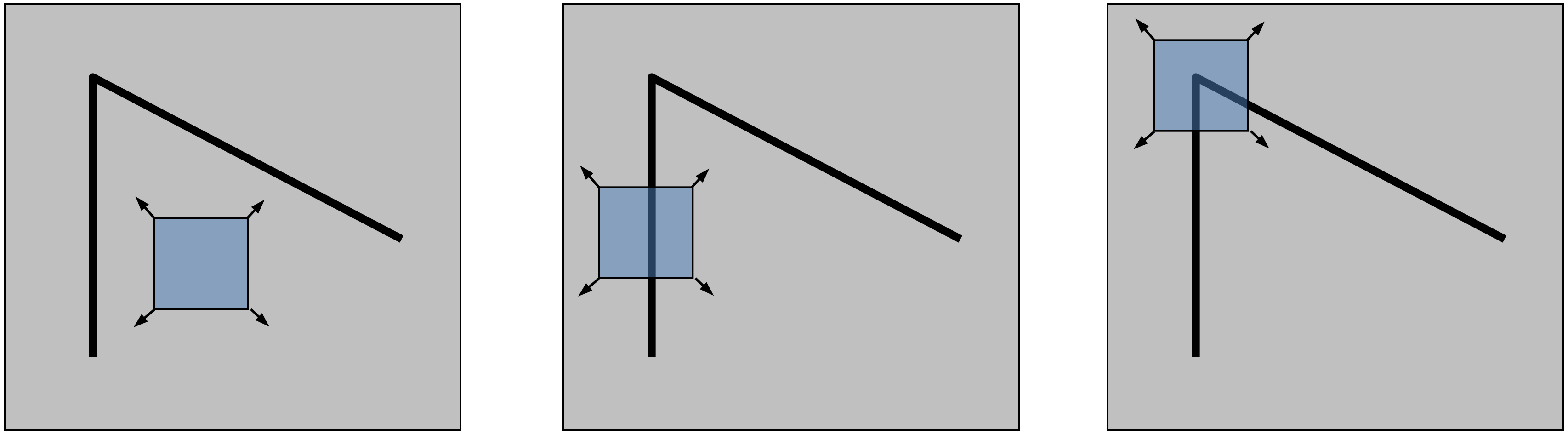
- What defines whether a feature is a good or bad candidate?



Credit: S. Seitz, D. Frolova, D. Simakov

Local measure of feature uniqueness

- How does the window change when you shift it?
- Shifting the window in any direction causes a big change



"flat" region:
no change in all directions

"edge":
no change along the edge direction

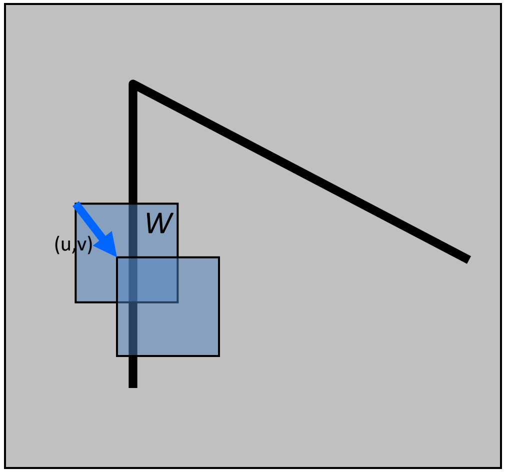
"corner":
significant change in all directions

Credit: S. Seitz, D. Frolova, D. Simakov

Harris corner 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" *E*(*u,v*):



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

- We are happy if this error is high
- Slow to compute exactly for each pixel and each offset (u,v)

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 approximation is good

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

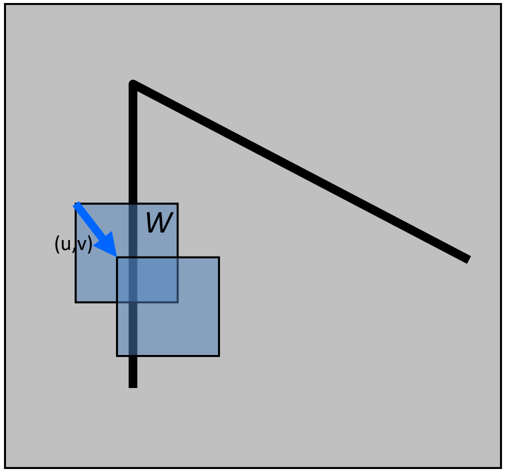
shorthand: $I_x = \frac{\partial I}{\partial x}$

Plugging this into the formula on the previous slide…

Corner detection: the math

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

- define an SSD "error" *E(u,v)*:

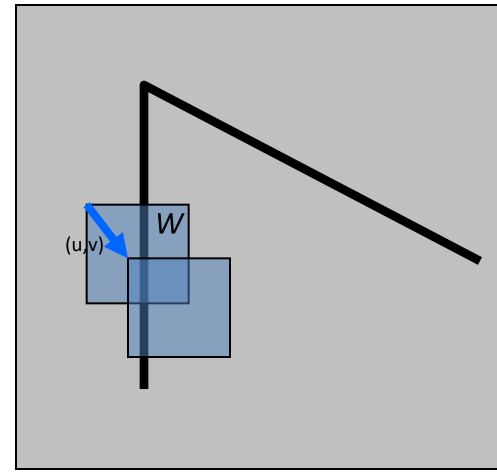


$$E(u,v) = \sum_{(x,y)\in W} [I(x+u,y+v) - I(x,y)]^2$$

$$\approx \sum_{(x,y)\in W} [I(x,y) + I_x u + I_y v - I(x,y)]^2$$

$$\approx \sum_{(x,y)\in W} [I_x u + I_y v]^2$$

Corner detection: the math



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

- define an SSD "error" *E(u,v)*:

$$E(u,v) \approx \sum_{(x,y)\in W} [I_x u + I_y v]^2$$

$$\approx Au^2 + 2Buv + Cv^2$$

$$A = \sum_{(x,y)\in W} I_x^2 \qquad B = \sum_{(x,y)\in W} I_x I_y \qquad C = \sum_{(x,y)\in W} I_y^2$$

Corner detection: the math

The surface $E(u,v)$ is locally approximated:

$$E(u,v) \approx Au^2 + 2Buv + Cv^2$$

$$\approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

$$B = \sum_{(x,y)\in W} I_x I_y$$

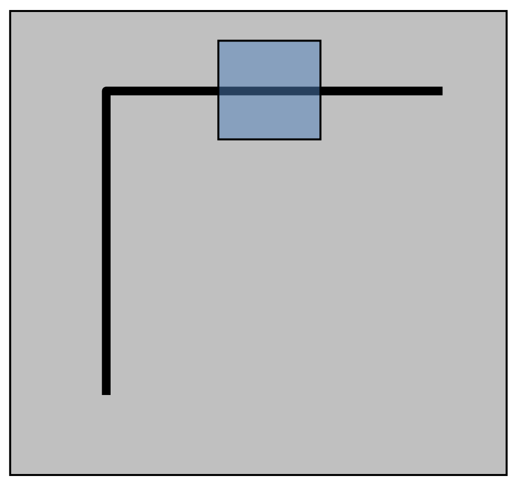
$$C = \sum_{(x,y)\in W} I_y^2$$

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

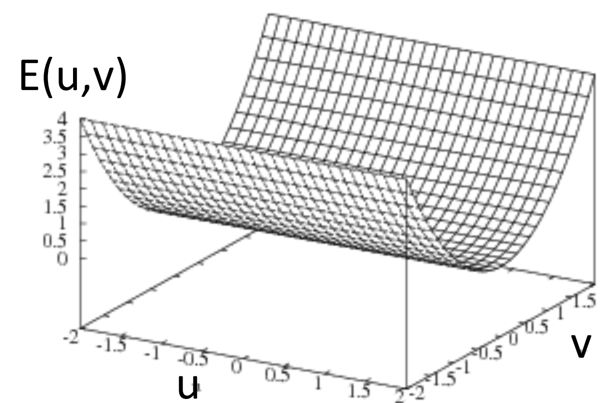
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$



Horizontal edge: $I_x = 0$

$$H = \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix}$$

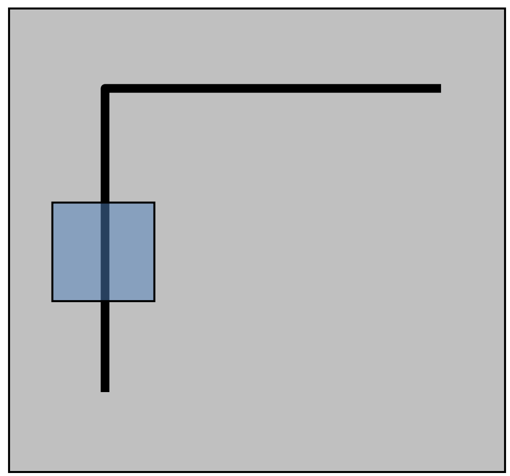


$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

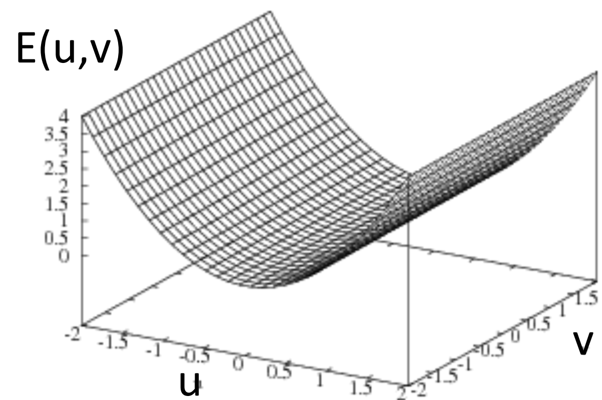
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$



Vertical edge: $I_y = 0$

$$H = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}$$



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:

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

Example

$$P^T = A = \begin{bmatrix} 0.1 & 0.3 \\ 0.9 & 0.7 \end{bmatrix}$$

Solve this equation:
X is eigenvector
λ contains eigenvalues

$$AX = \lambda X$$
$$(A - \lambda I)X = 0$$

By:

$$|A - \lambda I| = 0$$
$$\begin{vmatrix} 0.1 - \lambda & 0.3 \\ 0.9 & 0.7 - \lambda \end{vmatrix} = 0$$
$$(0.1 - \lambda)(0.7 - \lambda) - 0.9 \times 0.3 = 0$$
$$\lambda^2 - 0.8\lambda - 0.2 = 0$$
$$\lambda = 1 \; or \; \lambda = -0.2$$

Quiz

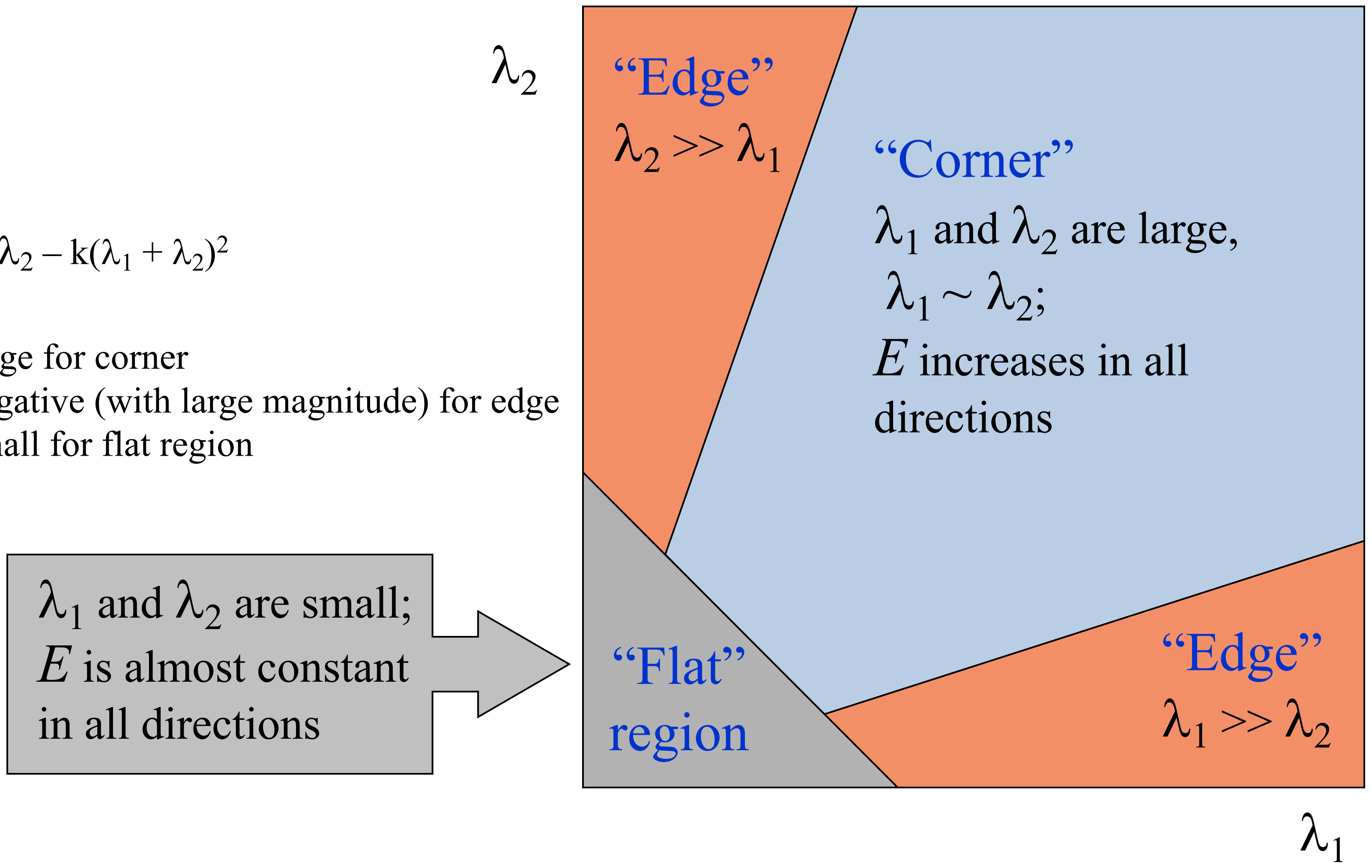
- Which of the following can be called as the "corners"
  - Pixels with large gradient magnitude
  - Pixels at the end of a straight line
  - Pixels at the junction of horizontal and vertical lines

Interpreting the eigenvalues

Classification of image points using eigenvalues of *H*:

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

R is large for corner
R is negative (with large magnitude) for edge
R is small for flat region



Other Versions of The Harris operator

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

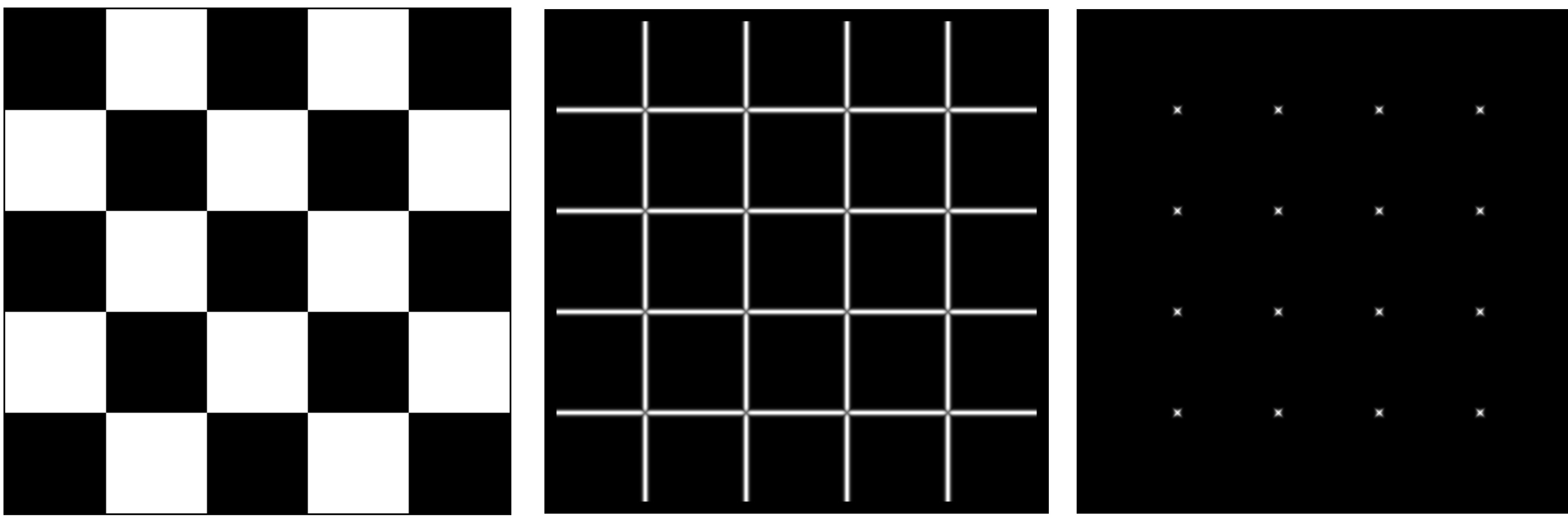
$$= \frac{determinant(H)}{trace(H)}$$

λ_{min} is a variant of the "Harris operator" for feature detection

Corner detection summary

Here's what you do

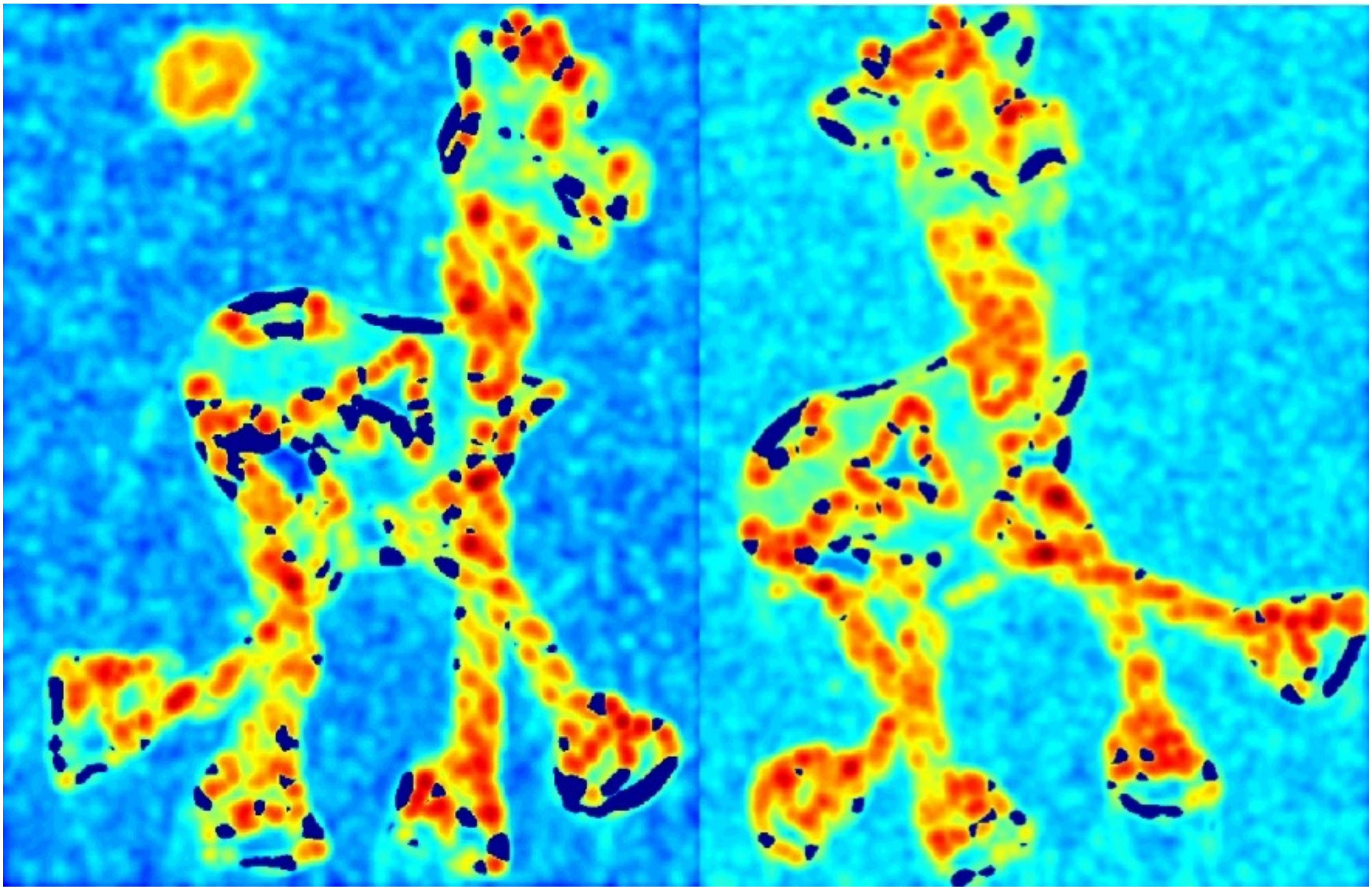
- Compute the gradient at each point in the image
- Create the *H* matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_{min} > threshold)
- Choose those points where λ_{min} is a local maximum as features



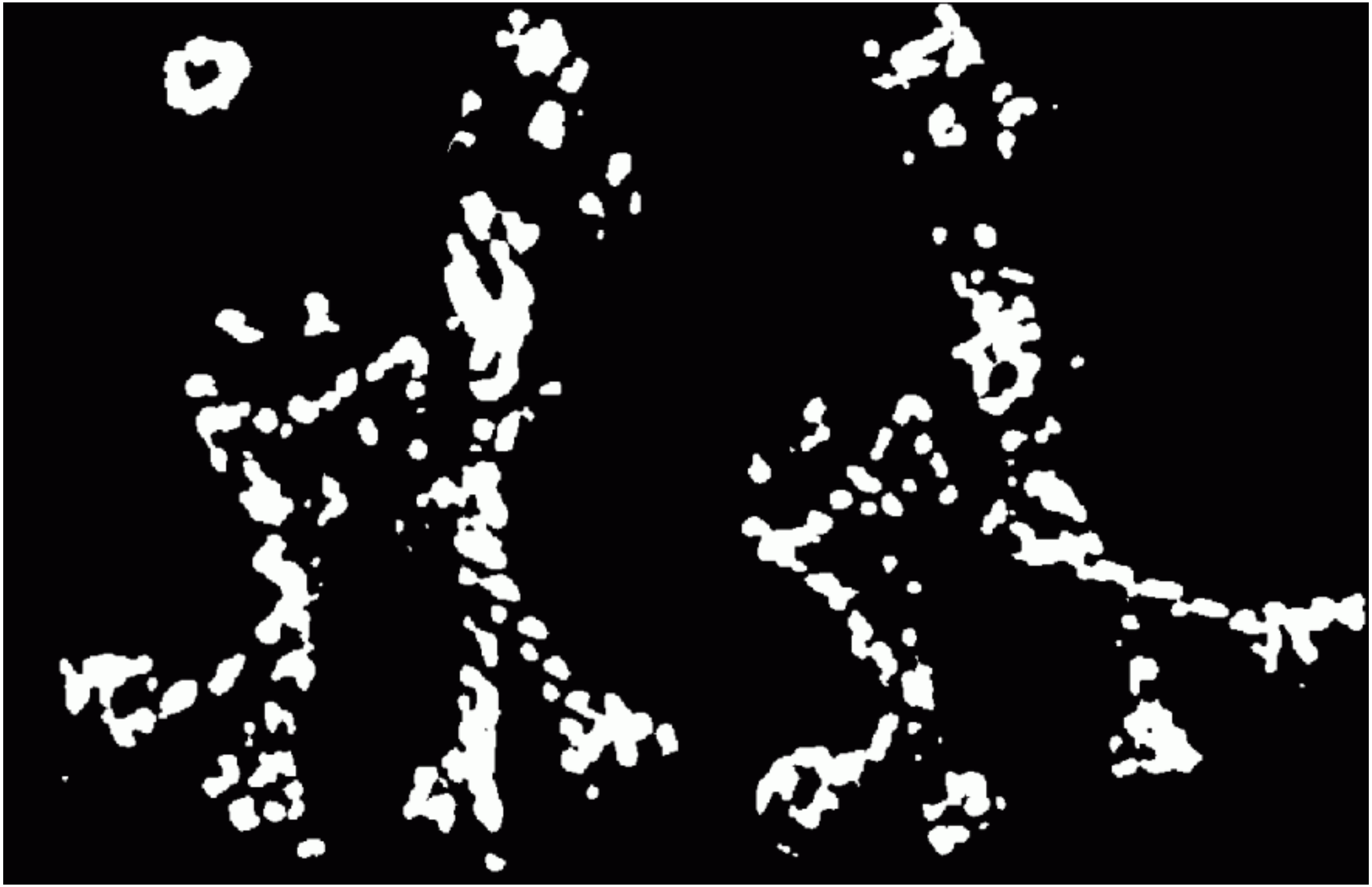
I $\lambda_{\max}$ $\lambda_{\min}$

Harris detector example

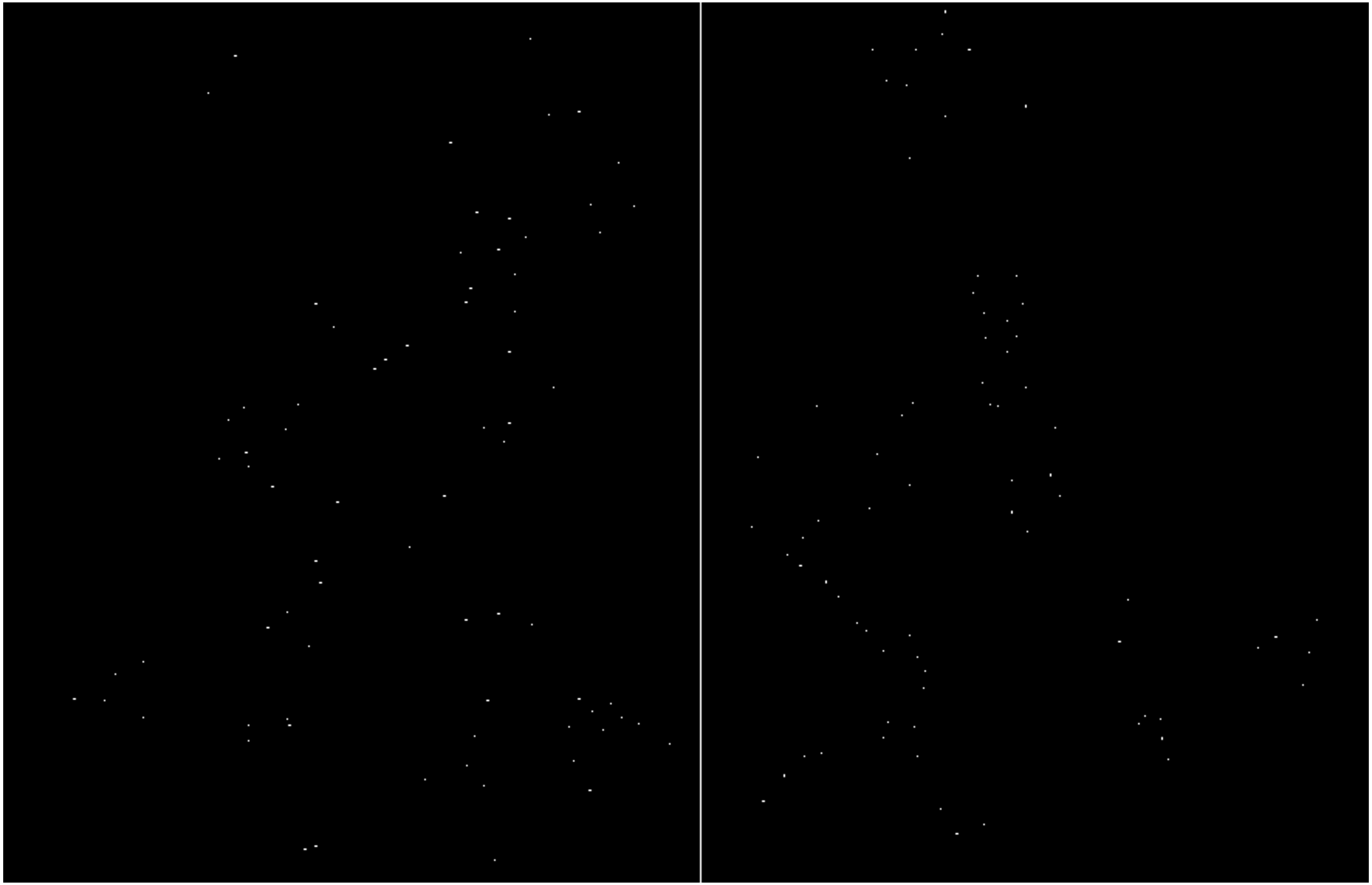
f value (red high, blue low)



Threshold (f > value)



Find local maxima of f



Harris features (in red)