ME/CS 132: Introduction to Vision-based Robot Navigation

Feature detection and matching

Roland Brockers
brockers@jpl.nasa.gov

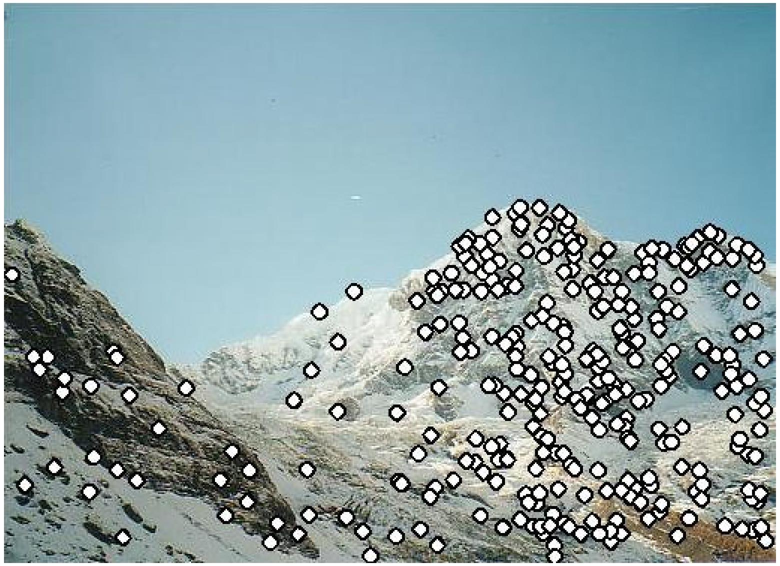
Feature detection and matching
Motivation

- How do we built a panorama ?

- Align (match) images
- Global methods sensitive to occlusion, lightning, perspective distortion

Image matching

- Find “good” local feature points in both images



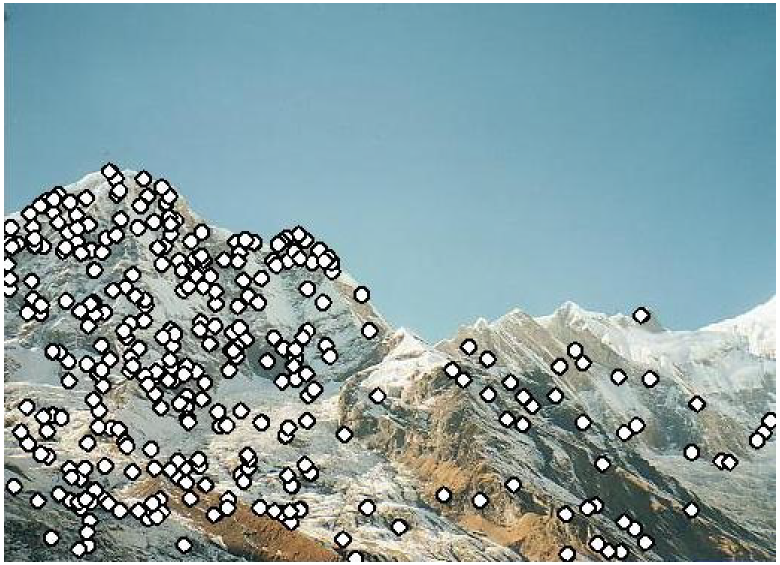


Image matching

- Find “good” local feature points in both images
- Find corresponding pairs

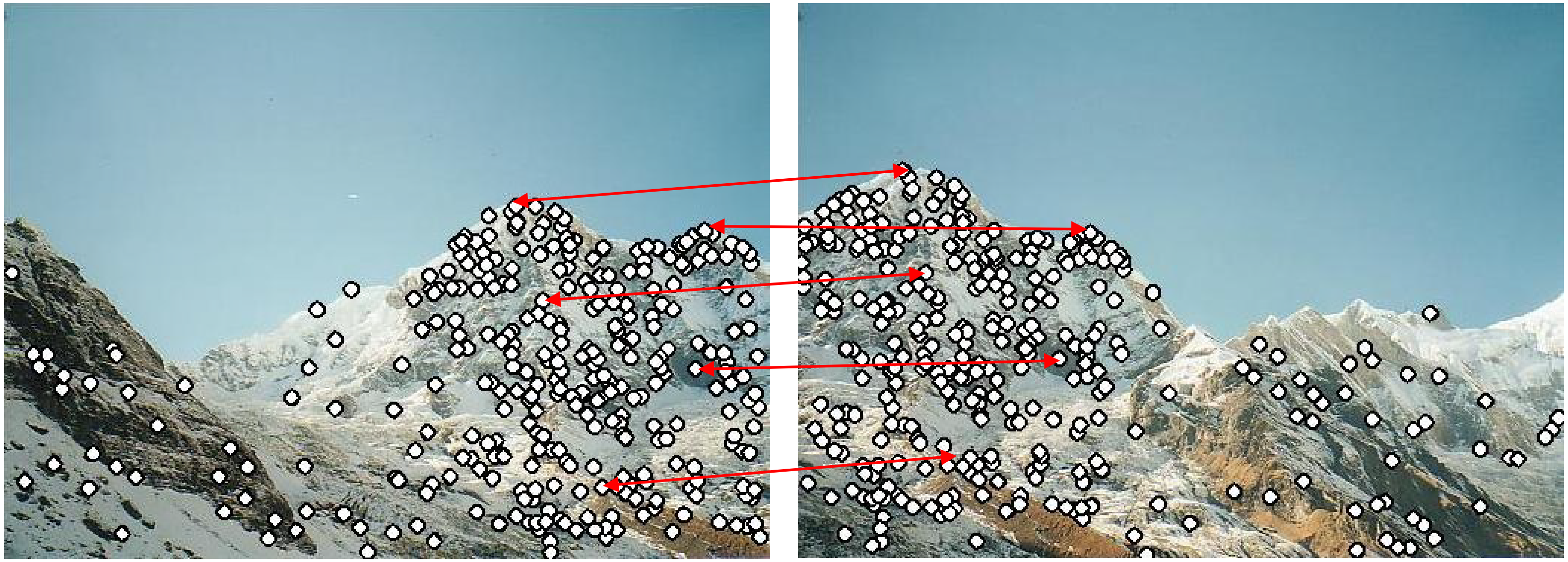


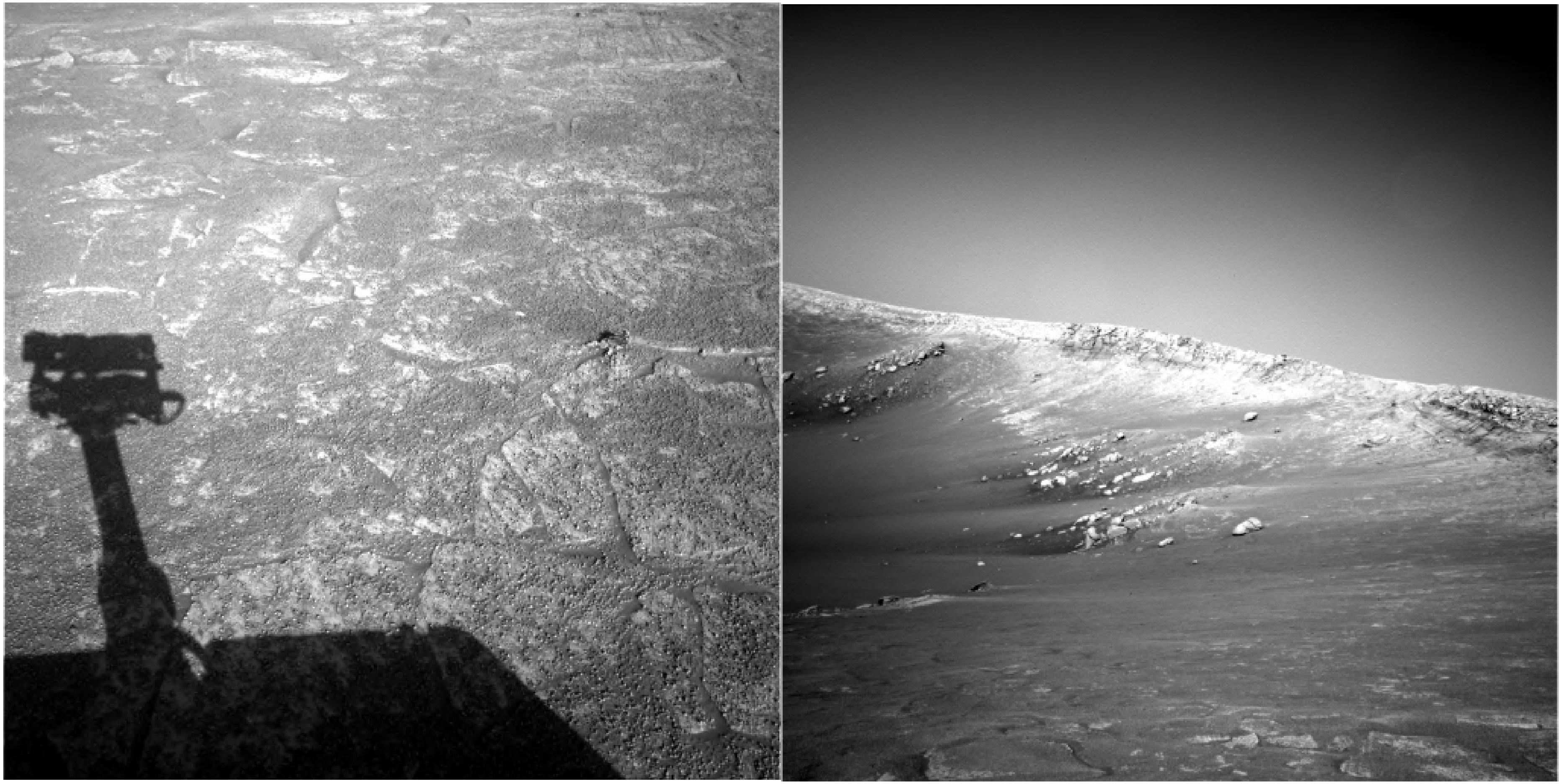
Image matching

- Find "good" local feature points in both images
- Find corresponding pairs
- Align images using corresponding pairs to correct for image distortion

Image matching

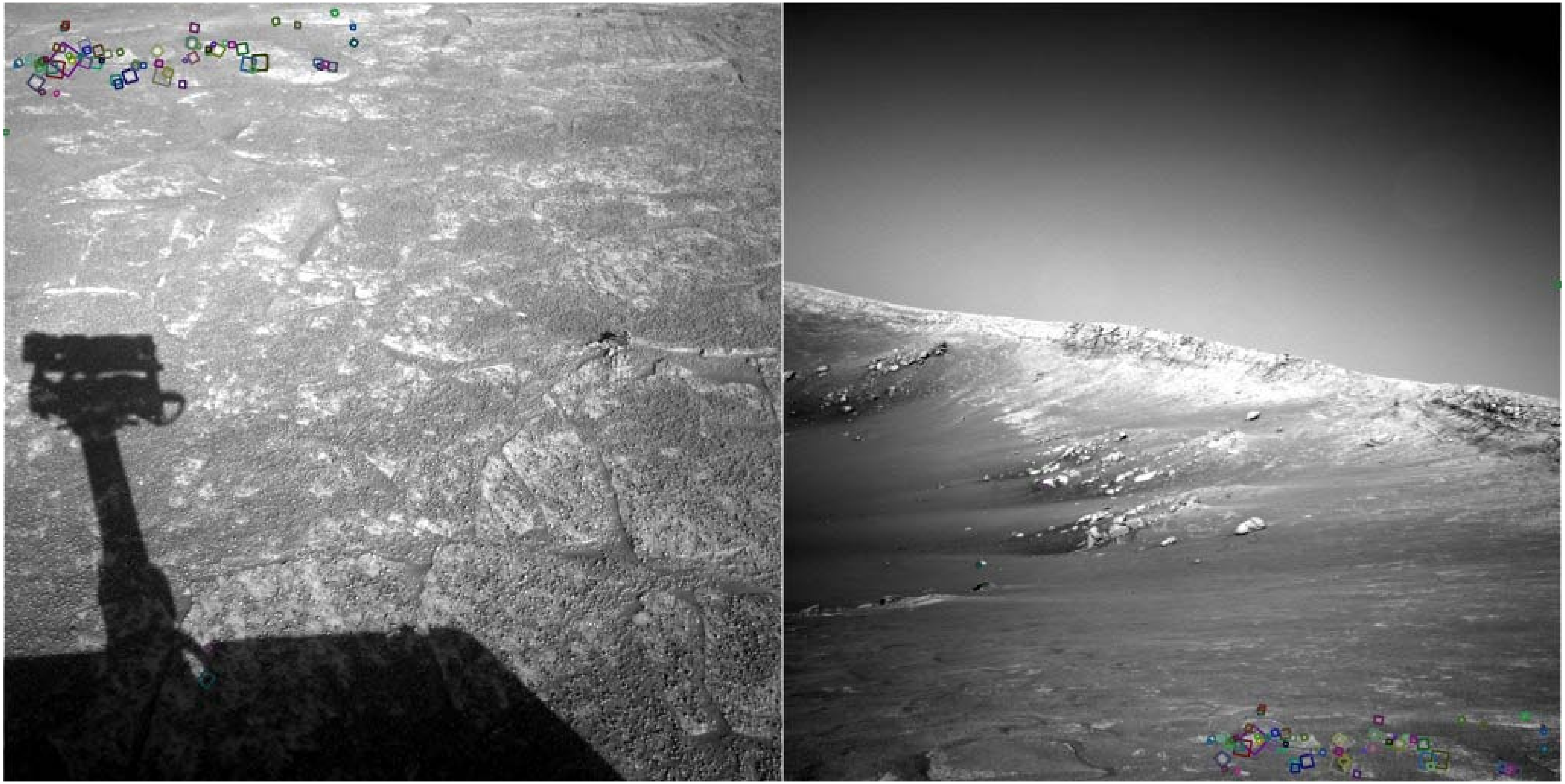
Image matching

Image matching



NASA Mars Rover images

Image matching

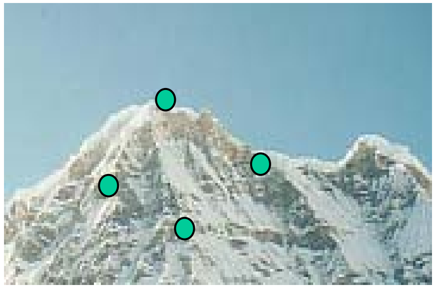


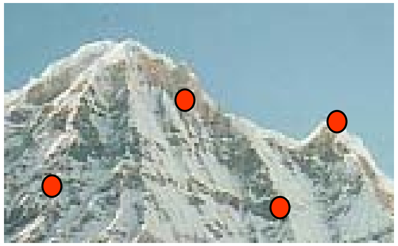
NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

Matching with features

Problem 1

- Detect the *same* point *independently* in both images





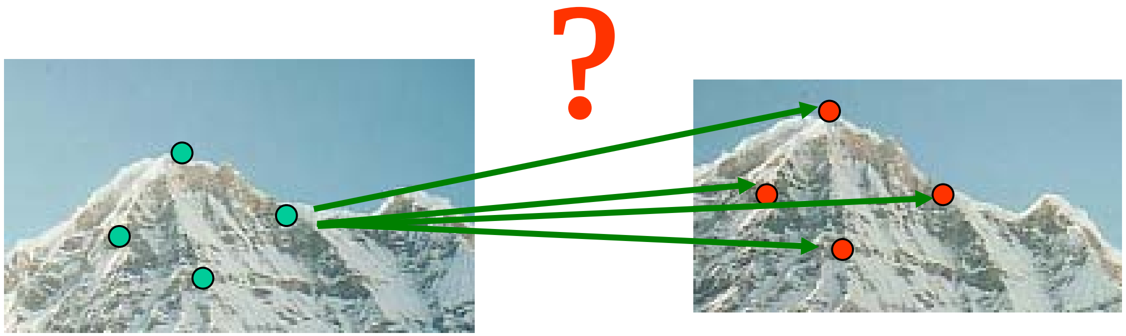
No matching possible

We need a repeatable detector !

Matching with features

Problem 2

- For each detected point, find the *correct* corresponding point



We need reliable and distinctive descriptor!

More motivation

Feature points are also used for:

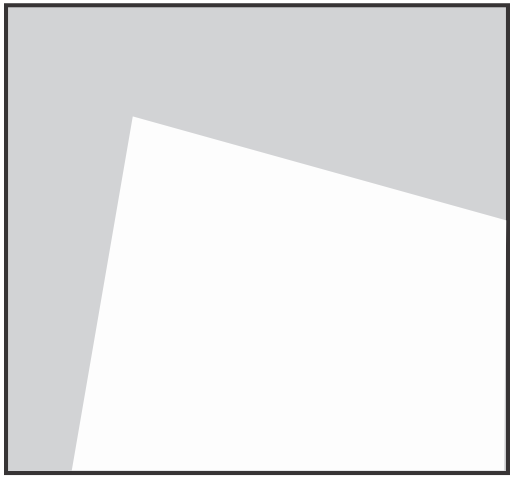
- Image alignment (homography, fundamental matrix)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- …

How to find good interest points

Image points that are locally unique!

E.g. consider a small local window

- What defines weather an image point is a good or bad candidate?

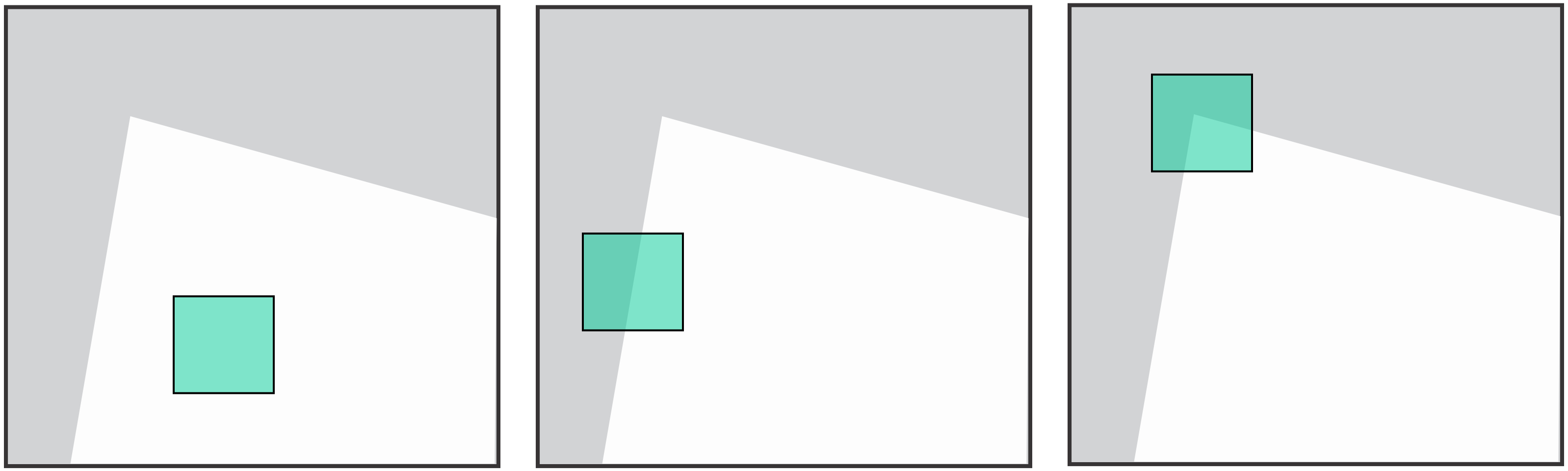


How to find good interest points

Image points that are locally unique!

E.g. consider a small local window

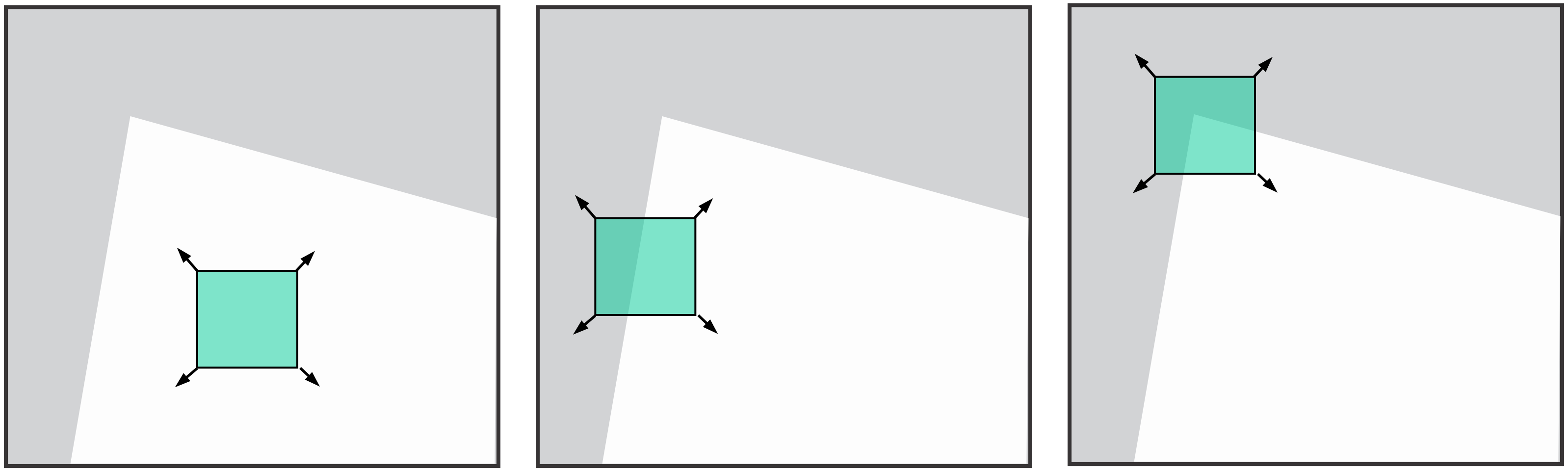
- What defines weather an image point is a good or bad candidate?



How to find good interest points

Local measure of uniqueness:

- How does the window change when you shift it?



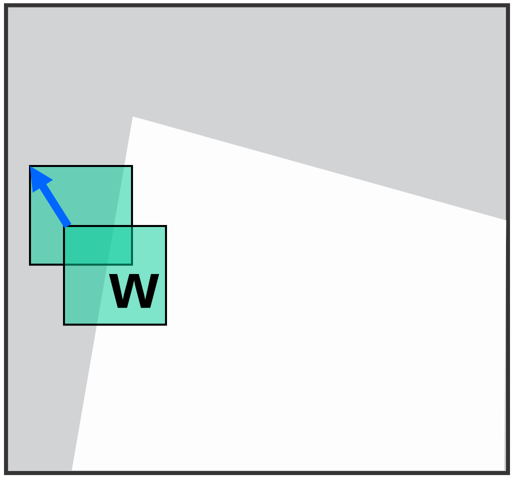
"flat" region:
no change in all directions

"edge":
no change along the edge direction

"corner":
significant change in all directions

Feature detection, a simple example

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



- How do the pixels in **w** change?
- Compare each pixel before and after movement by summing up the squared differences (SSD)
- Difference defines an "error" function E(u,v)

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

Small motion assumption

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

Taylor series expansion

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

$$I(x+u, y+v) \approx I(x,y) + I_x u + I_y v \qquad I_x = \partial I / \partial x, I_y = \partial I / \partial y$$

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

$$= \sum_{x,y} w(x,y)[uI_x + vI_y]^2$$

$$\Rightarrow E(u,v) = \sum_{x,y} w(x,y)\begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x I_x & I_x I_y \\ I_y I_x & I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Feature detection, a simple example

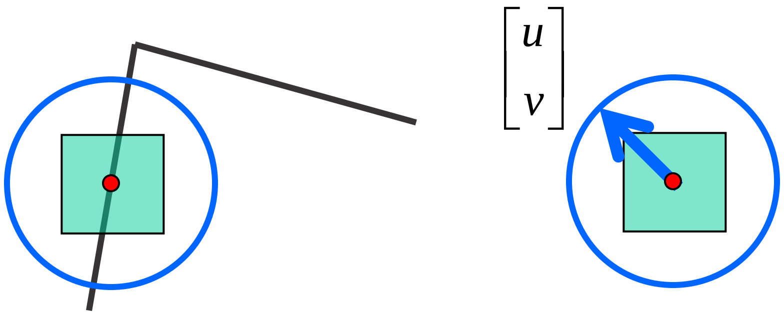
Expanding $I(x,y)$ in a Taylor series expansion, we have, for small shifts [*u,v*], a *bilinear* approximation:

$$E(u,v) \cong \begin{bmatrix} u & v \end{bmatrix} H \begin{bmatrix} u \\ v \end{bmatrix}$$

where *H* is a 2×2 matrix computed from image derivatives:

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

Feature detection, a simple example



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*

Quick eigenvalue/eigenvector review

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

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$$

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

- The eigenvalues are found by solving: $\det(\mathbf{A} - \lambda \mathbf{I}) = 0$
- In our case, **A** = **H** is a 2x2 matrix, so we have

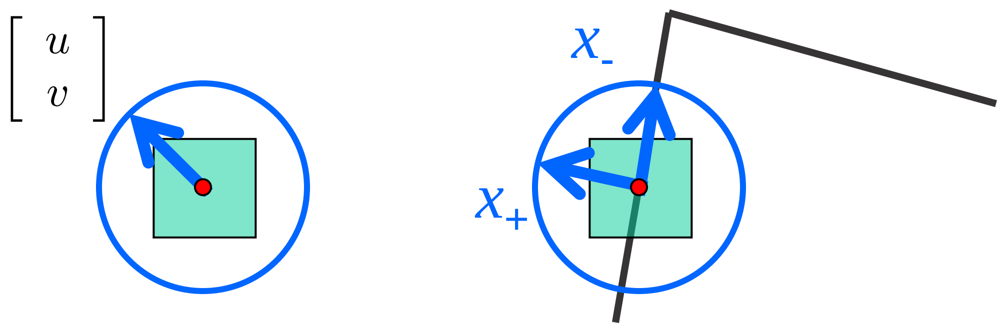
$$\det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

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

Once we know λ, **x** is defined by solving $\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$

Feature detection, a simple example

Eigenvalues and eigenvectors of **H**



Define shifts with smallest and largest change

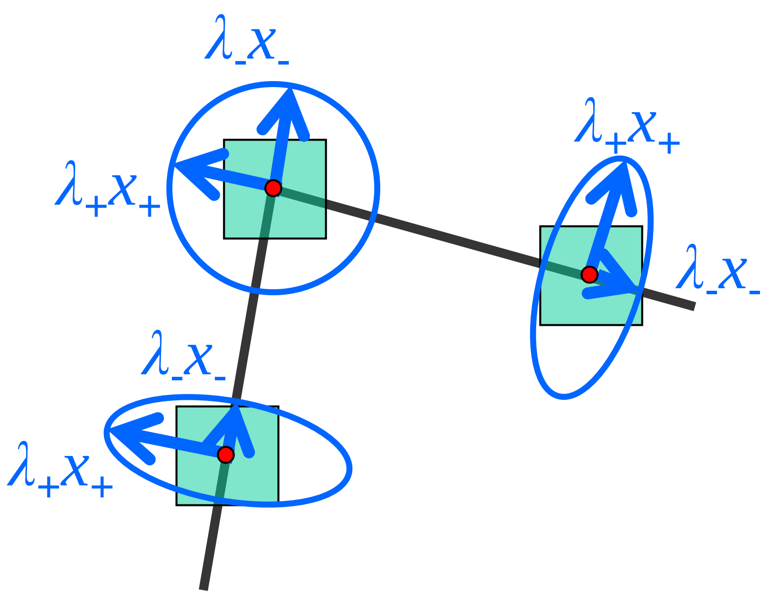
- x_+ = direction of **largest** increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of **smallest** increase in E.
- λ- = amount of increase in direction x_-

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

Feature detection, a simple example

- How are λ_+, x_+, λ_-, and x_+ relevant for feature detection?

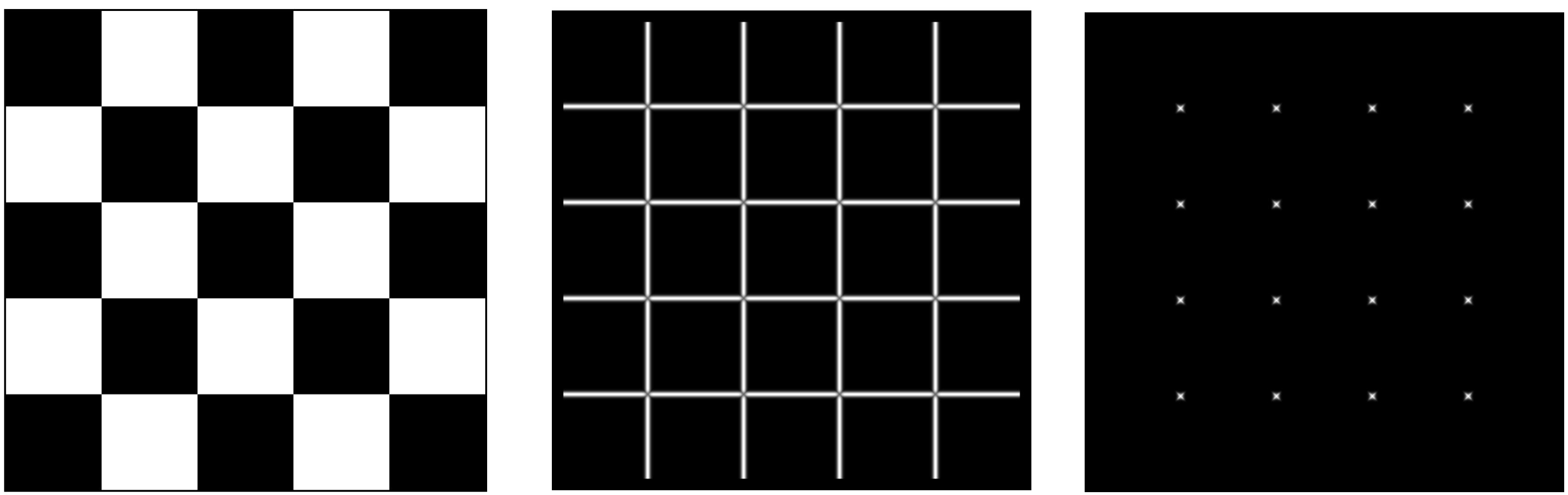
- Want *E(u,v)* to be ***large*** for small shifts in ***all*** directions
  - the *minimum* of *E(u,v)* should be large, over all unit vectors [u v]
  - this minimum is given by the smaller eigenvalue (λ_-) of *H*



Feature detection, a simple example

- How are λ_+, x_+, λ_-, and x_+ relevant for feature detection?

- 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_-

Feature detection summary

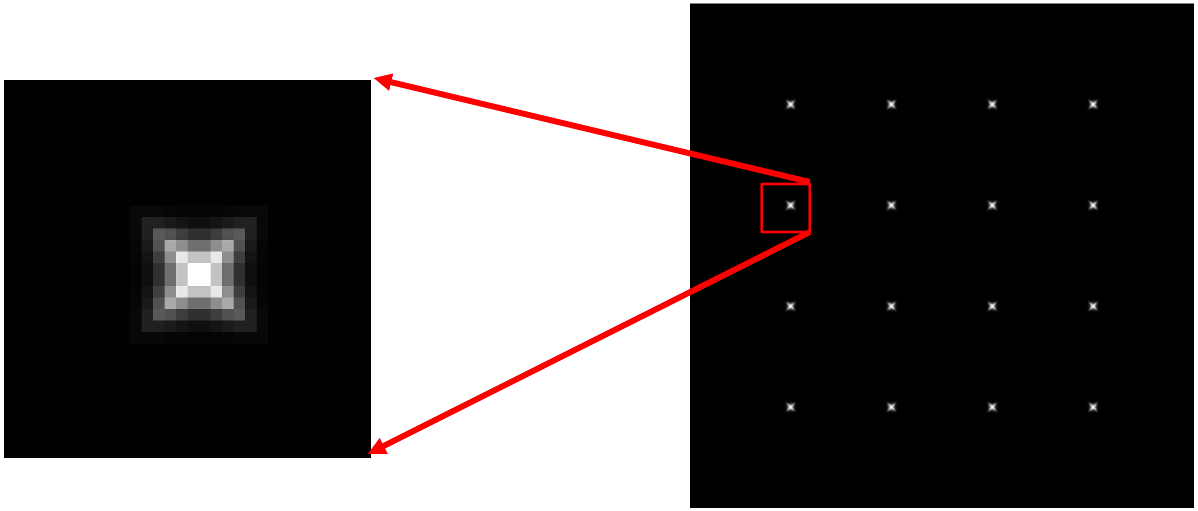
Basic feature detection

- 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

Feature detection summary

Basic feature detection

- 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



λ_-

Harris corner detector

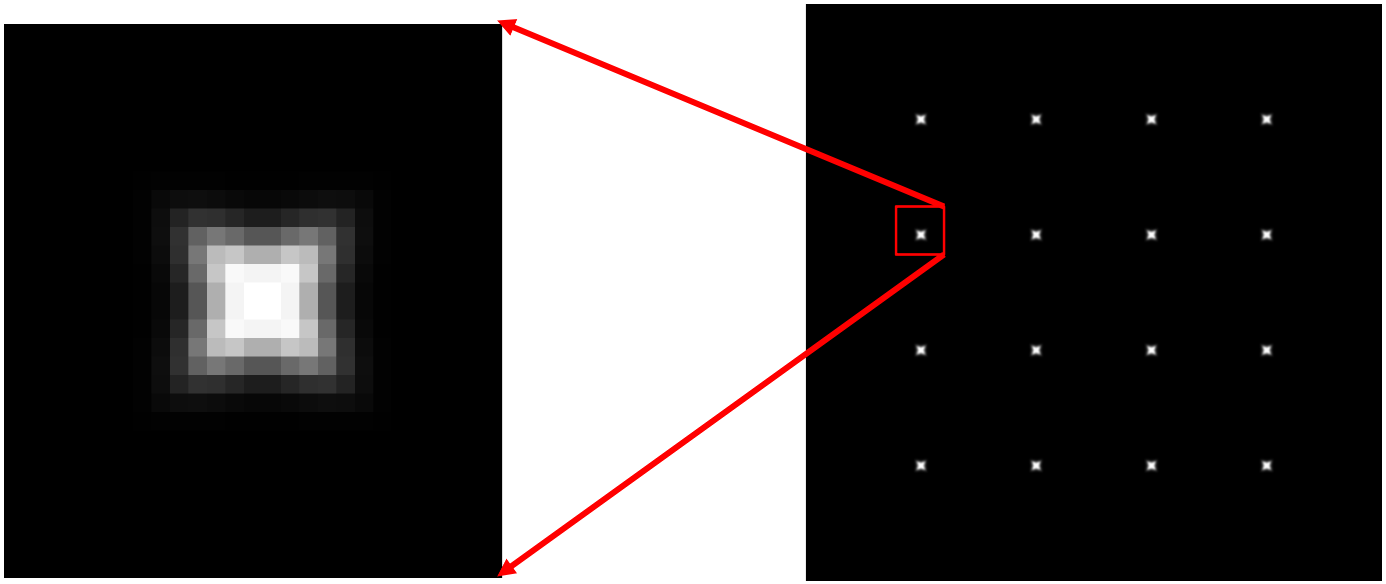
λ_{-} is a variant of the "Harris operator" for feature detection

$$f = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 = determinant(H) - k(trace(H))^2$$

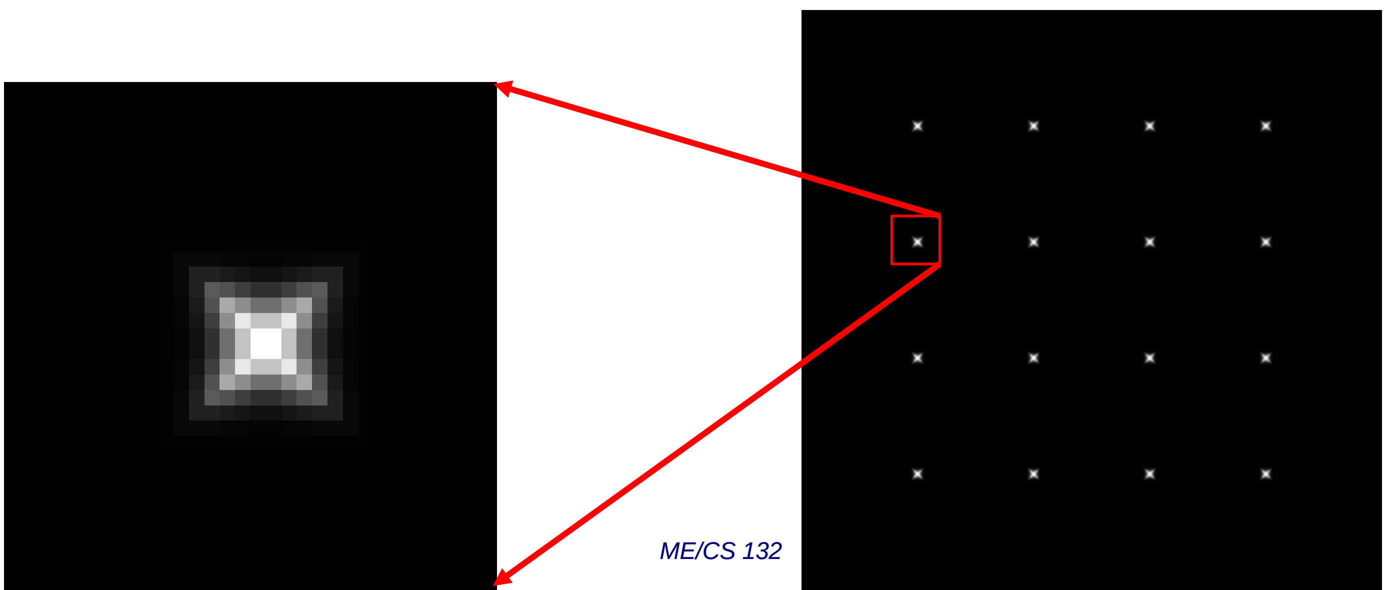
- The *trace* is the sum of the diagonals, i.e., *trace(H)* = $h_{11} + h_{22}$
- k is a empirical constant, k = 0.04 … 0.06
- Very similar to λ_{-} but less expensive (no square root)
- Called the "Harris Corner Detector" or "Harris Operator"
- Lots of other detectors, this is one of the most popular

C.Harris, M.Stephens. "A Combined Corner and Edge Detector". 1988

Harris corner detector



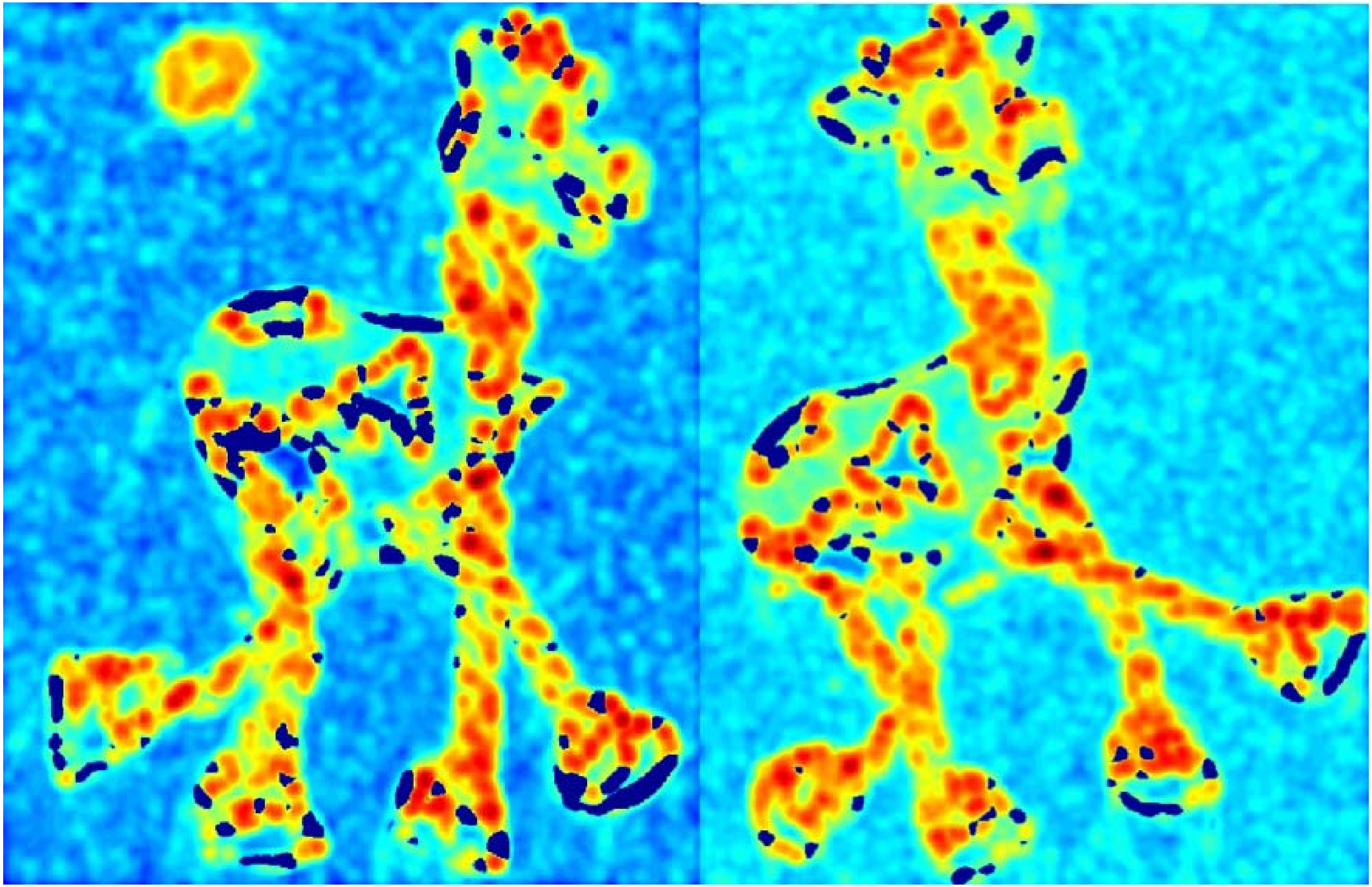
Harris operator



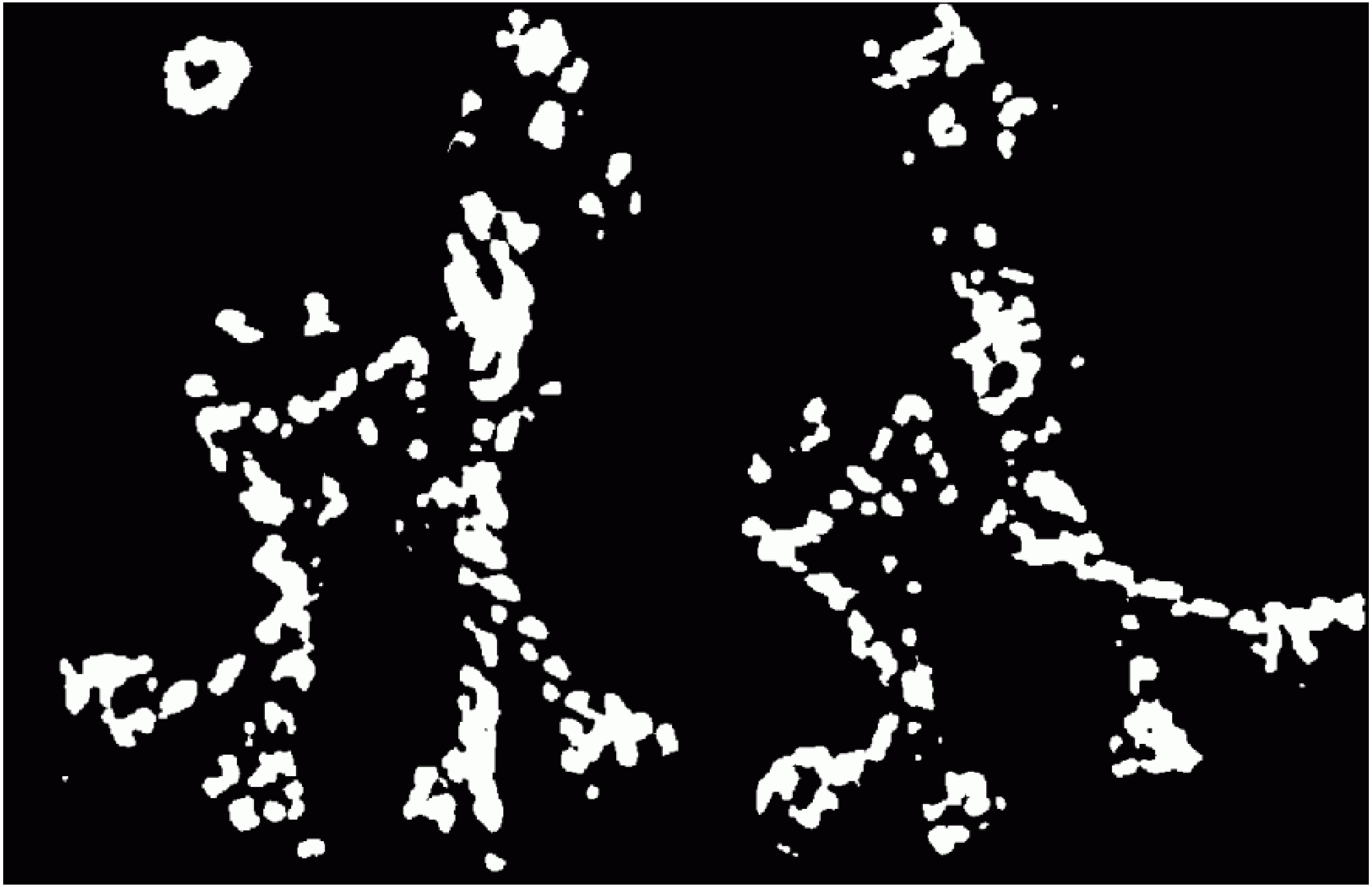
λ_-

Harris corner example

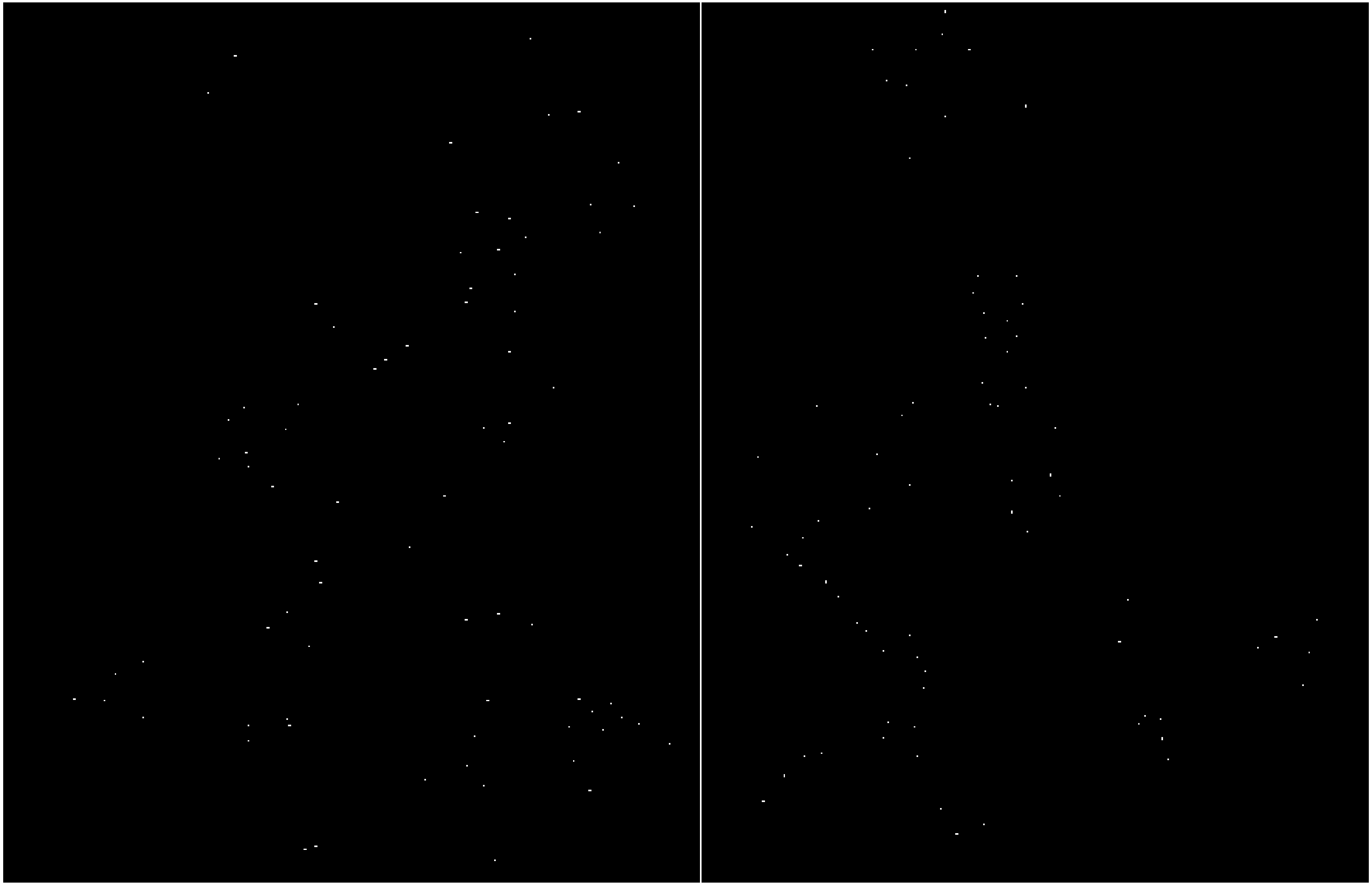
f value (red=high, blue=low)



Threshold (*f* > value)



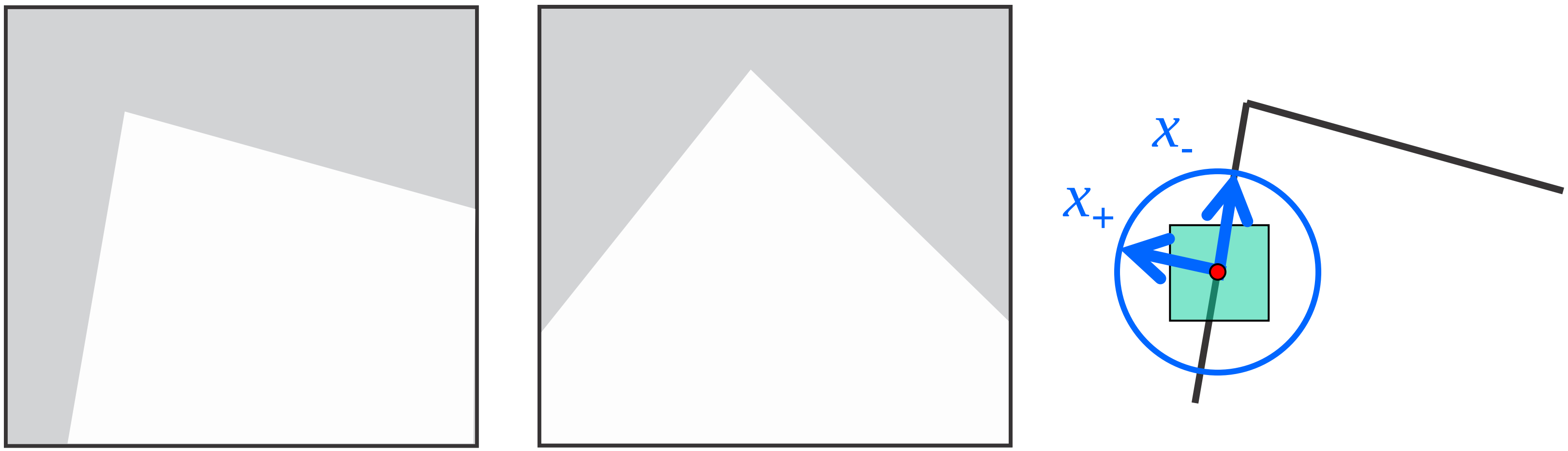
Find local maxima of *f*



Harris interest points (in red)

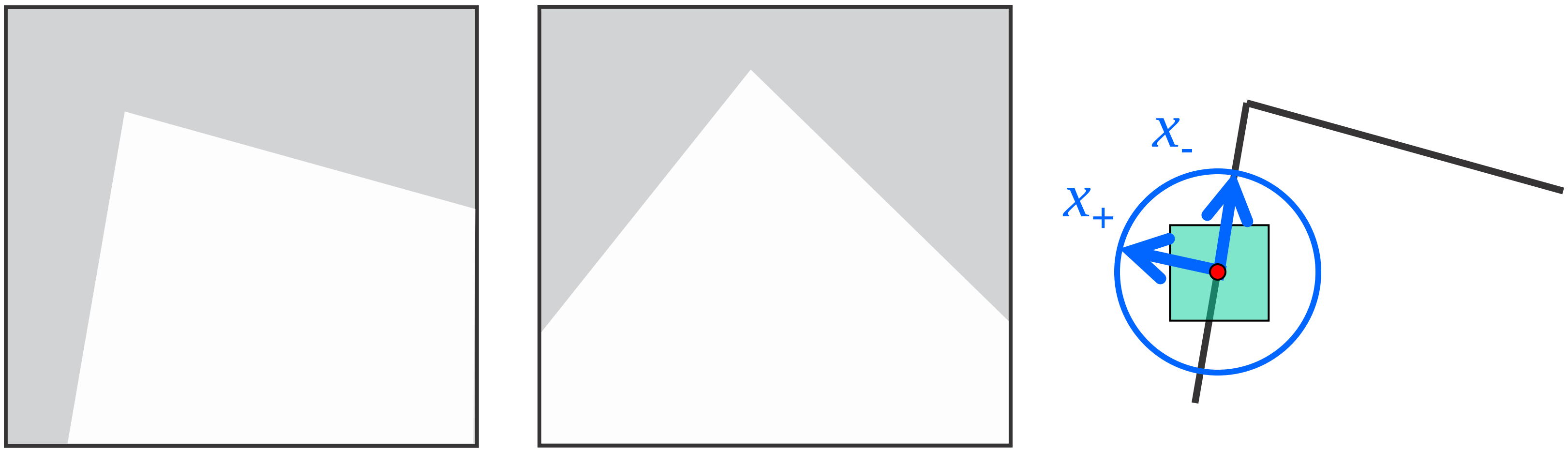
Harries Detector: Properties

Suppose you rotate the image, will you still pick up the same features?



Harries Detector: Properties

Suppose you rotate the image, will you still pick up the same features?



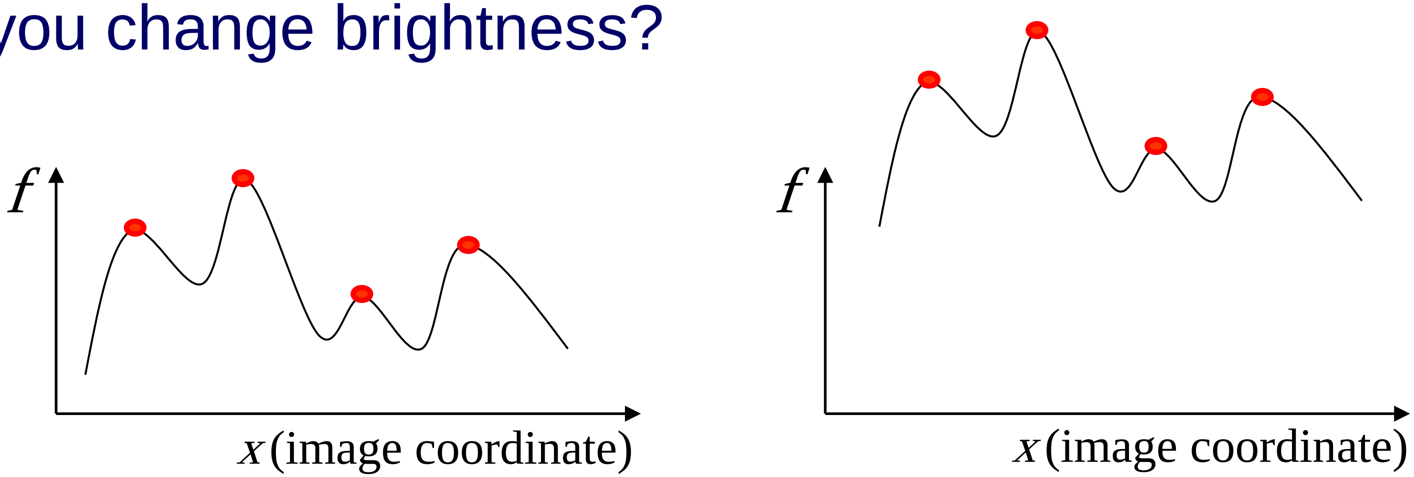
Rotation Invariance:

- Eigenvectors rotate, but eigenvalues remain the same

Corner response ***f*** is invariant to image rotation

Harries Detector: Properties

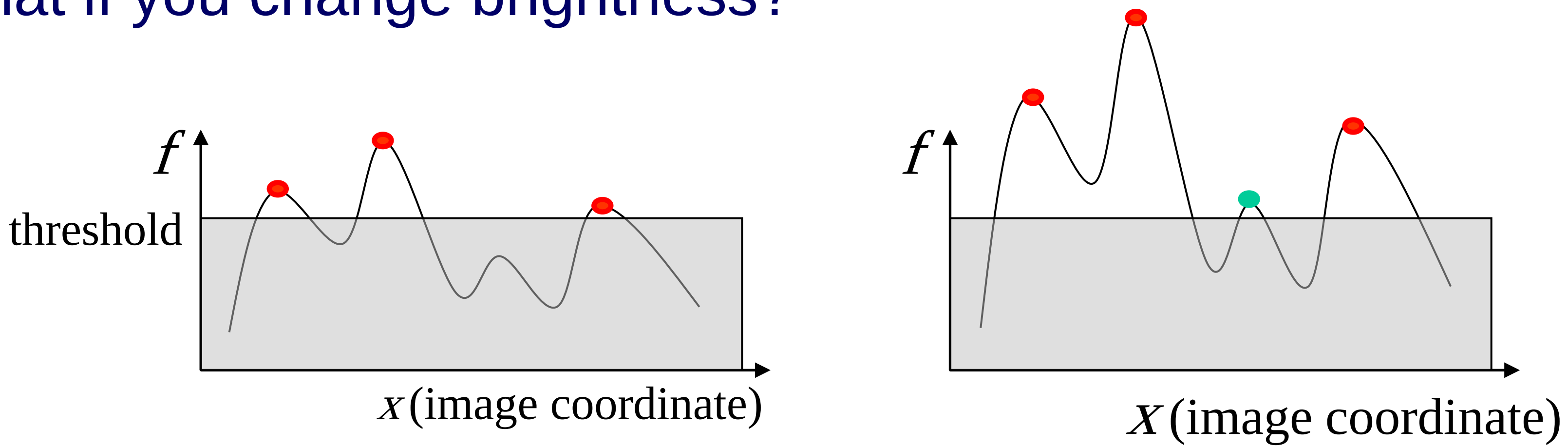
What if you change brightness?



- Invariant to intensity shifts: $I \rightarrow I + b$

Harries Detector: Properties

What if you change brightness?

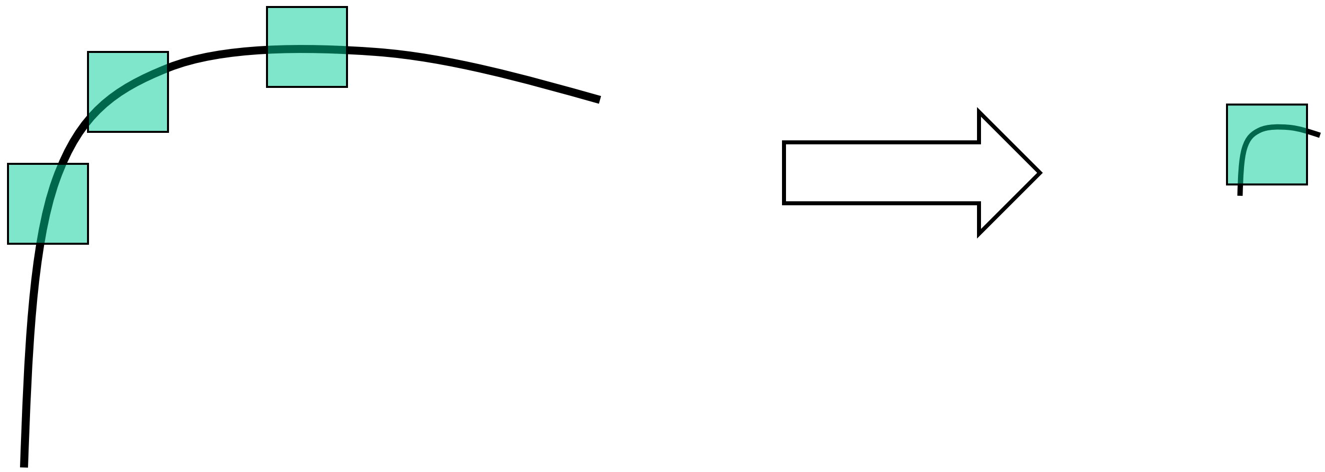


- Invariant to intensity shifts: $I \rightarrow I + b$
- Sensitive to intensity scale change $I \rightarrow aI$ because of fixed threshold on local maxima

Corner response ***f*** is only partial invariant to brightness changes

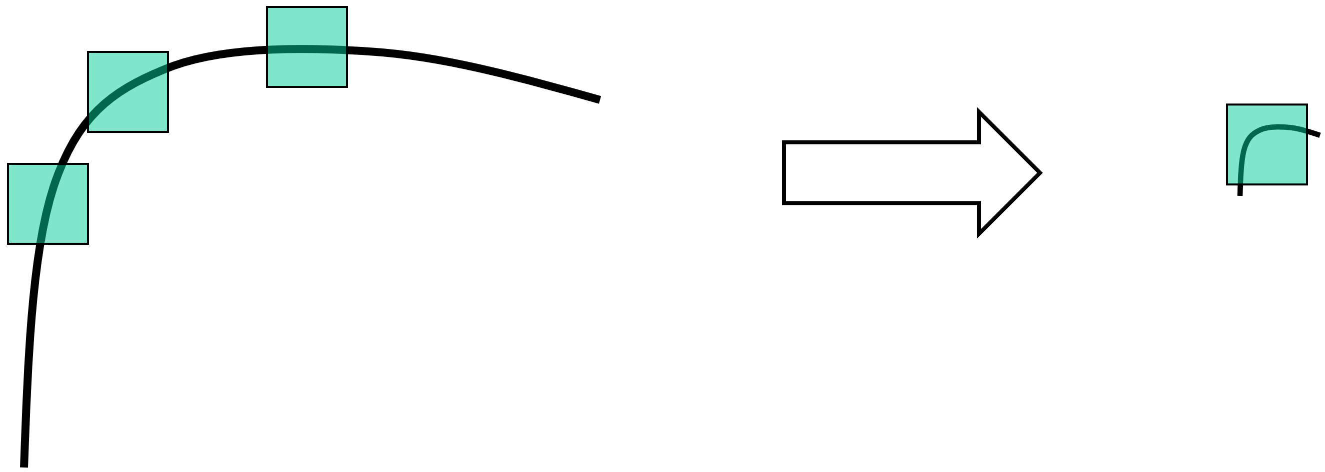
Harries Detector: Properties

Scale?



Harries Detector: Properties

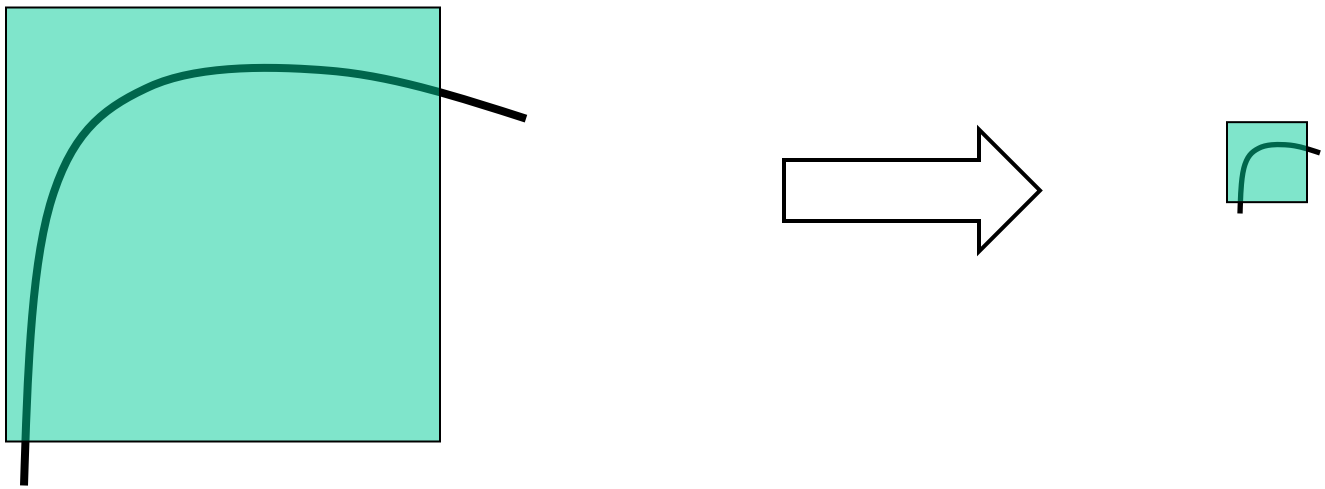
Scale?



Corner response f is not invariant to image scale

Work around?

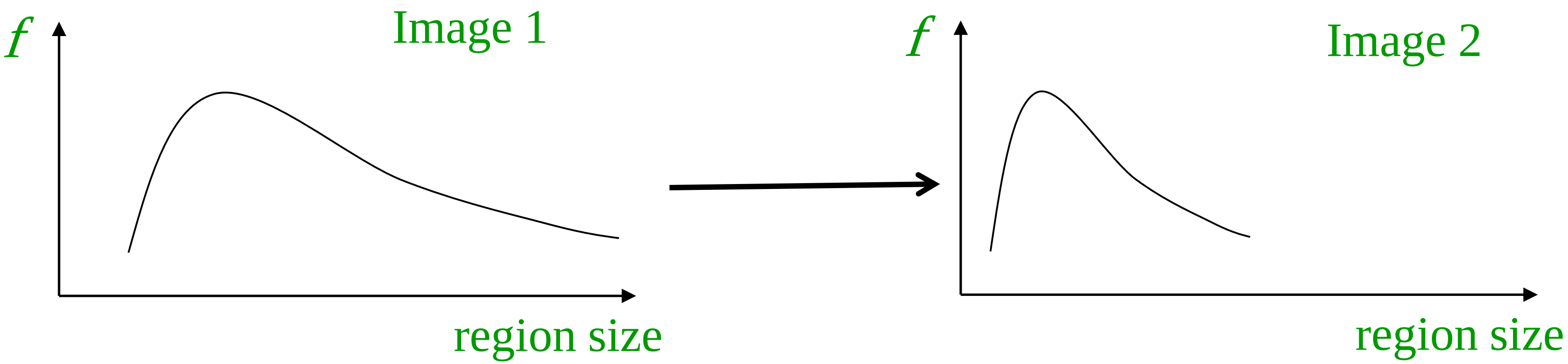
Scale invariance



How do we get the right window size *independently* in each image ?

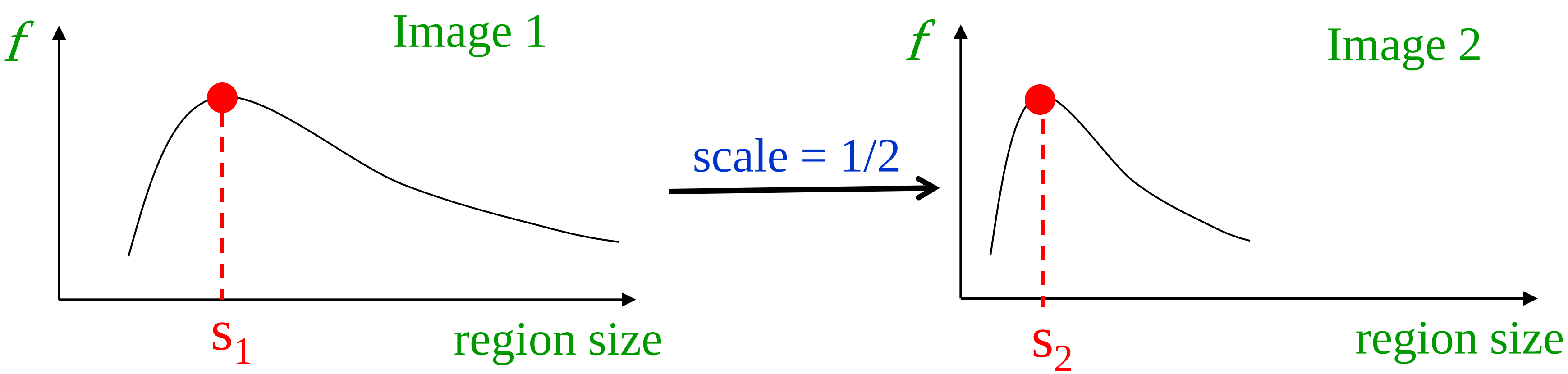
Scale invariance

- Design a function which is scale invariant
- E.g. average intensity, calculate the average intensity for different window sizes for a fixed image location



Scale invariance

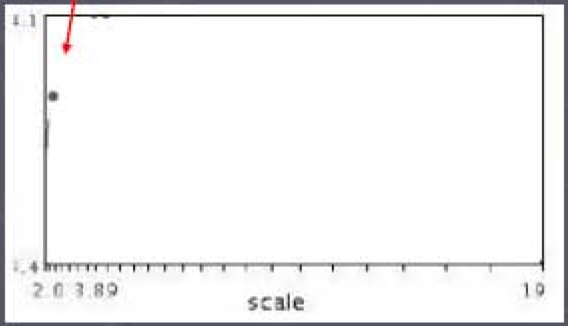
- Design a function which is scale invariant
- E.g. average intensity, calculate the average intensity for different window sizes for a fixed image location



Important: this scale invariant region size is found in each image independently!

Automatic scale selection

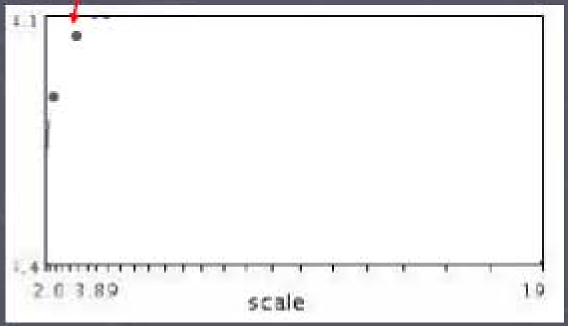
Lindeberg et al., 1996



$f(I_{i_1 \dots i_m}(x, \sigma))$

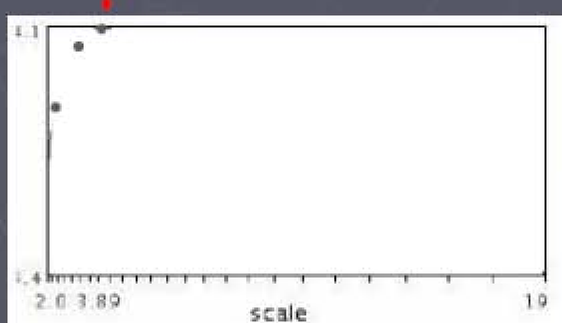
Slide from Tinne Tuytelaars

Automatic scale selection



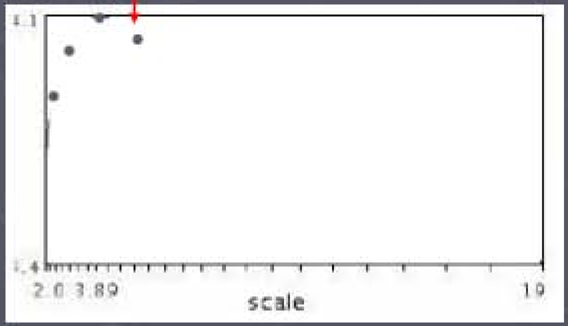
$f(I_{i_1 \dots i_m}(x, \sigma))$

Automatic scale selection



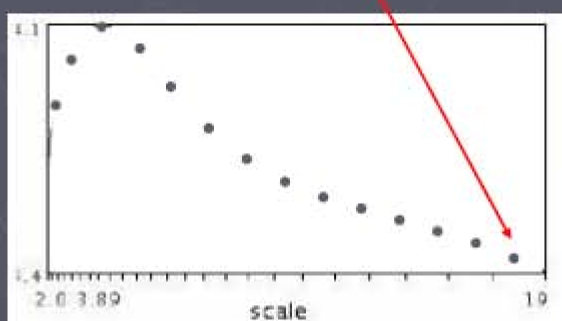
$f(I_{i_1 \ldots i_m}(x, \sigma))$

Automatic scale selection



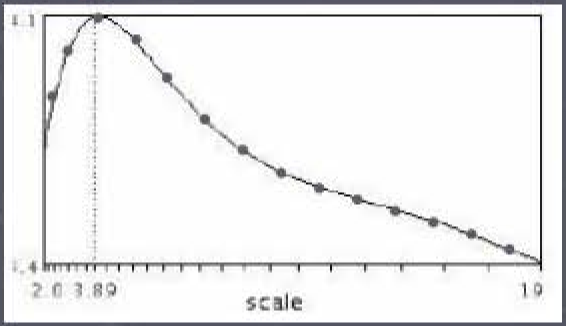
$f(I_{i_1 \dots i_m}(x,\sigma))$

Automatic scale selection



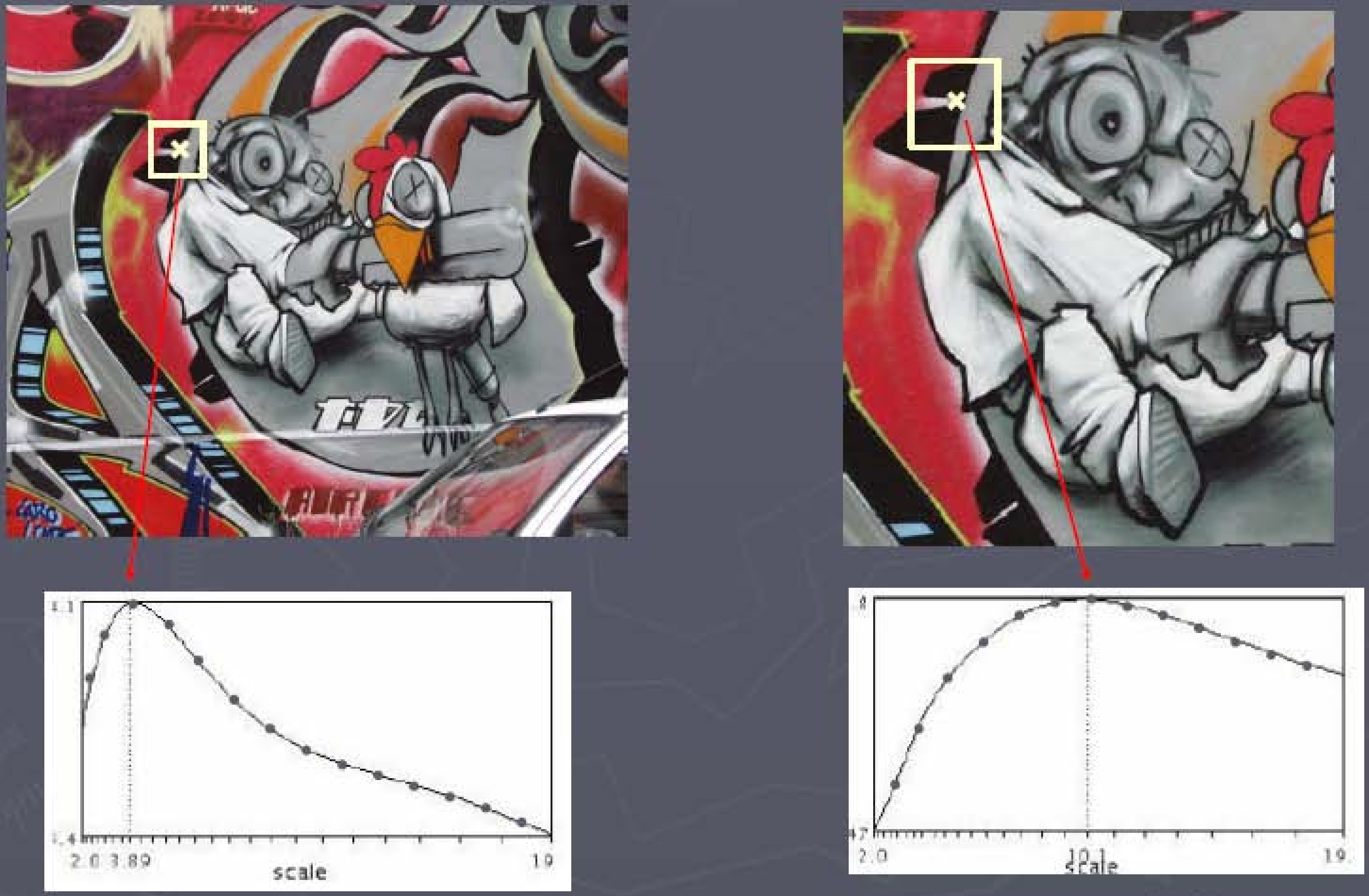
$f(I_{i_1 \ldots i_m}(x, \sigma))$

Automatic scale selection



$f(I_{i_1 \ldots i_m}(x, \sigma))$

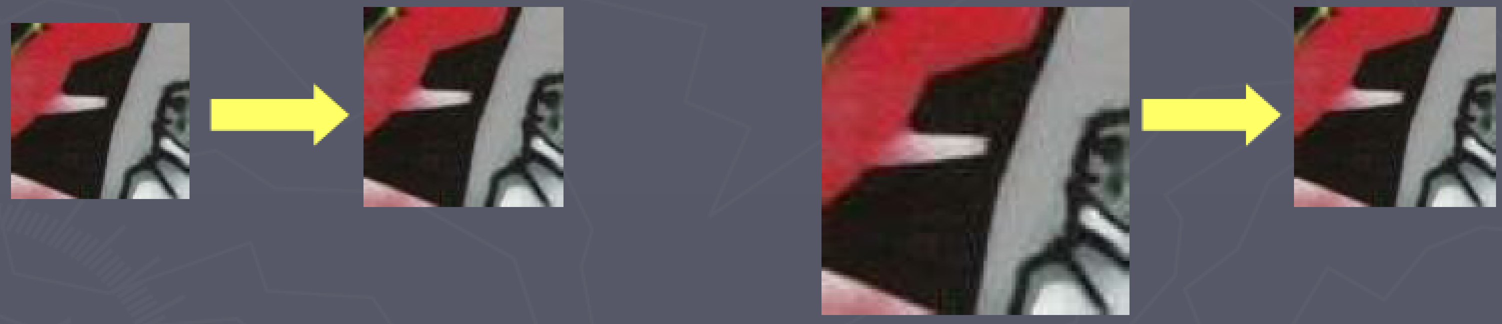
Automatic scale selection



$f(I_{i_1 \ldots i_m}(x, \sigma))$ $\qquad$ $f(I_{i_1 \ldots i_m}(x', \sigma'))$

Automatic scale selection

Normalize: rescale to fixed size



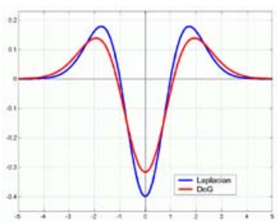
Scale invariance

Other functions for calculating scale

- Laplacian
- Difference of Gaussian (DoG)

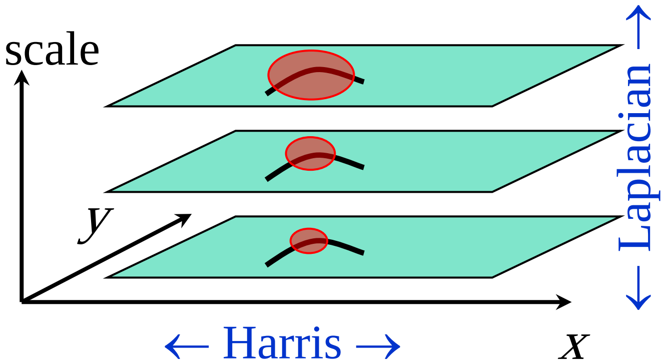
Harris-Laplacian:

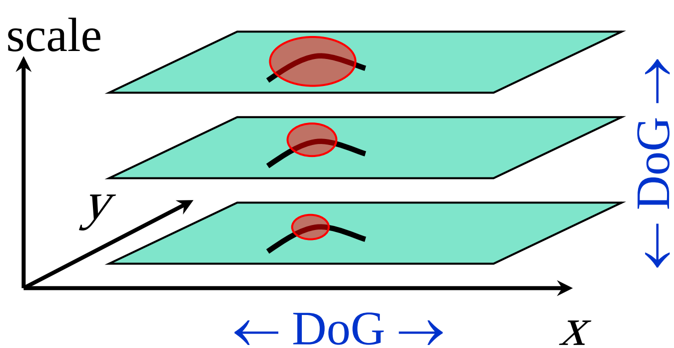
- Find local maximum of Harris corner detector in space (image coordinates)
- Find scale with convolution of image with laplace function



SIFT (Lowe)

- Find local maximum of DoG in space and scale

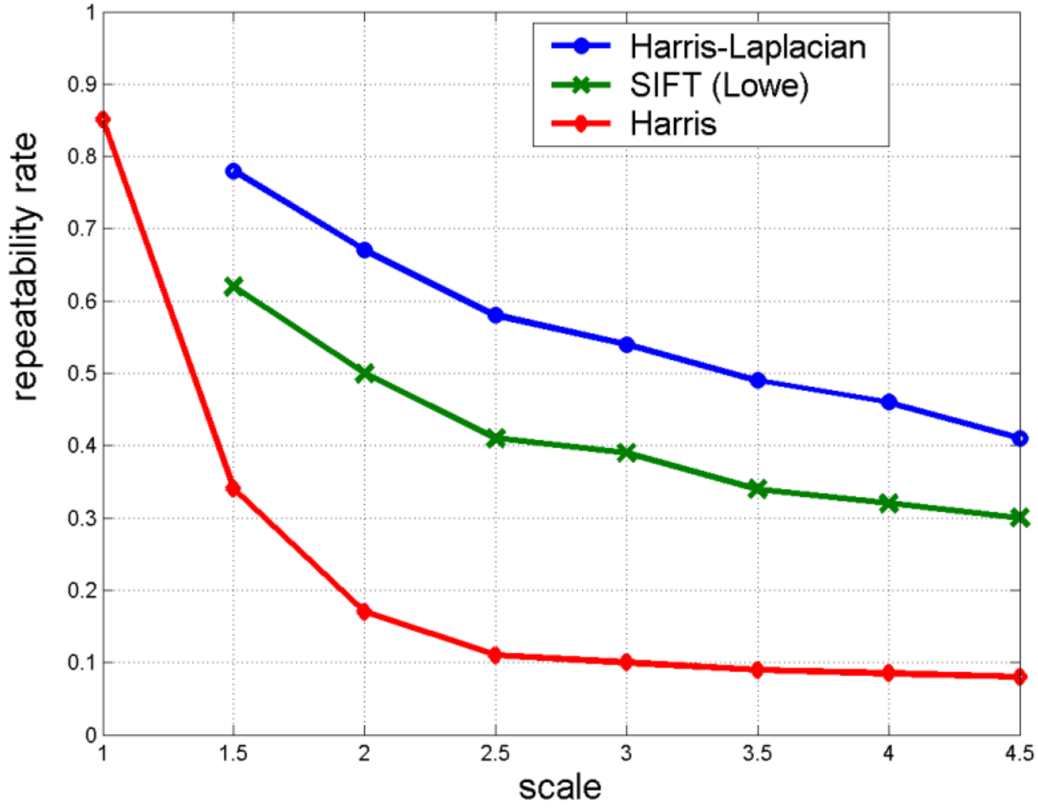




K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001
D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004

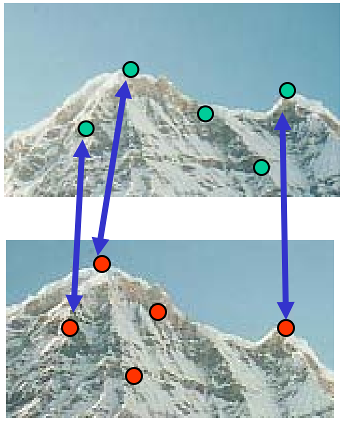
Scale invariance

Experimental evaluation of scale invariance



Repeatability rate:

$$\frac{\text{Number of corr.}}{\text{Number of possible corr.}}$$



K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001

Ideal features detector

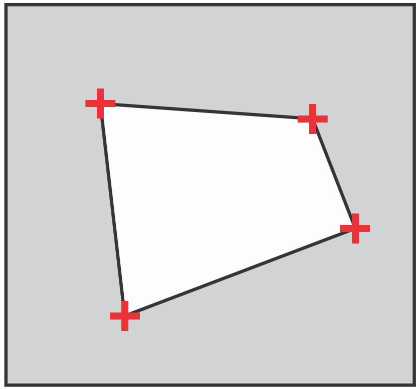
- Would always find the same feature points, regardless of image changes

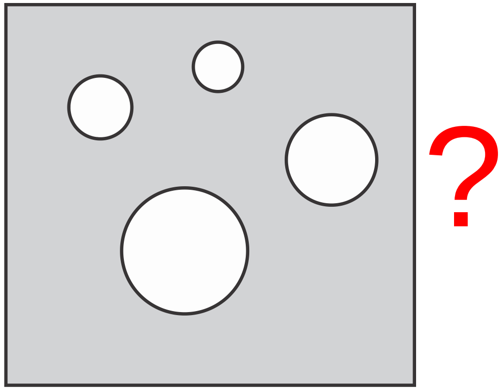
- Insensitive to
  - Scale
  - Rotation
  - Lighting
  - Perspective distortion
  - Partial occlusion

Other Feature Detectors

Corner detectors:

- Harris corners
- Susan FAST corners
- Forstner corners
- …

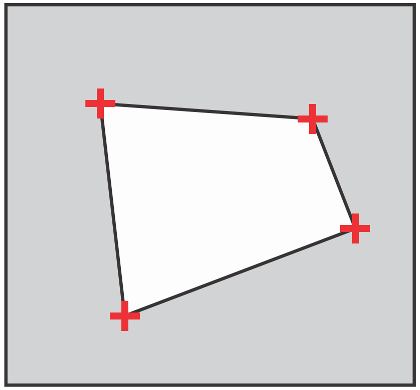




Other Feature Detectors

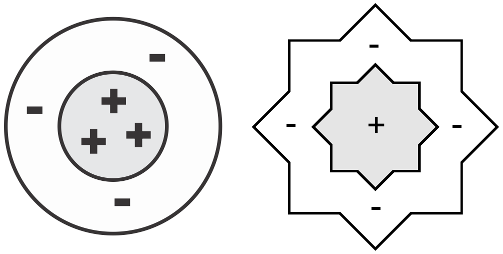
Corner detectors:

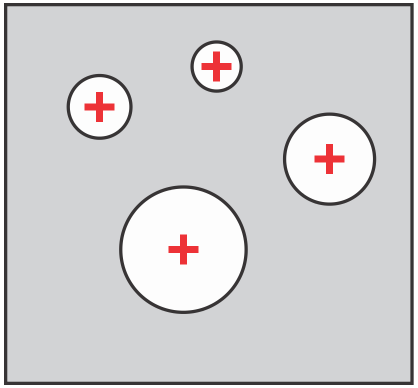
- Harris corners
- Susan FAST corners
- Forstner corners
- …



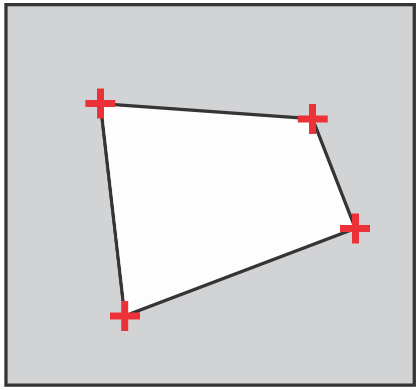
Blob detectors:

- CenSure
- STAR



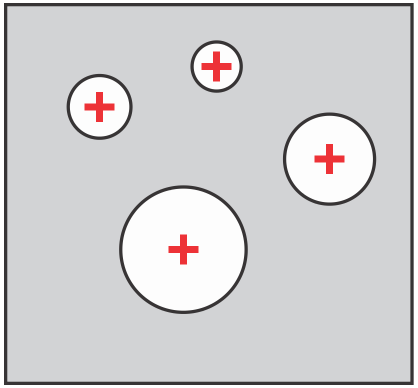


Other Feature Detectors



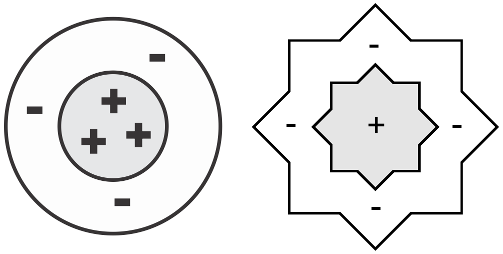
Corner detectors:

- Harris corners
- Susan FAST corners
- Forstner corners
- …



Blob detectors:

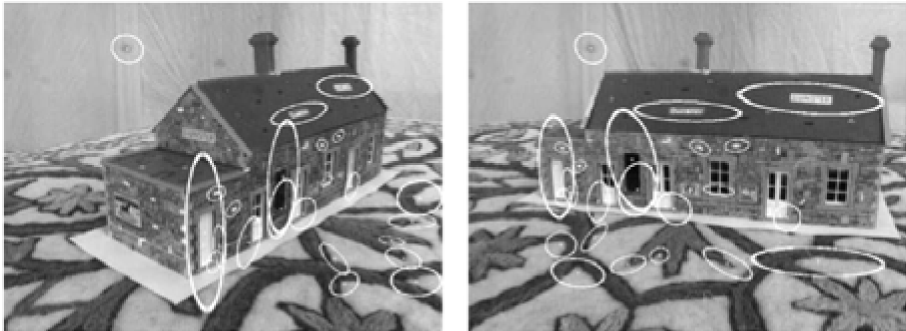
- CenSure
- STAR



Affine invariant detectors

(invariant to perspective distortion):

- Maximum stable extremal regions

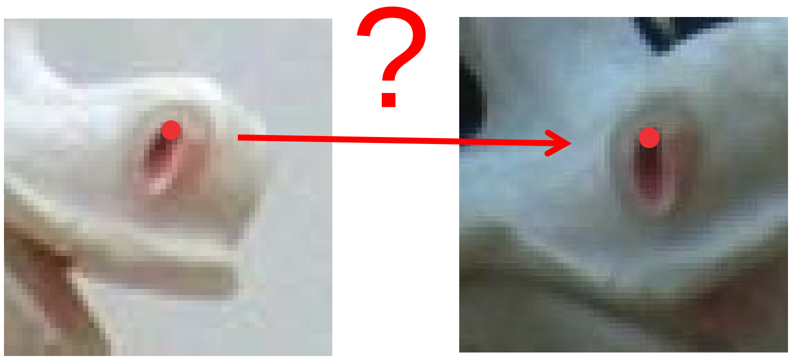


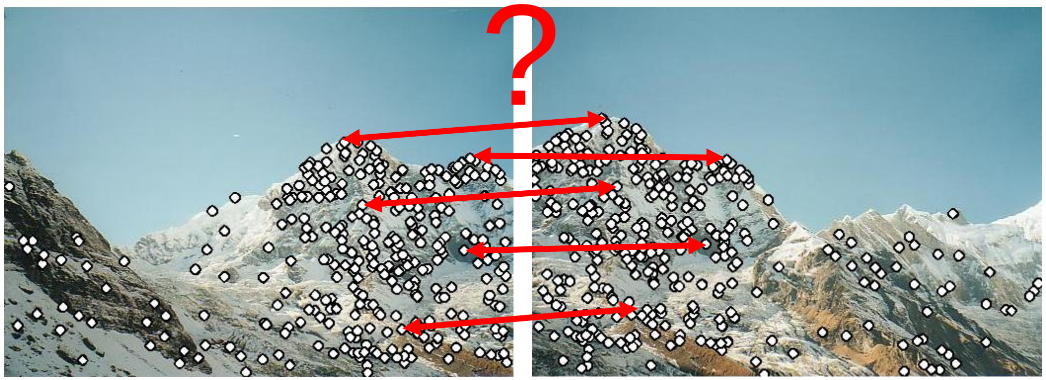
What next?

Now, that we have found the interesting points, how do we match features across different images?

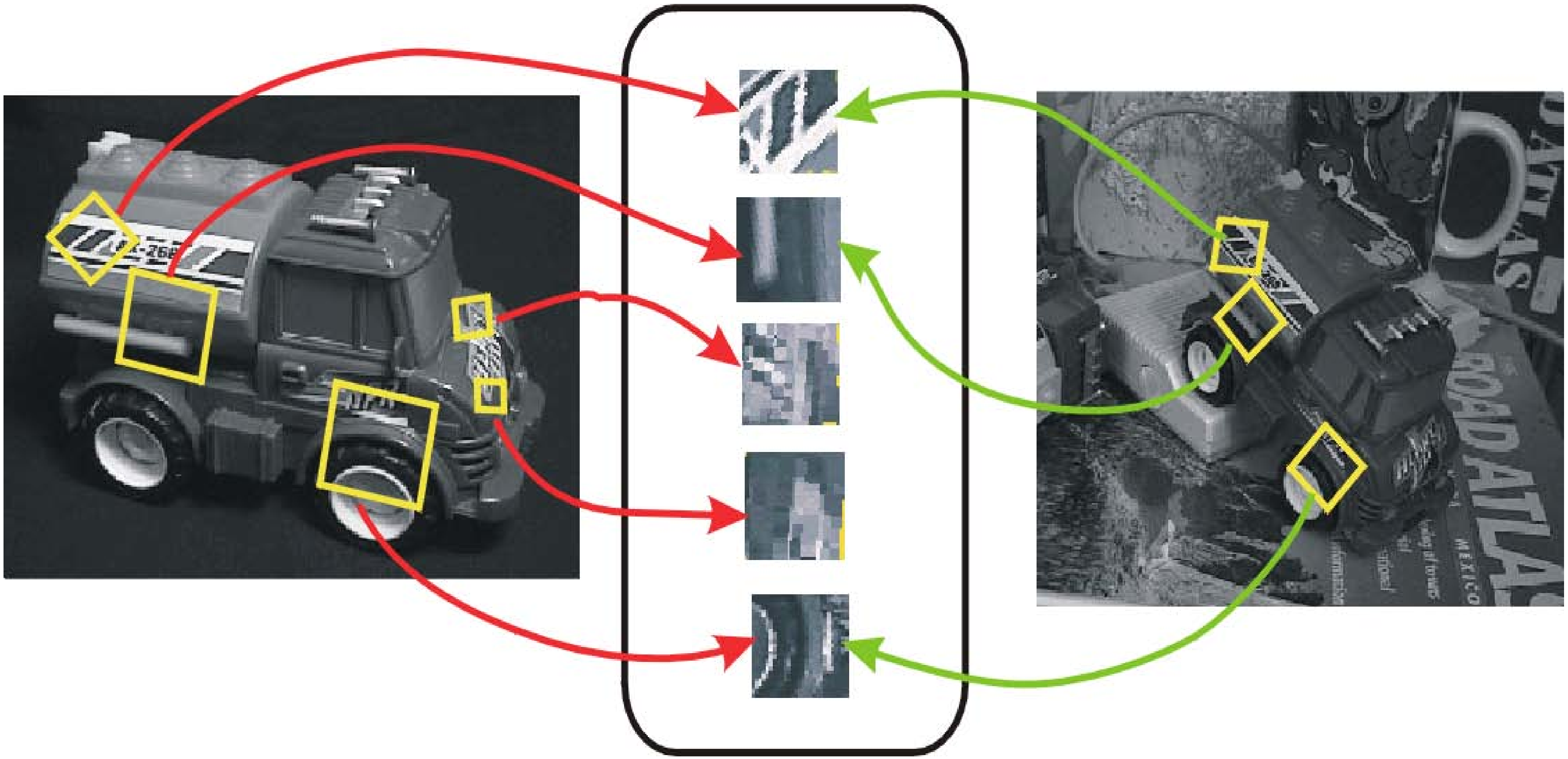
How do we compare two feature points?

- We need some *description* of the area around the feature !





Matching I

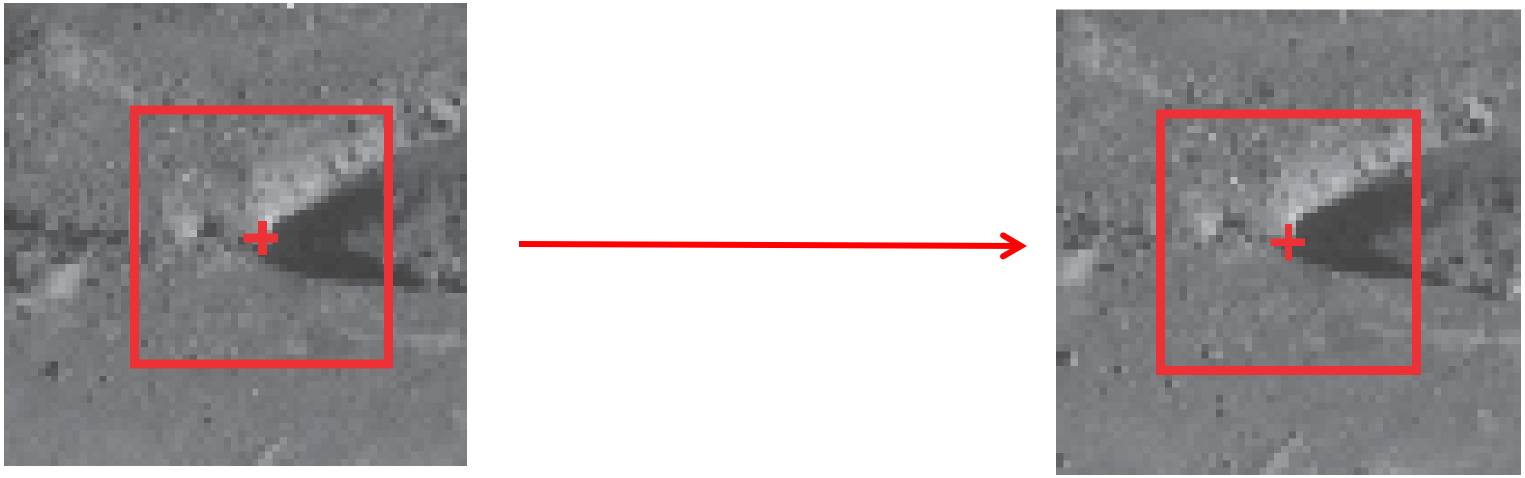


Feature Descriptors

Descriptors

Very simple example:

- Correlation of image intensities (SSD, NCC, SAD,…)



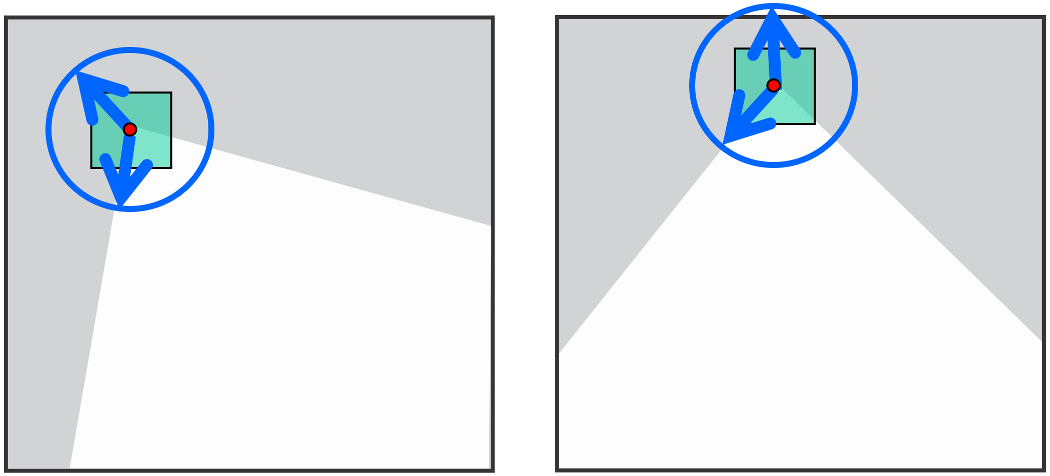
- NOT invariant to rotation, scale, illumination, ….
- Only, if images are very similar
(e.g. aerial images from altitude)

Descriptors

Rotation Invariance

- Harris corner response measure:

  depends only on the eigenvalues of the matrix H

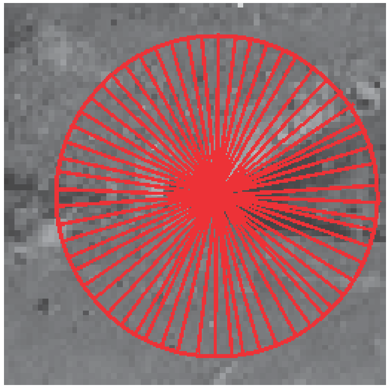


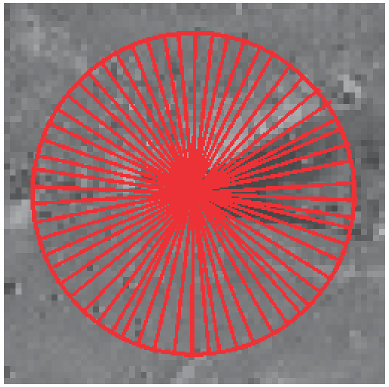
Descriptors

Rotation Invariance

- Image moments in polar coordinates

$$\left|m_{kl}\right| = \left|\iint r^k e^{-i\varphi l} I(r,\varphi) dr\, d\varphi\right|$$

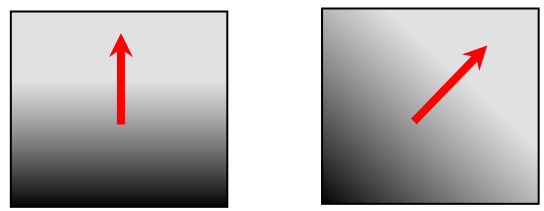




Descriptors

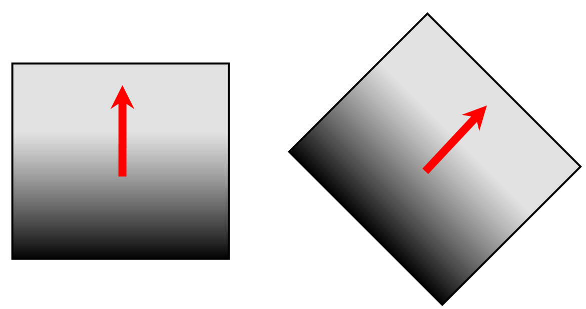
Rotation Invariance

- Find local orientation



Dominant direction of gradient

- Compute image derivatives relative to this orientation



K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001
D.Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. Accepted to IJCV 2004

Descriptors

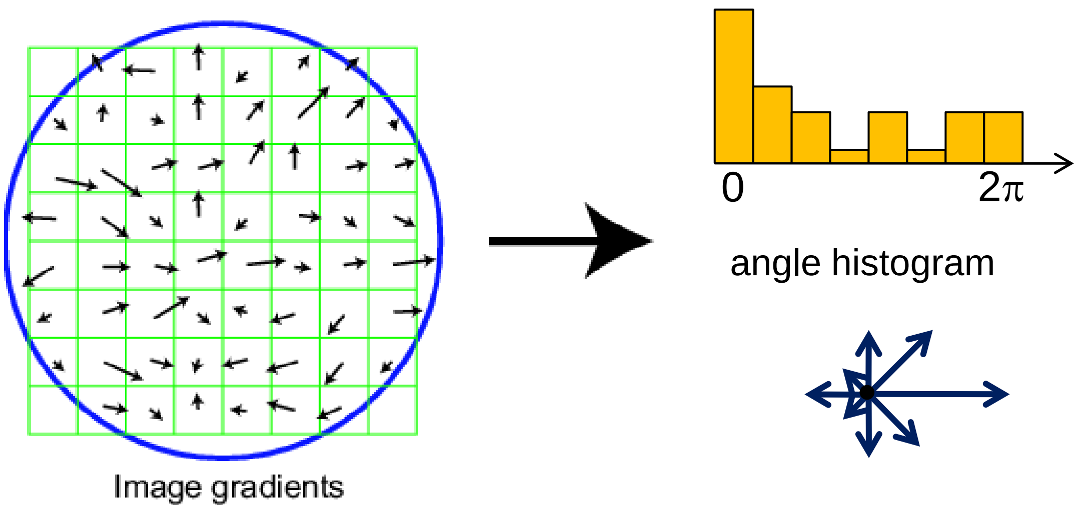
Scale Invariance

- Use scale determined by detector to normalize descriptor frame (scale descriptor window with detector window)

Descriptors

State of the art: SIFT (Scale Invariant Feature Transform)

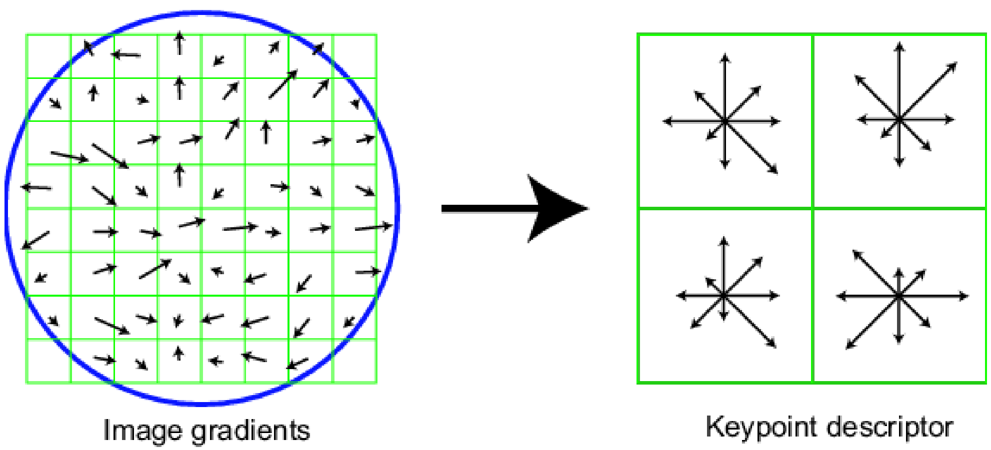
- Basic idea:
  - Take 16x16 square window around detected feature
  - Compute edge orientation (angle of the gradient - 90°) for each pixel
  - Throw out weak edges (threshold gradient magnitude)
  - Create histogram of surviving edge orientations



D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
http://www.cs.ubc.ca/~lowe/keypoints/

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor



D.Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. IJCV 2004
http://www.cs.ubc.ca/~lowe/keypoints/

SIFT

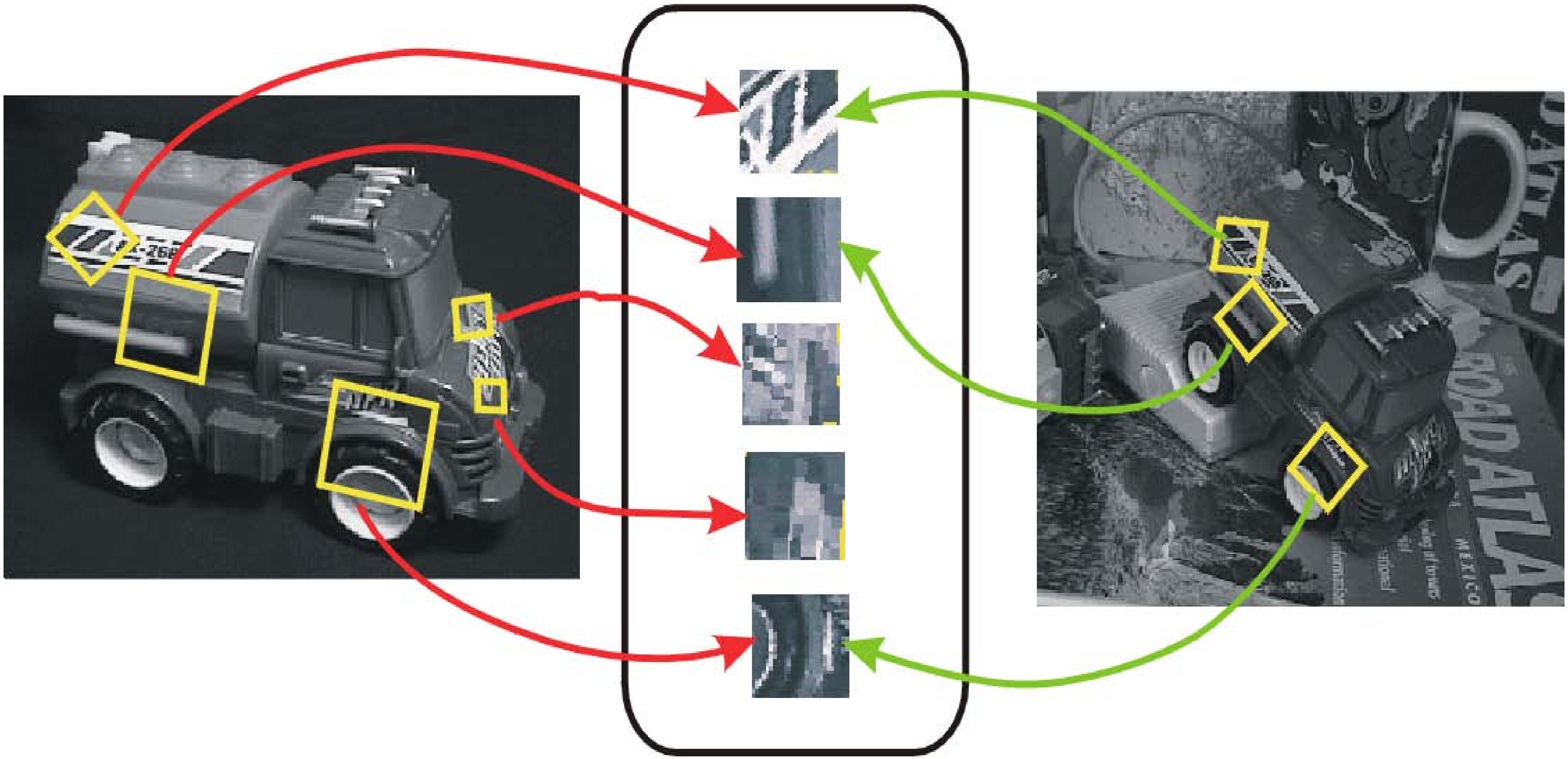
Very robust detector for image matching

- Handles changes in viewpoint (up to 60 degree out of plane rotation)
- Significant changes in illumination
- Fast and efficient
- Various speeded up versions running in real-time
  - SURF (SpeededUpRobustFeatures), 64 dimensional descriptor
  - DAISY, 16 dimensional descriptor

H. Bay, A. Ess, T. Tuytelaars, L. Van Gool: SURF: Speeded Up Robust Features, CVIU, 2008
E.Tola, V. Lepetit, P. Fua: A fast local descriptor for dense matching. CVPR 2008

Matching II

Now, that we found all the interesting points, and a good distinguishable description for each feature, how do we find the best match?



Matching II

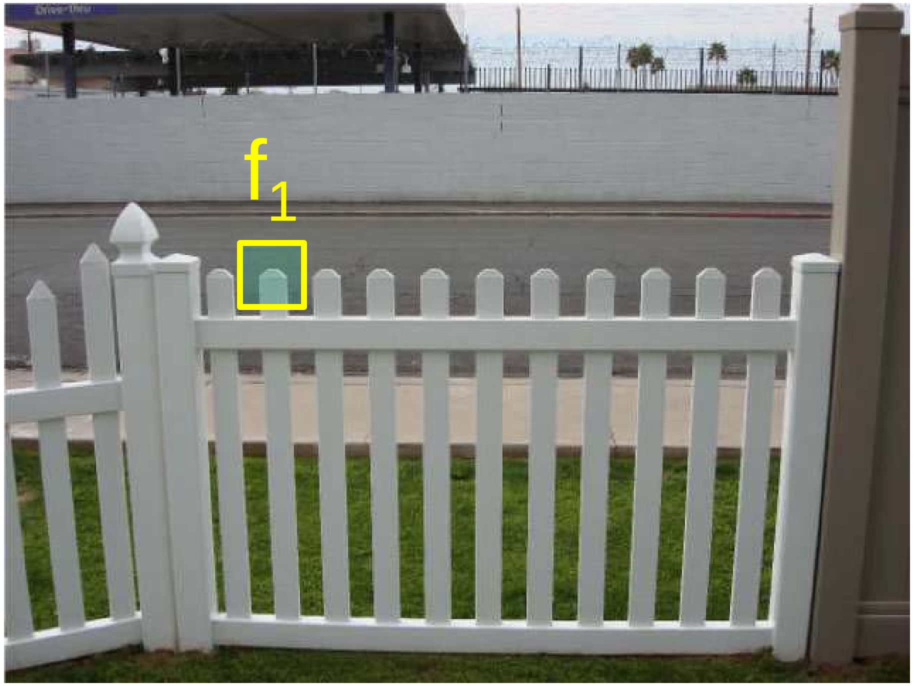
Given a feature in one image I_1 and a set of features in the other image I_2:

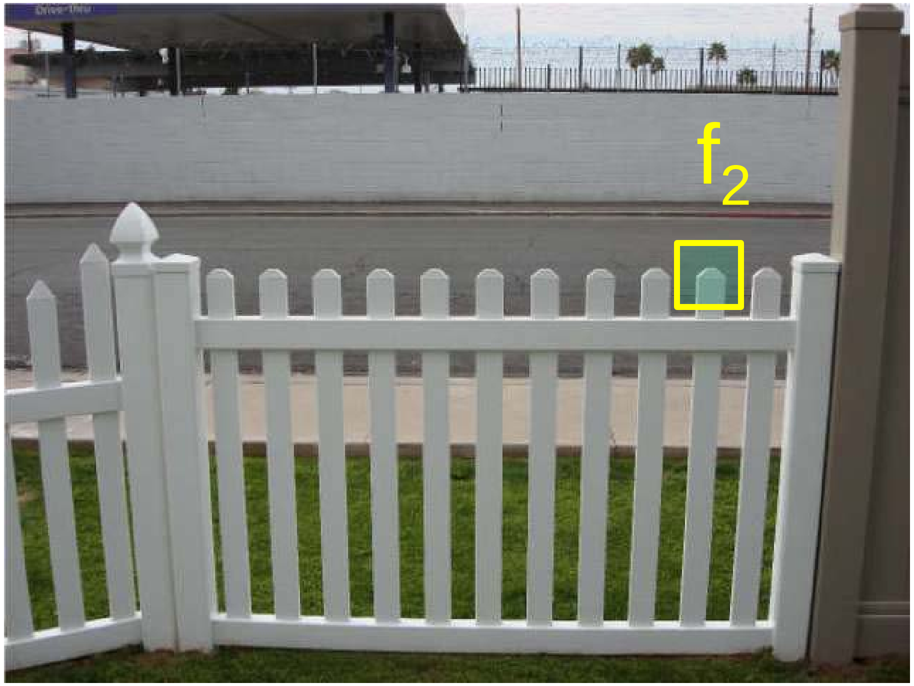
1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with minimum distance

Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
    - sum of square differences between entries of the two descriptors
    - can give good scores to very ambiguous (bad) matches

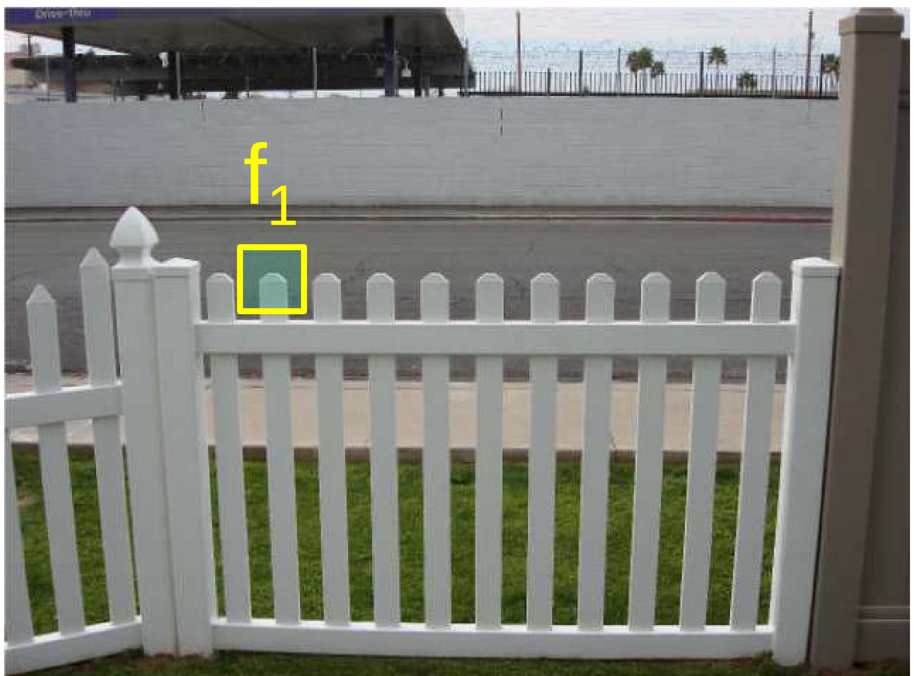


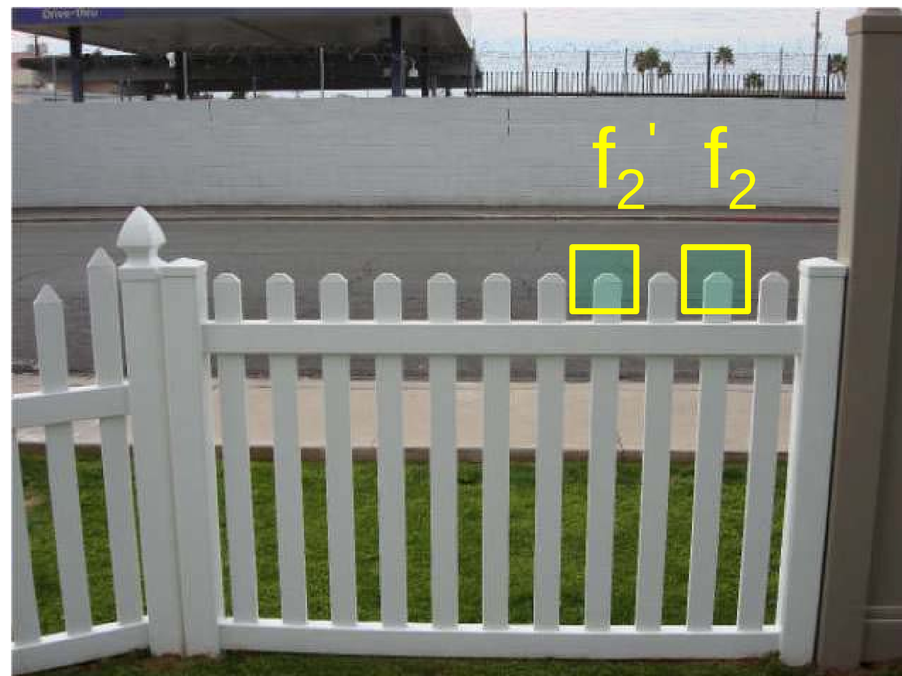


Feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = $SSD(f_1, f_2) / SSD(f_1, f_2')$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2^{nd} best SSD match to f_1 in I_2
  - gives small values for ambiguous matches



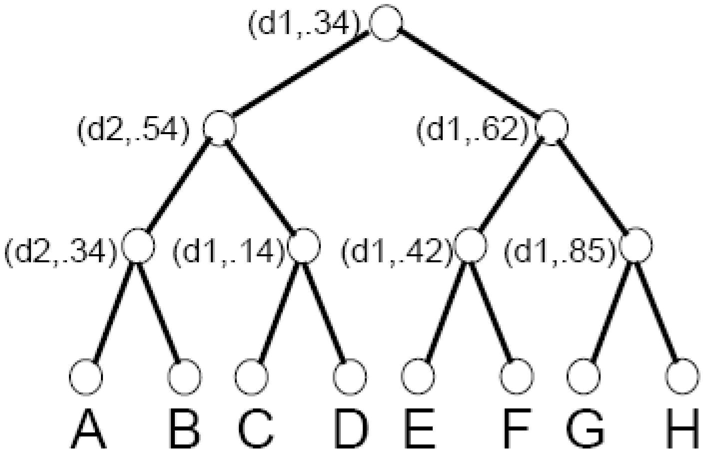


Feature distance

Unfortunately, comparing every feature in one image with every feature in the other image is $O(n^2)$ complex (imagine finding your image in a very large database)

Other search techniques:

- K-D trees, create a multi-dimensional search tree for all descriptors in each image
- Hashing



J. S. Beis, D. G. Lowe: Indexing without invariants in 3D object recognition. TPAMI 1999

References

- Chapter 4 of Szeliski's book

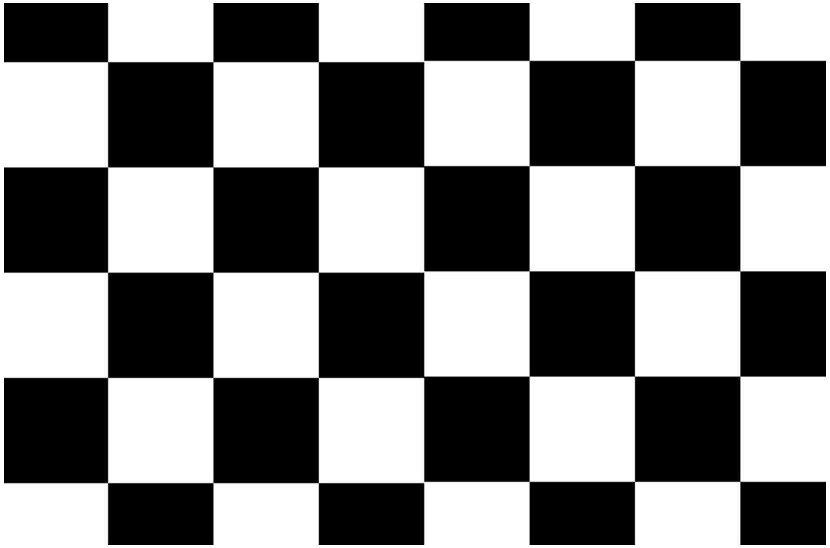
- C.Harris, M.Stephens: A Combined Corner and Edge Detector. In: Proceedings of the 4th Alvey Vision Conference, pp. 147–151,1988, http://www.bmva.org/bmvc/1988/avc-88-023.pdf
- C. Schmid, R. Mohr and C. Bauckhage : Evaluation of Interest Point Detectors. In: International Journal of Computer Vision, 37(2), 151-172, 2000 , (Evaluation and overview)
- M. Trajkovic and M. Hedley: Fast corner detection. *Image and Vision Comp.* **16** (2): 75–87, 1998 (FAST corners)
- M. Agrawal, K. Konolige, and M. R. Blas: Censure: Center surround extremas for realtime feature detection and matching. In *Proc. Europ. Conf. on Comp. Vis.*, ser. LNCS, vol. 5305, pp. 102–115, 2008 (CenSure blob detector)
- J.Matas et.al. "Distinguished Regions for Wide-baseline Stereo". Research Report of CMP, 2001, (Affine invariant detector - MSER)
- T.Tuytelaars, L.V.Gool. "Wide Baseline Stereo Matching Based on Local, Affinely Invariant Regions". BMVC 2000, (another affine invariant detector)
- K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001 (indexing for scale invariance)
- D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004 (SIFT)
- H. Bay, A. Ess, T. Tuytelaars, L. Van Gool: SURF: Speeded Up Robust Features, CVIU, 2008 (SURF descriptor)
- E.Tola, V. Lepetit, P. Fua: A fast local descriptor for dense matching. CVPR 2008 (DAISY descriptor)
- J. S. Beis, D. G. Lowe: Indexing without invariants in 3D object recognition. TPAMI 1999 (KD-trees)

Homework

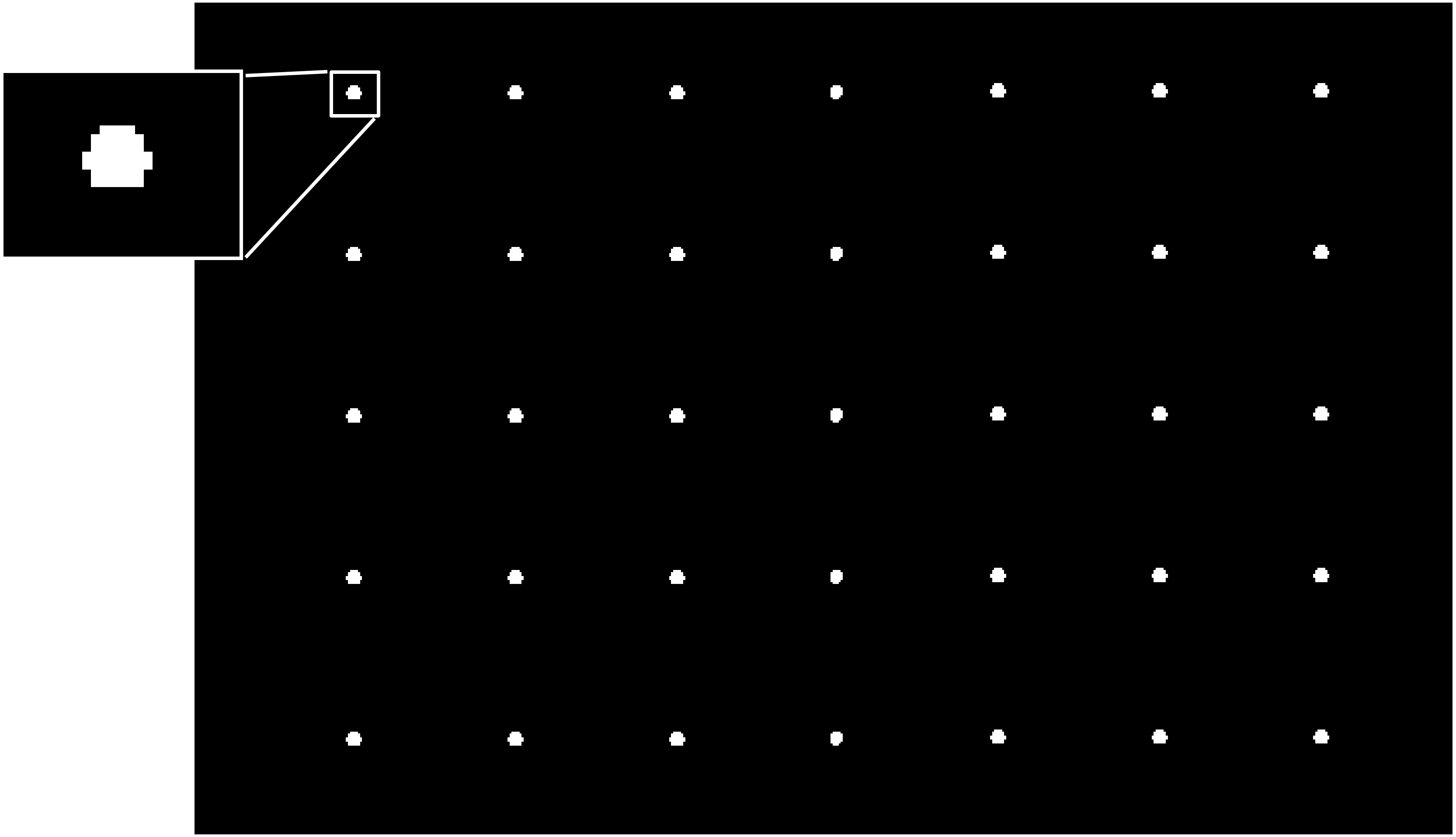
- Implement Harris corner feature detector
- Implement a feature matcher
- Maltlab or C/C++

Homework

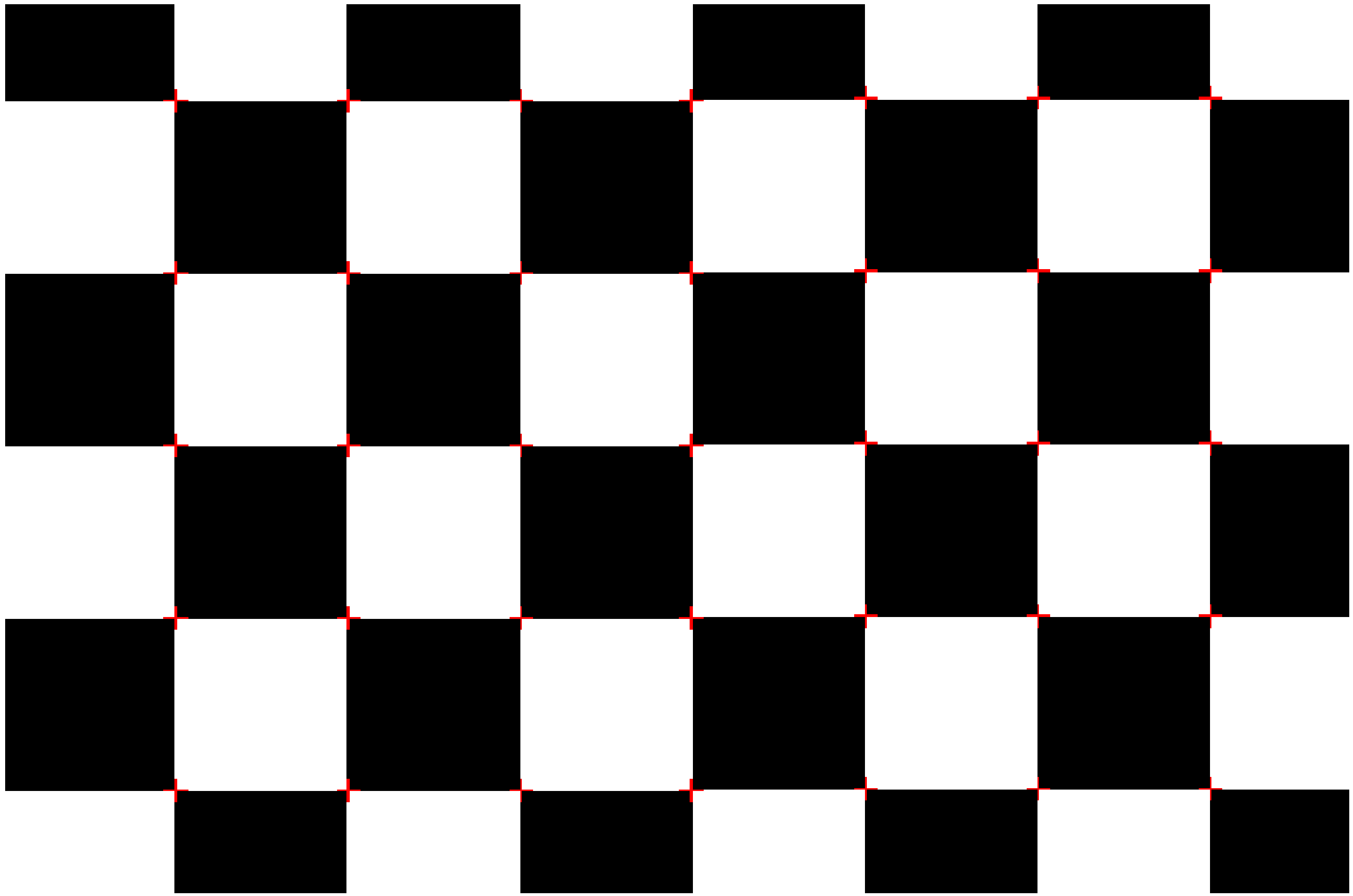
- Check your algorithm with the chess2 test image
- Apply a 50% threshold for f,
  which means $thr = 0.5 \cdot [\max(f) - \min(f)] + \min(f)$
  and a 15x15 Gaussian weighted window to calculate H



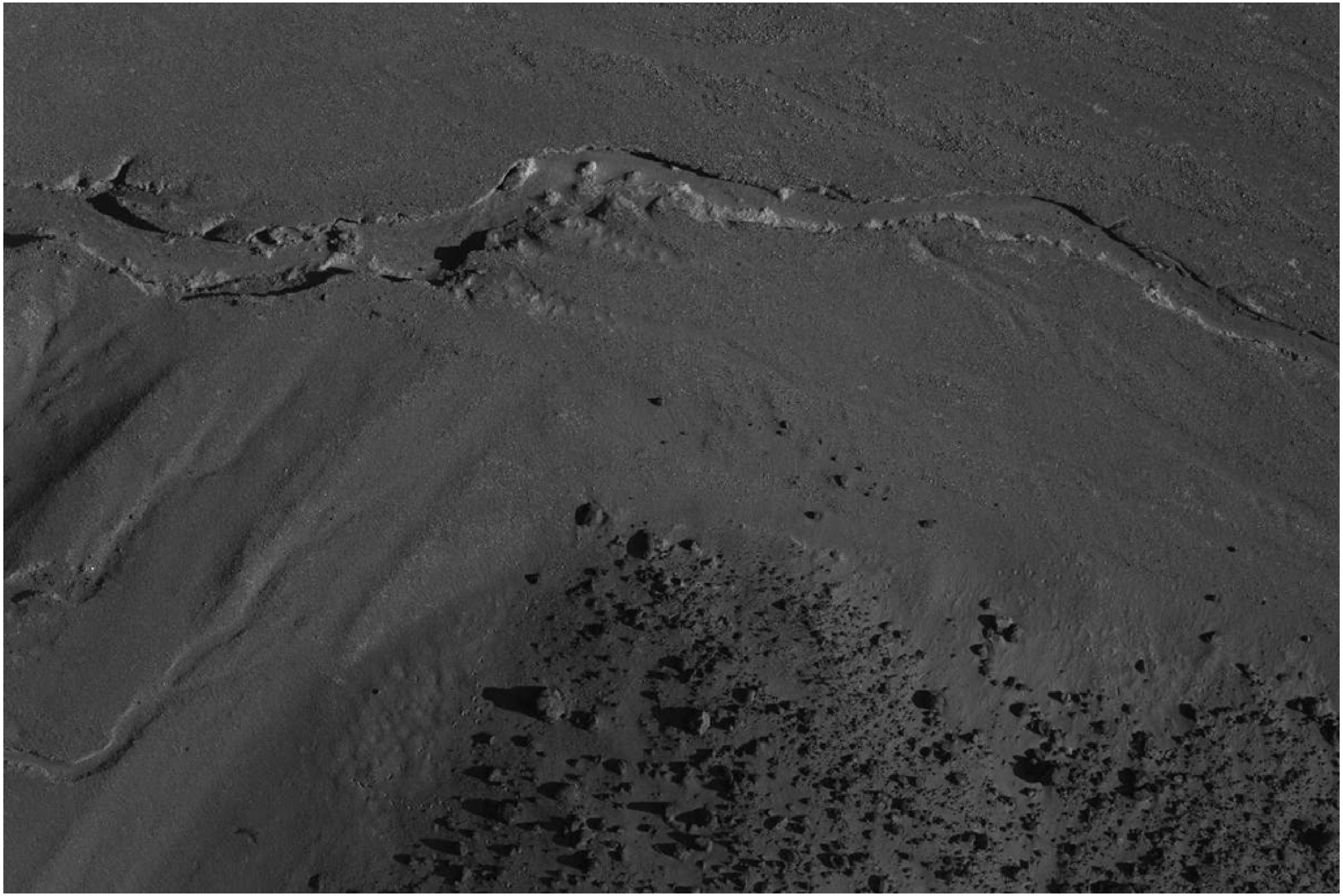
Homework



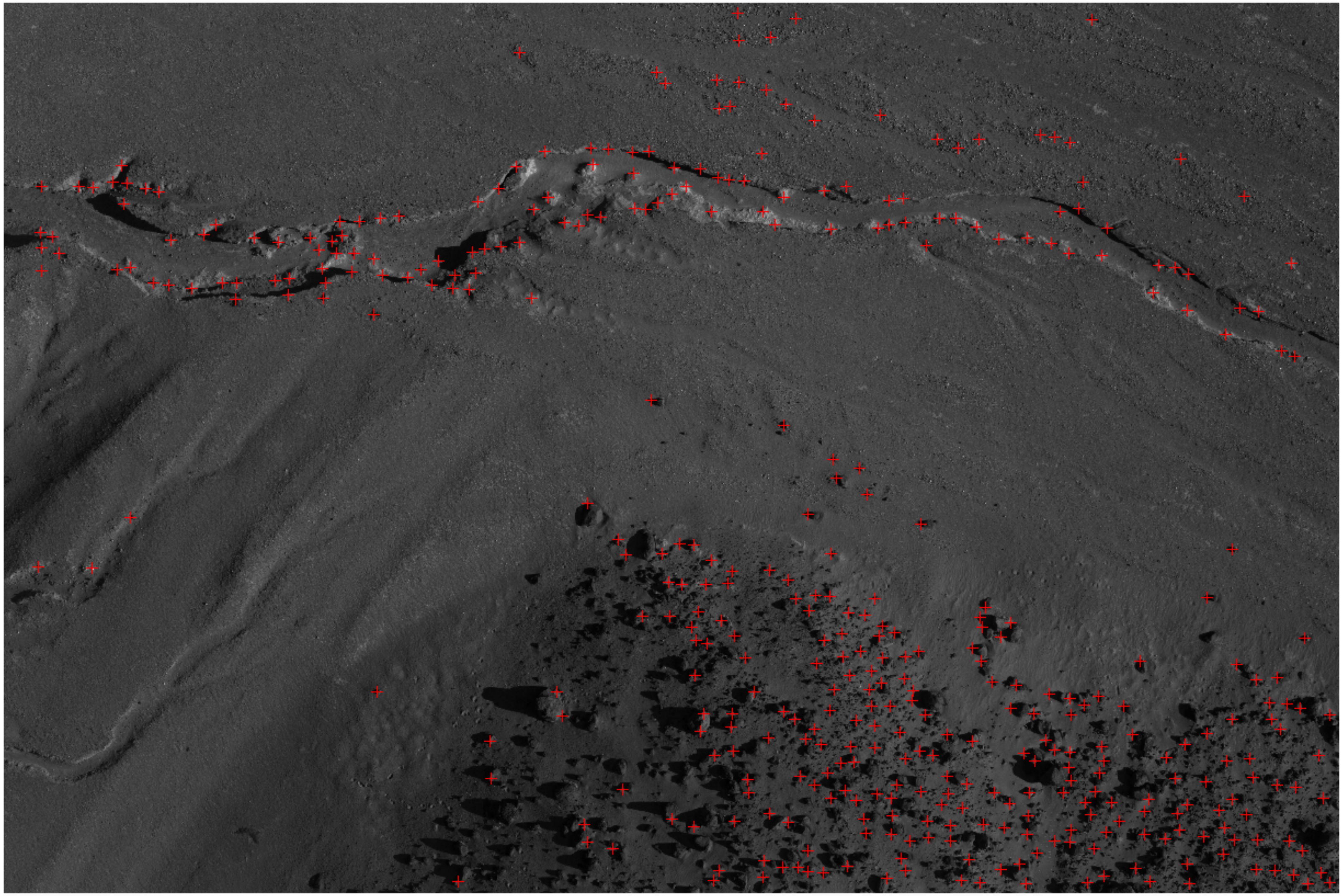
Homework



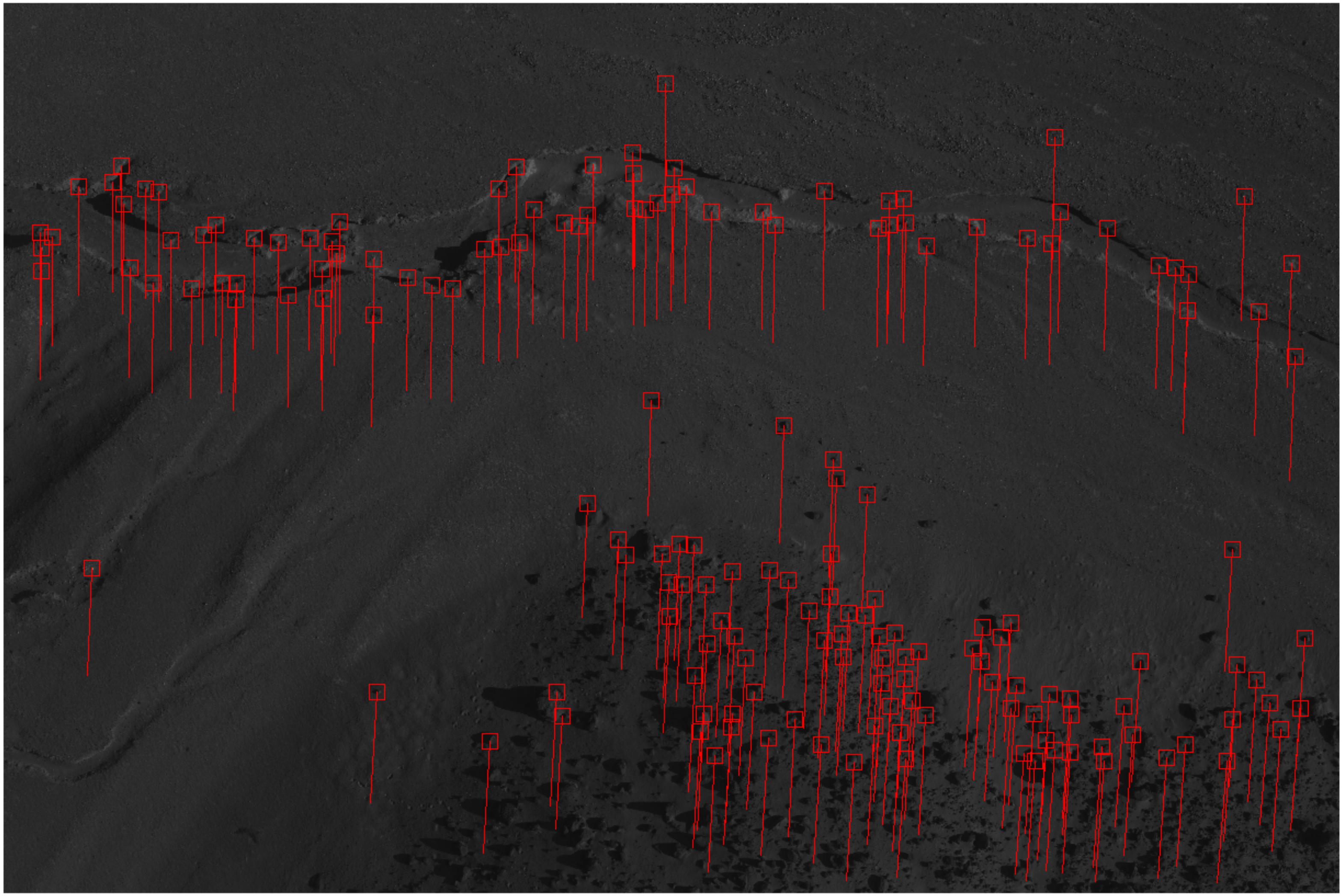
Mars Hill



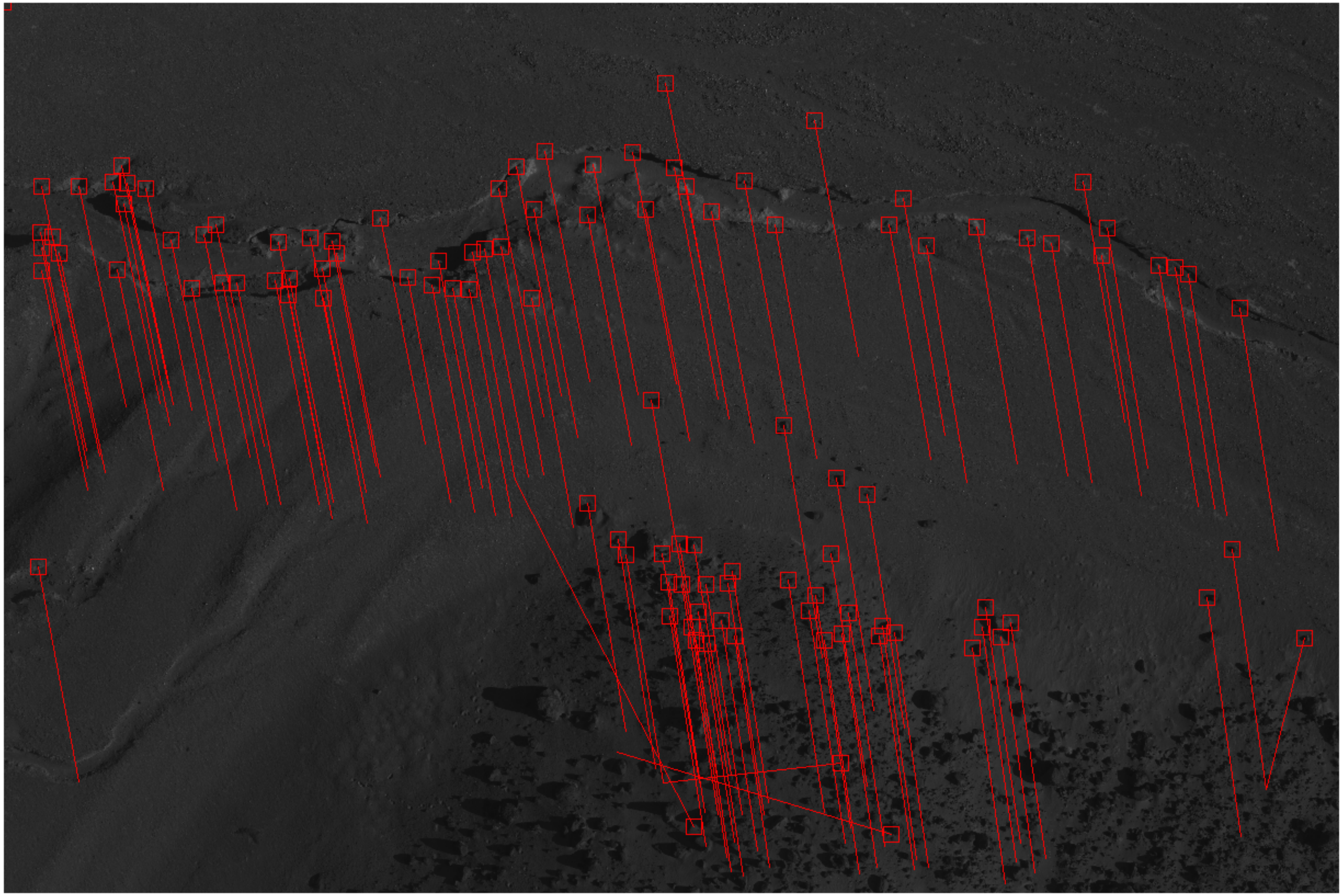
Homework



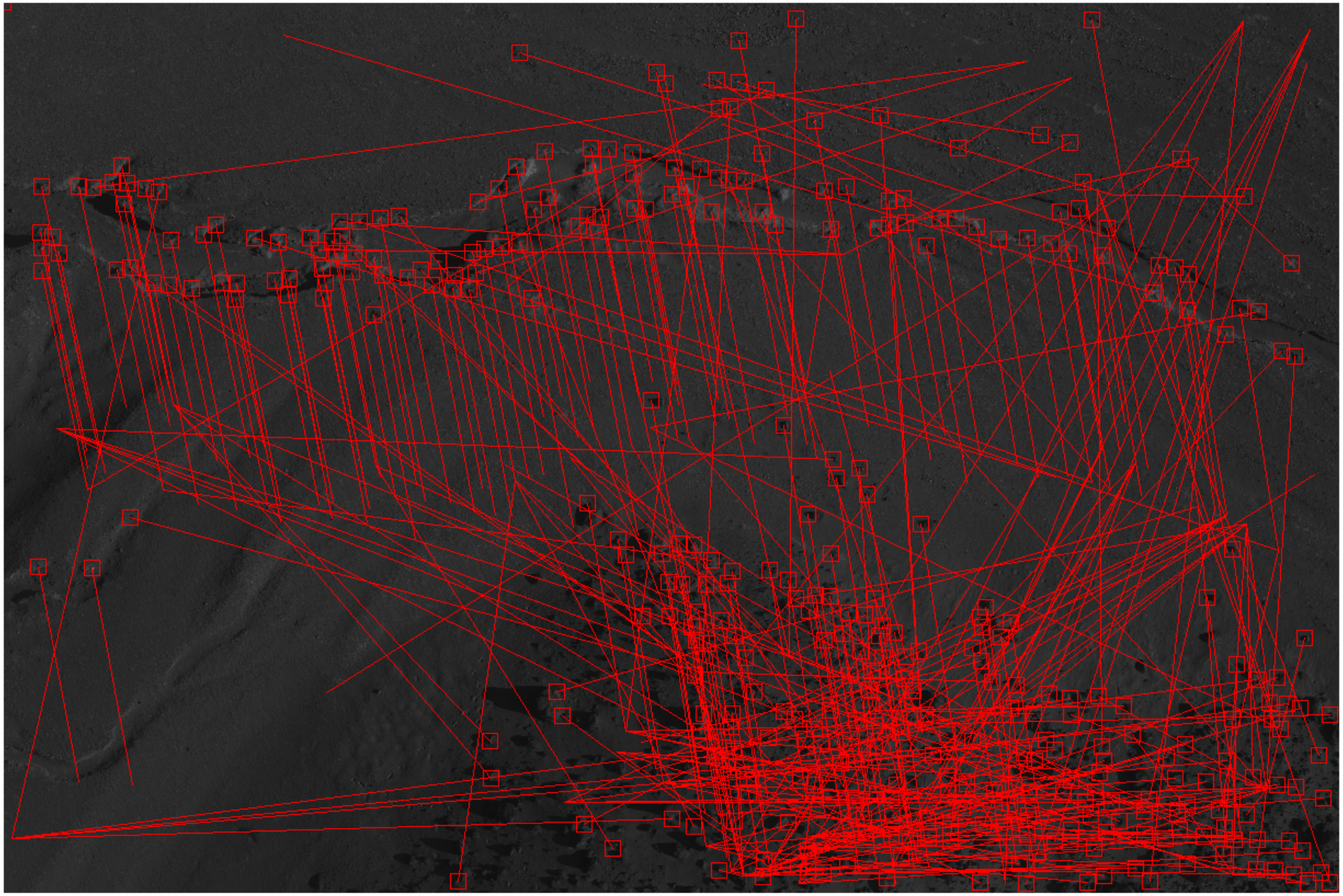
Homework



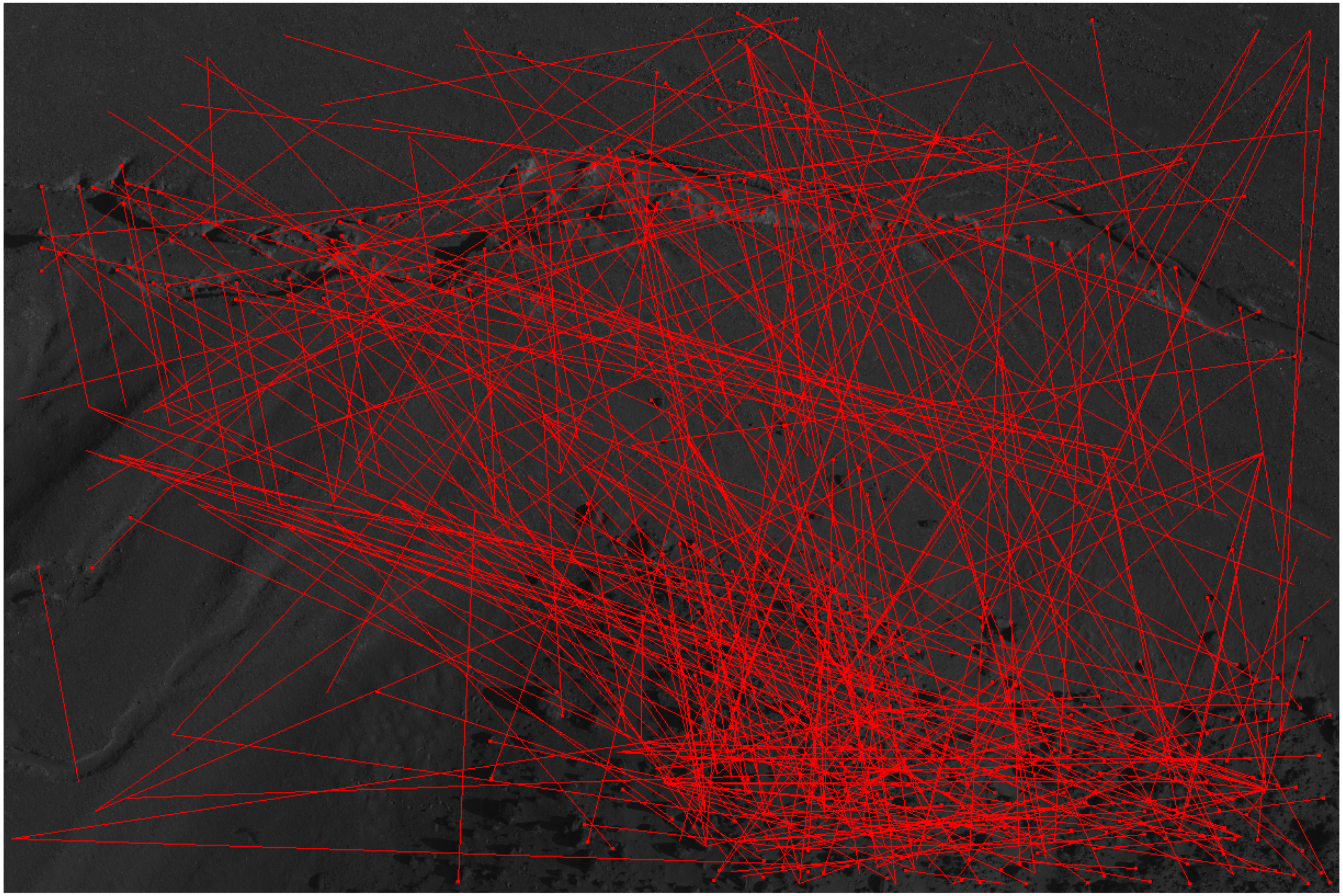
Homework



Homework



Homework



Homework