ECE 1390/2390 Image Processing and Computer Vision – Fall 2021

Feature detection and descriptors

Ahmed Dallal

Quiz 4 - Reminder

- Wed, 11/3
- Stereo geometry, camera calibration, and multiple views

Reading

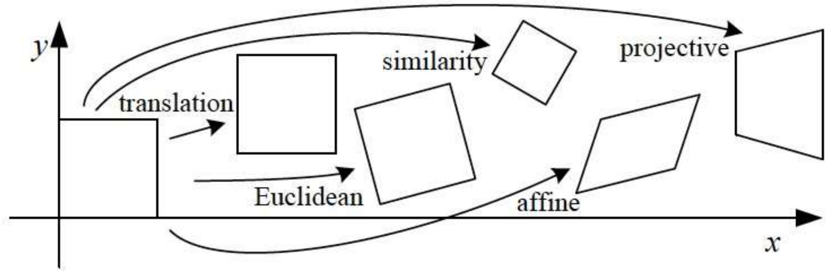
- Forsyth and Ponce: 5.3-5.4
  - Szeliski also covers this well – Section 4 – 4.1

- Paper: Distinctive Image Features from Scale-Invariant Keypoints (on Canvas)

Introduction to "features"

The basic image point matching problem

- Suppose I have two images related by some transformation. Or have two images of the same object in different positions.
- How to find the transformation of image 1 that would align it with image 2?



We want *Local*(1) *Features*(2)

- Goal: Find points in an image that can be:
  - Found in other images
  - Found precisely – well localized
  - Found reliably – well matched

We want *Local*[(1)] *Features*[(2)]

Why?

- Want to compute a fundamental matrix to recover geometry
- Robotics/Vision: See how a bunch of points move from one frame to another. Allows computation of how camera moved -> depth -> moving objects
- Build a panorama…

Suppose you want to build a panorama

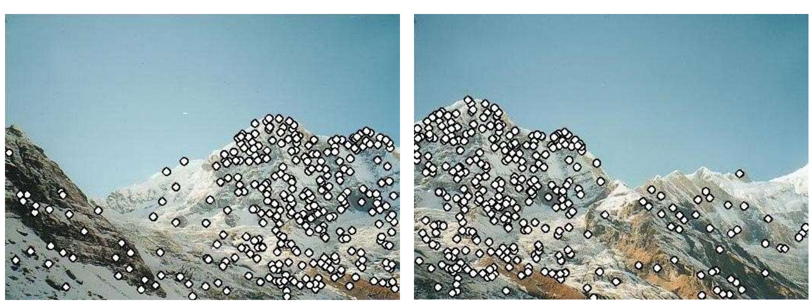
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

How do we build panorama?

- We need to match (align) images

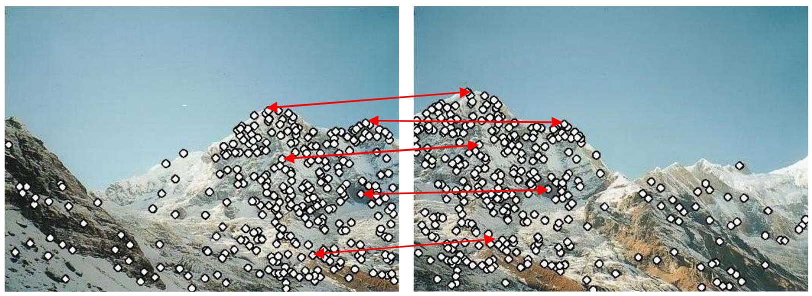
Matching with Features

- Detect features (feature points) in both images



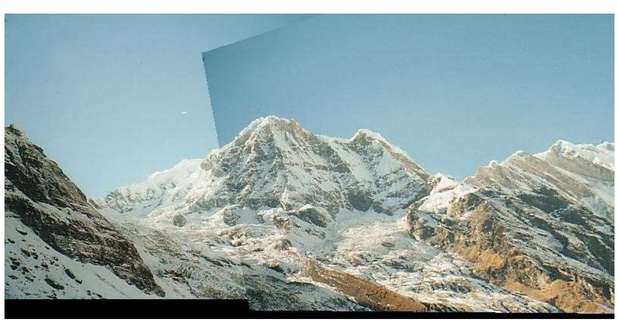
Matching with Features

- Detect features (feature points) in both images
- Match features - find corresponding pairs



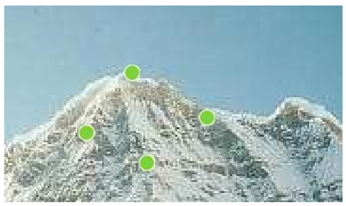
Matching with Features

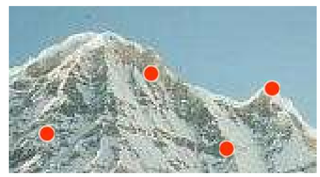
- Detect features (feature points) in both images
- Match features - find corresponding pairs
- Use these pairs to align images



Matching with Features

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



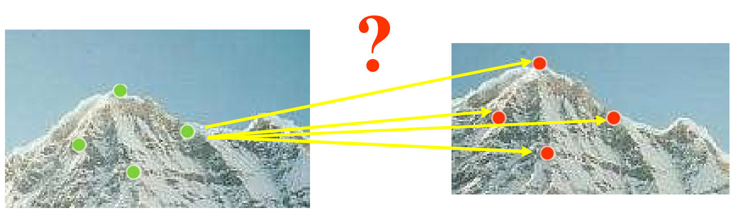


no chance to match!

We need a repeatable detector

Matching with Features

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

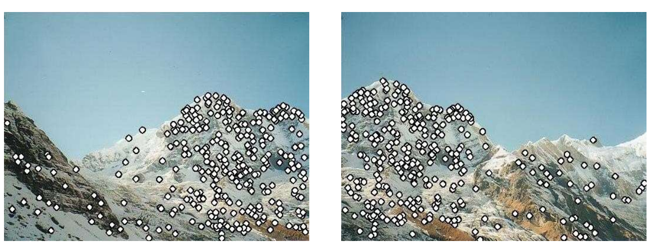


We need a reliable and distinctive ***descriptor***

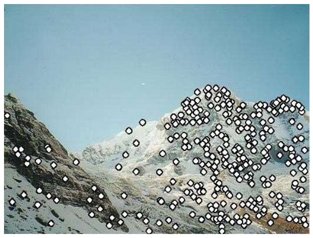
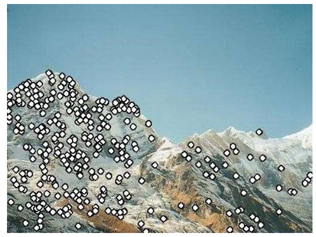
More motivation…

- Feature points are used also for:
  - Image alignment (e.g. homography or fundamental matrix)
  - 3D reconstruction
  - Motion tracking
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation
  - …other

Characteristics of good features



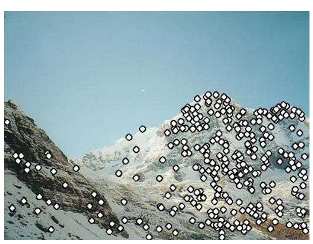
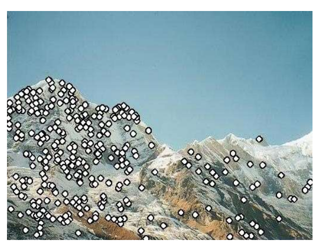
Characteristics of good features

Repeatability/Precision

- The same feature can be found in several images despite geometric and photometric transformations

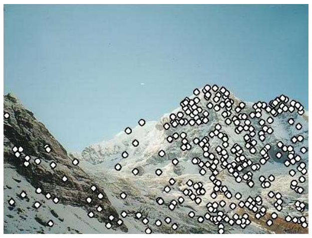
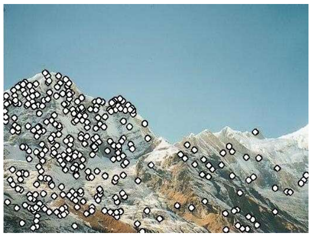
Characteristics of good features

Saliency/Matchability

- Each feature has a distinctive description

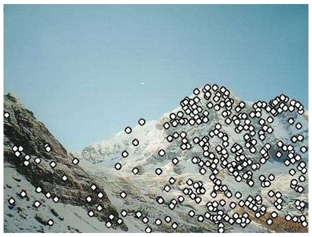
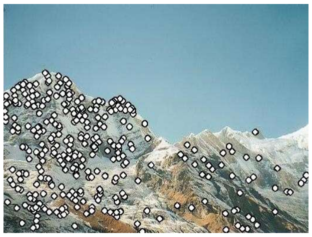
Characteristics of good features

Compactness and efficiency

- Many fewer features than image pixels

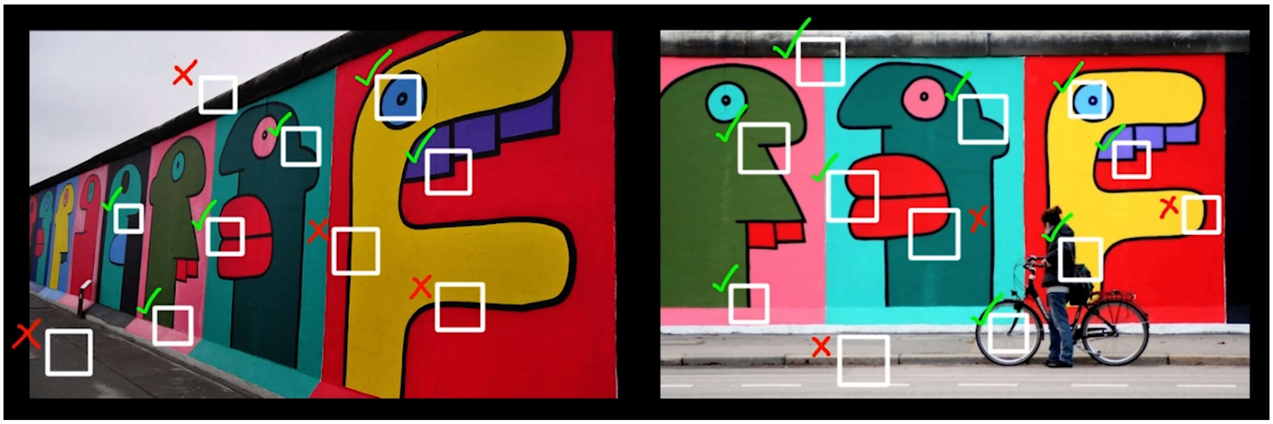
Characteristics of good features

Locality

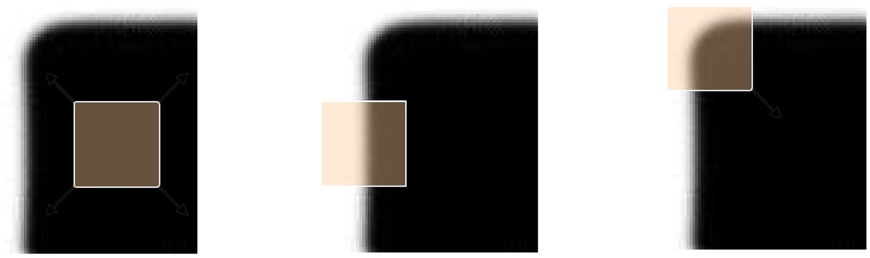
- Afeature occupies a relatively small area of the image; robust to clutter and occlusion

Good Features



Finding corners

Corner Detection: Basic Idea



"flat" region: no change in all directions

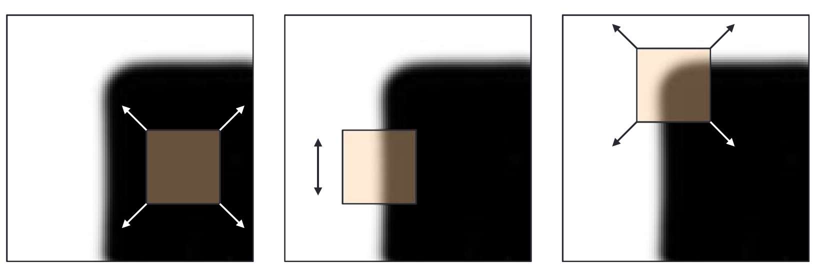
"edge": no change along the edge direction

"corner": significant change in all directions with small shift

Source: A. Efros

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

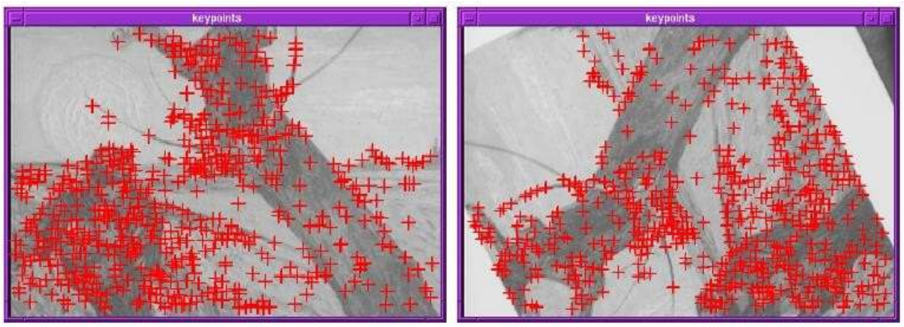
"edge": no change along the edge direction

"corner": significant change in all directions with small shift

Source: A. Efros

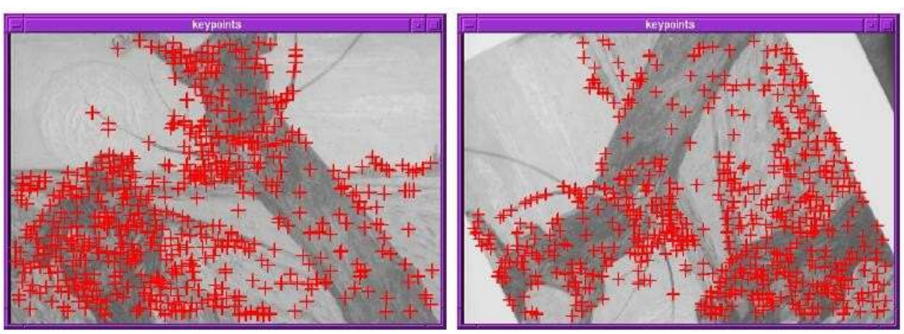
Finding *Corners*

- Key property: in the region around a corner, image gradient has two or more dominant directions



Finding Corners

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



Corner Detection: Mathematics

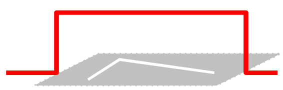
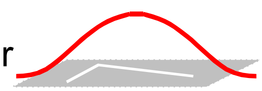
Change in appearance for the shift [*u*,*v*]:

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

Window function

Shifted intensity

Intensity

Window function $w(x,y) =$  or 

1 in window, 0 outside

Gaussian

Source: R. Szeliski

Corner Detection: Mathematics

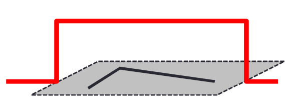
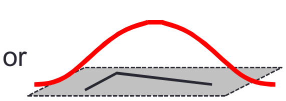
Change in appearance for the shift [*u*,*v*]:

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

Window function

Shifted intensity

Intensity

Window function $w(x,y) =$  or 

1 in window, 0 outside

Gaussian

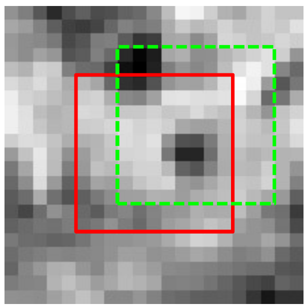
Source: R. Szeliski

Corner Detection: Mathematics

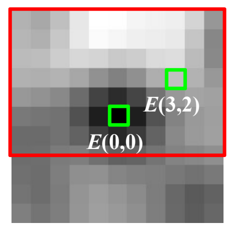
Change in appearance for the shift [*u,v*]:

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

$I(x, y)$

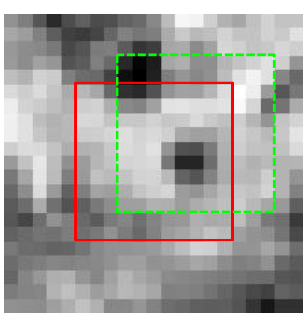


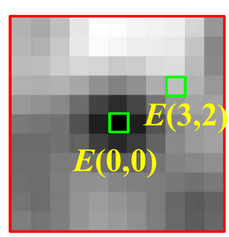
$E(u, v)$



Corner Detection: Mathematics

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





Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

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

We want to find out how this function behaves for ***small*** shifts (u,v near 0,0)

Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

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

We want to find out how this function behaves for ***small*** shifts (u,v near 0,0)

Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

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

Second-order Taylor expansion of $E(u,v)$ about (0,0) (local quadratic approximation for small u,v):

Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

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

$$F(\delta x) \approx F(0) + \delta x \cdot \frac{dF(0)}{dx} + \frac{1}{2}\delta x^2 \cdot \frac{d^2F(0)}{dx^2}$$

Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

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

$$F(\delta x) \approx F(0) + \delta x \cdot \frac{dF(0)}{dx} + \frac{1}{2}\delta x^2 \cdot \frac{d^2F(0)}{dx^2}$$

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

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

Second-order Taylor expansion of *E*(*u*,*v*) about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

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

Second-order Taylor expansion of *E*(*u*,*v*) about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

Need these derivatives…

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

- Second-order Taylor expansion of *E*(*u*,*v*) about (0,0):
- $E_u(u,v) = \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right]I_{\mathbf{x}}(x+u,y+v)$

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

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E_{uu}(u,v)=\sum_{x,y} 2w(x,y)I_{\mathbf{x}}(x+u,y+v)I_{\mathbf{x}}(x+u,y+v)$$

$$+\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_{\mathbf{xx}}(x+u,y+v)$$

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

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E_{uv}(u,v)=\sum_{x,y} 2w(x,y)I_y(x+u,y+v)I_x(x+u,y+v)$$

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

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

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

$$E_{uu}(u,v) = \sum_{x,y} 2w(x,y) I_x(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v) = \sum_{x,y} 2w(x,y) I_y(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xy}(x+u,y+v)$$

Evaluate E and its derivatives at (0,0):

$$E(u,v) \approx \overset{=0}{\boxed{E(0,0)}} + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

$$E_u(0,0) = \sum_{x,y} 2w(x,y)\underset{=0}{\boxed{\left[I(x,y) - I(x,y)\right]}} I_x(x,y)$$

$$E_{uu}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_x(x,y)$$
$$+ \sum_{x,y} 2w(x,y)\overset{=0}{\boxed{\left[I(x,y) - I(x,y)\right]}} I_{xx}(x,y)$$

$$E_{uv}(0,0) = \sum_{x,y} 2w(x,y) I_y(x,y) I_x(x,y)$$
$$+ \sum_{x,y} 2w(x,y)\overset{=0}{\boxed{\left[I(x,y) - I(x,y)\right]}} I_{xy}(x,y)$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \boxed{\frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}}$$

$$E(0,0) = 0 \qquad E_{uu}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_x(x,y)$$

$$E_u(0,0) = 0 \qquad E_{vv}(0,0) = \sum_{x,y} 2w(x,y) I_y(x,y) I_y(x,y)$$

$$E_v(0,0) = 0 \qquad E_{uv}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_y(x,y)$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

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

$$E(0,0) = 0 \qquad E_{uu}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_x(x,y)$$

$$E_u(0,0) = 0 \qquad E_{vv}(0,0) = \sum_{x,y} 2w(x,y) I_y(x,y) I_y(x,y)$$

$$E_v(0,0) = 0 \qquad E_{uv}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_y(x,y)$$

Corner Detection: Mathematics

The quadratic approximation simplifies to

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

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

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

The second moment matrix M:

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

Can be written (without the weight):

Each product is a rank 1 2x2

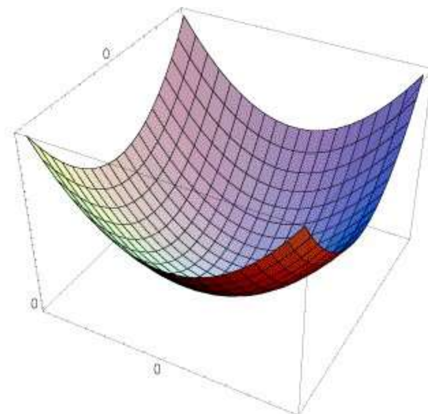
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \left(\begin{bmatrix} I_x \\ I_y \end{bmatrix} \begin{bmatrix} I_x & I_y \end{bmatrix} \right) = \sum \nabla I (\nabla I)^T$$

Interpreting the second moment matrix

The surface $E(u,v)$ is locally approximated by a quadratic form.

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

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

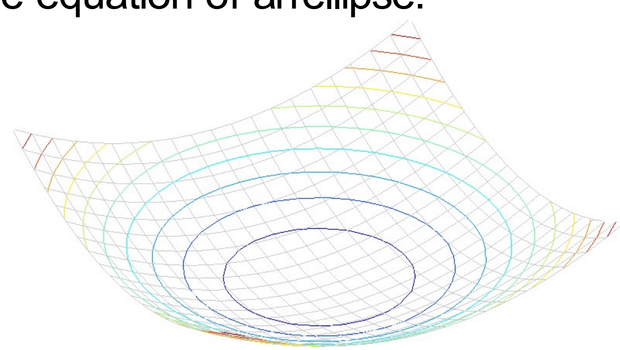


Interpreting the second moment matrix

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

$$\Sigma I_x^2 u^2 + 2\Sigma I_x I_y uv + \Sigma I_y^2 v^2 = k$$

This is the equation of an ellipse.

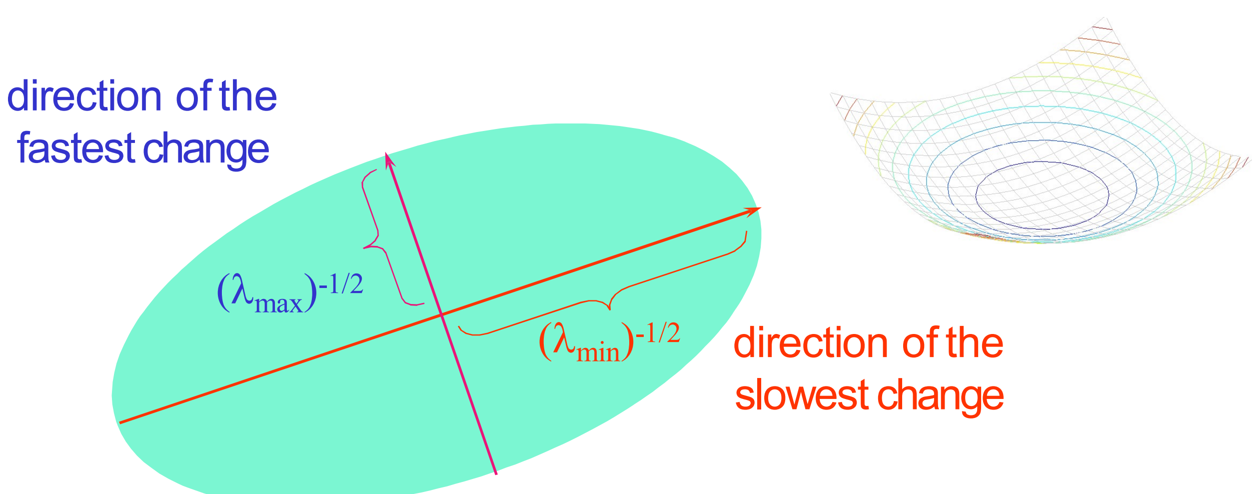


Interpreting the second moment matrix

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*

Interpreting the second moment matrix



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

Interpreting the second moment matrix

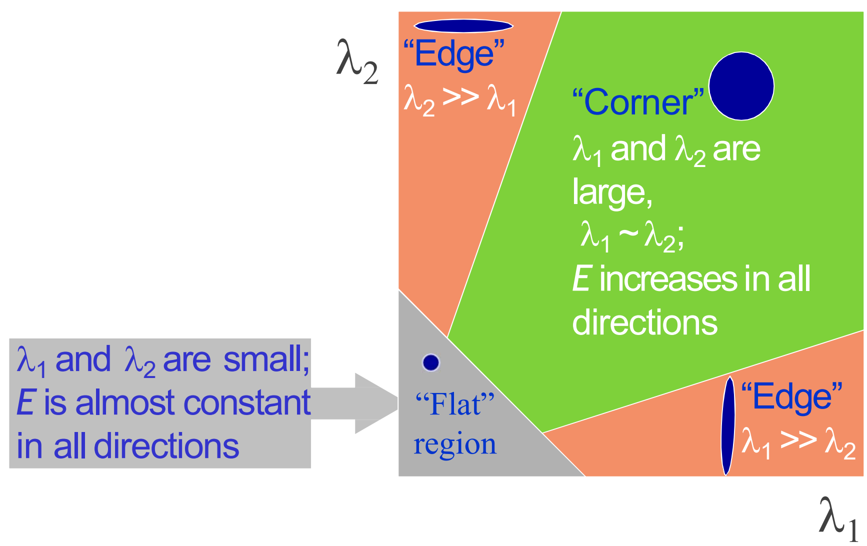
First, consider the axis-aligned case where gradients are either horizontal or vertical

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

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

Interpreting the eigenvalues

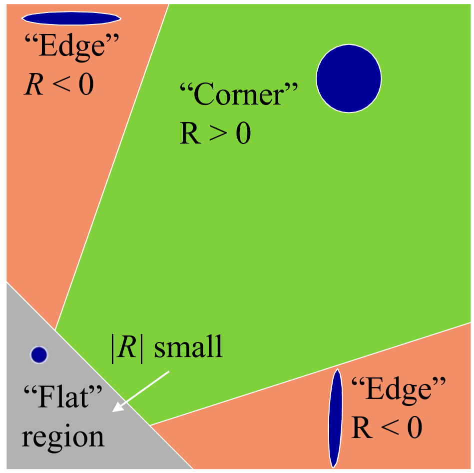
Classification of image points using eigenvalues of *M*:



Harris 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)



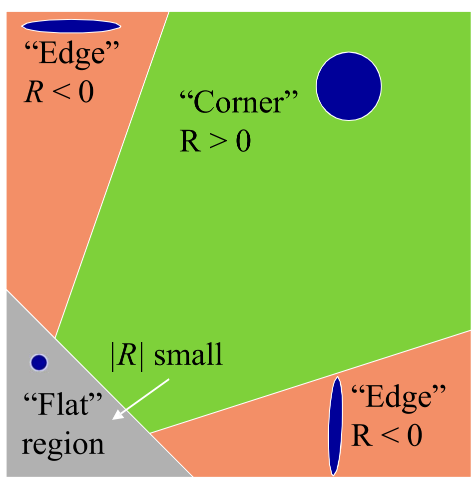
Harris corner response function

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

R is large for a corner

R is negative with large magnitude for an edge

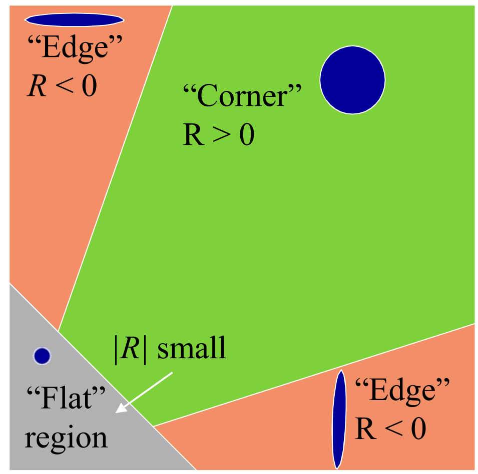
|*R*| is small for a flat region



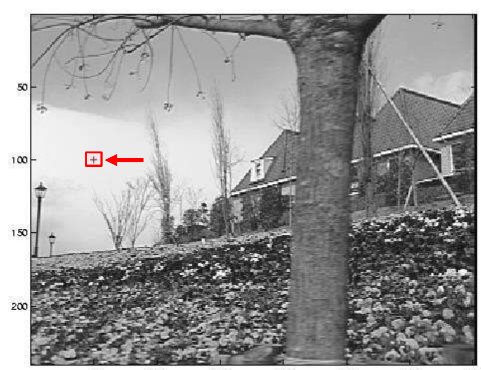
Harris corner response function

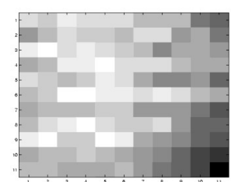
$$R = \det(M) - \alpha \operatorname{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

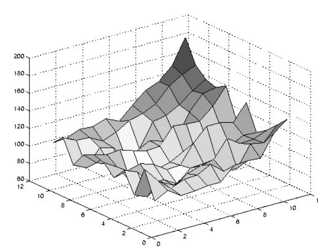
R depends only on eigenvalues of M, but don't compute them (no sqrt, so really fast even in the '80s).



Low texture region





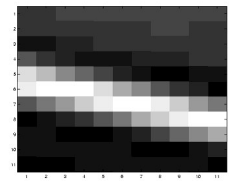


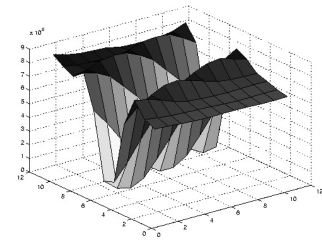
$$M = \sum \nabla I (\nabla I)^T$$

Gradients have small magnitude

=> small λ_1, small λ_2

Edge

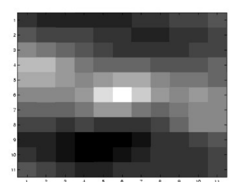


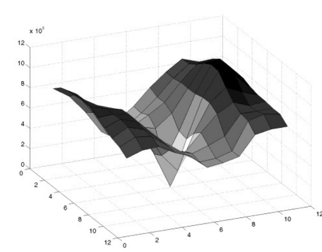


$$M = \sum \nabla I (\nabla I)^T$$

Large gradients, all the same
=> large λ_1, small λ_2

High textured region





$$M = \sum \nabla I (\nabla I)^T$$

Gradients different, large magnitudes
=> large λ_1, large λ_2

Harris detector: Algorithm

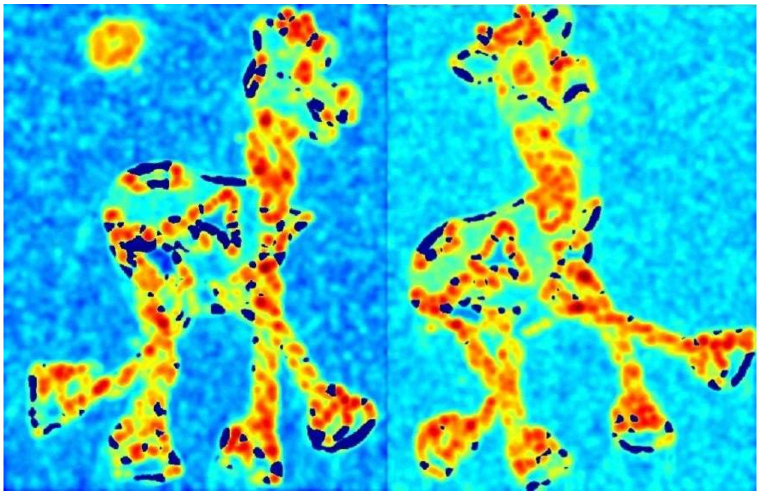
1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix M in a Gaussian window around each pixel
3. Compute corner response function R
4. Threshold R
5. Find local maxima of response function (nonmaximum suppression)

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

Harris Detector: Workflow

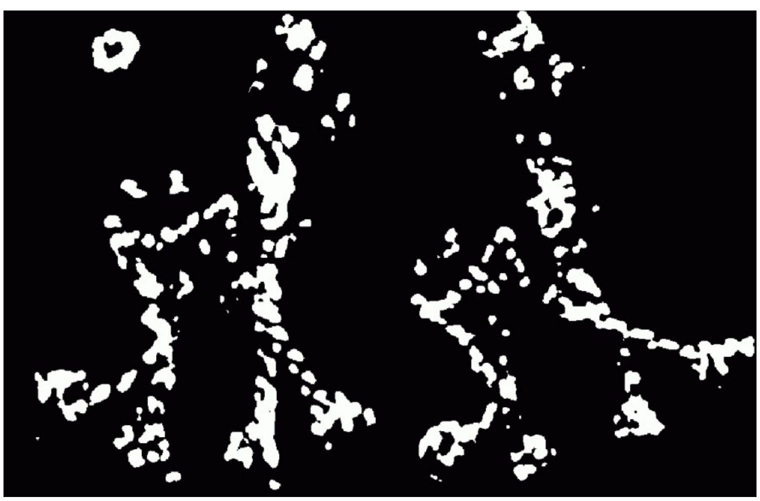
Harris Detector: Workflow

Compute corner response R



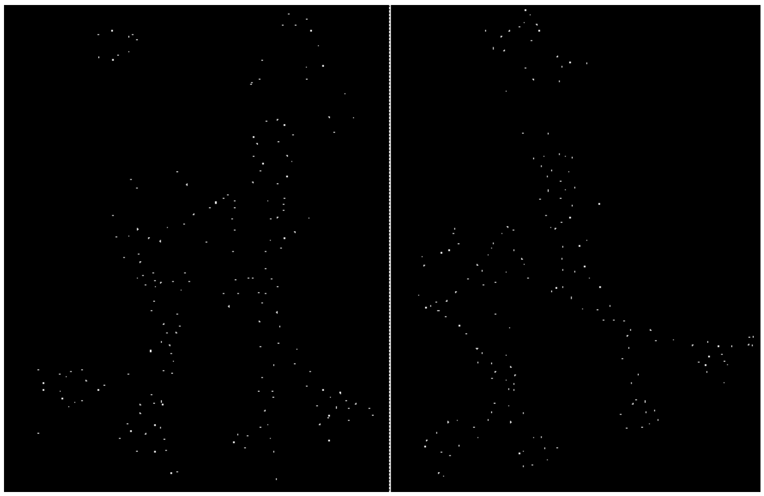
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

Take only the points of local maxima of R



Harris Detector: Workflow

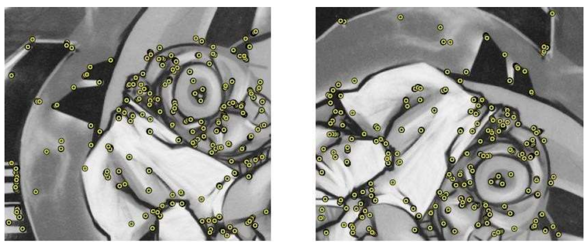
Other corners:

Shi-Tomasi '94:

"Cornerness" = min (λ_1, λ_2) Find local maximums

cvGoodFeaturesToTrack(...)

Reportedly better for region undergoing affine deformations



Other corners:

- Brown, M., Szeliski, R., and Winder, S. (2005):

$$\frac{\det M}{\operatorname{tr} M} = \frac{\lambda_0 \lambda_1}{\lambda_0 + \lambda_1}$$

- There are others…

Scale invariance