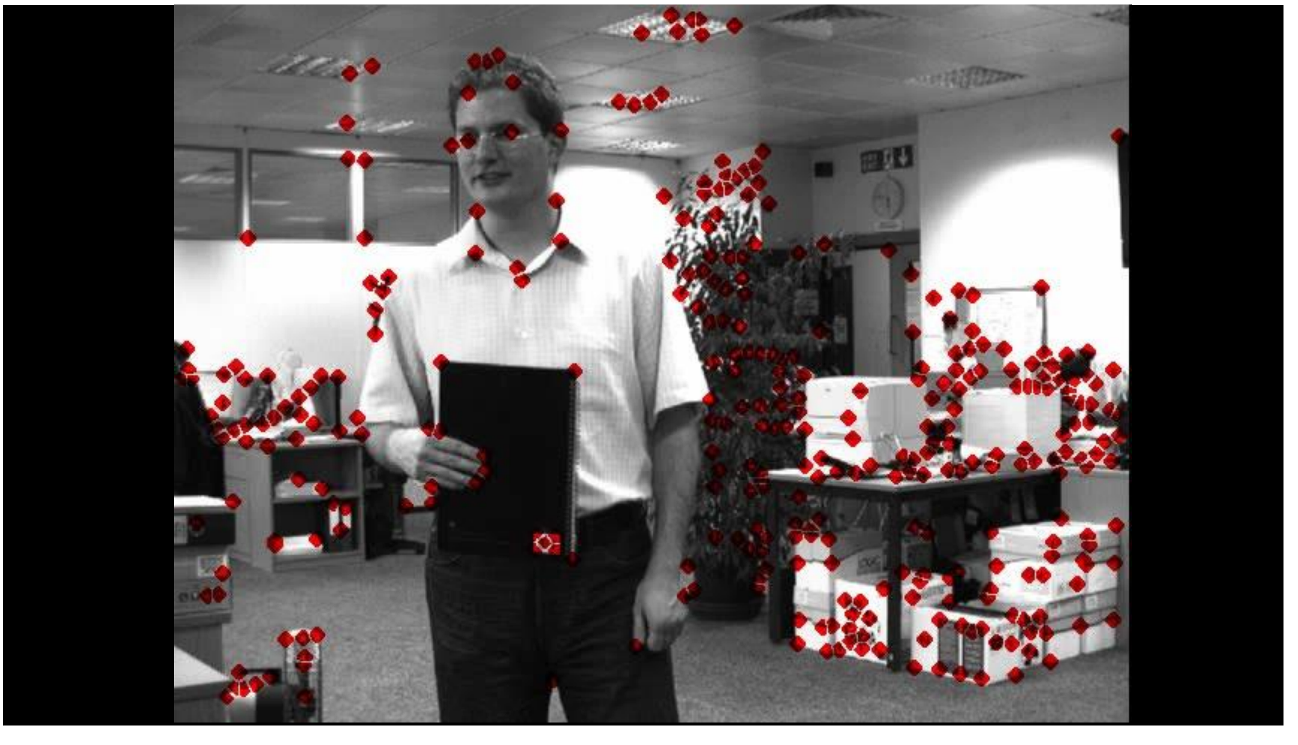
Introduction to Computer Vision

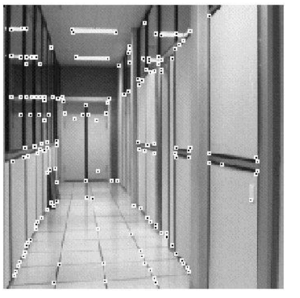
Feature points

Andrea Cavallaro



Applications

- Local features (feature points) are useful for
  - matching
  - tracking of objects
  - robot navigation
  - object recognition
  - pose estimation & camera calibration
  - scene classification
  - texture analysis
  - indexing and mage retrieval
  - video mining
  - augmented reality
  - …



Pair and discuss: how can we estimate the translation?

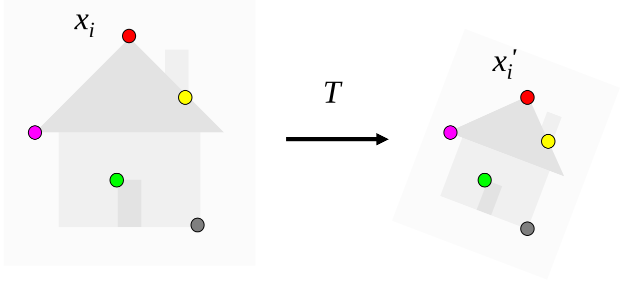
Image A

Image B

recall: geometric transformations

Alignment

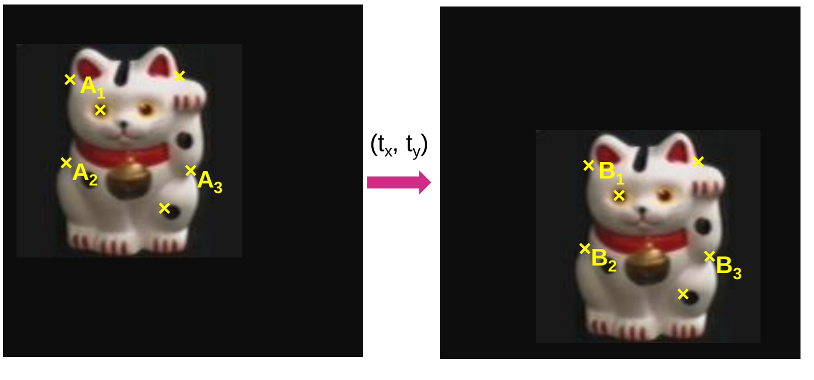
- Find the parameters of the transformation that best align matched points or
- Fit a model to a transformation between pairs of features (*matches*) in 2 images



find transformation T that minimizes

$$\sum_i \mathrm{residual}(T(x_i), x_i')$$

Example: solving for translation



Given matched points in {A} and {B}, estimate the translation of the object

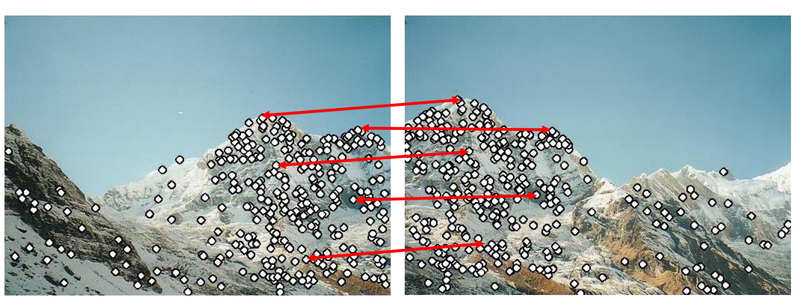
$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Least squares solution

Write down objective function
Derived solution: compute derivative → compute solution
Computational solution: write in form $Ax=b$ → solve using pseudo-inverse $x = A^{-1}b$ or eigenvalue decomposition

Application: panorama stitching

Feature matching



“What stuff in the left image matches with stuff on the right?”

Source: Steve Seitz and Rick Szeliski

Building a panorama

1. Detect feature points in both images
2. Find corresponding pairs
3. Find a parametric transformation (homography)
4. Warp (right image to left image)

$$\forall \text{ matching pair } \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{left} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{right}$$

Feature matching for panorama stitching

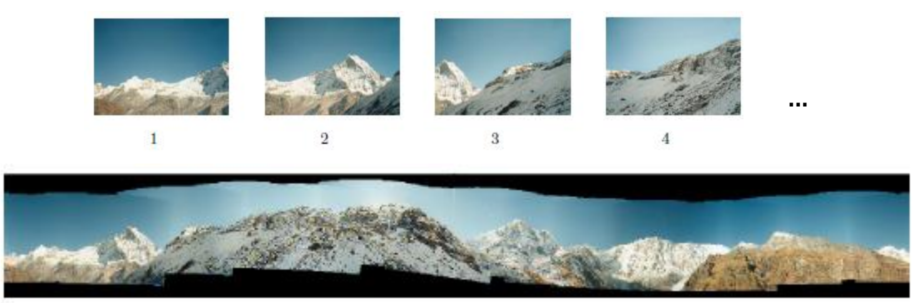
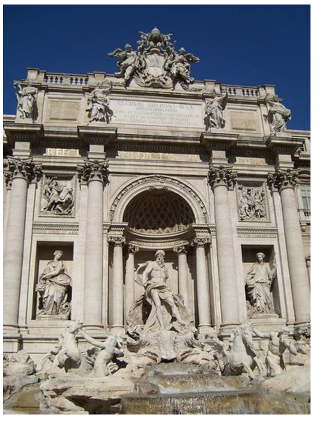


Image matching



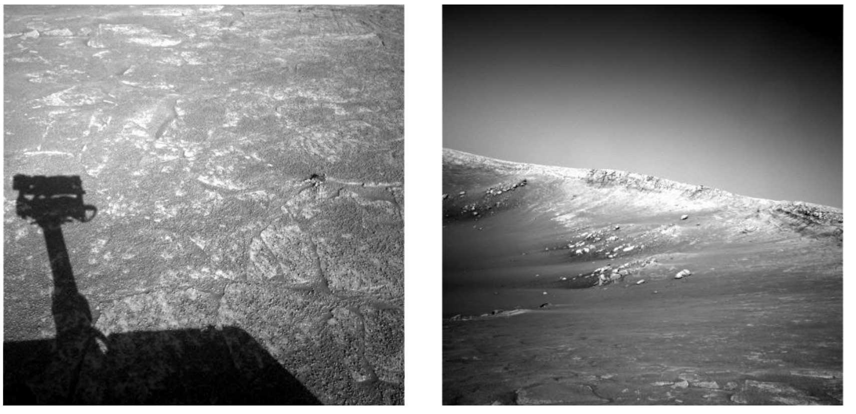
"What stuff in the left image matches with stuff on the right?"

Image matching: harder case

"What stuff in the left image matches with stuff on the right?"

Image matching: even harder case

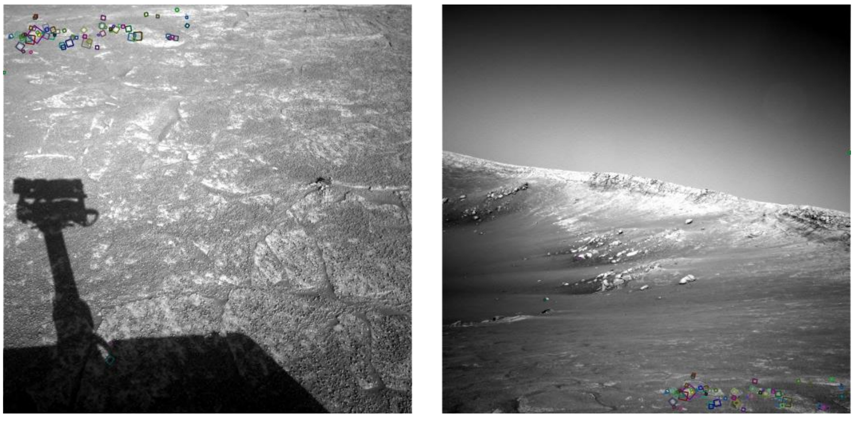
NASA Mars Rover images



"What stuff in the left image matches with stuff on the right?"

Interest point matching

NASA Mars Rover images with SIFT feature matches



Figures by Noah Snavely

Deformations

- Let us compare two images I_1 and I_2
  - I_2 may be a transformed version of I_1
  - What kind of transformations are we likely to encounter?

zoom + rotation

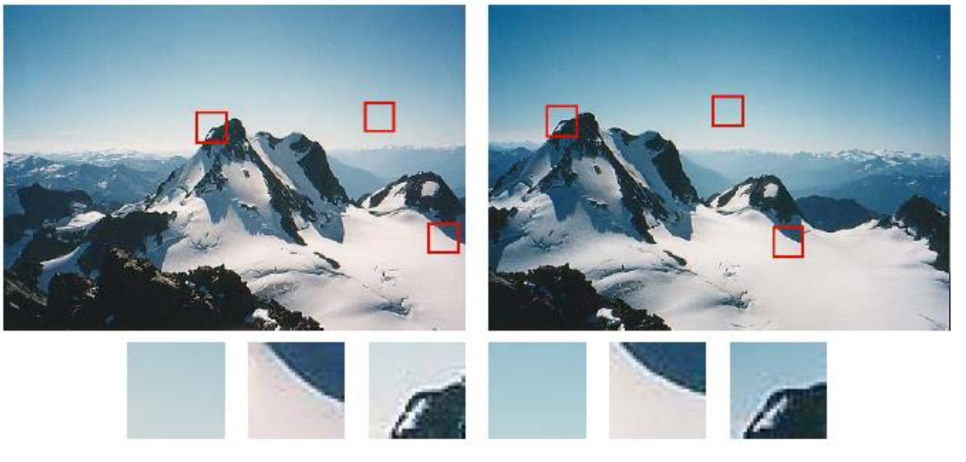
blur

change in view point

change in light

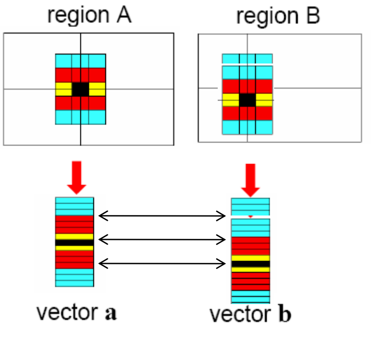
What features to use?

- Two images of the same scene to be aligned by matching features
  - what would be good features considering the transformations we are likely to encounter?

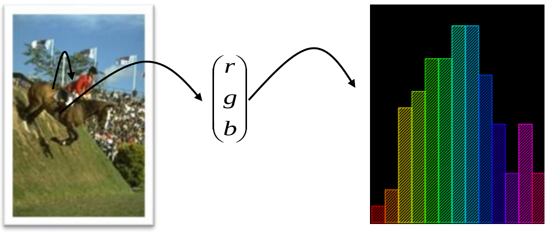


Simplest descriptions of neighbourhood around interest point

- List of intensities to form a feature vector
  - very sensitive to even small shifts, rotations



- Colour histogram
  - may be not distinctive enough (no spatial information)



each pixel is described by a color vector

distribution of these is described by a histogram

Pair and discuss: what makes a feature unique?

- Look for image regions that are unique,
  i.e. that lead to unambiguous matches in other images (uniqueness)

Local features

- Image pattern that differs from its neighbours in terms of
  - intensity
  - colour
  - texture

- Local features can be
  - points/corners
  - small image patches/regions

- Typically, measurements are taken from a region centred on a local feature and converted into descriptors

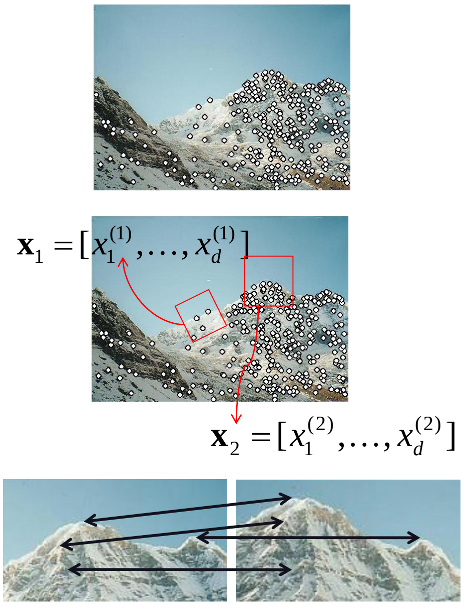
Local features

- Importance of local features for object recognition by the human visual system
  - removing the corners from images impedes human recognition, while
  - removing most of the straight edge information does not

Source: T. Tuytelaars & K. Mikolajczyk

Local features: main components

- Detection
  - identify the interest points
- Description
  - extract vector feature descriptor surrounding each interest point
- Matching
  - Determine correspondence between descriptors in two views



Source: Kristen Grauman

Properties of good features

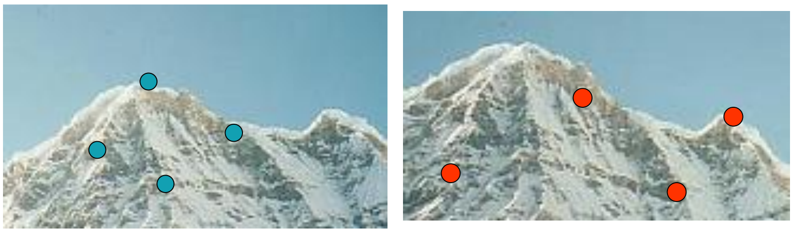
- Repeatability
  - given two images of the same object or scene, taken under different viewing conditions, a high percentage of the features detected on the scene part visible in both images should be found in both images
- Distinctiveness
  - the intensity patterns underlying the detected features should show sufficient variation so that features can be distinguished and matched
- Locality
  - the features should be local, so as to reduce the probability of occlusion and to allow simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions

Properties of good features

- Quantity
  - the number of detected features should be sufficiently large such that a reasonable number of features are detected even on small objects
- Accuracy
  - the detected features should be accurately localized, in image location and with respect to scale
- Efficiency
  - the detection of features in a new image should allow for time-critical applications

Interest operator repeatability

- Ability to detect (at least some of) the same points in both images

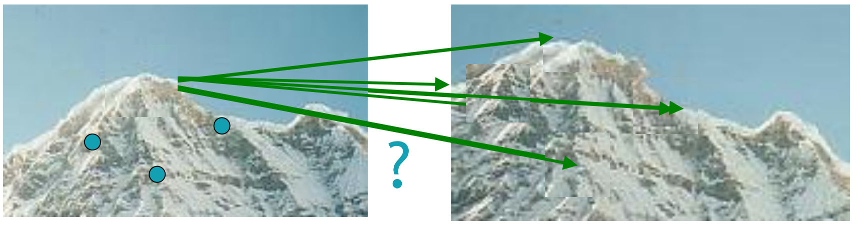


No chance to find true matches!

- Yet ability to run the detection procedure independently per image

Descriptor distinctiveness

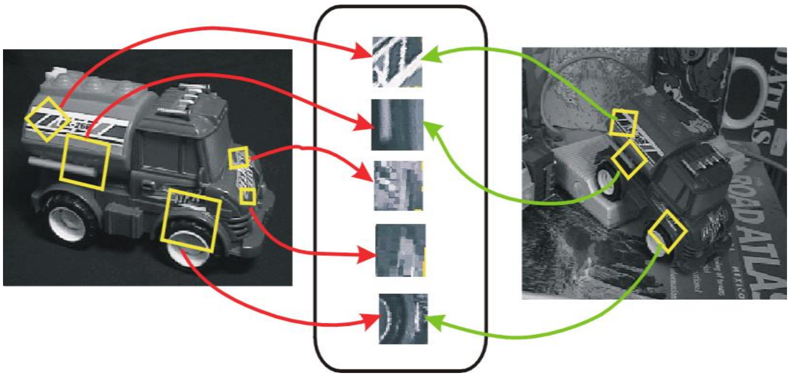
- Ability to reliably determine which point goes with which



- Must provide some invariance to geometric and photometric differences between the two views

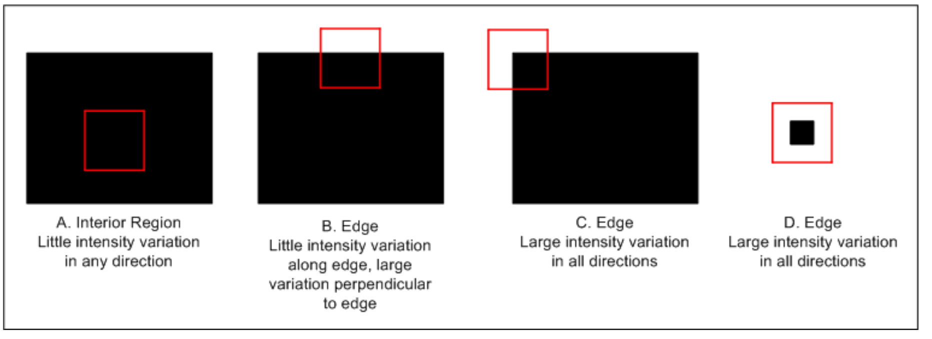
Local invariant features

- Finding points and representing their patches should produce similar results even when conditions vary
  - photometric invariance: brightness, exposure, …
  - geometric invariance: translation, rotation, scale, …



Local measures of uniqueness

- Suppose we only consider a small window of pixels
  - what defines whether a feature is a good or bad candidate?



Corners

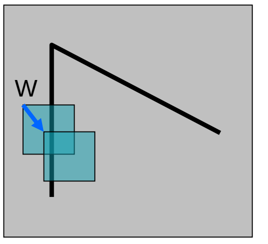
- Corner
  - where two edges come together
  - where the image gradient has significant components in the x and y direction
  - can establish corners from the gradient rather than the edge images
  - corresponds to point in both the world and image spaces

- Point features / interest points / corners / feature points

- Variance-based interest operators
  - to select image points
  - interest (corner) points
  - measure: usually the variance of the intensity values on a small window
  - the simplest is Moravec

Variance-based interest point detection

- When shifting the window W by (u,v)
  - how do the pixels in W change?
  - compare each pixel before and after the shift by summing the squared differences (SSD)



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

shifted intensity

intensity

Moravec interest point detector

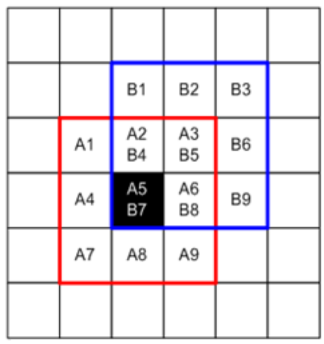
- Defines interest points as
  - points where there is a large intensity variation in every direction

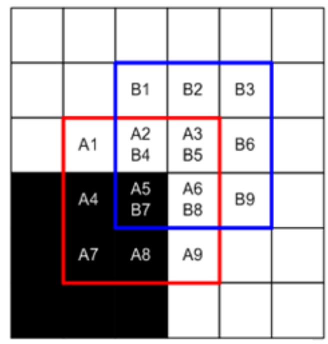
- Measures the intensity variation by
  - placing a small square window (e.g. 3x3, 5x5, 7x7) centred at a given point
  - shifting this window by one pixel in each of the eight principle directions
    - horizontally, vertically, and four diagonals
  - intensity variation for a given shift is calculated by taking the sum of squares of intensity differences of corresponding pixels in these two windows

Moravec detector: example

- Calculate intensity variation for 3x3 window in the upper right diagonal direction
  - for an isolated black pixel on a white background
  - and on an ideal corner

$$E = \sum_{i=1}^{9} (A_i - B_i)^2 = 2x255^2 \qquad E = \sum_{i=1}^{9} (A_i - B_i)^2 = 3x255^2$$





Moravec interest point detector

- Moravec operator
  - gives a measure of cornerness to each pixel in the image
  - this measure is the minimum intensity value found over the eight shift directions
- Moravec operator applied to each pixel in an image creates a cornerness map
  - corners are the local maxima in the cornerness map

X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
X	X	0	0	0	0	0	0	0	0	0	1	1	1	X	X
X	X	0	0	0	0	0	1	1	0	0	1	2	1	X	X
X	X	0	0	0	0	0	2	1	0	0	1	1	1	X	X
X	X	0	0	0	0	0	0	0	0	0	0	0	0	X	X
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

cornerness map for an image when a 3x3 window is used

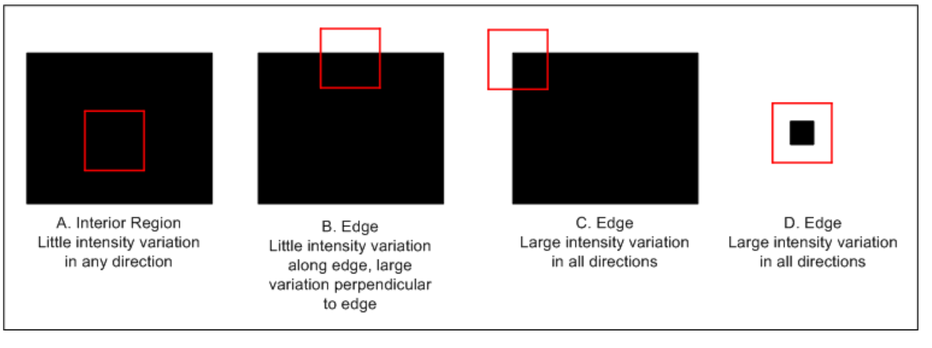
Moravec interest point detector

- Computationally efficient

- Sensitive to noise
  - many local maxima do not correspond to corners
  - isolated pixels can be detected as corners

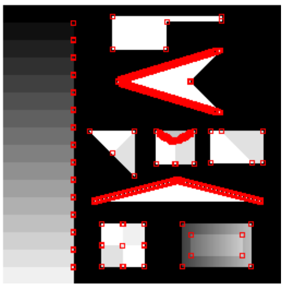
- Using a larger window size
  - makes the detector more robust
  - true corner: larger intensity variation than the isolated pixel
  - setting all cornerness values below a threshold to zero helps
  - choosing this threshold is difficult
    - must be set high ‘enough’ to avoid these false corners
    - but low ‘enough’ to retain as many true corners as possible

Moravec interest point detector

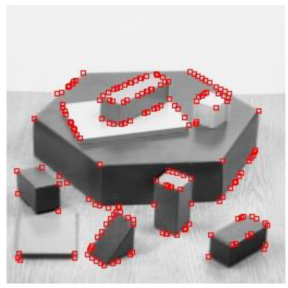


Four types of window positions that were considered by the Moravec operator

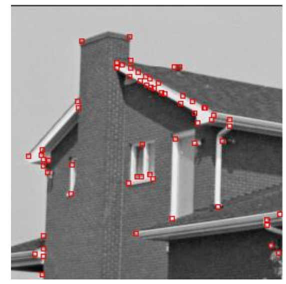
Moravec operator with a 3x3 window: examples



threshold near zero



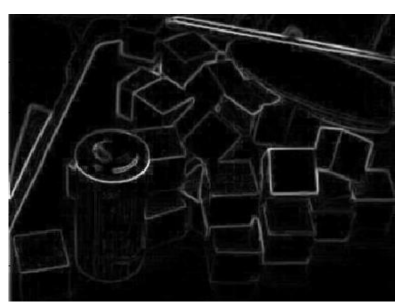
threshold chosen to detect most of the corners while minimizing the number of false corners detected



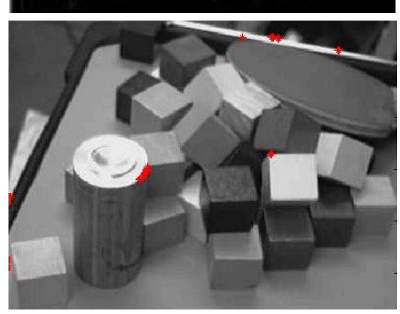
threshold chosen high to avoid detecting corners due to the texture of the house

Examples: detected interest points with different thresholds

original image



Moravec operator applied



Example: high & low threshold

original

Moravec interest points
for a low threshold

Example: high & low threshold

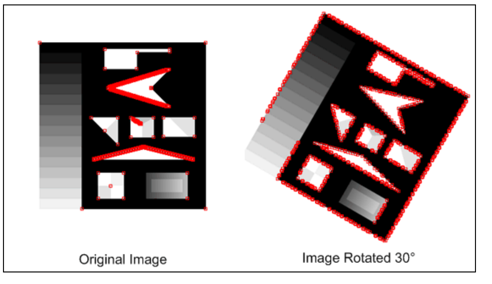
original

Moravec interest points
for a high threshold

Moravec interest point detector: limitations

- Noisy response
  - estimation of the local intensity variation can be improved with a Gaussian window, i.e. by placing more weight on the intensity differences of pixel pairs near the centre of the window (the difference of pixel pairs near the centre is a better indication of the local variance)

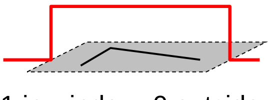
- Poor repeatability due to the anisotropic response of operator
  - intensity variation calculated only at a discrete set of shifts in the eight principle directions
  - the operator is not rotationally invariant



Variance-based interest point detection

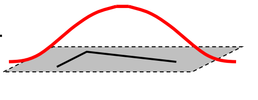
$$E(u,v)=\sum_{x,y}\underbrace{w(x,y)}_{\text{window function}}\left[\underbrace{I(x+u,y+v)}_{\text{shifted intensity}}-\underbrace{I(x,y)}_{\text{intensity}}\right]^2$$

$w(x,y)$ =



1 in window, 0 outside

or

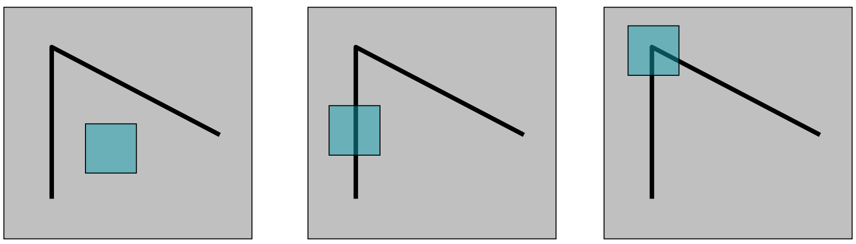


Gaussian

Source: R. Szeliski

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

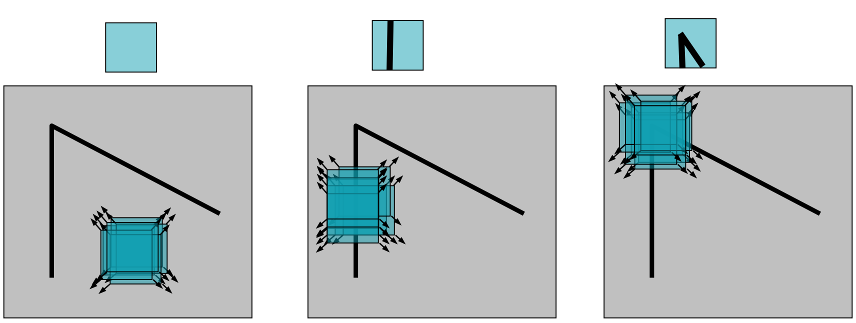
- Let us only consider a small window



Source: Darya Frolova, Denis Simakov, Weizmann Institute.

Local measures of uniqueness

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



flat region:
no change in any directions

edge:
no change along the edge direction

corner:
significant change in all directions

Source: Darya Frolova, Denis Simakov, Weizmann Institute.

Interest point detection using derivatives

- Taking the sum of differences between corresponding pixels in the two windows is an approximation of the image gradient
  - For an arbitrary shift *(u,v)* we can state the intensity variation as:

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

  - 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

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

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

Interest point detection using derivatives

- Plugging this into the original formula:

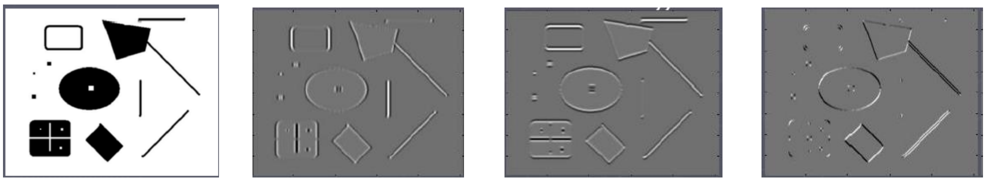
$$\begin{aligned} E(u,v) &= \sum_{(x,y)\in W} [I(x+u,y+v) - I(x,y)]^2 \\ &\approx \sum_{(x,y)\in W} \left[I(x,y) + [I_x \ I_y] \begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)\right]^2 \\ &\approx \sum_{(x,y)\in W} \left[[I_x \ I_y] \begin{bmatrix} u \\ v \end{bmatrix}\right]^2 \end{aligned}$$

- This can be re-written as:

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

Interest point detection using derivatives

- M is 2 x 2 matrix of image derivatives
  - averaged in neighborhood of a point
  - depends on image properties



$$I_x \Leftrightarrow \frac{\partial I}{\partial x} \qquad I_y \Leftrightarrow \frac{\partial I}{\partial y} \qquad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$$

Interest point detection using derivatives

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

- M contains all the differential operators describing the geometry of the image surface at a given point *(x,y)*
- We want E to be large for small shifts in all directions
- Eigenvalues and eigenvectors of M
  - define shifts with the smallest and largest change (E value)
  - will be proportional to the principle curvatures of the image surface and form a rotationally invariant description of M

Quick eigenvalue/eigenvector review

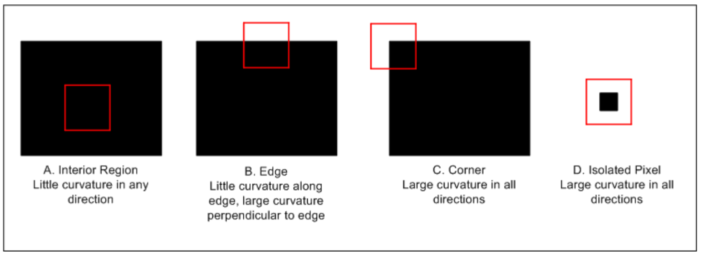
The eigenvectors of a matrix A are the vectors x that satisfy: $Ax = \lambda x$

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

- the eigenvalues are found by solving: $det(A - \lambda I) = 0$
- in our case, $A = M$ is a 2x2 matrix, so $\det\begin{bmatrix} m_{11}-\lambda & m_{12} \\ m_{21} & m_{22}-\lambda \end{bmatrix} = 0$
- the solution: $\lambda_{\pm} = \frac{1}{2}\left[(m_{11}+m_{22}) \pm \sqrt{4m_{12}m_{21}+(m_{11}-m_{22})^2}\right]$

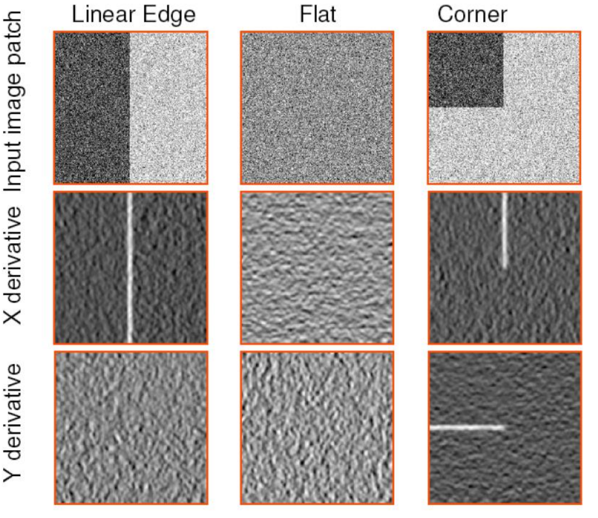
Once you know λ, find x by solving $\det\begin{bmatrix} m_{11}-\lambda & m_{12} \\ m_{21} & m_{22}-\lambda \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = 0$

Interpreting the eigenvalues



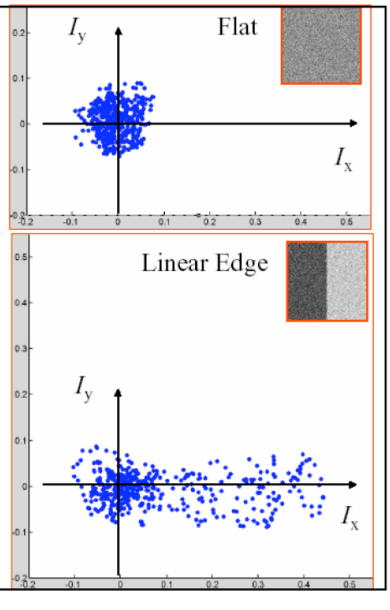
- In position A
  - image intensity will be relatively constant within the window
  - little curvature in the surface within this window
  - both eigenvalues will be relatively small
- In position B
  - significant curvature perpendicular to the edge and little curvature along the edge
  - one of the eigenvalues will be large and the other small
- Both positions C & D have significant curvature in both directions
  - both eigenvalues will be large

Interpreting the image derivatives

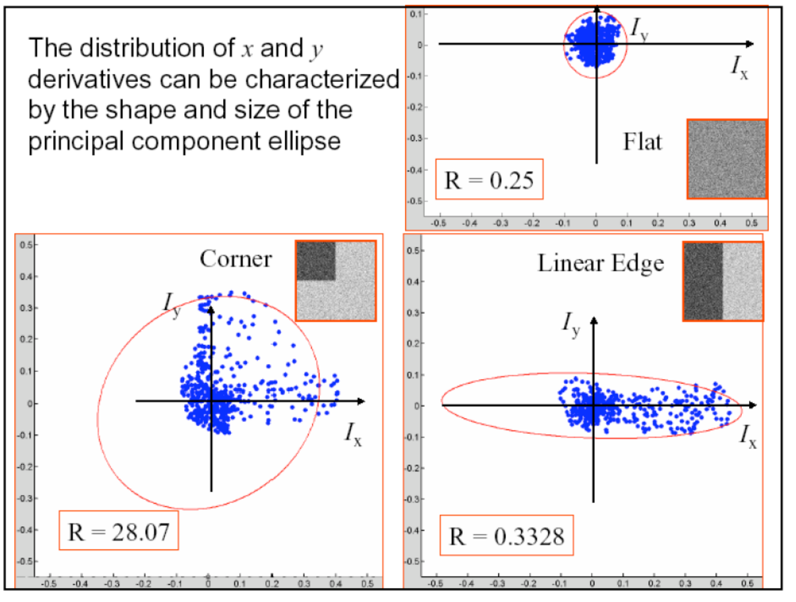


Interpreting the image derivatives

The distribution of the x and y derivatives is very different for all three types of patches

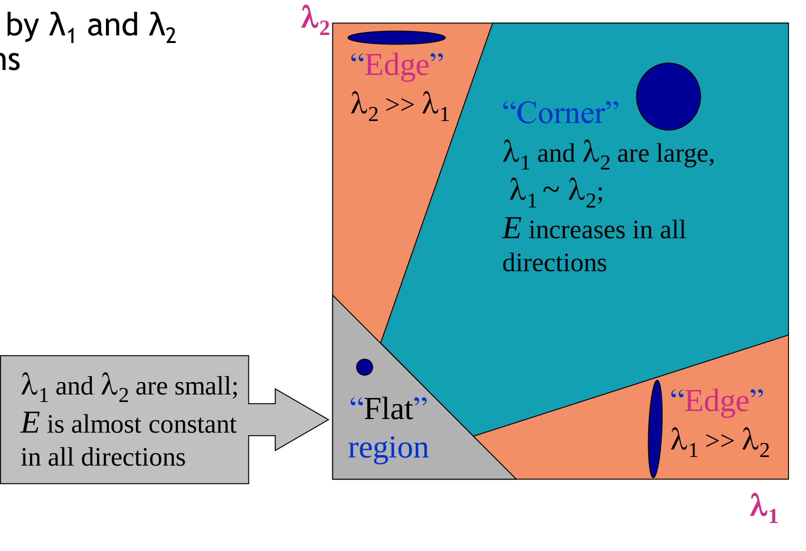


Interpreting the image derivatives



Interpreting the eigenvalues

- Let λ_1 and λ_2 be the eigenvalues of M
- The plane described by λ_1 and λ_2 is divided into regions



Harris corner detector

- Cornerness of a point is calculated as

$$R(x_0, y_0) = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α scale constant (0.04 to 0.06)

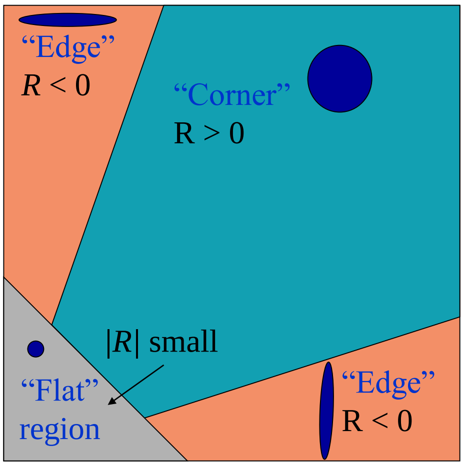
- Computed using two approximations

$$R(x_0, y_0) = \det(M) - \alpha\, \mathrm{trace}(M)^2$$

M : 2×2 matrix computed from image derivatives

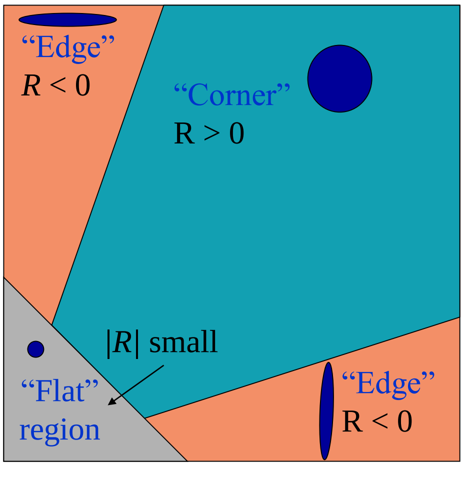
$$M(x_0, y_0) = \sum_{x,y \in \mathrm{neighborhood}(x_0, y_0)} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

weight function



Harris corner detector

- R depends only on eigenvalues of M
- Corner: R is large
- Edge: R is negative, with large magnitude
- Flat region: |R| is small



Harris corner detector

- Based on the second moment matrix
  - used for describing local image structures
  - the matrix describes the gradient distribution in a local neighbourhood of a point
  - the eigenvalues of this matrix represent the principal signal changes in two orthogonal directions in a neighbourhood around the point
- Corners
  - locations in the image for which the image signal varies significantly in both directions (i.e. for which both eigenvalues are large)
- Cornerness measure
  - combines the two eigenvalues in a single measure
  - is computationally less expensive

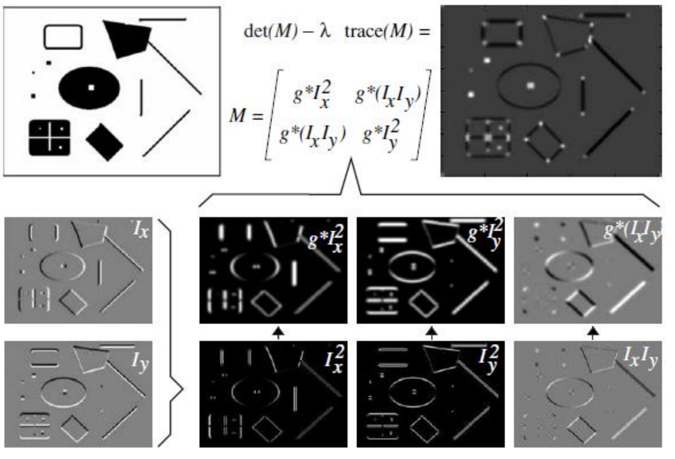
Harris corner detector

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

α: (0.04 to 0.06)

- Since the determinant of a matrix is equal to the product of its eigenvalues and the trace corresponds to the sum
  - high values of the cornerness measure correspond to both eigenvalues being large
  - adding the second term with the trace reduces the response of the operator on strong straight contours
  - computing this measure based on the determinant and the trace is computationally less demanding than computing the eigenvalues
- Thresholding is still required to distinguish between corners and other points

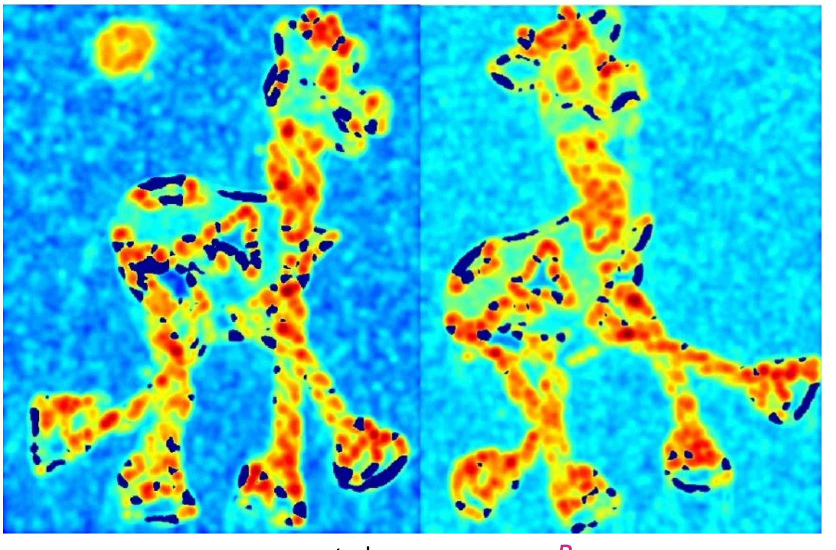
Harris detector: example



Components of the second moment matrix and Harris cornerness measure

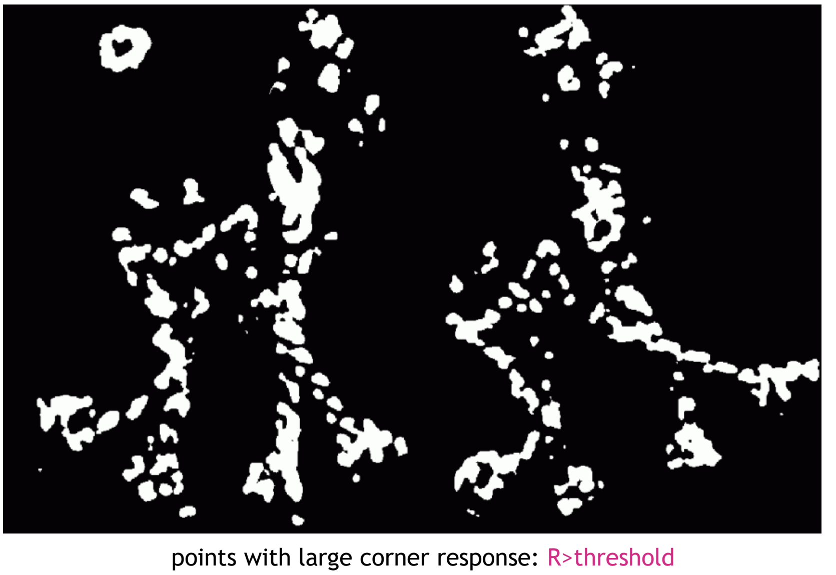
Harris corner detector: example

Harris corner detector: example



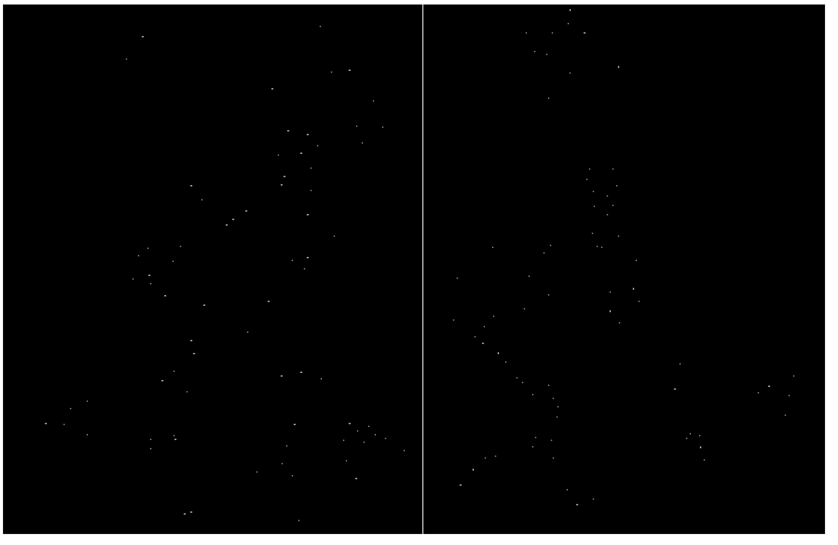
computed corner response R

Harris corner detector: example



points with large corner response: R>threshold

Harris corner detector: example

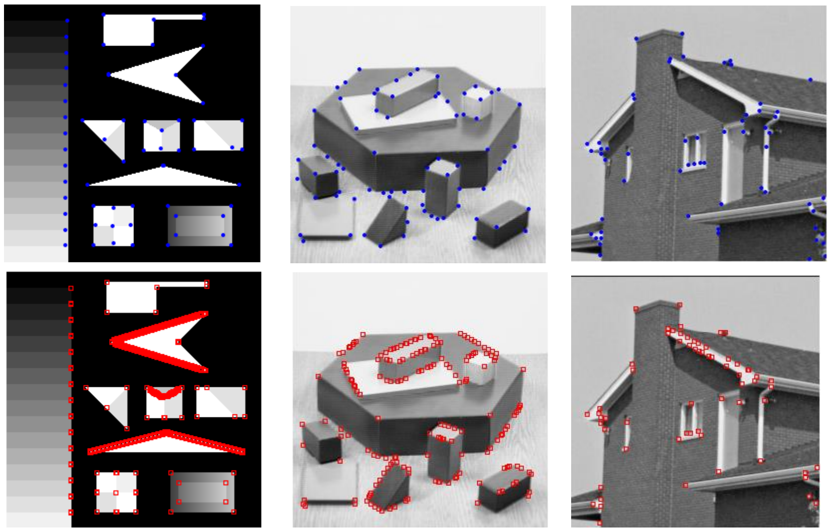


only the points of local maxima of R

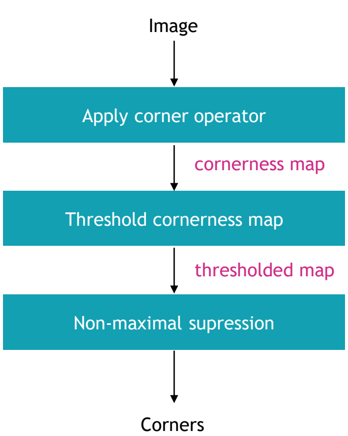
Harris corner detector: example

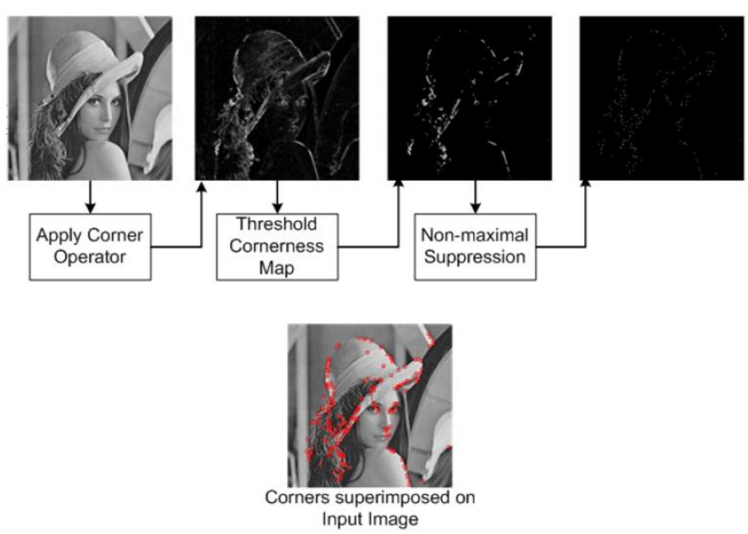
detected corners

Harris vs. Moravec



Corner detectors: summary



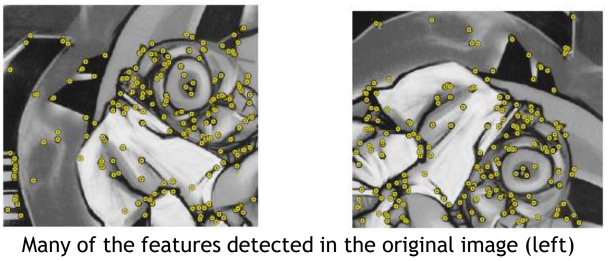


Corner detection: invariance

- Will we still pick up the same features if we change the brightness of the image / rotate the image by some angle / scale the image?

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

Harris corner detection: invariance

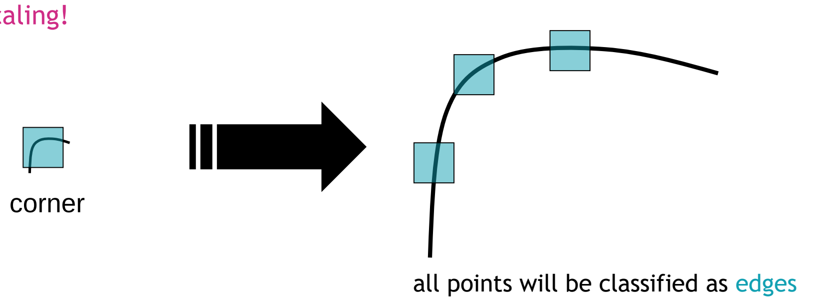


Many of the features detected in the original image (left) have also been found in the rotated version (right)

- The repeatability of the Harris detector under rotations is high
- Features are typically found at locations which are informative (i.e. with a high variability in the intensity pattern)

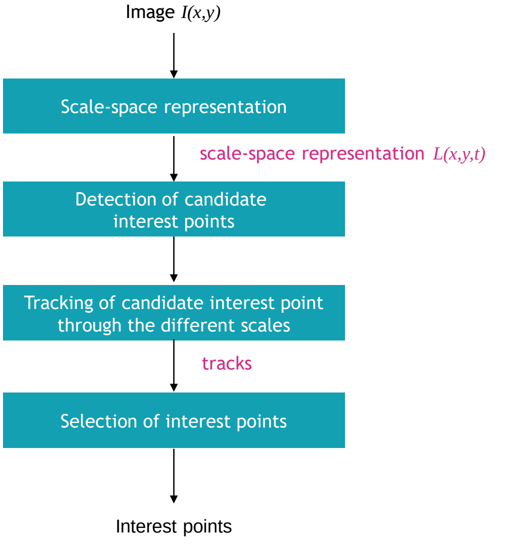
Harris corner detection: invariance

- Corner locations are
  - partially invariant to intensity change
  - covariant w.r.t. translation
  - covariant w.r.t. rotation
  - not covariant to scaling!



Solution: scale-space approach

Scale-space approach



Scale-space approach

- Scale-space representation
  - parametric set of images gradually smoothed
  - the most appropriate scale for extracting the interest points is not known

Original | Scale t = 5 | Scale t = 10 | Scale t = 30

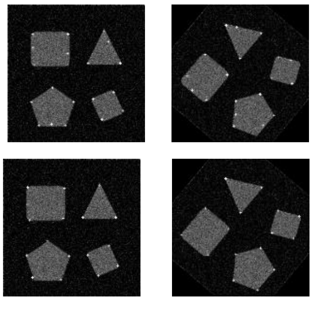
- Criterion for the selection of interest points: life-span of a track
  - the life-span of points due to noise is short
  - the life-span of interest points is long

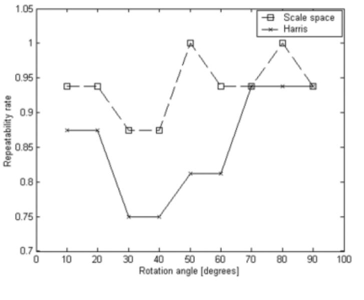
Evaluation: Harris vs. scale-space

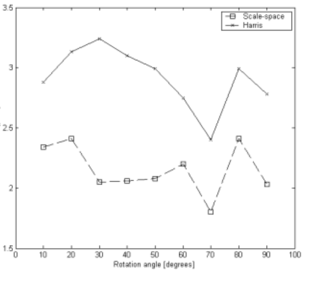
- Repeatability
  - Stability of interest points with viewpoint changes and/or geometric transformations
  - Repeatability = (# of interest points repeated) / (total # of interest points)
- Accuracy
  - High accuracy in localisation
  - Measure of the precision of estimated interest point compared to the 'real' interest point
  - Accuracy → Root Mean Square Error

Harris

scale-space







Scale Invariant Feature Transform (SIFT)

- Extracts highly distinctive features that are
  - invariant to rotation and scaling
  - partially invariant to changes in illumination
  - partially invariant to affine transformations

- SIFT: main steps
  - scale-space extrema detection
    - extract scale and rotation invariant interest points (i.e. keypoints)
  - keypoint localization
    - determine location and scale for each interest point
  - orientation assignment
    - assign one or more orientations to each keypoint
  - keypoint descriptor
    - use local image gradients at the selected scale

Scale invariant detection

- Functions (kernels) for determining scale

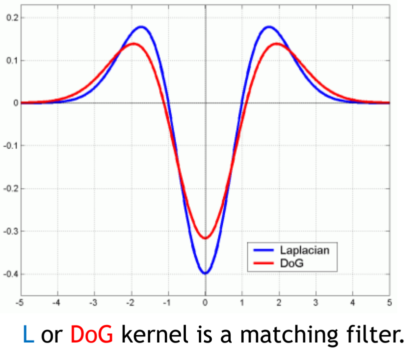
$$L = \sigma^2 \left(G_{xx}(x,y,\sigma) + G_{yy}(x,y,\sigma) \right)$$

Laplacian

$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

Difference of Gaussians

$$G(x,y,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



L or DoG kernel is a matching filter.
It finds blob-like structure.
It is also successful in getting characteristic scale of other structures, e.g. corners.

$$f = \text{Kernel} * \text{Image}$$

Keypoint detection

- Image: convolved with Gaussian filters at different scales

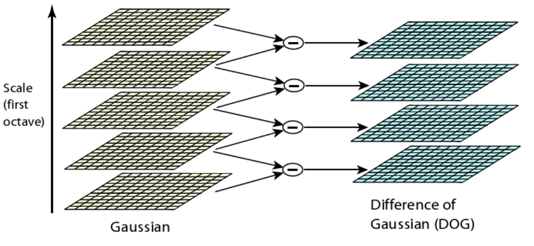
$$L(x, y, k\sigma) = G(x, y, k\sigma) * I(x, y) \qquad k\sigma: \text{scale (blur)}$$

- Difference of successive Gaussian-blurred images are taken (DoG)
- DoG image: $D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma)$

$G(k^2\sigma) * I$

$G(k\sigma) * I$

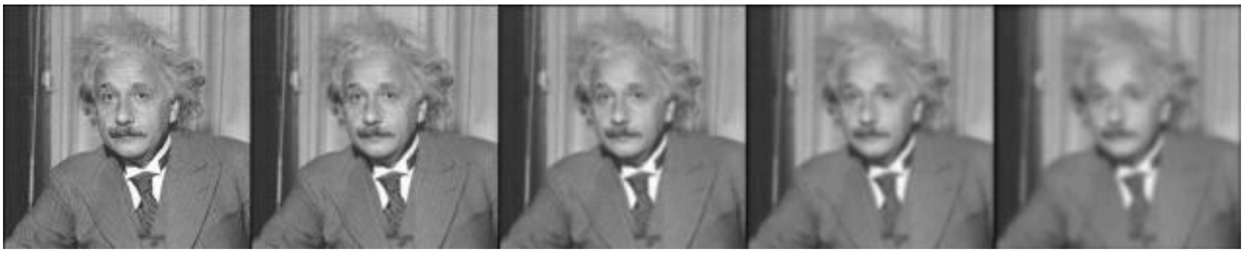
$G(\sigma) * I$



$$D(\sigma) \equiv (G(k\sigma) - G(\sigma)) * I$$

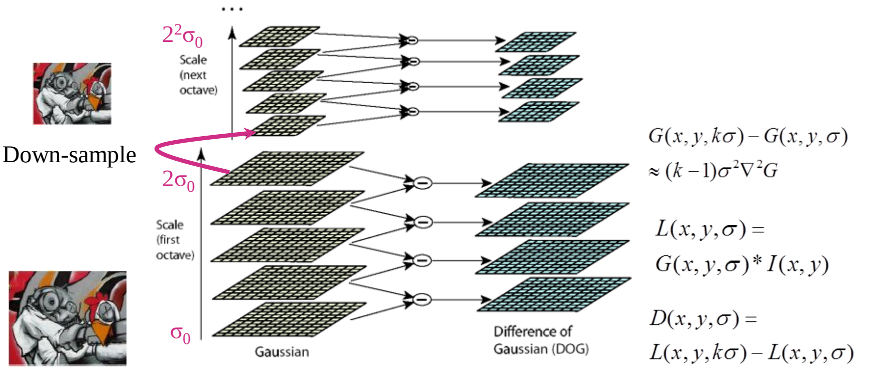
- Keypoints are maxima/minima of the DoG that occur at multiple scales

Difference of Gaussians: example



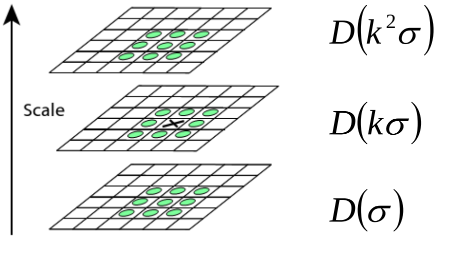
Scale-space extrema detection

- DoG images are grouped by octaves
  - octave: corresponds to doubling the value of σ
  - fixed number of scales (i.e. levels) per octave



Keypoint detection

- Keypoints: local minima/maxima of the DoG images across scales
  - choose all extrema in DoG pyramid within 3x3x3 neighborhood
  - compare each pixel in the DoG images to its eight neighbors at the same scale and nine corresponding neighboring pixels in each of the neighboring scales
    - comparisons with the nearest 26 neighbours in a discretized scale-space volume *(x, y, scale)*
  - if pixel value is the max/min among all compared pixels → selected as candidate keypoint



Keypoint localisation

- Scale-space extrema detection
  - produces too many keypoint candidates & some are unstable
    → discard low contrast keypoints & those located on edges
- Next step: accurate location, scale, and ratio of principal curvatures
  - determine the location & scale of keypoints to sub-pixel & sub-scale accuracy by fitting a 3D quadratic function at each keypoint

$$X_i = (x_i, y_i, \sigma_i) \rightarrow X_i + \Delta X$$

  - compute its maxima → yields a non integer position (in x,y)

Recall

- Harris interest point filtering uses the 2nd order moment matrix to reject points lying on edges

$$M(x_0, y_0) = \sum_{x,y \in \text{neighborhood}(x_0,y_0)} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

$$R(M) = det(M) - \alpha\, trace^2(M)$$

or

$$R(M) = \lambda_1 \lambda_2 - \alpha\, (\lambda_1 + \lambda_2)^2$$

Keypoint filtering

- Reject points with bad contrast
  - reject keypoint if $|D(X_i + \Delta X)| < 0.03$
  - assumes that image values have been normalized in [0,1]
- Laplacian gives strong response on edges → additional filtering is needed
  - eigenvalues of the Hessian matrix are computed & their strengths evaluated
- Reject edges
  - SIFT uses the Hessian matrix for efficiency $\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$

α: largest eigenvalue (λ_{max}); β: smallest eigenvalue (λ_{min}) → proportional to principal curvatures

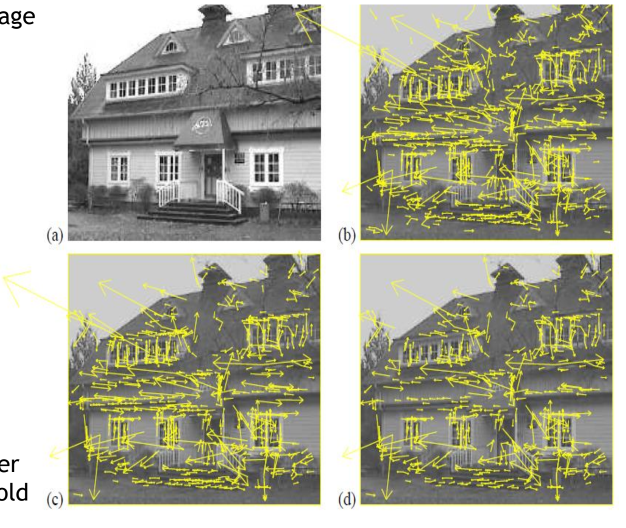
$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$
$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

→ $$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r},$$

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

$r = \alpha/\beta$
SIFT uses $r = 10$

Keypoint localisation: example



233x189 image

832 DoG extrema

729 extrema left after low-contrast threshold

536 left after testing ratio based on Hessian

Orientation assignment

- By assigning a consistent orientation, the descriptor can be orientation invariant
- Create histogram of gradient directions, within a region around the keypoint, at selected scale
- Let, for a keypoint, L be the Gaussian image with the closest scale
  - compute gradient magnitude and orientation using finite differences:

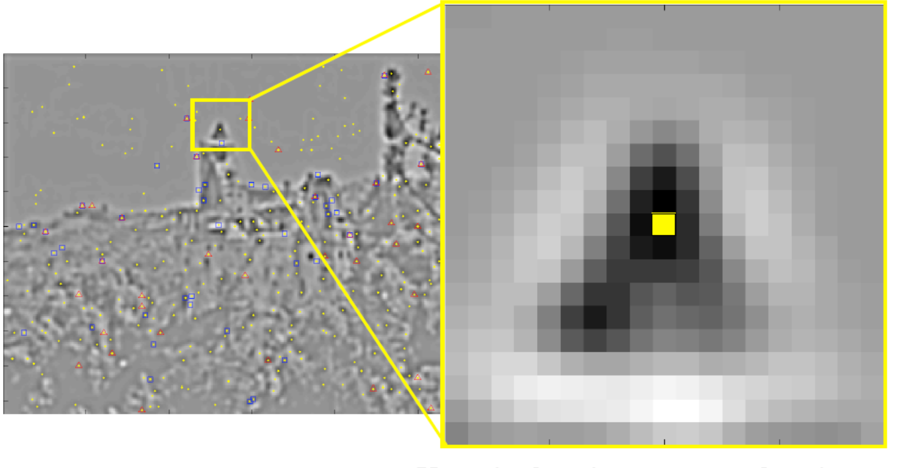
$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \qquad GradientVector = \begin{bmatrix} L(x+1,y) - L(x-1,y) \\ L(x,y+1) - L(x,y-1) \end{bmatrix}$$

$$m(x,y) = \sqrt{(L(x+1,y)-L(x-1,y))^2 + (L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y) = a\tan 2((L(x,y+1)-L(x,y-1))/(L(x+1,y)-L(x-1,y)))$$

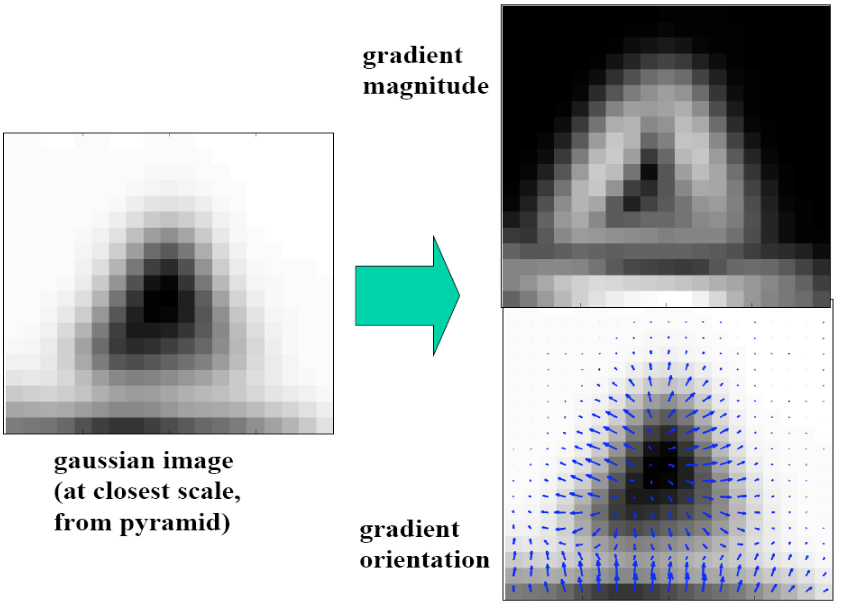
- Histogram entries are weighted by
  - gradient magnitude and
  - a Gaussian function with σ = 1.5 x (scale of the keypoint)

Orientation assignment

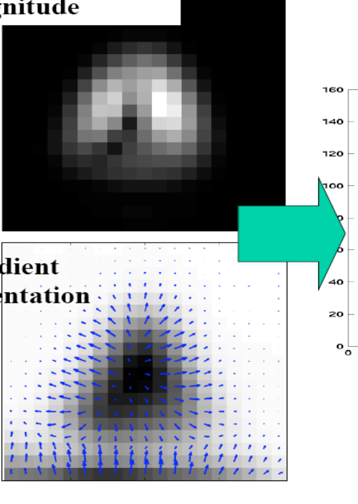


•Keypoint location = extrema location
•Keypoint scale is scale of the DOG image

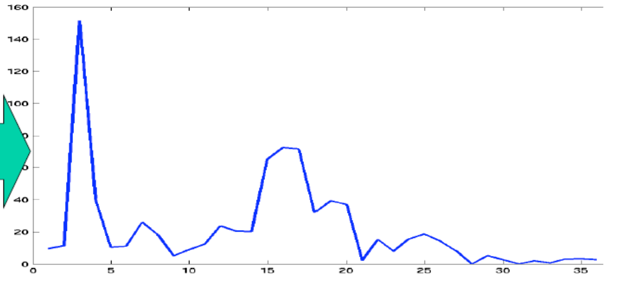
Orientation assignment



Orientation assignment

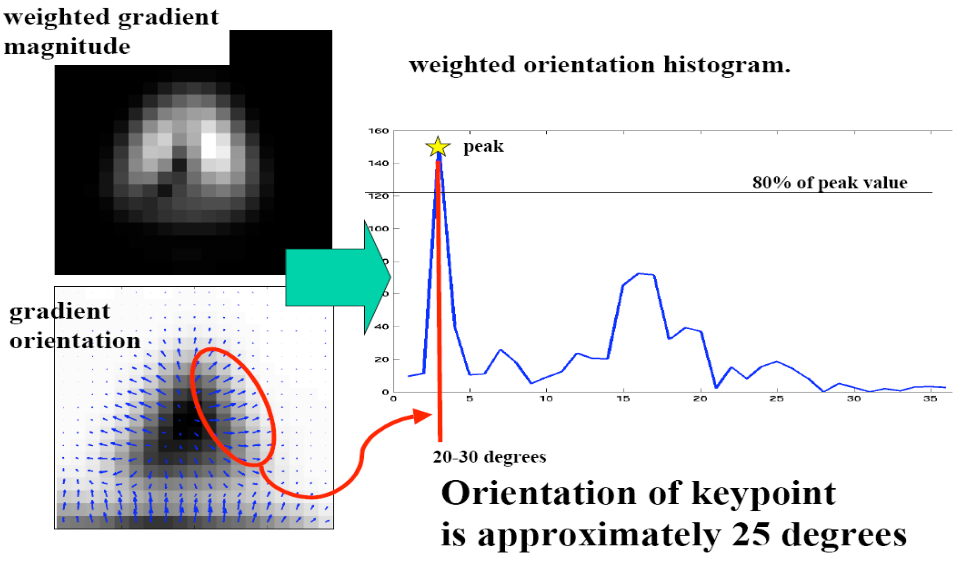


weighted orientation histogram.
Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.



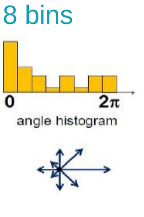
36 buckets
10 degree range of angles in each bucket, i.e.
0 <=ang<10 : bucket 1
10<=ang<20 : bucket 2
20<=ang<30 : bucket 3 …

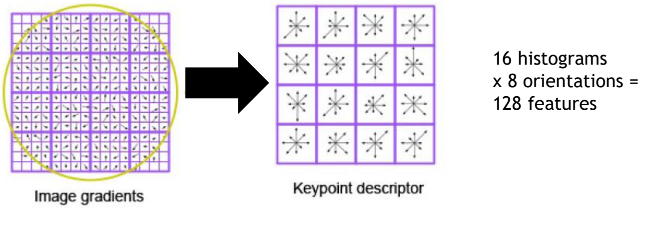
Orientation assignment



SIFT descriptor

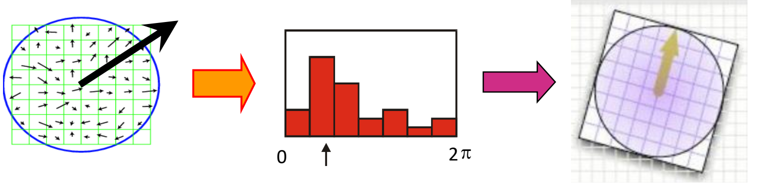
- Take a 16 x16 window around the detected interest point
- Divide into a 4x4 grid of cells
- Compute histogram in each cell (4x4 samples) in 8 directions
  - Gaussian weighting around center, with σ = 0.5 x (that of the scale of a keypoint)
  - 4x4x8 = 128 dimensional feature vector





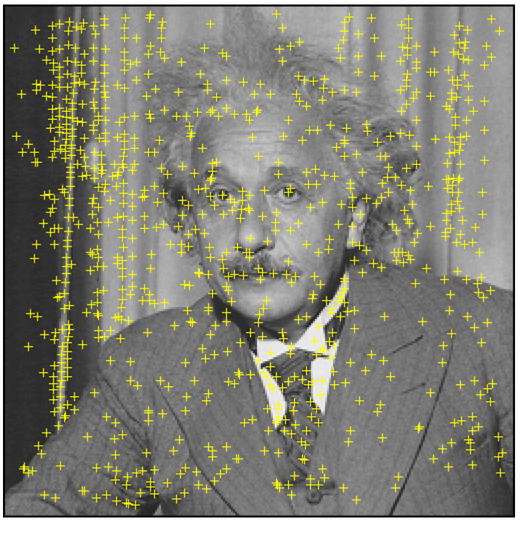
SIFT descriptor

- Assign canonical orientation at peak of smoothed histogram
  - fit parabola to better localize peak



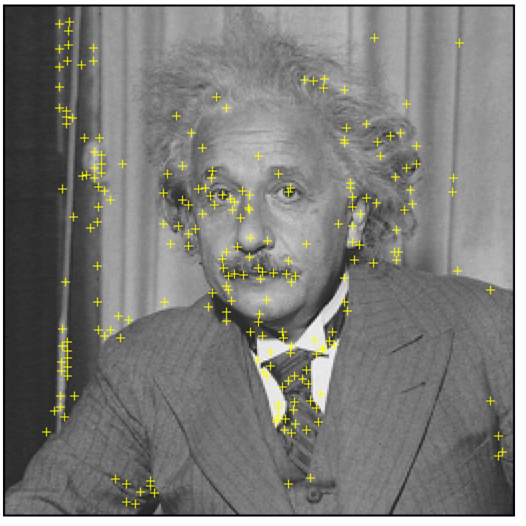
- If other peaks are within 80% of highest peak → assign multiple orientations
  - significantly improves stability of matching
  - about 15% of keypoints has multiple orientations assigned

SIFT keypoint localisation: example



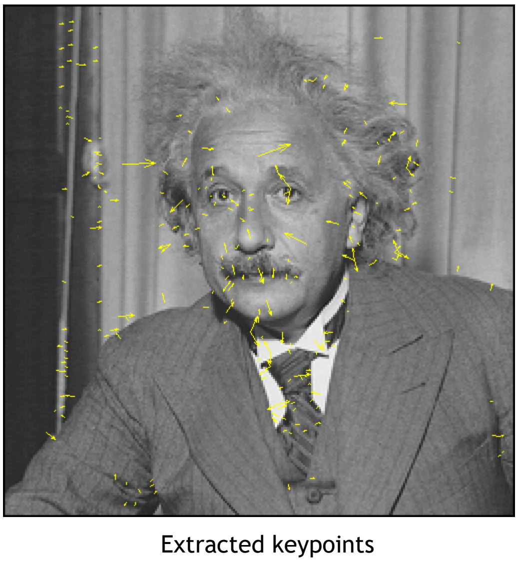
Maxima

SIFT keypoint localisation: example



Remove low contrast and edges

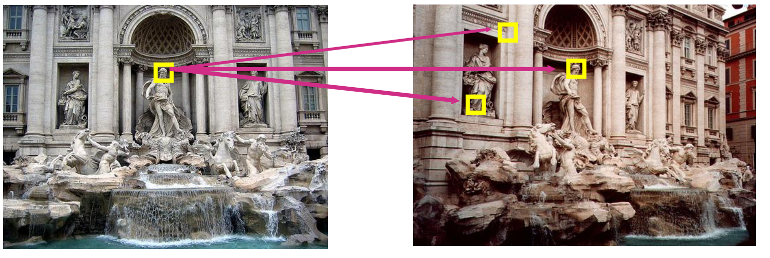
SIFT keypoint localisation: example



Extracted keypoints
(arrows indicate scale and orientation)

Matching SIFT features

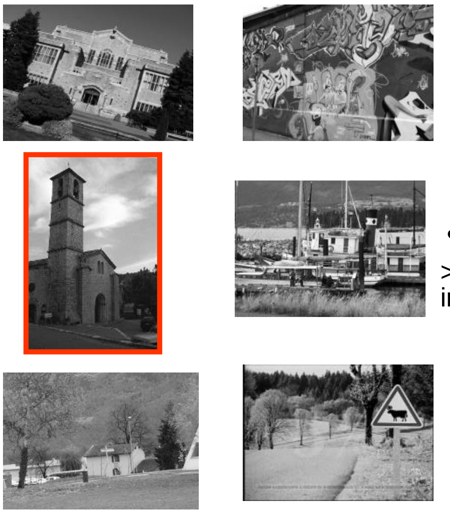
- Given a feature in I_1, how to find the best match in I_2?
    1. Define distance function that compares two descriptors
    2. Test all the features in I_2, find the one with min distance



I_1 I_2

Image retrieval: example

change in viewing angle



. . .

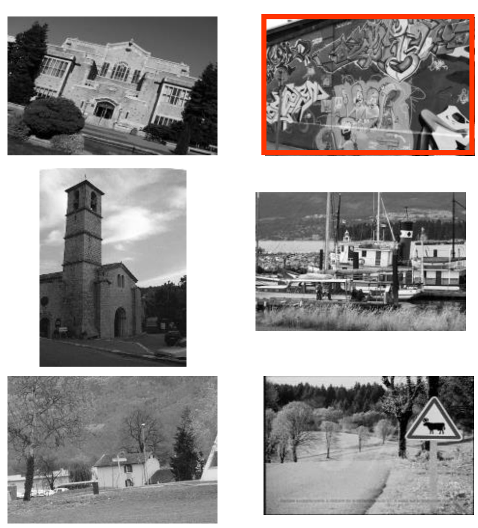
> 5000 images

Example: matches

22 correct matches

Image retrieval: example 2

change in viewing angle
+ scale change



. . .
> 5000
images

Example: matches

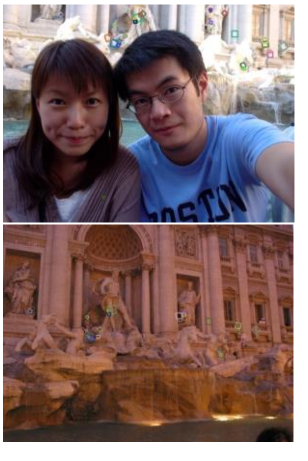
33 correct matches

Properties of SIFT

- Very robust
  - handles changes in viewpoint
    - up to about 60 degree out of plane rotation
  - handles significant changes in illumination
    - day vs. night
- Fast and efficient
  - can run in real time

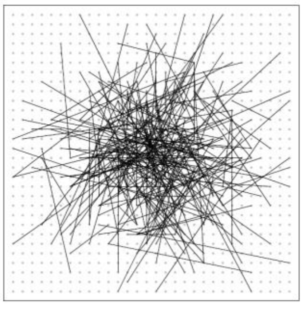
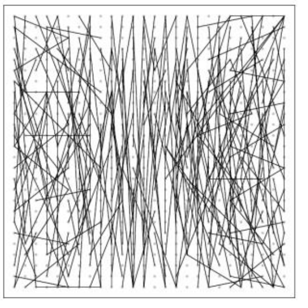
- SURF
  - Speeded Up Robust Features
  - faster than SIFT

  SURF: Speeded Up Robust Features,
  Computer Vision and Image Understanding, 2008



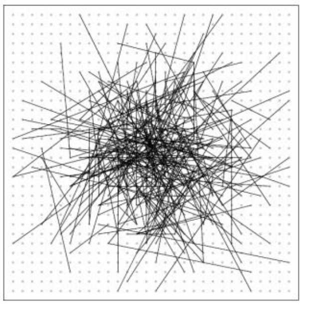
Binary descriptors

- Principles
  - computed from local image patches from a set of pairwise intensity comparisons
  - each bit: the result of one comparison
  - similarity measure: Hamming distance

http://cs.unc.edu/~jheinly/binary_descriptors.html

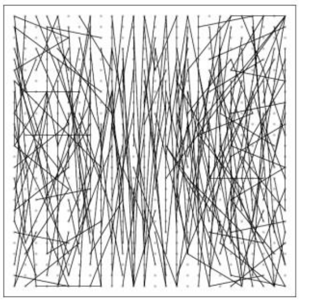
BRIEF



- Binary Robust Independent Elementary Features
  - sampling pattern
    - 128, 256, or 512 comparisons → 128, 256, or 512 bits
  - sample points
    - selected randomly from an isotropic Gaussian distribution centred at the feature location
  - low computational cost
  - low storage requirements

BRIEF: Binary Robust Independent Elementary Features. ECCV 2010

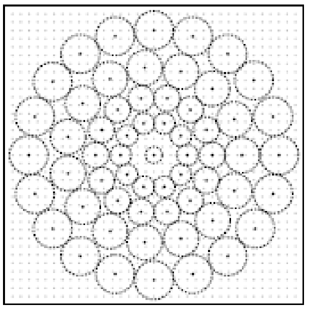
ORB



- Oriented FAST and Rotated BRIEF
  - overcomes the lack of rotation invariance of BRIEF
  - local orientation
    - vector between the feature location and the intensity centroid (i.e. weighted average of intensities in the patch)
  - 256 pairwise intensity comparisons
    - constructed via machine learning
      - maximizing the descriptor's variance
      - minimizing the correlation under various orientations

ORB: An Efficient Alternative to SIFT or SURF. ICCV 2011

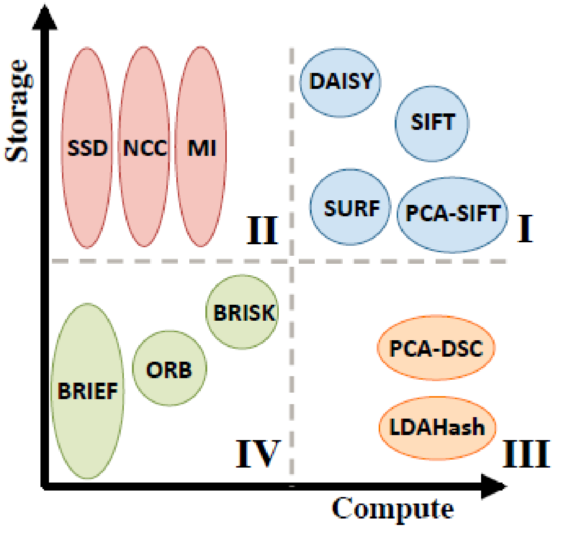
BRISK



- Binary Robust Invariant Scalable Keypoints
  - scale invariant
    - detects keypoints in a scale-space pyramid
    - non-maxima suppression and interpolation across all scales
  - rotation invariant
  - symmetric pattern
    - sample points in concentric circles surrounding the feature
    - each sample point represents a Gaussian blurring of its surrounding
    - standard deviation of blurring increases with distance from centre
  - high computational cost
  - high storage requirements

BRISK: Binary Robust Invariant Scalable Keypoints. ICCV 2011

Comparison



Comparative Evaluation of Binary Features. ECCV 2012