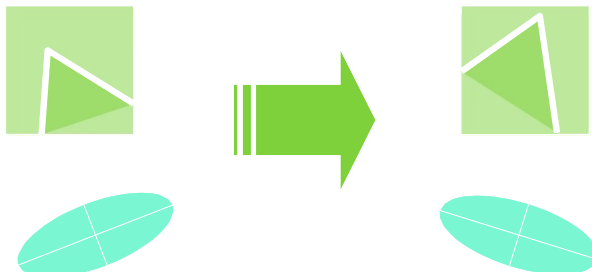
Scale invariance

Harris Detector: Some Properties

Rotation invariance?

Harris Detector: Some Properties

Rotation invariance?



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

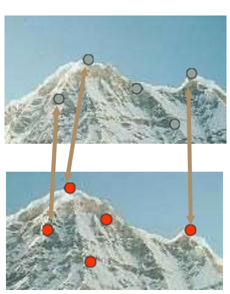
Corner response R is invariant to image rotation

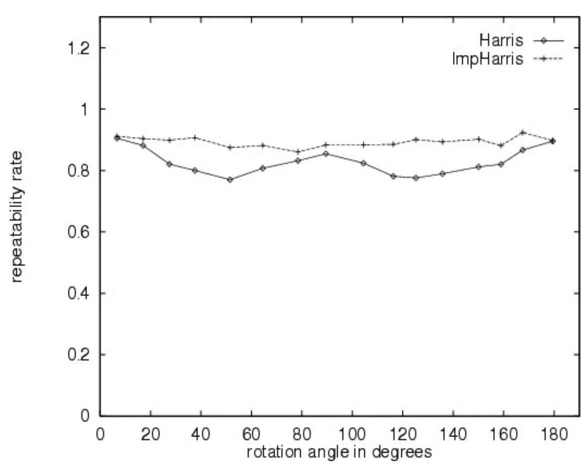
Rotation Invariant Detection

Harris Corner Detector

Repeatability rate:

$$\frac{\#\text{ correspondences}}{\#\text{ possible correspondences}}$$





C.Schmid et.al. "Evaluation of Interest Point Detectors". IJCV 2000

Harris Detector: Some Properties

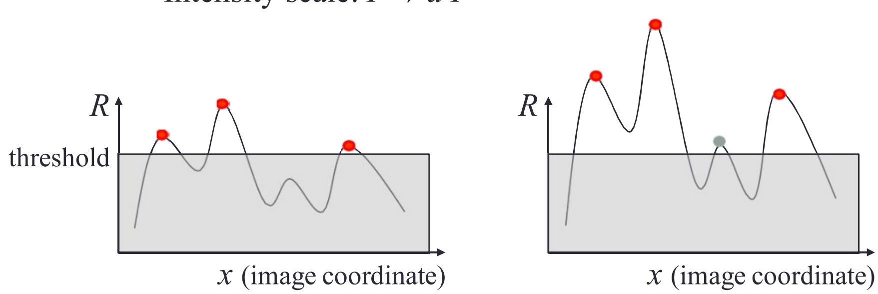
- Invariance to image intensity change?

Harris Detector: Some Properties

- Partial invariance to additive and multiplicative intensity changes (threshold issue for multiplicative)

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

✓ Intensity scale: $I \rightarrow a\, I$

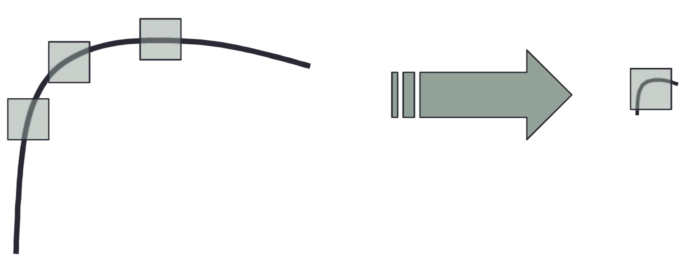


Harris Detector: Some Properties

- Invariant to image scale?

Harris Detector: Some Properties

- **Not invariant to *image scale*!**

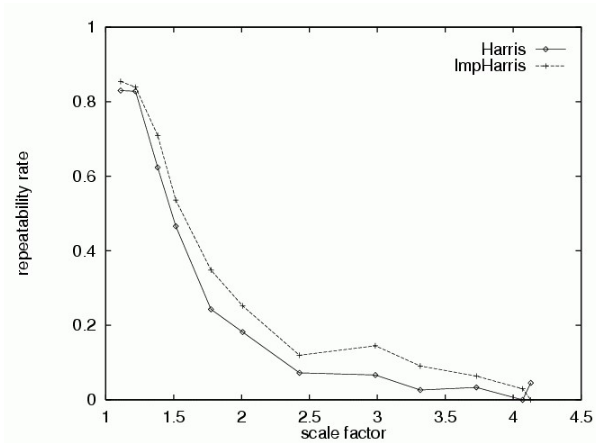


All points will be classified as edges

Corner !

Harris Detector: Some Properties

Not invariant to image scale:

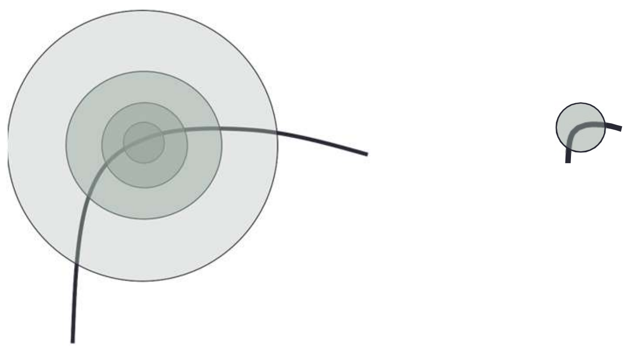


- Quality of Harris detector for different scale changes

**IF* we want scale invariance...*

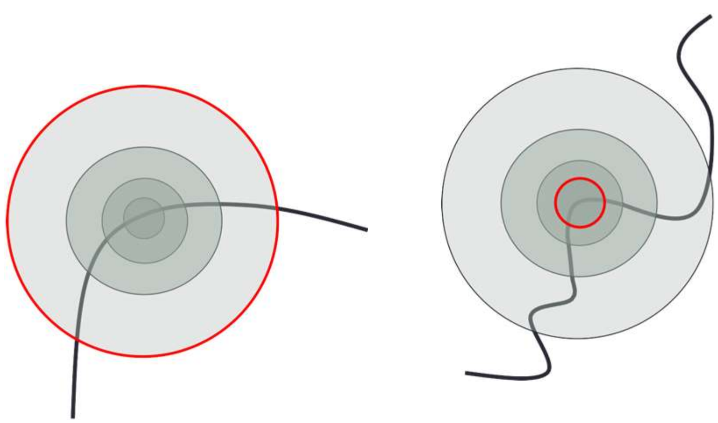
Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale Invariant Detection

- The problem: how do we choose corresponding circles **independently** in each image?



Scale Invariant Detection

- Solution:
  - Design a function on the region (circle), which is "scale invariant" - not affected by the size but will be the same for "corresponding regions, " even if they are at different sizes/scales.

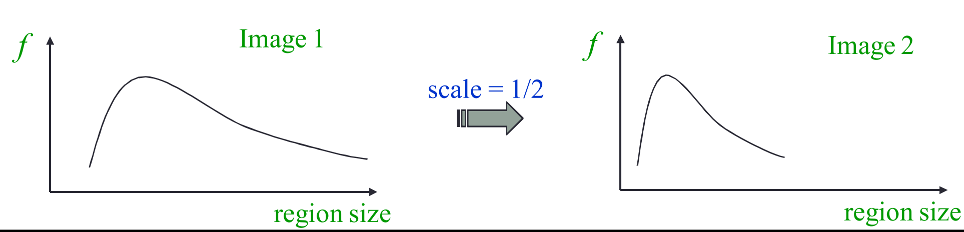
Example: Average intensity. For corresponding regions (even of different sizes) it will be the same.

Scale Invariant Detection

- Solution:
  - Design a function on the region (circle), which is "scale invariant" (the same for corresponding regions, even if they are at different scales)

Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

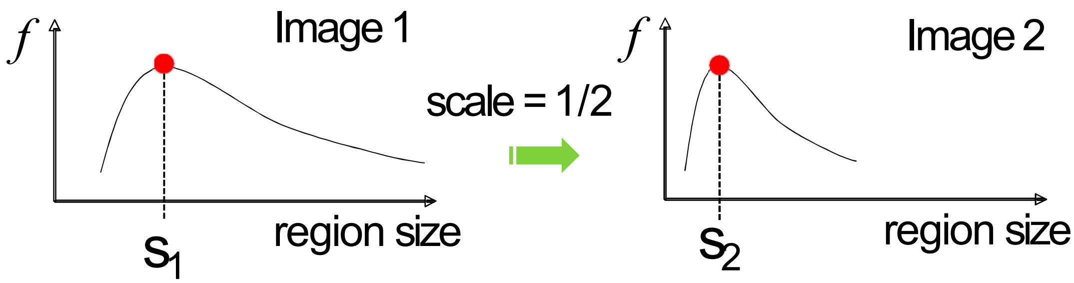
For some given point in one image, we can consider it as a function of region size (circle radius)



Scale Invariant Detection

One method:

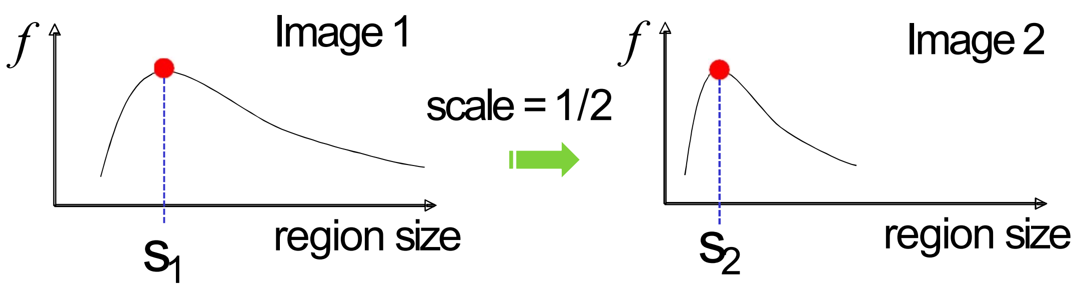
- At a point, compute the scale invariant function over different size neighborhoods (different scales).
- Choose the scale for each image at which the function is a maximum



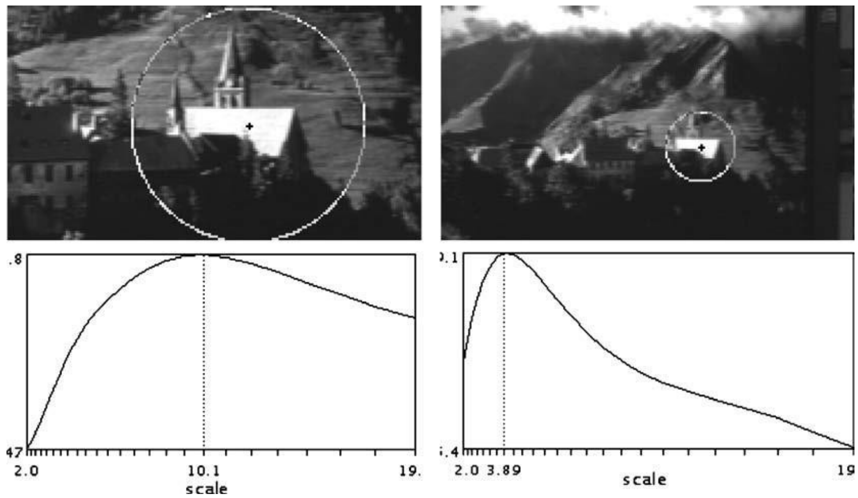
Scale Invariant Detection

One method:

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

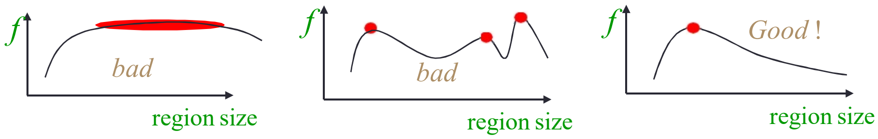


Scale sensitive response



Scale Invariant Detection

- A "good" function for scale detection: has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

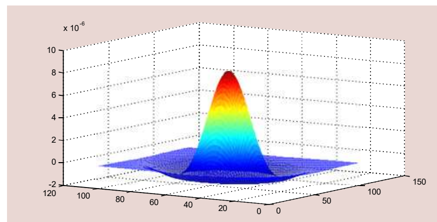
Scale Invariant Detection

Function is just application of a kernel: $f = \text{Kernel} * \text{Image}$

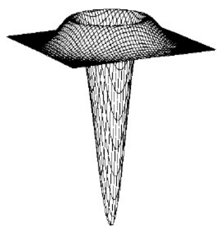
(Laplacian of Gaussian - LoG)

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

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u, v)$$



Scale Invariant Detection

- Functions for determining scale

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

Kernels:

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

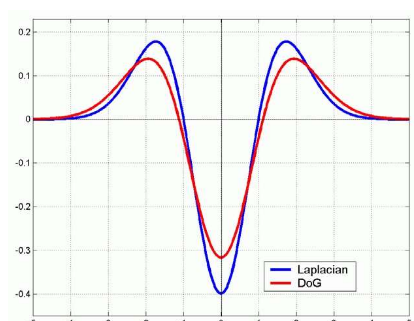
(Laplacian)

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

(Difference of Gaussians)

where Gaussian

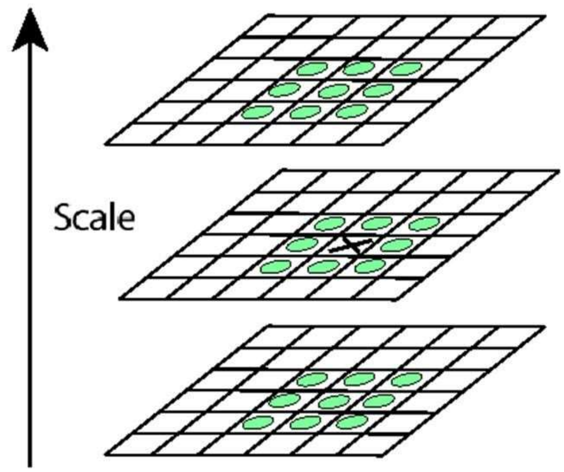
$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



Note: both kernels are invariant to *scale* and *rotation*

Key point localization

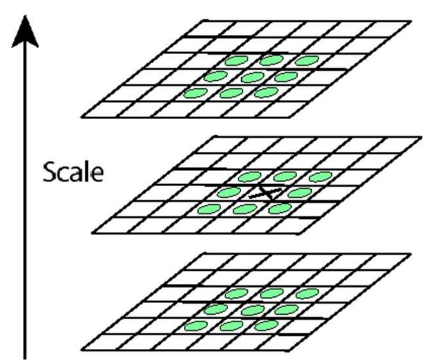
- General idea: find robust extremum (maximum or minimum) both in space and in scale.



Key point localization

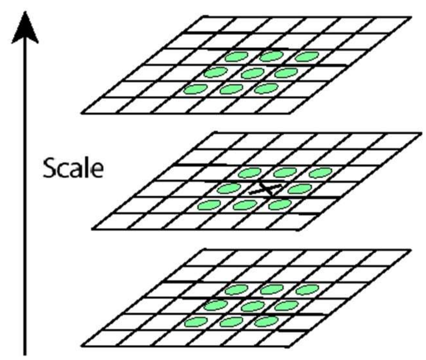
- SIFT: **S**cale **I**nvariant **F**eature **T**ransform

- Specific suggestion: use DoG pyramid to find maximum values (remember edge detection?) – then eliminate “edges” and pick only corners.

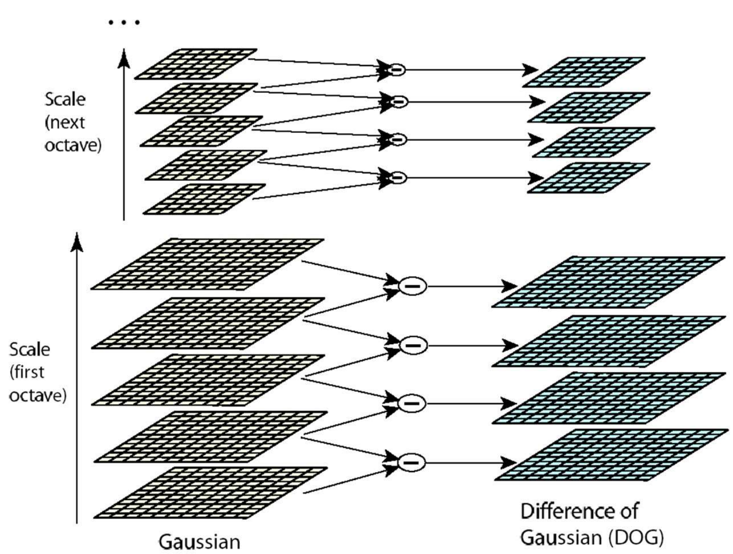


Key point localization

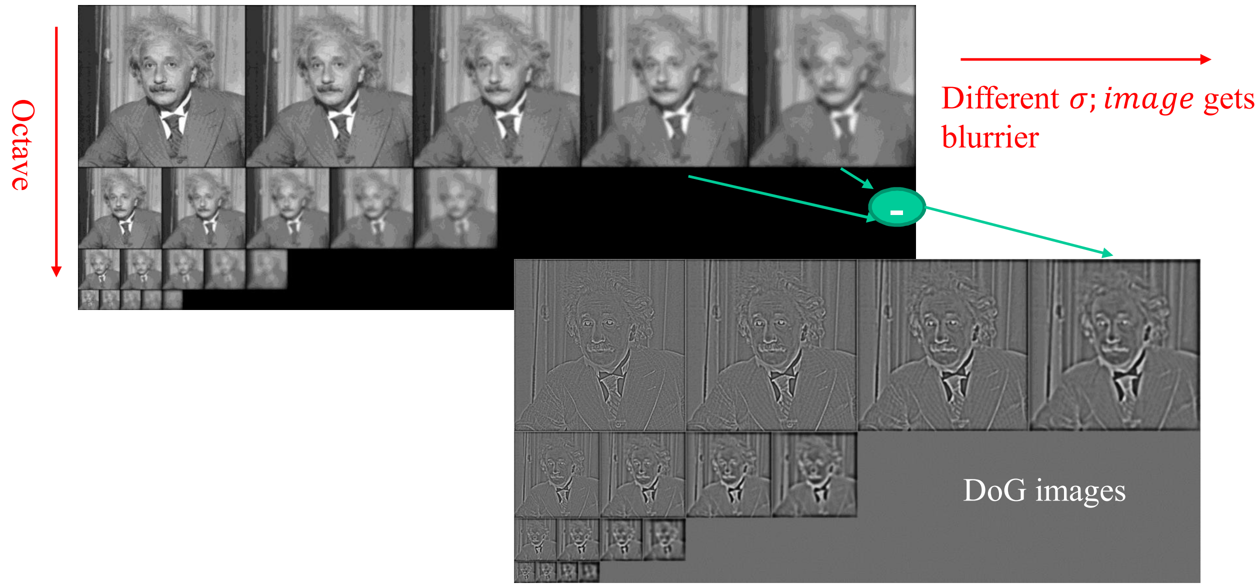
(Each point is compared to its 8 neighbors in the current image and 9 neighbors each in the scales above and below.)



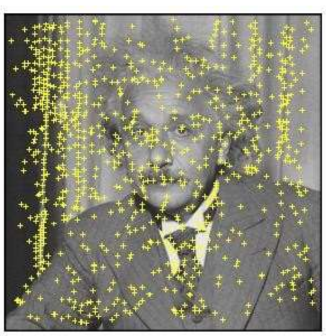
Scale space processed one octave at a time

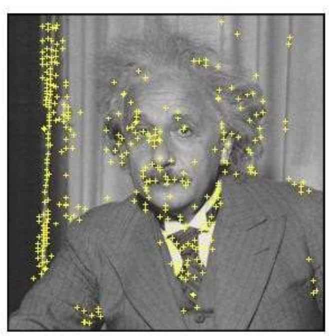


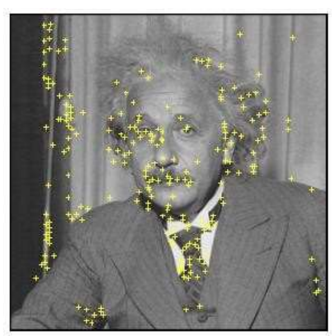
Extrema at different scales



Remove low contrast, edge bound

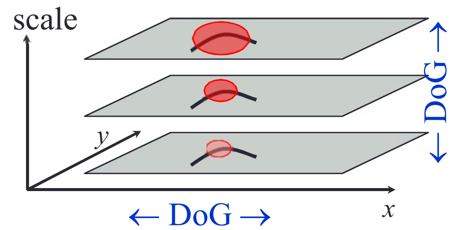

Extrema points


Contrast > C

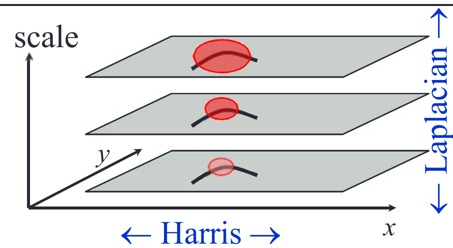

Not on edge

Scale Invariant Detectors

- SIFT (Lowe)[1]
  Find local maximum of:
  – Difference of Gaussians in space and scale



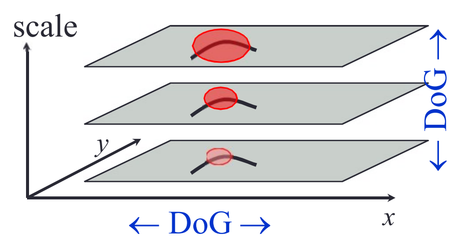
- Harris-Laplacian[2]
  Find local maximum of:
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale



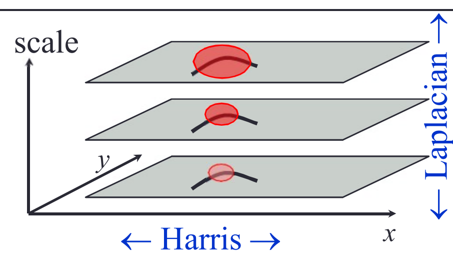
[1]D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
[2]K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

Scale Invariant Detectors

- SIFT (Lowe)[1]
  Find local maximum of:
  – Difference of Gaussians in space and scale



- Harris-Laplacian[2]
  Find local maximum of:
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale

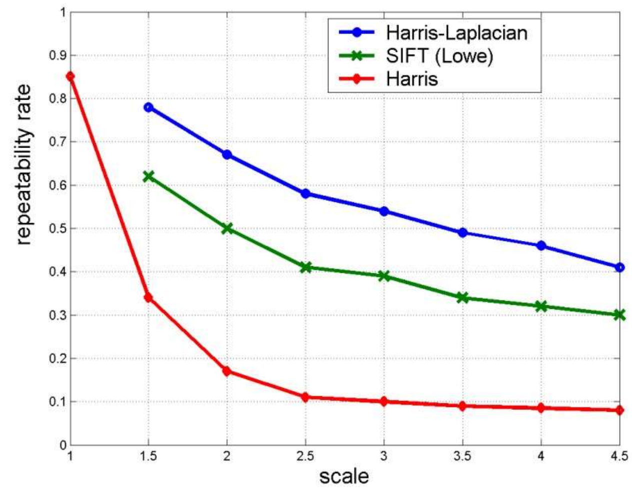


[1]D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
[2]K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

Scale Invariant Detectors

Repeatability rate:

$$\frac{\#\ \text{correspondences}}{\#\ \text{possible correspondences}}$$



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

Scale Invariant Detection: Summary

- Given: Two images of the same scene with a *large* scale difference between them
- Goal: Find the same interest points independently in each image
- Solution: Search for maxima of suitable functions in scale and in space (over the image)

Scale Invariant Detection: Summary

- Given: two images of the same scene with a *large* scale difference between them
- Goal: find the same interest points independently in each image
- Solution: search for maxima of suitable functions in scale and in space (over the image)

Methods:

1. SIFT [Lowe]: maximize Difference of Gaussians over scale and space
2. Harris-Laplacian [Mikolajczyk, Schmid]: maximize Laplacian over scale, Harris' measure of corner response over the image

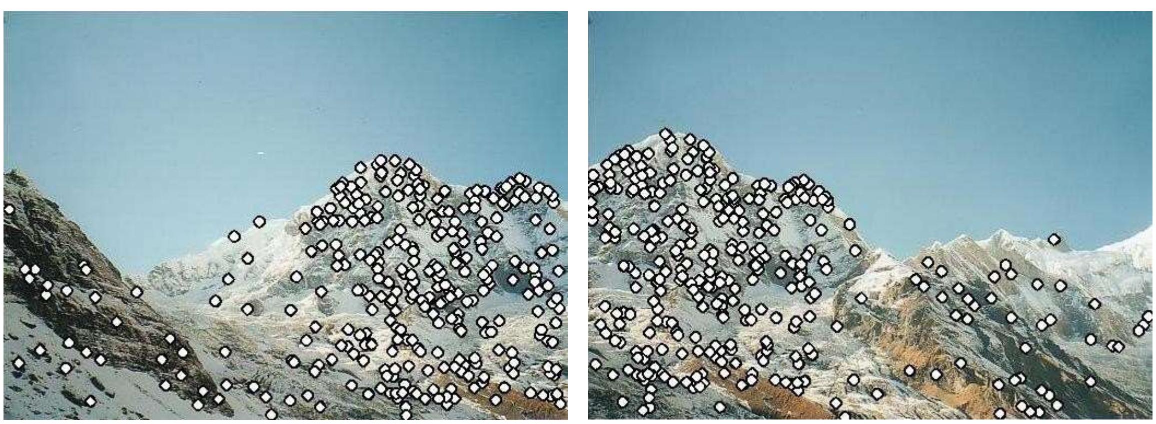
SIFT descriptor

Objectives

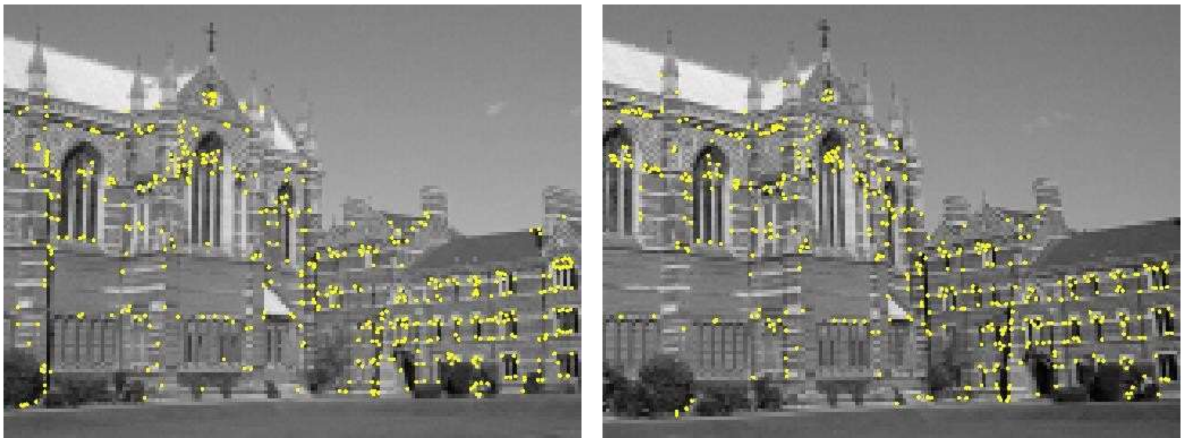
- Features recap:
  - Goal is to find corresponding locations in two images.
  - Last time: find locations that can be accurately located and likely to be found in both images even if photometric or slight geometric changes.
  - This time– find possible (likely?) correspondences between points
  - Next: which of the guessed, plausible correspondences are correct

Point Descriptors

- Last time: How to detect interest points



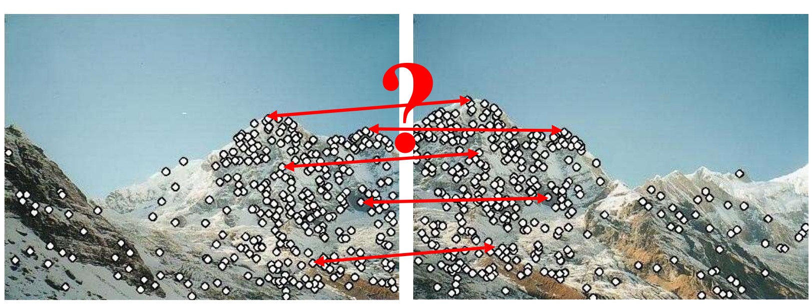
Harris detector



Interest points extracted with Harris (~ 500 points)

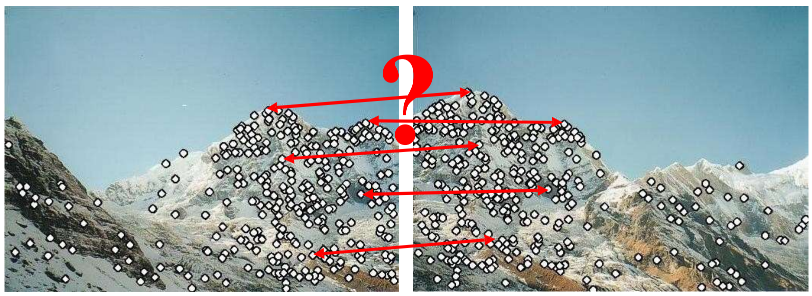
Point Descriptors

- Now: How to match them?



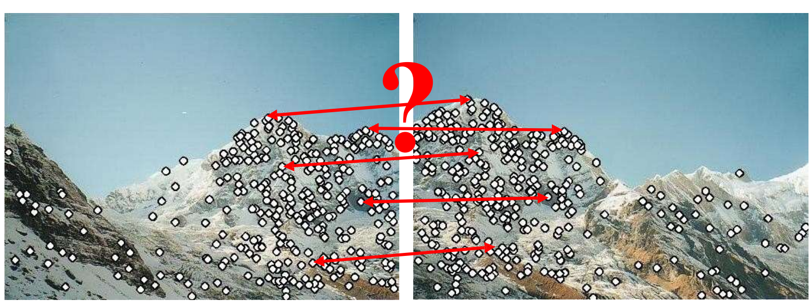
Point Descriptors

- We need to describe them – a "*descriptor*"



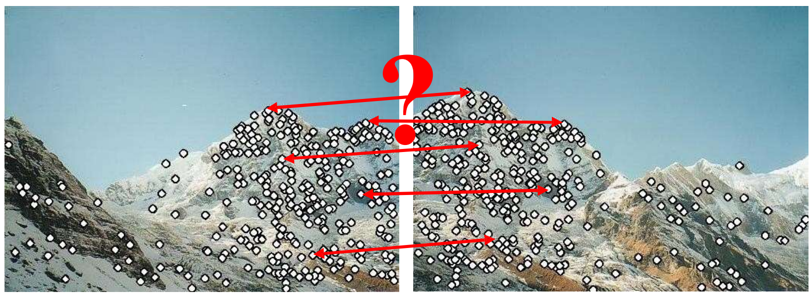
Criteria for Point Descriptors

- We want the descriptors to be the (almost) same in both image – *invariant*.



Criteria for Point Descriptors

- We also need the descriptors to be *distinctive*.



Simple solution?

- Harris gives good detection – and we also know the scale.
- Why not just use correlation to check the match of the window around the feature in image 1 with every feature in image2?

Simple solution? *Not so good!*

- Not so good because:
  - Correlation is not rotation invariant - why do we want this?
  - Correlation is sensitive to photometric changes.
  - Normalized correlation is sensitive to non-linear photometric changes and even slight geometric ones.
  - Could be slow – check all features against all features

SIFT: Scale Invariant Feature Detection

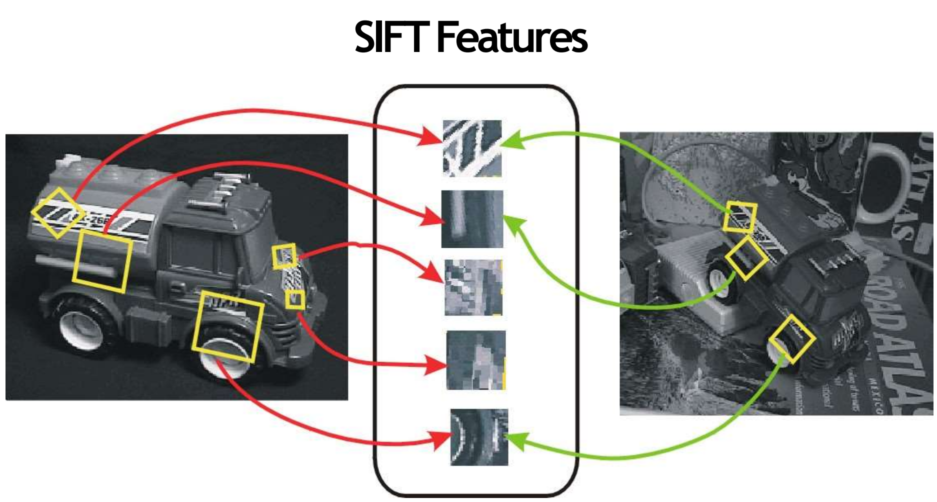
- Motivation: The Harris operator was not invariant to scale and correlation was not invariant to rotation.

- For better image matching, Lowe's goals were:
  - To develop an interest operator – a *detector* – that is invariant to scale and rotation.
  - Also: create a descriptor that was robust to the variations corresponding to typical viewing conditions. *The descriptor is the most-used part of SIFT.*

Idea of SIFT

- Image content is represented by a constellation of local features that are invariant to translation, rotation, scale, and other imaging parameters

Idea of SIFT

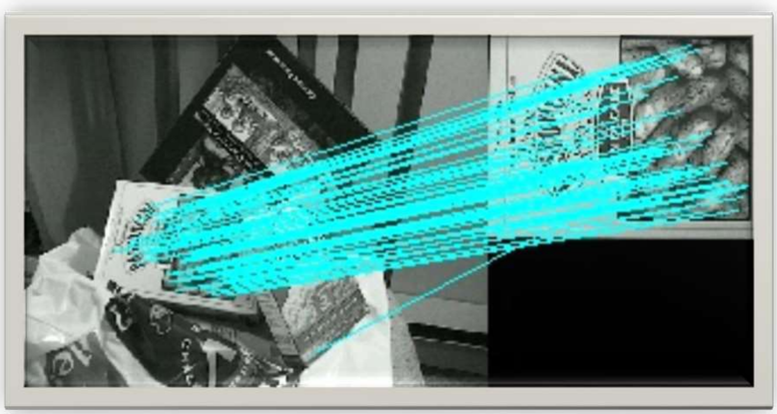


Another version of the problem…

Want to find

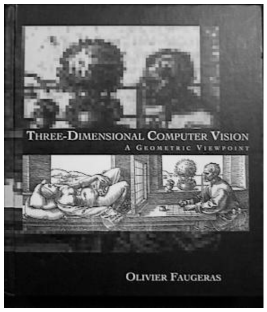
…in here

Another version of the problem…



Another version of the problem…

Want to find



… in here

Overall SIFT Procedure

- Scale-space extrema detection
- Keypoint localization

Or use Harris-Laplace or other method

- Orientation assignment
- Keypoint description

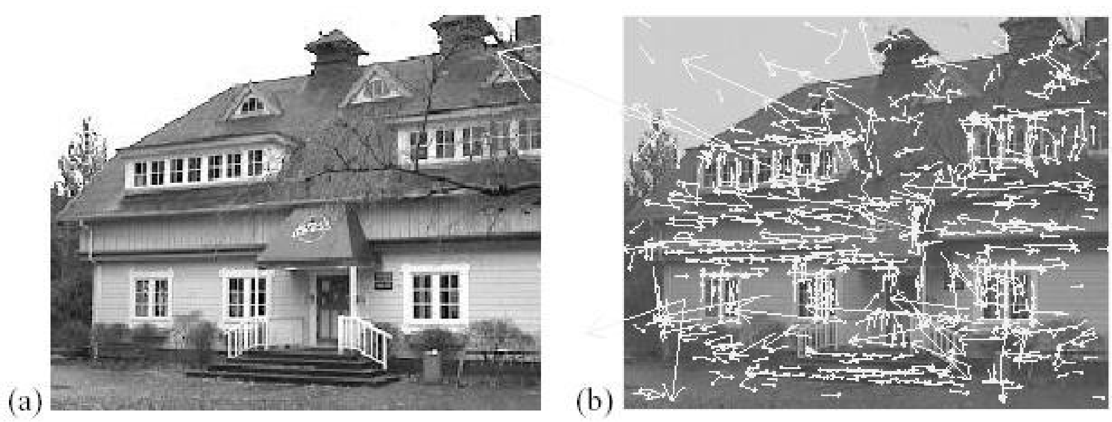
Overall Procedure at a High Level

- Scale-space extrema detection
  - Search over multiple scales and image locations
- Keypoint localization
  - Define a model to determine location and scale. Select keypoints based on a measure of stability.

Use Harris-Laplace or other method

- Orientation assignment
  - Compute best orientation(s) for each keypoint region.
- Keypoint description
  - Use local image gradients at selected scale and rotation
  - to describe each keypoint region.

Example of keypoint detection



(a) 233x189 image
(b) 832 DOG extrema

Overall SIFT Procedure

1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment

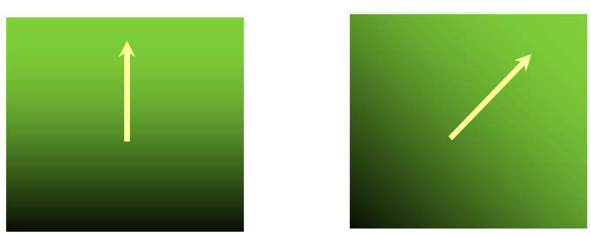
   Compute best orientation(s) for each keypoint region.

4. Keypoint description

   Use local image gradients at selected scale and rotation to describe each keypoint region.

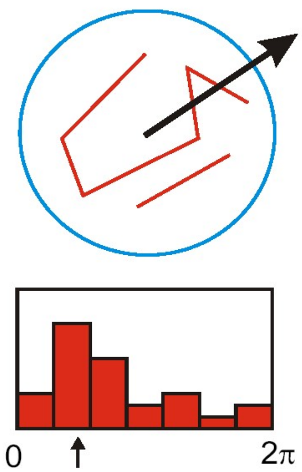
Descriptors Invariant to Rotation

- Find the dominant direction of gradient –that is the base orientation.



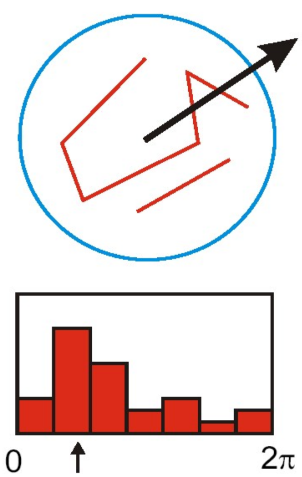
Compute image derivatives relative to this orientation

Orientation assignment



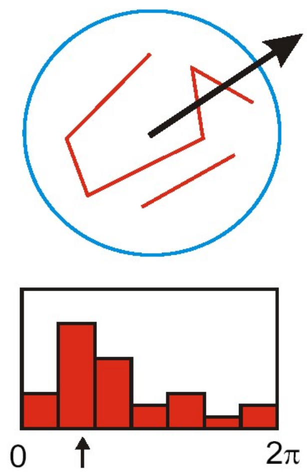
- Create histogram of local gradient directions at *selected* scale – 36 bins

Orientation assignment



- Assign canonical orientation at peak of smoothed histogram

Orientation assignment



- Each *keypoint* now specifies stable 2D coordinates (x, y, scale, orientation) – invariant to those.

4. Keypoint Descriptors

- Next is to compute a descriptor for the local image region about each keypoint that is:
  - Highly ***distinctive***
  - As ***invariant*** as possible to variations such as changes in viewpoint and illumination

But first… normalization

- Rotate the window to standard orientation
- Scale the window size based on the scale at which the point was found.

SIFT vector formation

Compute a feature vector based upon:

- histograms of gradients

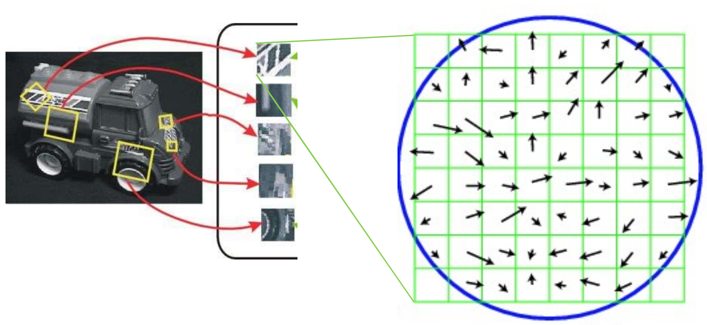
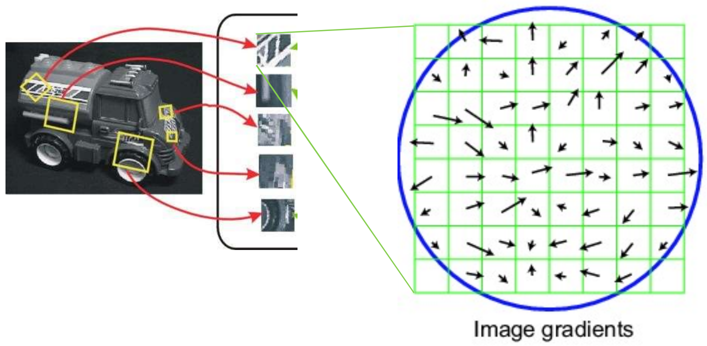


Image gradients

SIFT vector formation

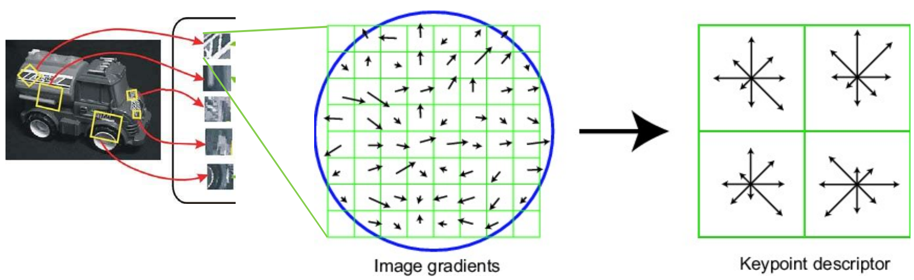
Compute a feature vector based upon:

- Gradient weighted by a centered Gaussian,



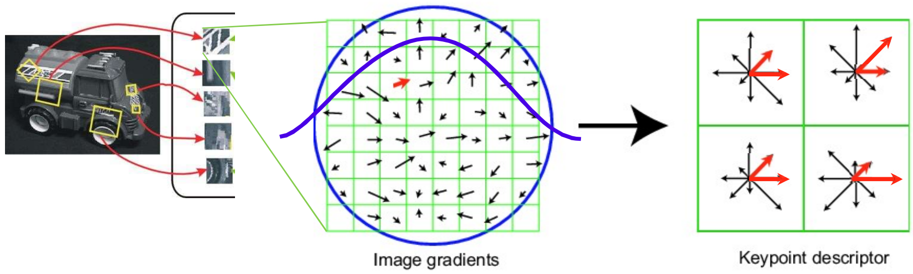
SIFT vector formation

- 4x4 array of gradient orientation histograms over 4x4 pixels
  - not really histogram, weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



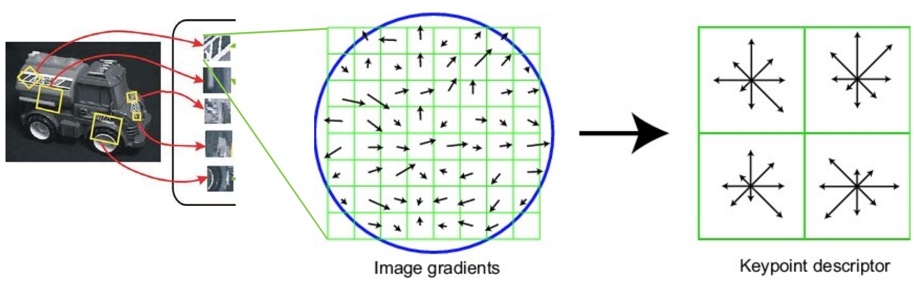
showing only 2x2 here but is 4x4

Ensure smoothness



Reduce effect of illumination

- 128-dim vector normalized to magnitude 1.0
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after rotation normalization, clamp gradients >0.2



Evaluating the SIFT descriptors

- Database images were subjected to rotation, scaling, affine stretch, brightness and contrast changes, and added noise.
- Feature point detectors and descriptors were compared before and after the distortions.
- Mostly looking for stability with respect to change.

Sensitivity to number of histogram orientations

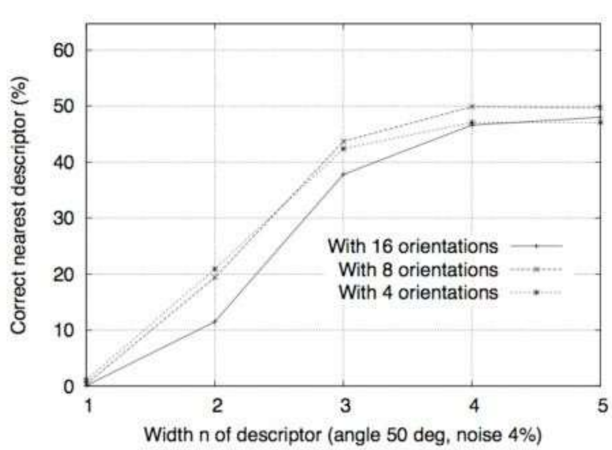
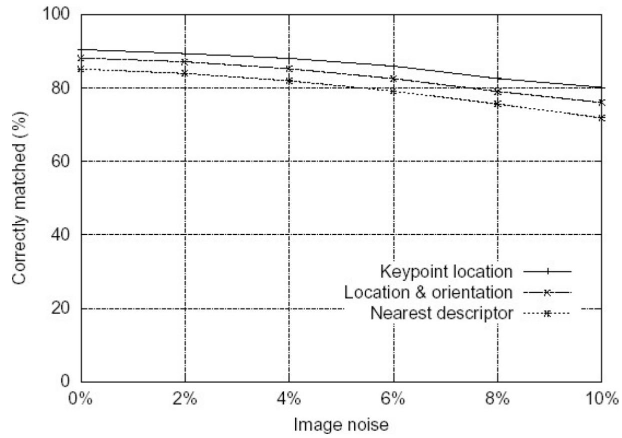


Figure 8: This graph shows the percent of keypoints giving the correct match to a database of 40,000 keypoints as a function of width of the $n \times n$ keypoint descriptor and the number of orientations in each histogram. The graph is computed for images with affine viewpoint change of 50 degrees and addition of 4% noise.

Feature stability to noise

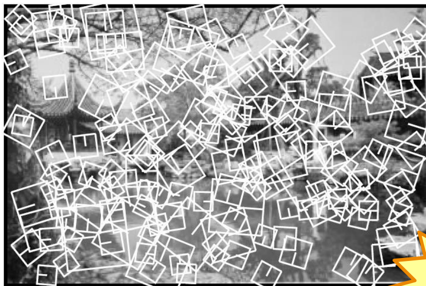
- Match features after random change in image scale & orientation, with differing levels of image noise
- Find nearest neighbor in database of 30,000 features



Experimental results - summary

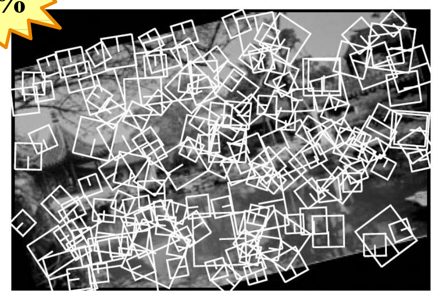
Image transformation	Location and scale match	Orientation match
Decrease constrast by 1.2	89.0 %	86.6 %
Decrease intensity by 0.2	88.5 %	85.9 %
Rotate by 20°	85.4 %	81.0 %
Scale by 0.7	85.1 %	80.3 %
Stretch by 1.2	83.5 %	76.1 %
Stretch by 1.5	77.7 %	65.0 %
Add 10% pixel noise	90.3 %	88.4 %
All previous	**78.6 %**	**71.8 %**

Experimental results

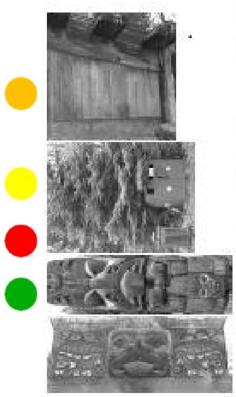


Original image

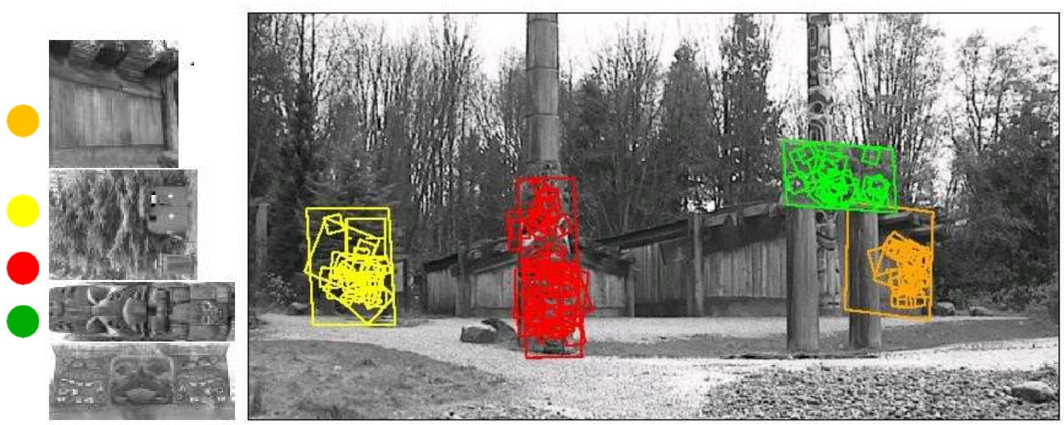
Keypoints on image after rotation (15°), scaling (90%), horizontal stretching (110%), change of brightness (-10%) and contrast (90%), and addition of pixel noise



SIFT matching object pieces (for location)



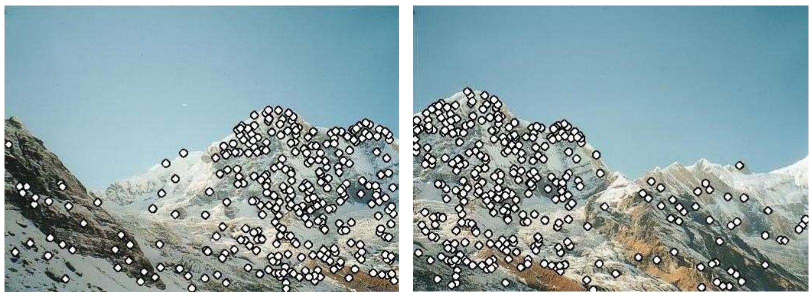
SIFT matching object pieces (for location)



Matching feature points (a little)

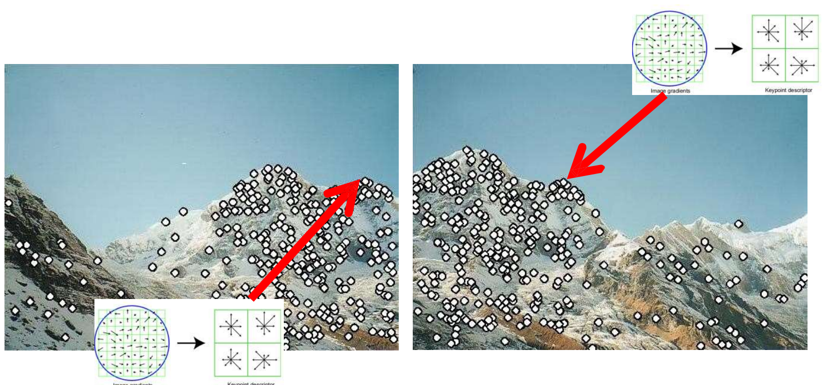
Feature Points

- We know how to detect points



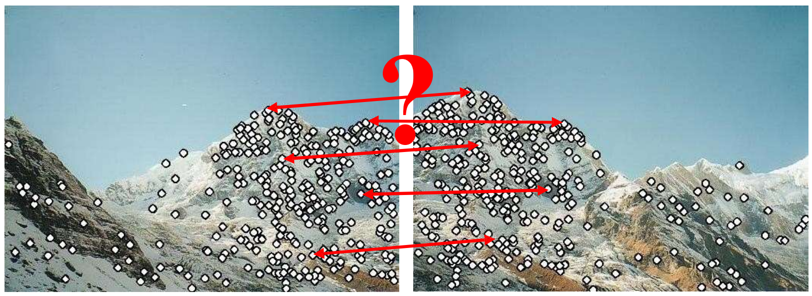
Feature Points

- We know how to describe them



Feature Points

- Next question: How to match them?



How to match feature points?

- Could just do nearest-neighbor search
  - You will!

- But that's really expensive…SIFT tests have 10,000's of points!

How to match feature points?

- Other methods
  - *k*-D tree and
  - Best Bin First (BBF)
  - Haar wavelet
  - Hashing
- Result: Can give speedup by factor of *100-1000* while finding nearest neighbor (of interest) 95% of the time

3D Object Recognition

Train:

1. Extract outlines with background subtraction
2. Compute "keypoints" – interest points and descriptors.

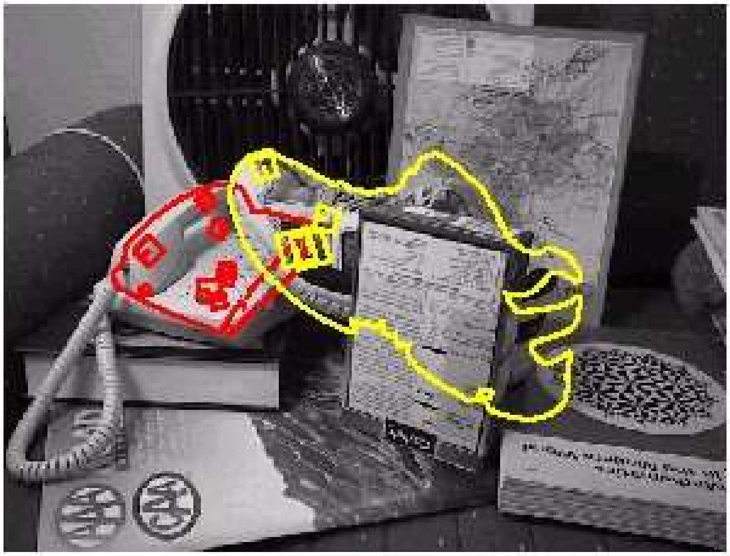
3D Object Recognition

Test:

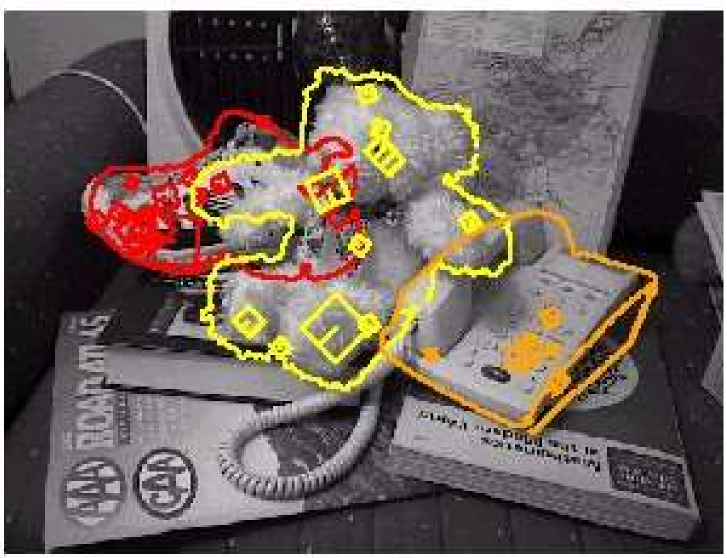
1. Find possible matches.
2. Search for consistent solution – such as *affine*. *(How many points?!?!?)*

Results

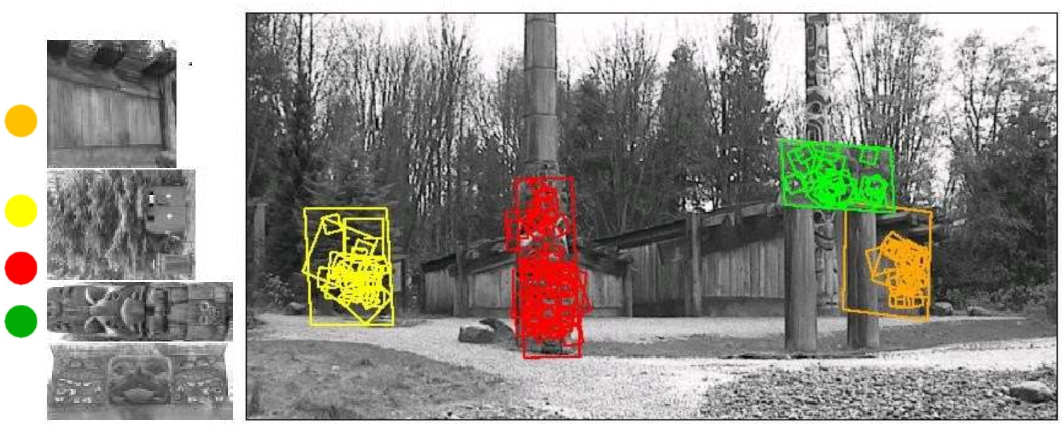
Recognition under occlusion



Recognition under occlusion



Locating object pieces



(From last lesson)

SIFT in Sony Aibo (Evolution Robotics)

SIFT usage:

- Recognize charging station
- Communicate with visual cards

http://www.sony-aibo.com/aibo-models/sony-aibo-ers-7/