Computer Vision
and
Machine Learning
(Keypoint detector and descriptor)

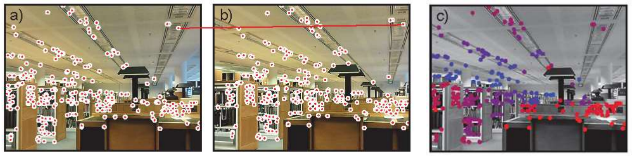
Bhabatosh Chanda
chanda@isical.ac.in

Binocular stereo reconstruction

1. Compute image features.
2. Compute feature descriptors.
3. Find initial matches.
4. Compute fundamental matrix.
5. Refine matches.
6. Estimate essential matrix.
7. Decompose essential matrix.
8. Estimate 3D points.

Correspondence problem

- How to detect points in the scene (object) whose coordinates need to be determined.
- How to establish correspondence between points (in different camera frames) which are images of same scene point.
- How to perform reliable and efficient search.

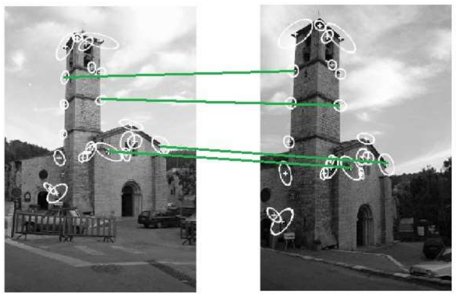


Detector and descriptor

- Harris corner detector
- Histogram of Oriented Gradients (HOG)
- Scale Invariant Feature Transform (SIFT)

Motivation: Matching Problem

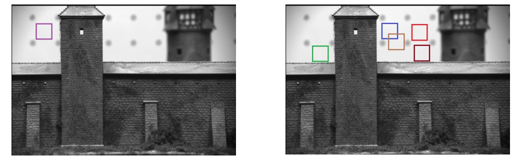
Vision tasks such as stereo and motion estimation require finding corresponding features across two or more views.



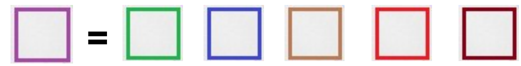
Robert Collins, Penn State Univ.

Not all Patches Created are Equal!

- A blindly picked up patch may not help perform the task

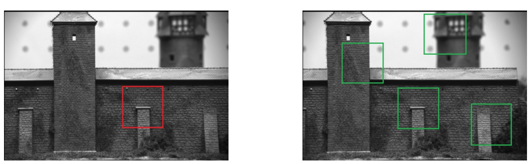


- Feature-less or attribute-less patches are highly probable in image and match with many such similar ones



Motivation: Patch Matching

- Elements to be matched are image patches of fixed size



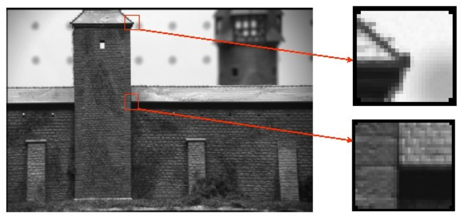
- Task: find the best (most similar) patch in a second image

Motivation: Patch Matching

- Elements to be matched are image patches of fixed size

- A distinctive patch is a good patch for matching

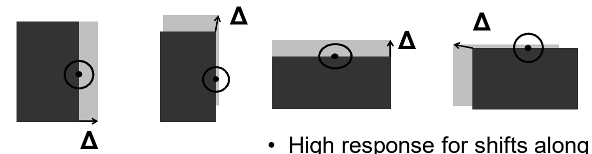
What is corner?

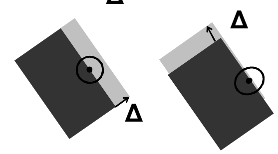


- High curvature point on contour (edge)
- Junction of contours
- Usually stable features with respect to view points
- Large variation in neighborhood of the point in almost all directions

Harris corner detector

- Key idea: Measure changes over a neighborhood due to a shift and then analyze dependency on shift direction.
- Direction dependency of the response for lines

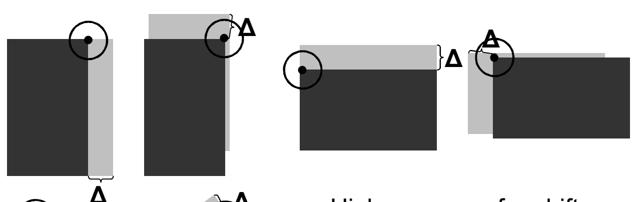


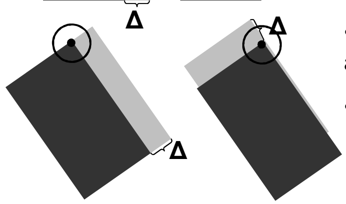


- High response for shifts along the edge direction.
- Low response for shifts orthogonal to edge direction.
- **Anisotropic response**

Key idea: *continued …*

- Orientation dependence of the shift response for corners





- High response for shifts along all directions
- **Isotropic response**

Harris corner: formulation

- An image patch or neigborhood W is shifted by a shift vector $\boldsymbol{\Delta} = [\Delta x, \Delta y]$.
- A corner does not have the ***aperture problem*** and therefore should show high shift response for all orientation of $\boldsymbol{\Delta}$.
- Sum of squared intensity difference between the original and the shifted image over the neighborhood W is

$$S_W(\boldsymbol{\Delta}) = \sum_{(x_i, y_i)\in W} \left(f(x_i, y_i) - f(x_i + \Delta x, y_i + \Delta y) \right)^2$$

Harris corner: formulation

- Sum of squared intensity difference between original and shifted image over *W* is

$$S_W(\mathbf{\Delta}) = \sum_{(x_i,y_i)\in W} \left(f(x_i,y_i) - f(x_i+\Delta x, y_i+\Delta y) \right)^2$$

- Apply Taylor expansion

$$f(x_i+\Delta x, y_i+\Delta y) = f(x_i,y_i) + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial y}\Delta y + \frac{\partial^2 f}{\partial x^2}\frac{(\Delta x)^2}{2!} + \frac{\partial^2 f}{\partial y^2}\frac{(\Delta y)^2}{2!} + \cdots$$

$$f(x_i+\Delta x, y_i+\Delta y) \approx f(x_i,y_i) + \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

Harris corner *[Continued …]*

$$S(x,y,\mathbf{\Delta}) = \sum_{(x_i,y_i)\in W} \left(f(x_i,y_i) - f(x_i,y_i) - \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2$$

$$= \sum_{(x_i,y_i)\in W} \left(-\begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2$$

$$= \sum_{(x_i,y_i)\in W} \left(\begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2$$

$$= \sum_{(x_i,y_i)\in W} \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \left(\begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} \\ \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \right) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

Harris corner [*Continued …]*

$$= \sum_{(x_i,y_i)\in W} \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \left(\begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} \\ \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \right) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$= \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \left(\sum_{(x_i,y_i)\in W} \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} \\ \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial f(x_i,y_i)}{\partial x} & \frac{\partial f(x_i,y_i)}{\partial y} \end{bmatrix} \right) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$= \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \begin{bmatrix} \sum_{(x_i,y_i)\in W} \left(\frac{\partial f(x_i,y_i)}{\partial x} \right)^2 & \sum_{(x_i,y_i)\in W} \frac{\partial f(x_i,y_i)}{\partial x}\frac{\partial f(x_i,y_i)}{\partial y} \\ \sum_{(x_i,y_i)\in W} \frac{\partial f(x_i,y_i)}{\partial x}\frac{\partial f(x_i,y_i)}{\partial y} & \sum_{(x_i,y_i)\in W} \left(\frac{\partial f(x_i,y_i)}{\partial y} \right)^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$= \mathbf{\Delta}^T \mathbf{A}_w(x,y)\mathbf{\Delta}$$

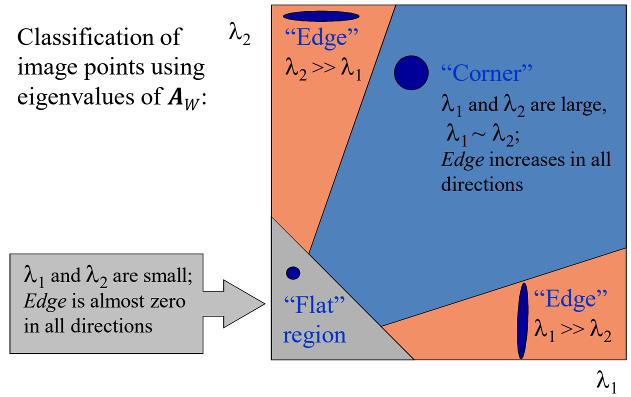
The matrix $\mathbf{A}_W$ is called the *Harris matrix.*

Harris matrix

- The Harris matrix $\boldsymbol{A}_W$ is symmetric and positive semi-definite.
- PCA of $\boldsymbol{A}_W$ gives eigen vector (e_1, e_2) and eigen value (λ_1, λ_2).
- Three distinct situations:
  - Both λ_1 and λ_2 are small $\Rightarrow$ a flat region
  - One λ is large and other is small $\Rightarrow$ existence of edge
  - Both λ_1 and λ_2 are large $\Rightarrow$ existence of corner

Harris Detector: Implementation

Classification of image points using eigenvalues of $\boldsymbol{A}_W$:



Harris Detector: Implementation

Measure of corner response:

$$R = \det \boldsymbol{A}_W - k(trace\boldsymbol{A}_W)^2$$

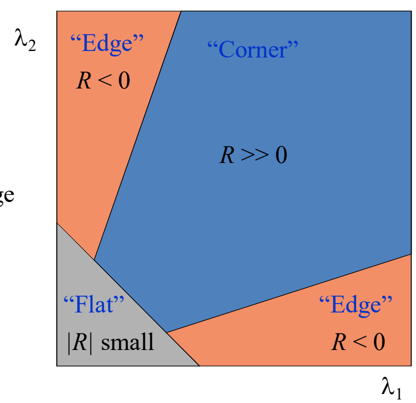
where

$$\det \boldsymbol{A}_W = \lambda_1 \lambda_2$$
$$trace\boldsymbol{A}_W = \lambda_1 + \lambda_2$$

(k – empirical constant, $k = 0.04 - 0.06$)

Harris Detector: Mathematics

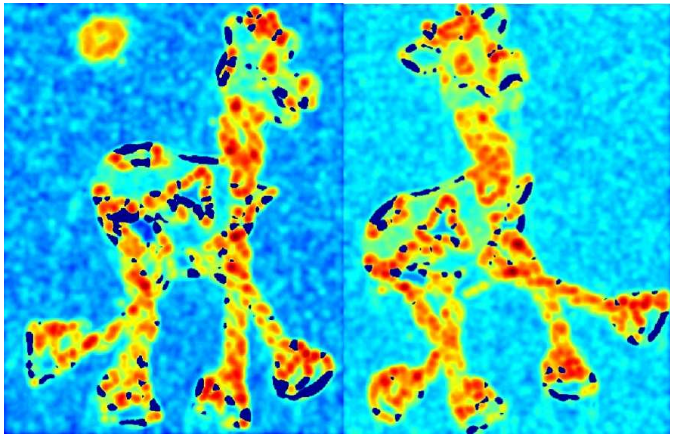
- R depends only on eigenvalues of $\boldsymbol{A}_W$
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Harris Detector: Workflow

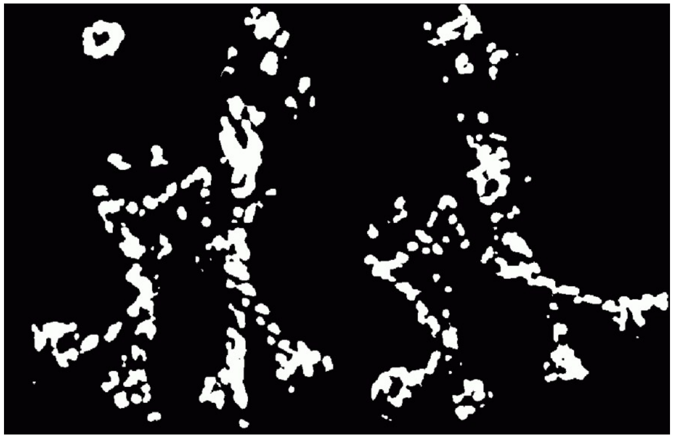
Harris Detector: Workflow

Compute corner response R



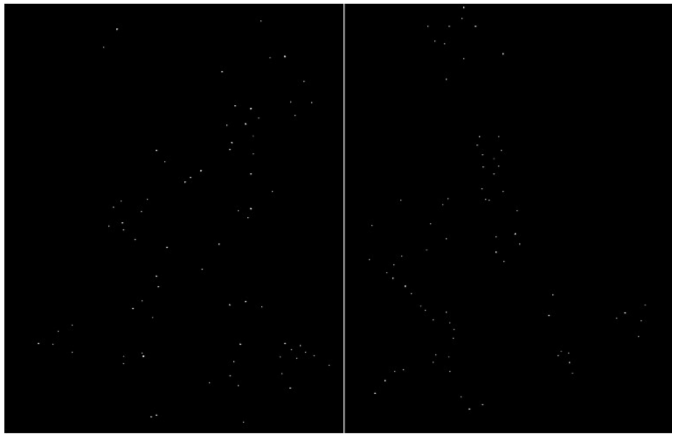
Harris Detector: Workflow

Find points with large corner response: $R >$ threshold



Harris Detector: Workflow

Take only the points of local maxima of R



Harris Detector: Workflow

Histogram of Oriented Gradient (HOG)

HOG feature extraction

- Compute horizontal gradient $\frac{\partial f}{\partial x}$ and vertical gradient $\frac{\partial f}{\partial y}$ after smoothing
- Compute gradient orientation $\theta = tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$ and magnitude $|\nabla f| = \sqrt{\frac{\partial f}{\partial x}^2 + \frac{\partial f}{\partial y}^2}$
  - For color image, pick the color channel with the highest gradient magnitude for each pixel.

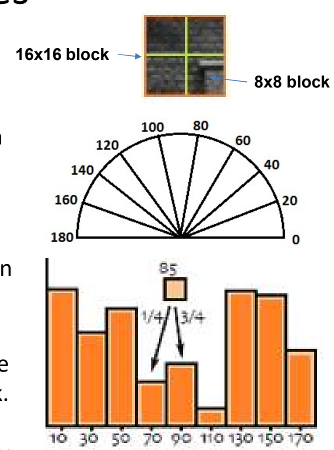
HOG feature: Example

For a 64x128 image,

- Divide the image into 16x16 blocks of 50% overlap.
  - 7x15=105 blocks in total
- Each block should consist of 2x2 cells with size 8x8.
- Quantize the gradient orientation into 9 bins
  - The vote is the gradient magnitude
  - Interpolate votes between neighbouring bin centre.
  - The vote can also be weighted with Gaussian kernel to rationalize the pixels near the edges of block.
- Concatenate histograms (dimension: 105x4x9=3,780)

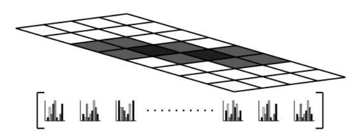
Votes

- Each block consists of 2x2 cells with size 8x8
- Quantize the gradient orientation into 9 bins (0-180)
- The vote is the gradient magnitude
- Interpolate votes linearly between neighbouring bin centres.
- The vote can also be weighted with Gaussian to down weight the pixels near the edges of the block.



Final Feature Vector

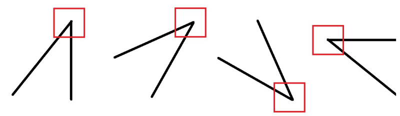
- Concatenate histograms
  - Make it a 1D vector of length 3780.



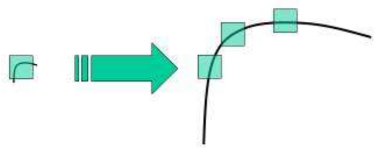
- Visualization

Harris corner detector

- Harris corner detector is rotation invariant

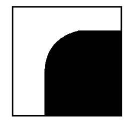


- But it is not scale invariant (as corners are not!!)

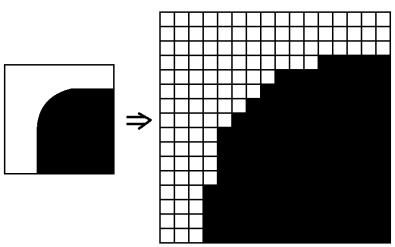


- **<u>Invariance:</u> features (*transform* (image)) = features (image)**

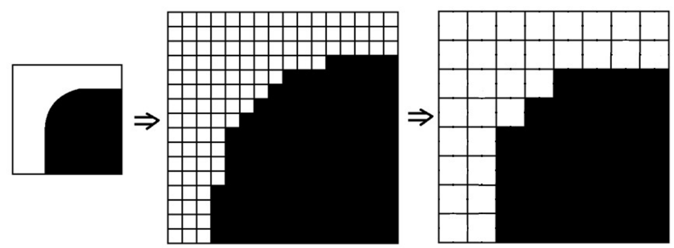
Corner at different scales



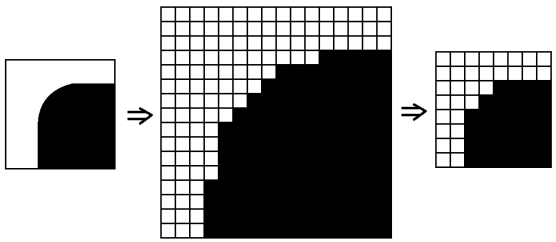
Corner at different scales



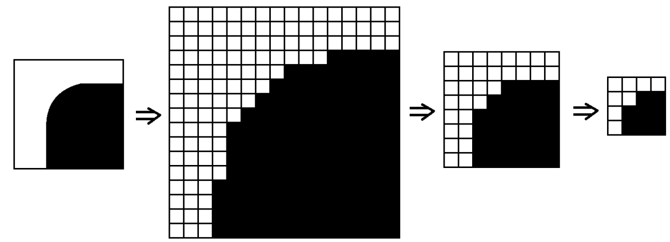
Corner at different scales



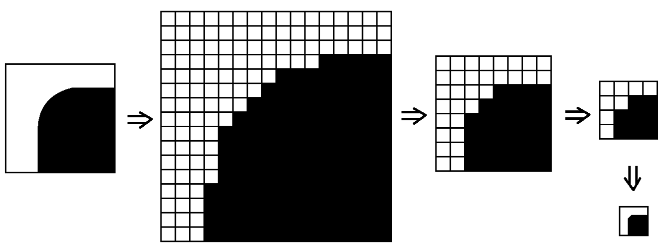
Corner at different scales



Corner at different scales

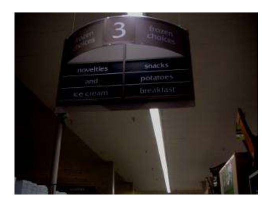


Corner at different scales



Type of invariance

- Illumination



Type of invariance

- Illumination
- Scale

Type of invariance

- Illumination
- Scale
- Rotation

Type of invariance

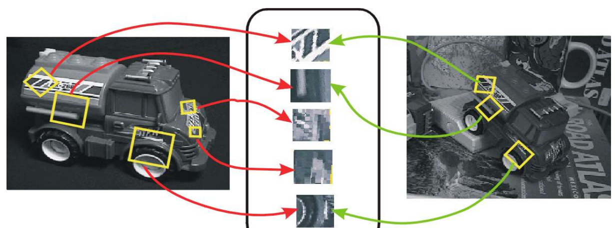
- Illumination
- Scale
- Rotation
- Affine

SIFT

- Scale Invariant Feature Transform (SIFT)
- David Lowe, **Distinctive Image Features from Scale-Invariant Keypoints**, IJCV, 2004.
- Lowe aimed to create a descriptor that was robust to the variations corresponding to typical viewing conditions. Descriptor is the most-used part of SIFT.
- SIFT transforms image data into scale-invariant coordinates (location) corresponding to local features.

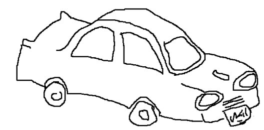
Idea of SIFT

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

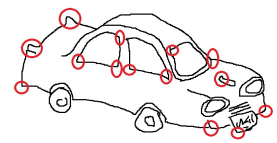


SIFT Features

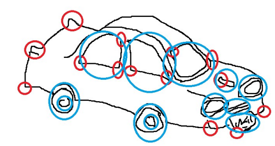
Corner vs. SIFT



Car: original



Corner only



SIFT

Correlation and convolution

- Correlation between *f(x)* and *g(x)* isdefined as

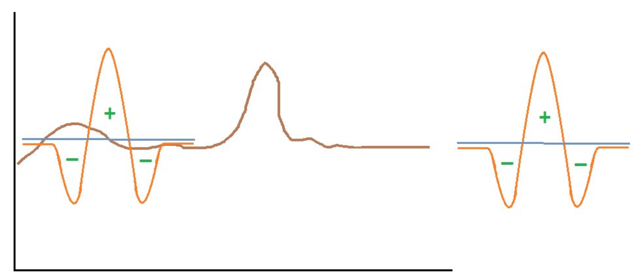
$$h(x) = \sum_{u \in D_g} f(u)g(u+x)$$

- Convolution between *f(x)* and *g(x)* is defined as

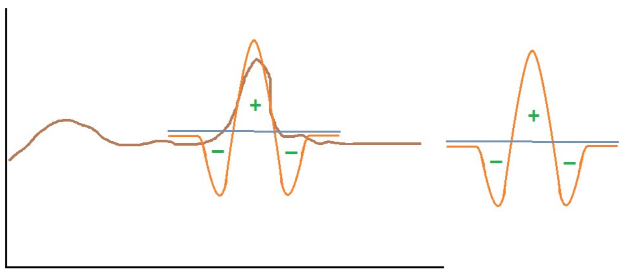
$$h(x) = \sum_{u \in D_g} f(u)g(u-x)$$

- If *h(x)* be symmetric about y-axis, correlation may obtained by computing convolution.

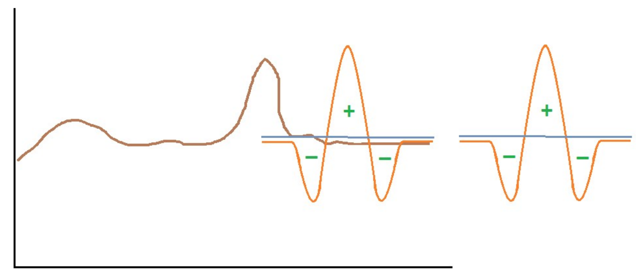
Correlation with normalized LoG



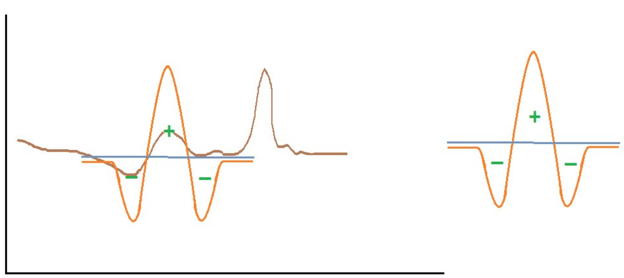
Correlation with normalized LoG



Correlation with normalized LoG



Correlation with normalized LoG



SIFT algorithm: Major steps

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

Step 1. Scale-space extrema detection

- Goal: Identify locations and scales that can be repeatably found under different views of the same scene or object.
- Method: search for stable features across multiple scales using a continuous function of scale.
  - Prior work has shown that under a variety of assumptions, the best function is a Gaussian function.
  - The scale space of an image is a function *L(x,y,kσ)* that is produced from the convolution of a Gaussian kernel (at different scales) with the input image.

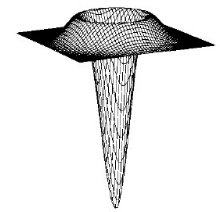
Method: Scale-space extrema detection

- Find the points, whose surrounding patches (at some scale) are distinctive.
- A plausible method is to convolve the image *I(x,y)* with Laplacian of Gaussian.
  - Scale normalized (x by scale²)
  - Proposed by Lindeberg (1994)

$$\nabla^2 S = \nabla^2 (G_\sigma * I) = \nabla^2 G_\sigma * I$$

where $\nabla^2 G(x,y,\sigma) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$

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



Method: Scale-space extrema detection

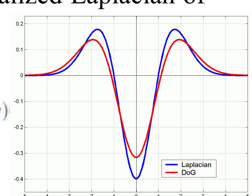
- Gaussian is an *ad hoc* solution of heat diffusion equation

$$\frac{\partial^2 G}{\partial x^2} = \sigma \nabla^2 G$$

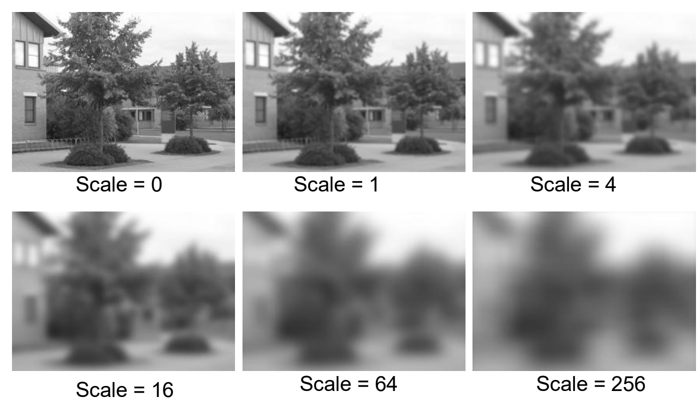
- Hence $G(x,y,k\sigma) - G(x,y,\sigma) \approx (k-1)\sigma^2 \nabla^2 G.$
- An approximation to the scale-normalized Laplacian of Gaussian is DoG:

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$$

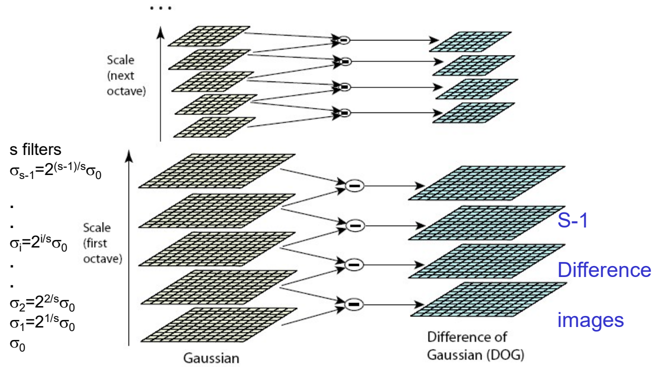
$$\begin{aligned} D(x,y,\sigma) &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ &= L(x,y,k\sigma) - L(x,y,\sigma). \end{aligned}$$



Scale-space representation



Lowe's pyramid scheme



The parameter **s** determines the number of images per octave.

number of octaves = 4, number of scale levels = 5, σ_0=1.6. k=$\sqrt{2}$

Example: Subsampling with Gaussian smoothing

Original G 1/2 G 1/4

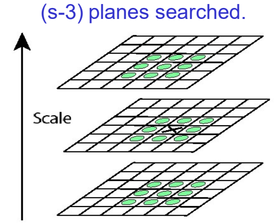
Lowe's Pyramid Scheme

- Scale space is separated into octaves:
  - Octave 1 uses scale σ
  - Octave 2 uses scale 2σ
  - so on
- In each octave, the initial image is repeatedly convolved with Gaussians to produce a set of scale space images.
- Adjacent Gaussians are subtracted to produce the DOG
- After each octave, the Gaussian image is down-sampled by a factor of 2 to start the next level.

Step 2: Keypoint localization

- Detect maxima and minima of difference-of-Gaussian in scale space
- Each point is compared to its 8 neighbors in the current image and 9 neighbors each in the scales above and below

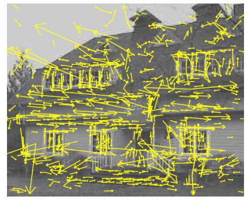
(s-1) difference images.
top and bottom ignored.
(s-3) planes searched.



For each max or min found, output is the **location** and the **scale**.

Keypoint localization

(a) 233x189 image



(b) 832 DOG extrema

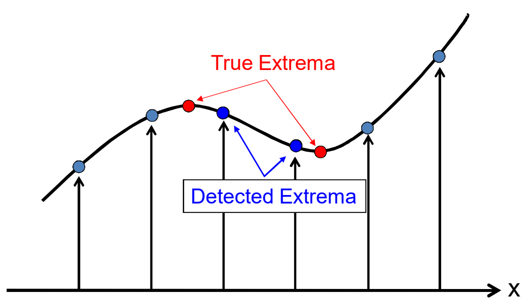
- Too many keypoints, some are unstable:
  - points with low contrast (sensitive to noise)
  - points that are localized along an edge

Keypoint localization

- Once a keypoint candidate is found, perform a detailed fit to nearby data to determine
  - Exact location, scale, and ratio of principal curvatures
- In initial work, keypoints are found at location and scale of a central sample point.
- In refinement work, they fit a 3D quadratic function to improve interpolation accuracy.
- The Hessian matrix was used to eliminate edge responses.

Keypoint localization

- The Problem:



Keypoint Localization

Low contrast points elimination:

- Fit keypoint at x to nearby data using quadratic approximation

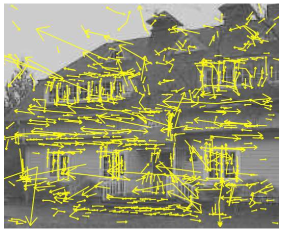
$$D(x+h) = D(x) + \frac{\partial D^T}{\partial x} h + \frac{1}{2} h^T \frac{\partial^2 D^T}{\partial x^2} h$$

- $x = (x, y, \sigma)$ and $h = (\Delta x, \Delta y, \Delta \sigma)$
- Calculate the local extrima $x + h = \hat{x}$ of the fitted function.

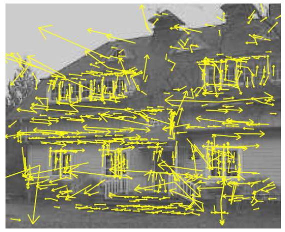
$$\hat{x} = -\left[\frac{\partial^2 D}{\partial x^2}\right]^{-1} \frac{\partial D}{\partial x}$$

- Discard local extrema $|D(\hat{x})| < 0.03$

Keypoint Localization



(a) 832 DOG extrema



(b) 729 after deleting weak extrema

729 out of 832 are left after contrast thresholding

Eliminating the Edge Response

- Reject flats: $|D(\hat{\mathbf{x}})|$ < 0.03
- Reject edges: $\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$
- Let α and β be the eigenvalues with $\alpha > \beta$.

$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$

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

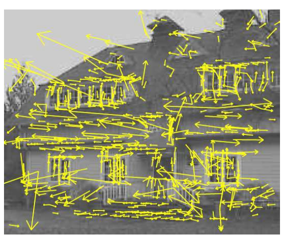
- Let r = α/β, so $\alpha = r\beta$ and r > 1.0

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

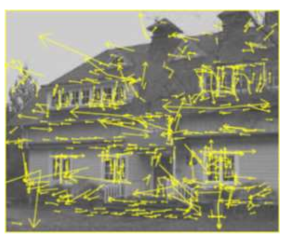
$(r+1)^2/r$ is at a min when the 2 eigenvalues are equal.

- Good choice r > 10

Keypoint Localization



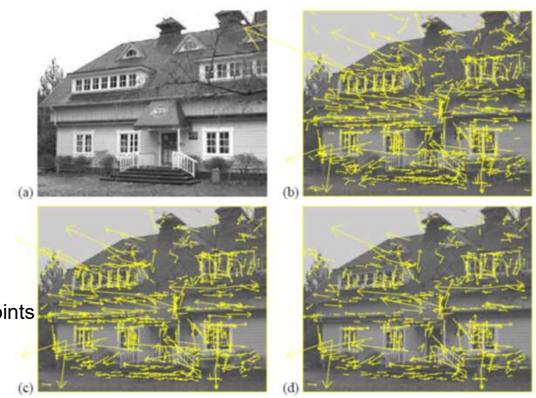
(a) 729 DOG extrema



(b) 536 after ratio thresholding

536 out of 832 are left after contrast and ratio thresholding

Keypoint Localization



832 keypoints

729 keypoints

536 keypoints

Step 3: Orientation assignment

- Assign an orientation to each keypoint, the keypoint descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation.
- A neigbourhood is taken around the keypoint location depending on the scale.
- Compute magnitude and orientation of gradient on the Gaussian smoothed images.

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

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Orientation assignment

- A histogram is formed by quantizing the (360 degrees) orientations into 36 bins.
- Vote is gradient magnitude and Gaussian-weighted window with σ =1.5 times the scale of the keypoint.
- Peaks (and also 80% of it) in the histogram are considered to compute orientation of the patch.
- At the same location, there could be multiple keypoints with different orientations.

Step 4: Keypoint Descriptors

- At this point, each keypoint has
  - location
  - scale
  - orientation
- Next is to compute a descriptor for the local image region about each keypoint that is
  - highly distinctive
  - invariant (as much as possible) to variations or changes in viewpoint and illumination

Normalization

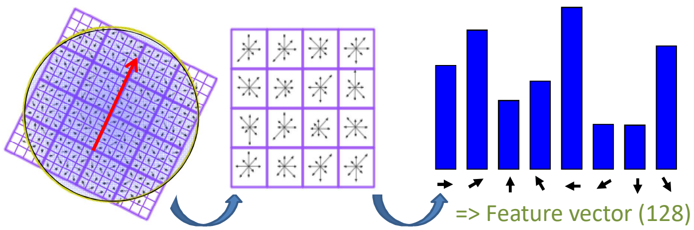
- Rotate the window to standard orientation
- Scale the window size based on the scale at which the point was found.

Remaining goal:

- Define **local** descriptor invariant to remaining variations:
  - Illumination
  - 3D Viewpoint

Keypoint descriptor

- Create 16 gradient histograms (8 bins)
  - Weighted by magnitude and Gaussian window (σ is half the window size)
  - Histogram and gradient values are interpolated and smoothed



Lowe's Keypoint Descriptor

- Use the normalized region about the keypoint
- Compute gradient magnitude and orientation at each point in the region
- Weight them by a Gaussian window overlaid on the circle
- Create an orientation histogram over the 4X4 subregions of the window
- 4X4 descriptors over 16X16 sample array were used. 4X4 times 8 directions gives a vector of 128 values.

Thank you !

Any question?