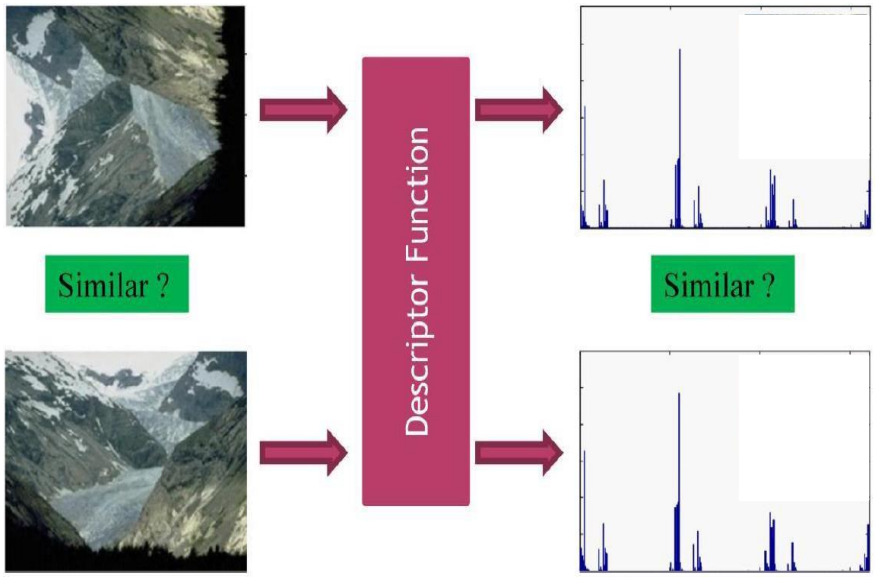
Feature Detection and Discription



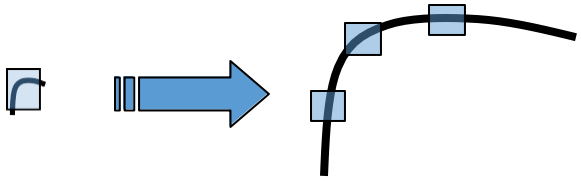
Computer Vision

Adduru U G Sankararao, IIIT Sri City

Previous Class

- Keypoint detection: repeatable and distinctive
  - Corners,
  - Invariant to scale, rotation, etc.

- Harris Corner Detection
  - Rotation Invariant
  - Partial Intensity Change Invariant
  - *Not Invariant to Scale*

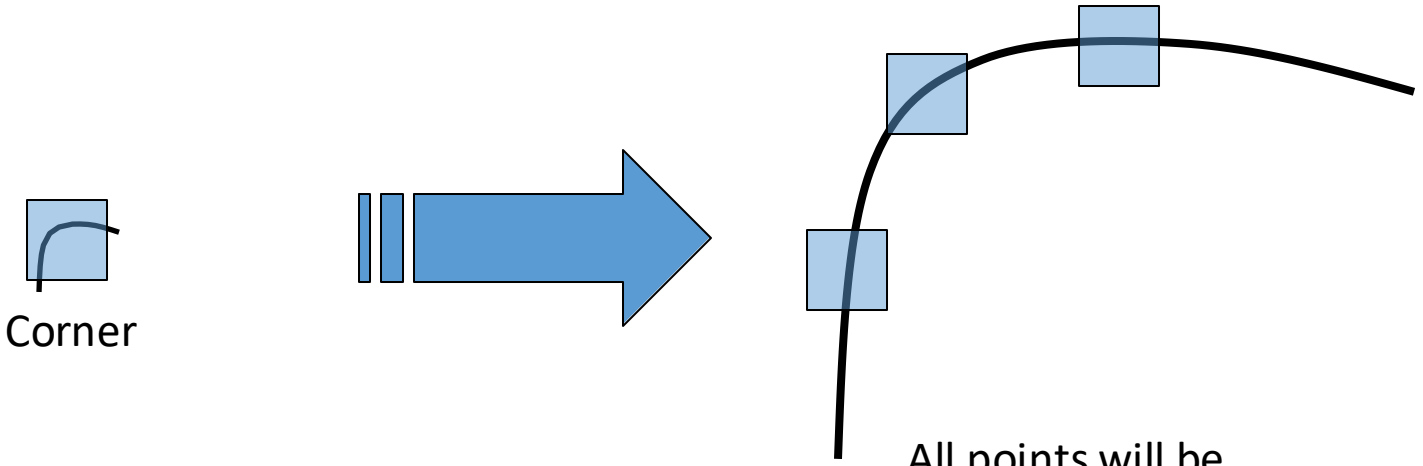


This lecture

- Scale Space, Image Pyramids, Filter Banks
- Scale Invariant Feature Transform (SIFT)

Harris Detector: Invariance Properties

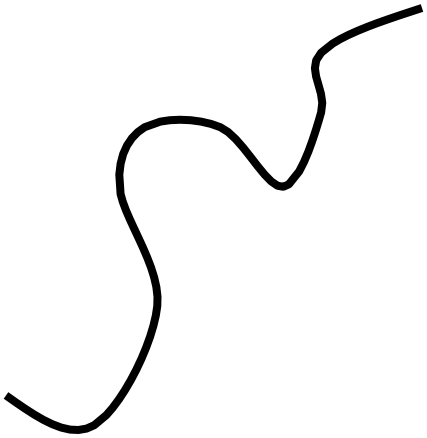
- Scaling



Not invariant to scaling

Scale invariant detection

Suppose you're looking for corners

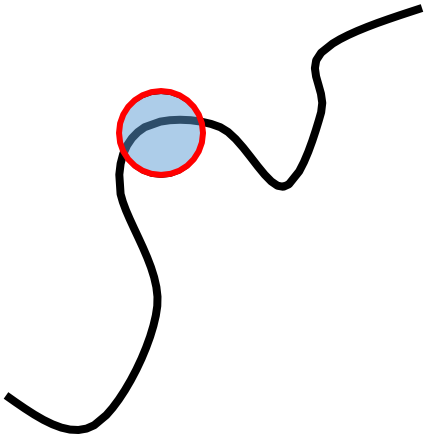


Key idea: find scale that gives local maximum of *f*

- in both position and scale
- One definition of *f*: the Harris operator

Scale invariant detection

Suppose you're looking for corners

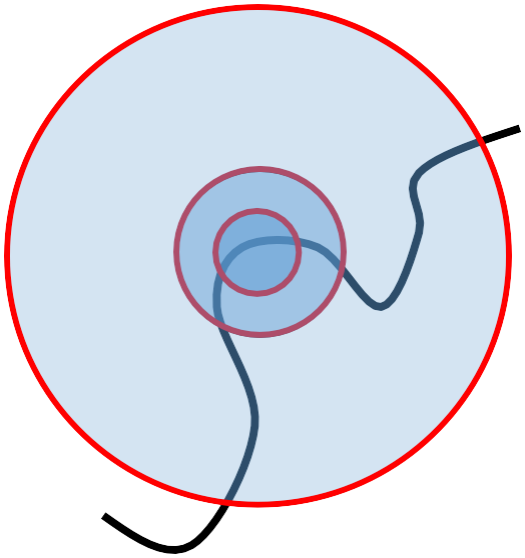


Key idea: find scale that gives local maximum of *f*

- in both position and scale
- One definition of *f*: the Harris operator

Scale invariant detection

Suppose you're looking for corners



Key idea: find scale that gives local maximum of *f*

- in both position and scale
- One definition of *f*: the Harris operator

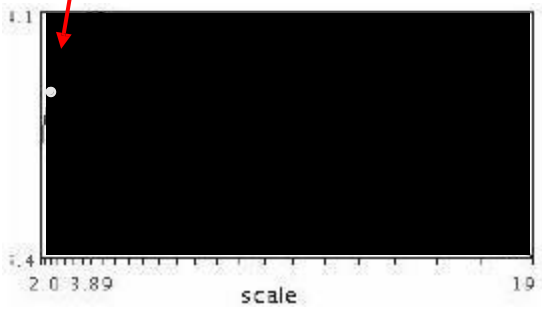
Automatic Scale Selection

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

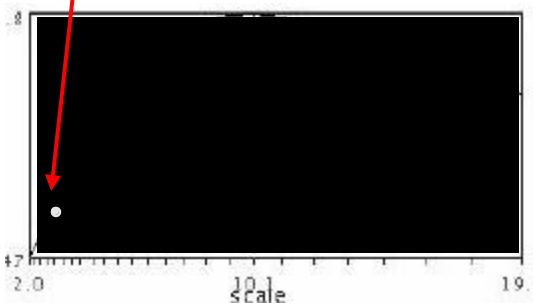
How to find corresponding patch sizes?

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



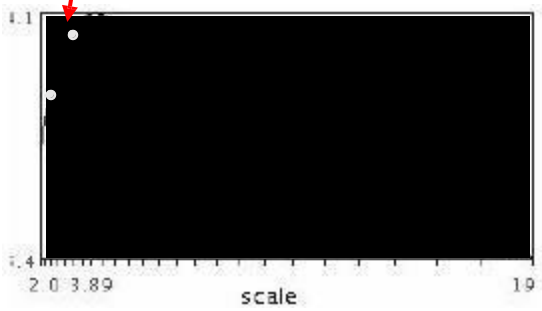
$f(I_{i_1 \ldots i_m}(x,\sigma))$



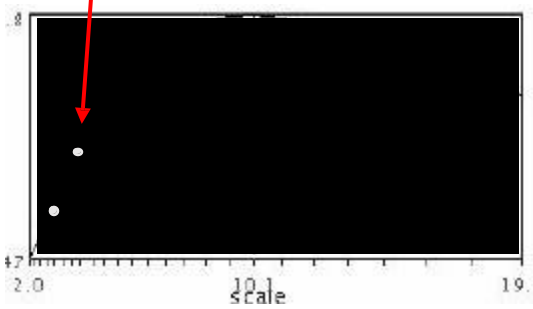
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



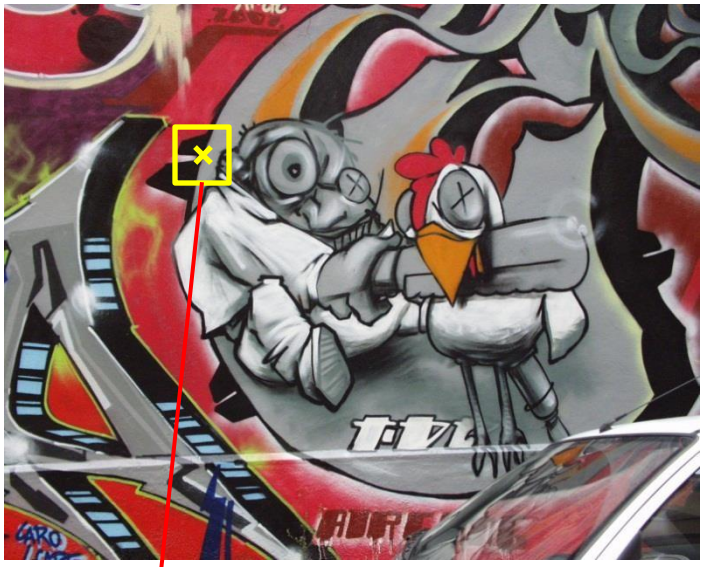
$f(I_{i_1 \ldots i_m}(x,\sigma))$

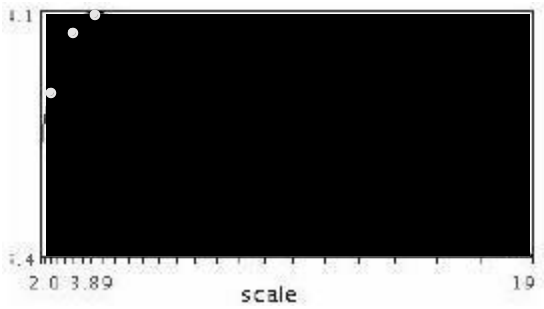


$f(I_{i_1 \ldots i_m}(x',\sigma))$

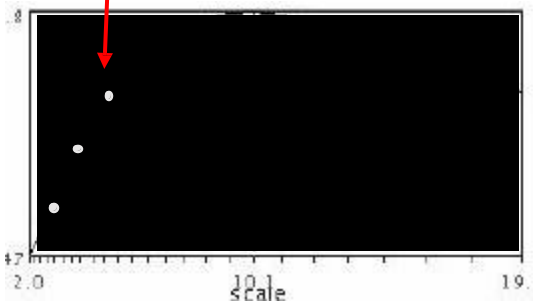
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





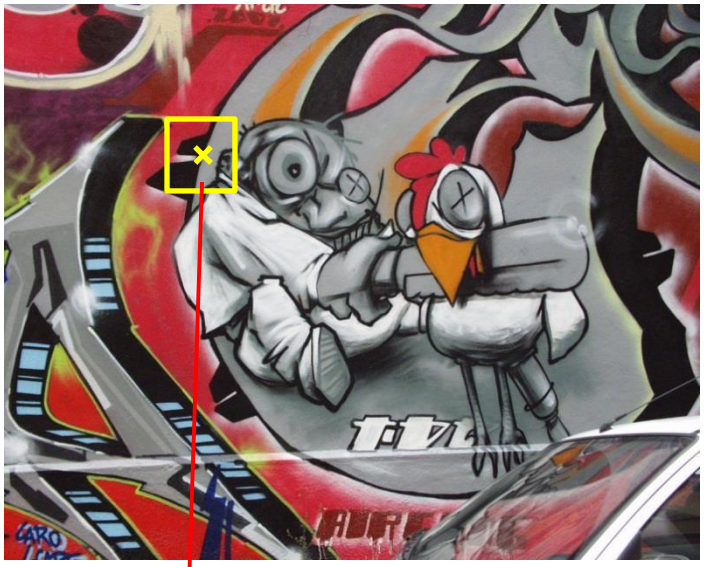
$f(I_{i_1 \ldots i_m}(x,\sigma))$

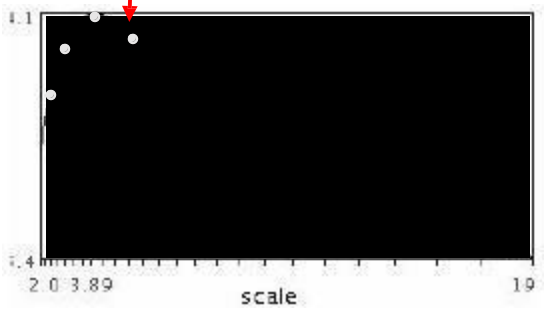


$f(I_{i_1 \ldots i_m}(x',\sigma))$

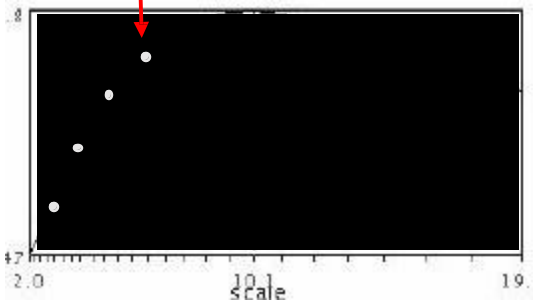
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





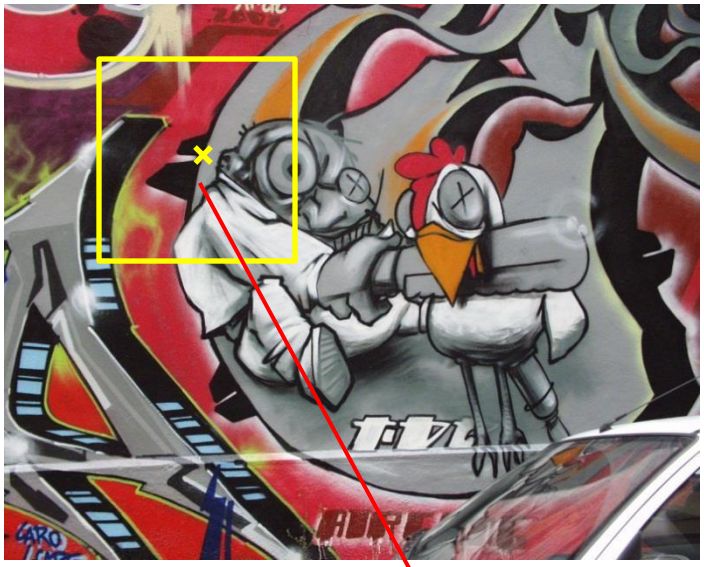
$f(I_{i_1 \ldots i_m}(x,\sigma))$

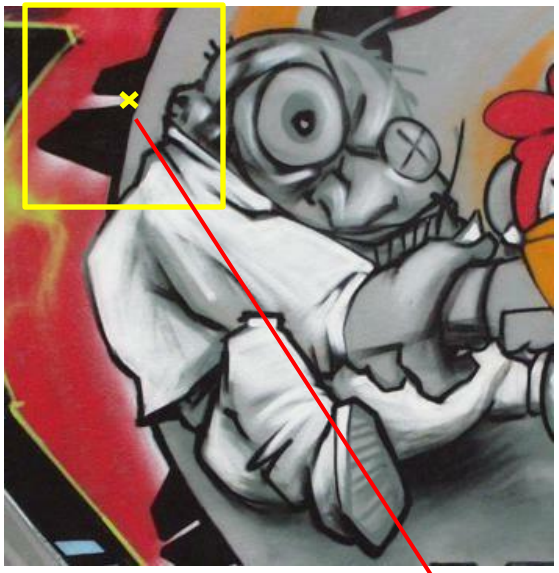


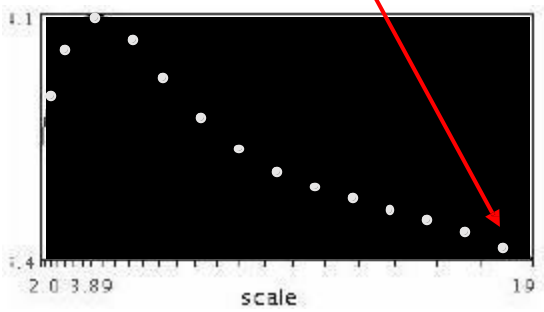
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Automatic Scale Selection

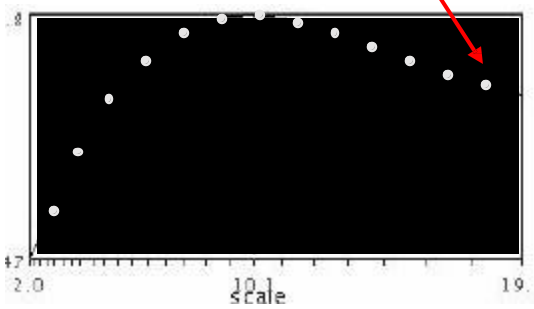
- Function responses for increasing scale (scale signature)







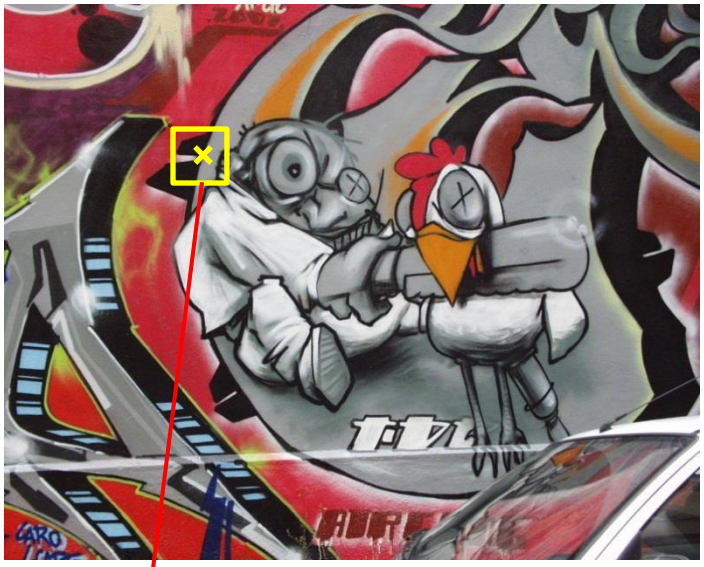
$f(I_{i_1 \ldots i_m}(x,\sigma))$

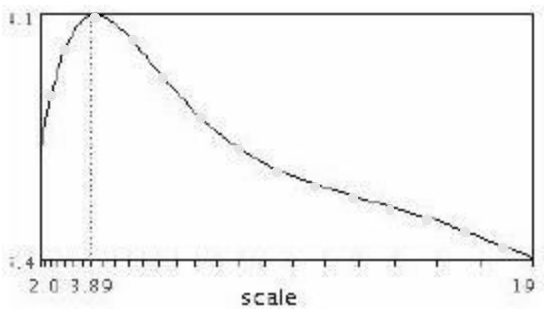


$f(I_{i_1 \ldots i_m}(x',\sigma))$

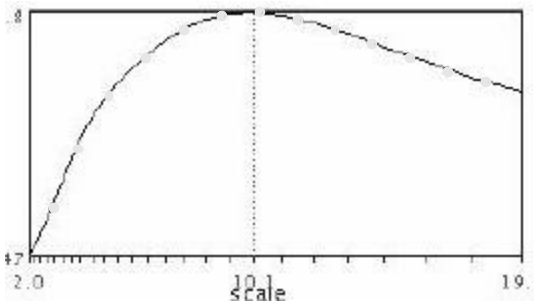
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





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



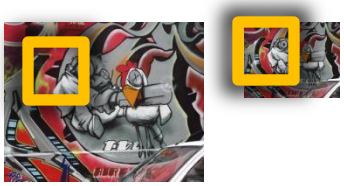
$f(I_{i_1 \ldots i_m}(x',\sigma'))$

K. Grauman, B. Leibe

Is there a better way to do this?

Automatic Scale Selection:Implementation

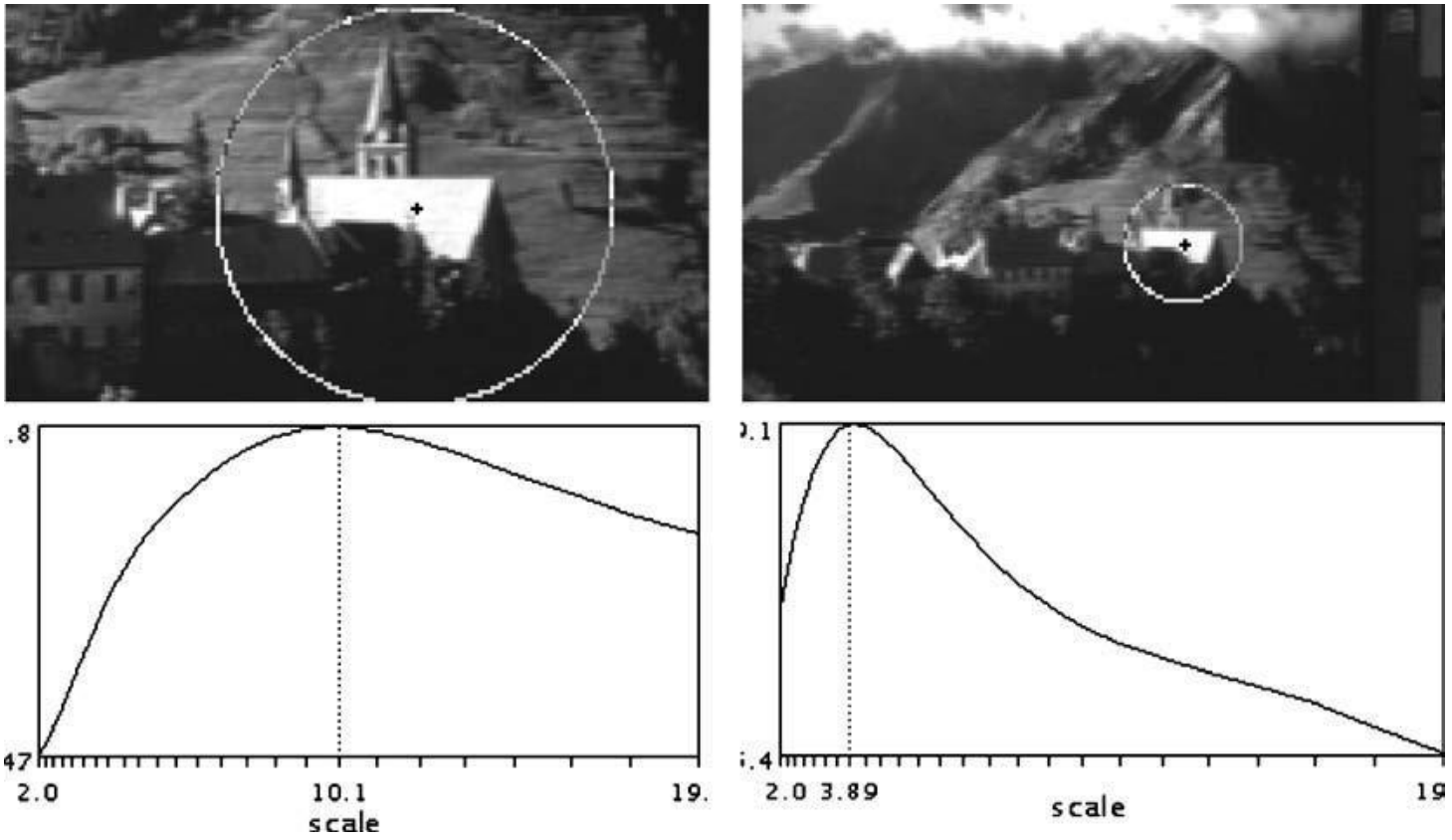
- Instead of computing f for larger and larger windows, we can implement using a fixed window size with a Gaussian pyramid



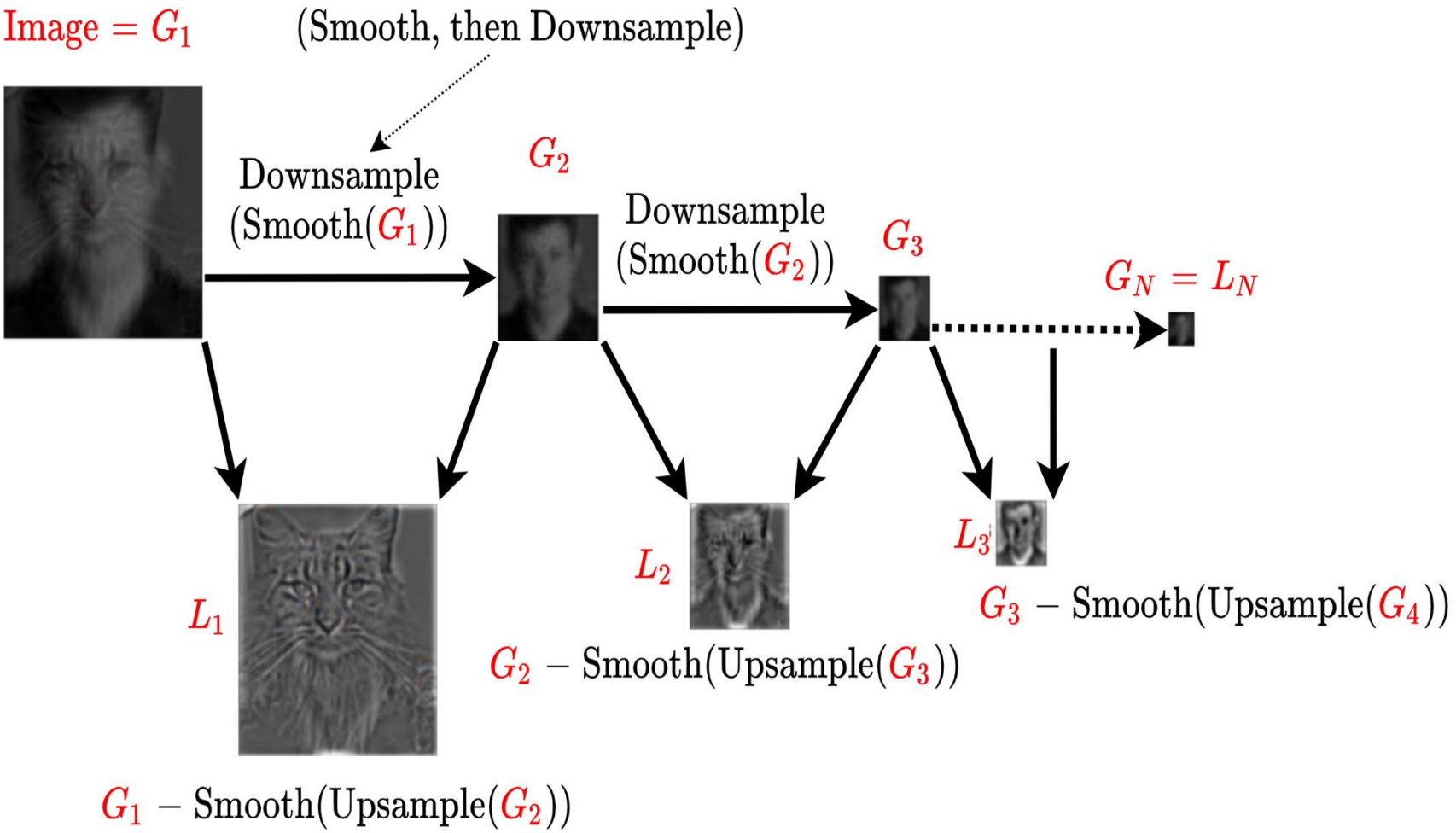
(sometimes need to create in-between levels, e.g. a ¾-size image)

Keypoint detection with scale selection

- We want to extract keypoints with characteristic scale that is *covariant* with the image transformation



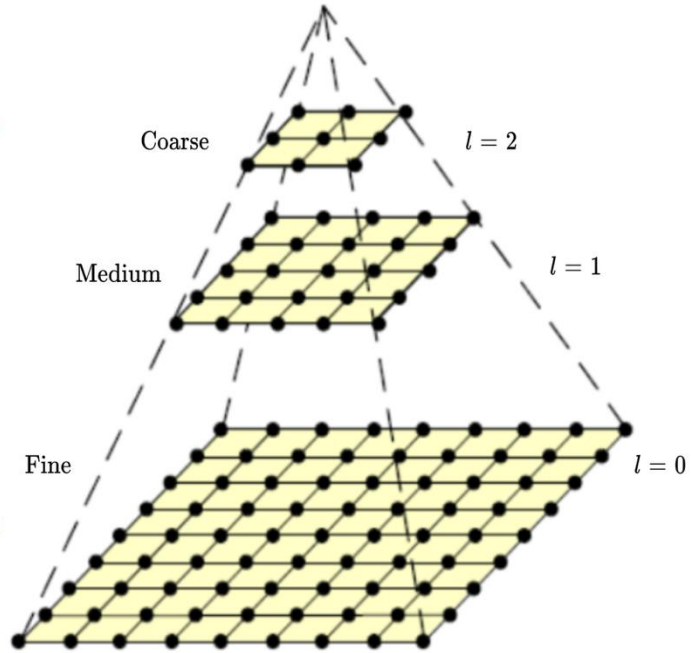
Gaussian and Laplacian Pyramid



Credit: Derek Hoiem

Images Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration: Coarse-to-fine Image Registration

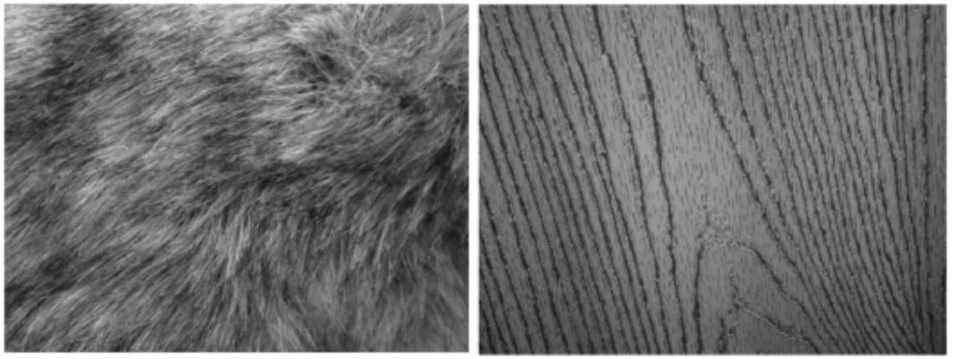


- *Compute Gaussian pyramid.*
- *Align with coarse pyramid.*
- *Successively align with finer pyramids.*
  - *Search smaller range.*

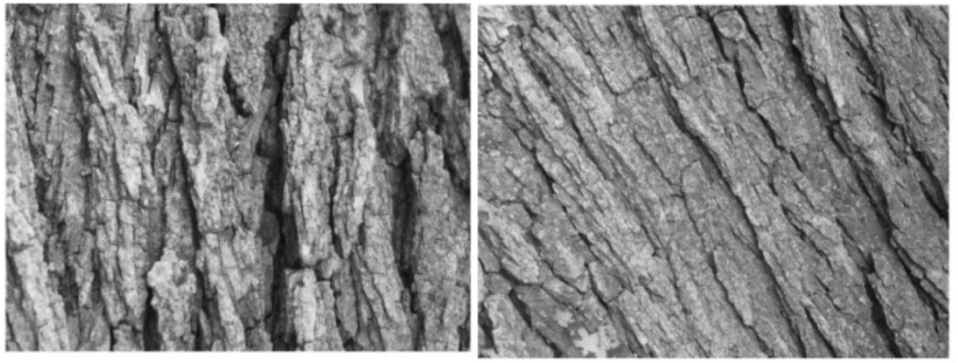
Textures in Images

- Regular or stochastic patterns caused by bumps, grooves and/or markings.
- Gives us information about spatial arrangement of colours or intensities in an image.

Different Materials



Different Orientation

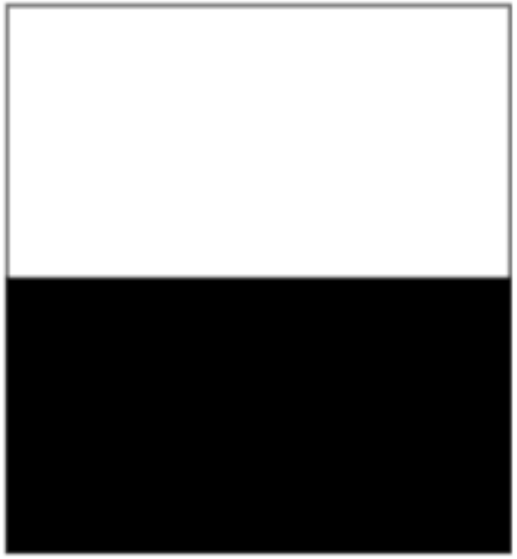


Different Scales

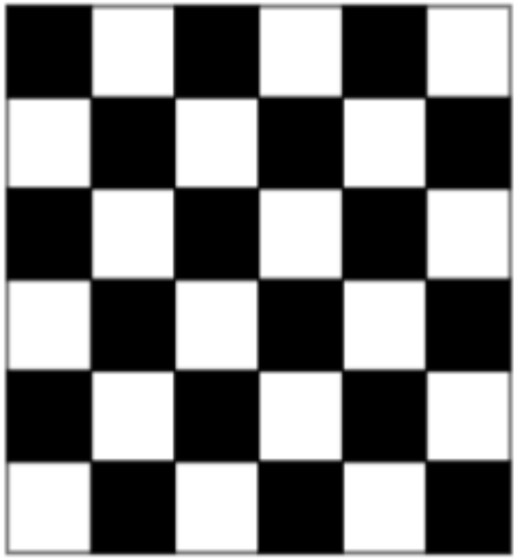
Textures in Images

- Conveys more information that can be exploited to match regions of interest in images.

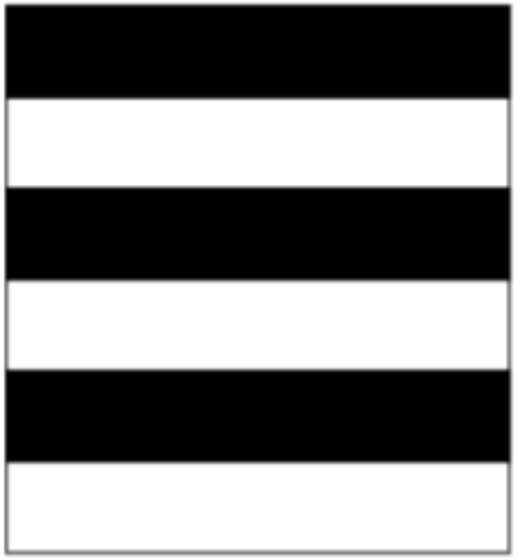
Histogram conveys 50% white pixels and 50% black pixels



(Block Pattern)



(Checkerboard Pattern)



(Striped Pattern)

Drastically different textures

Credit: Linda G Shapiro

Textures in Images

- Conveys more information that can be exploited to match regions of interest in images.

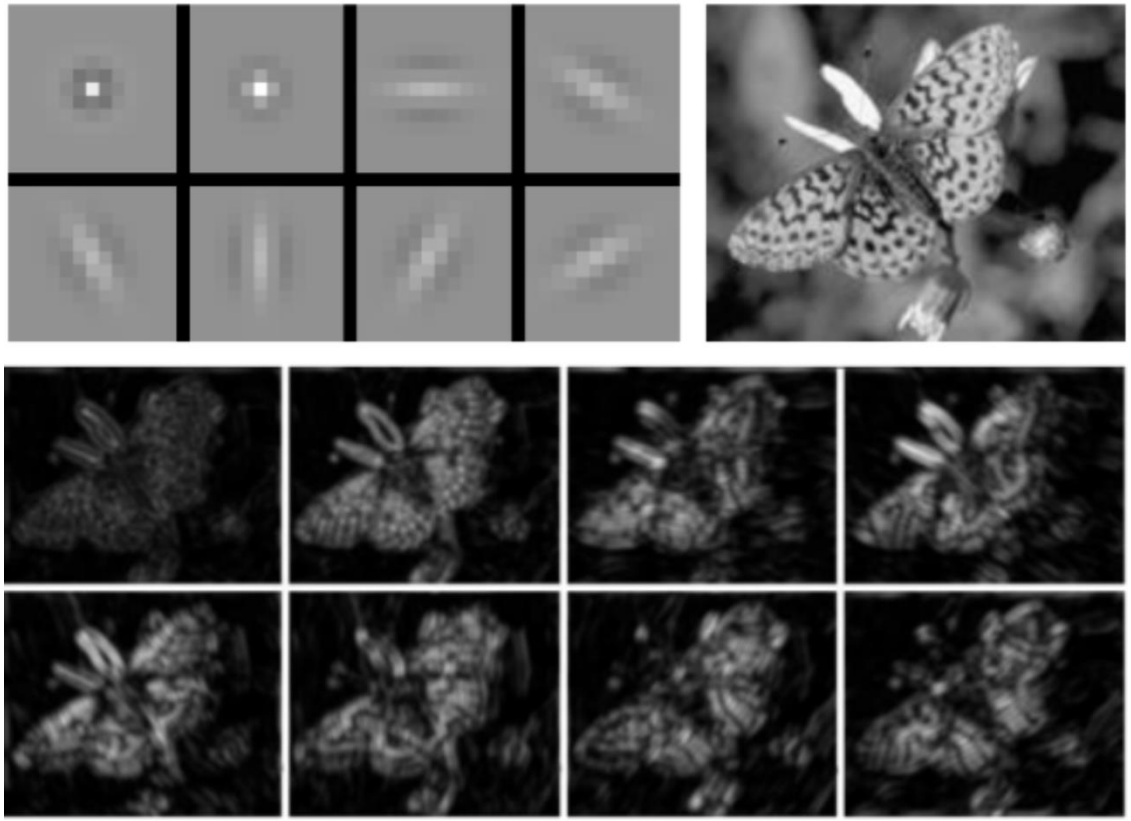
How to represent textures?

- Compute responses of blobs and edges at various orientations and scales.
- Ways to process:
  - Record simple statistics (e.g., mean, std.) of absolute filter responses.
  - Take vectors of filter responses at each pixel and cluster them.

Credit: Linda G Shapiro

Filter Banks

- An array of bandpass filters that separates the input signal into multiple components, each one carrying a single frequency sub-band of the original signal.
- Process image with each filter and keep responses (or squared/abs responses).

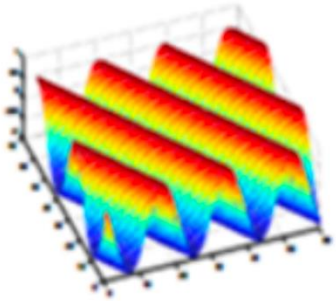


Credit: Derek Hoiem

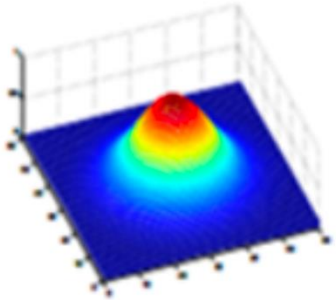
Gabor Filters

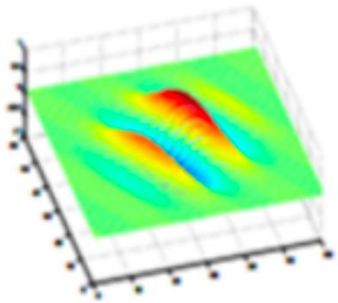
- Special classes of bandpass filters (i.e., they allow a certain `band' of frequencies and reject the others).
- A Gabor filter can be viewed as a sinusoidal signal of particular frequency and orientation, modulated by a Gaussian wave.

A 2-D Gaussian



A sinusoid oriented 30° with x-axis



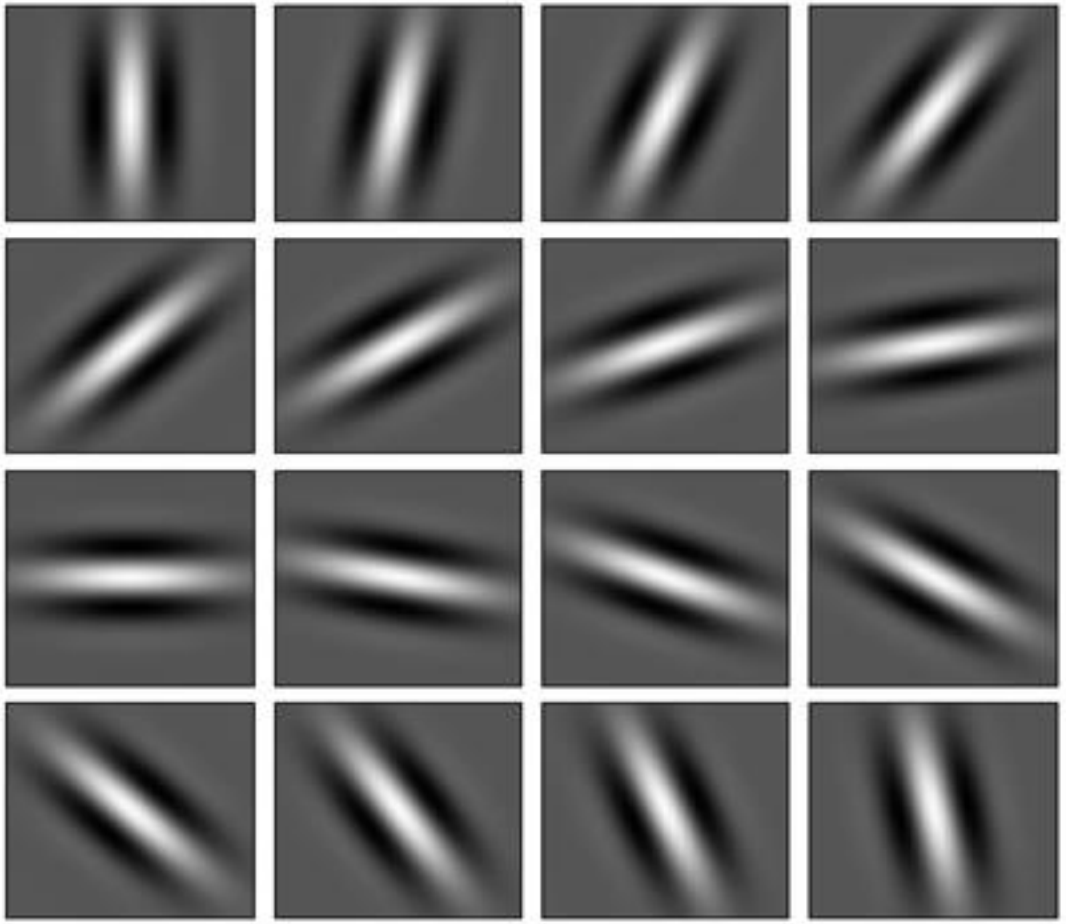


A corresponding 2-D Gabor Filter

A 2-D Gabor filter obtained by modulating the sine wave with a Gaussian

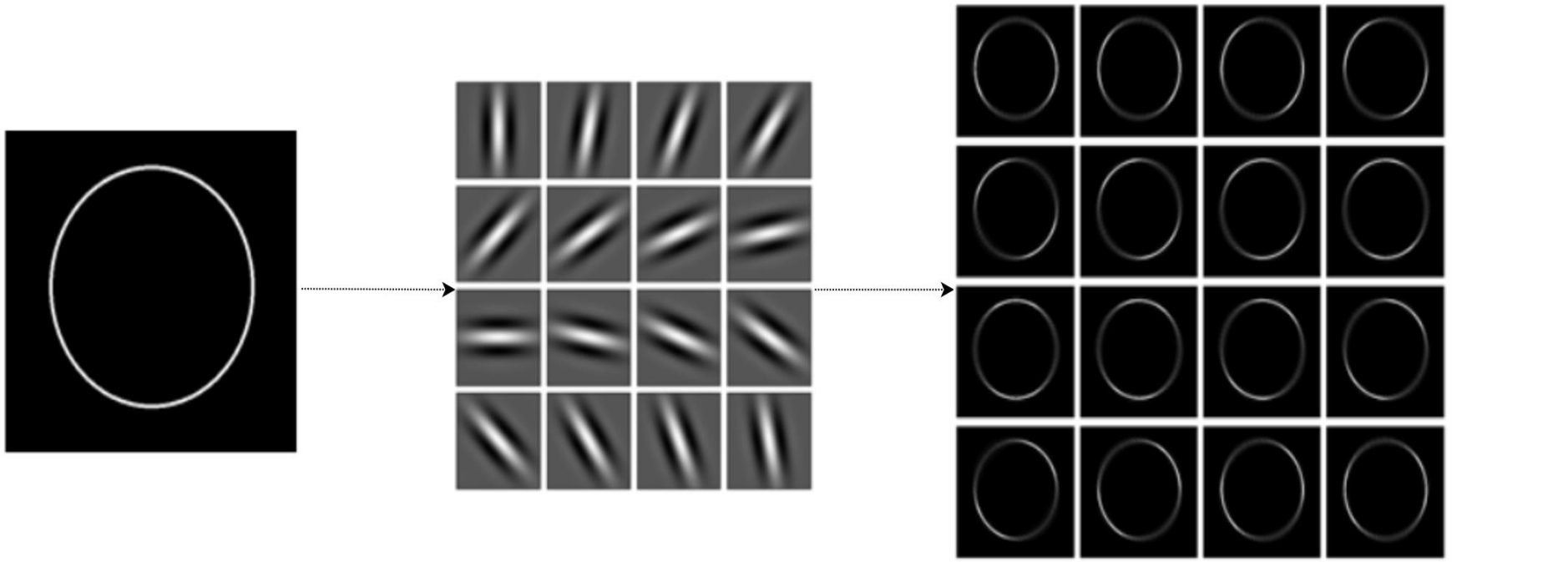
Gabor Filter Banks

- Bank of 16 Gabor filters at an orientation of 11.250 (i.e. if the first filter is at 00.00, then the second will be at 11.25, the third will be at 22.50, and so on.)



Gabor Filter Banks

- Bank of 16 Gabor filters at an orientation of 11.250 (i.e. if the first filter is at 00.00, then the second will be at 11.25, the third will be at 22.50, and so on.)



Input Image of a Circle

A bank of 16 Gabor filters

Output Image of the circle after passed through indiviual Gabor filters

Scale Invariant Feature Transform (SIFT)

- David G. Lowe, Distinctive Image Features from Scale-invariant Keypoints, IJCV 2004: Over 50000 citations
- Transforms image data into scale-invariant coordinates
- Fundamental to many core vision problems/applications:
  - Recognition, Motion tracking, Multiview geometry

Widely Used Local Descriptors

- SIFT – Scale Invariant Feature Transform

Distinctive image features from scale-invariant keypoints

DG Lowe - International journal of computer vision, 2004 - Springer

… the assigned orientation, **scale**, and loca- tion for each **feature**, thereby providing **invariance** to these … **Invariant** Fea- ture **Transform** (SIFT), as it **transforms** image data into **scale-invariant** coordinates relative … that densely cover the image over the full range of **scales** and locations …

☆ 99 Cited by 50252 Related articles All 179 versions

- LBP – Local Binary Pattern

Multiresolution gray-scale and rotation invariant texture classification with local binary patterns

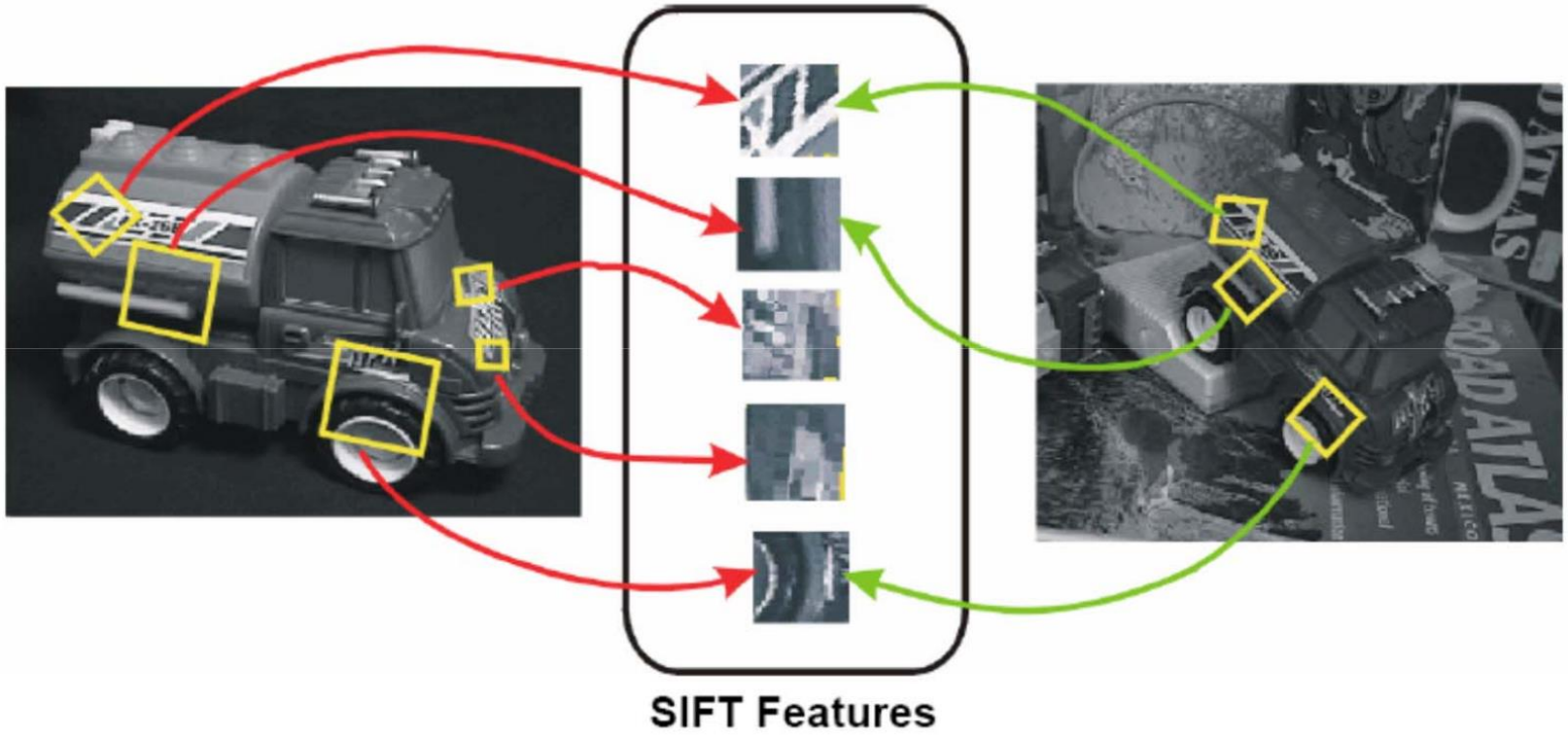
T Ojala, M Pietikainen… - … Transactions on **pattern** …, 2002 - ieeexplore.ieee.org

Presents a theoretically very simple, yet efficient, multiresolution approach to gray-scale and rotation invariant texture classification based on **local binary patterns** and nonparametric discrimination of sample and prototype distributions. The method is based on recognizing …

☆ 99 Cited by 12251 Related articles All 16 versions

SIFT: Invariant Local Features

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, shear.



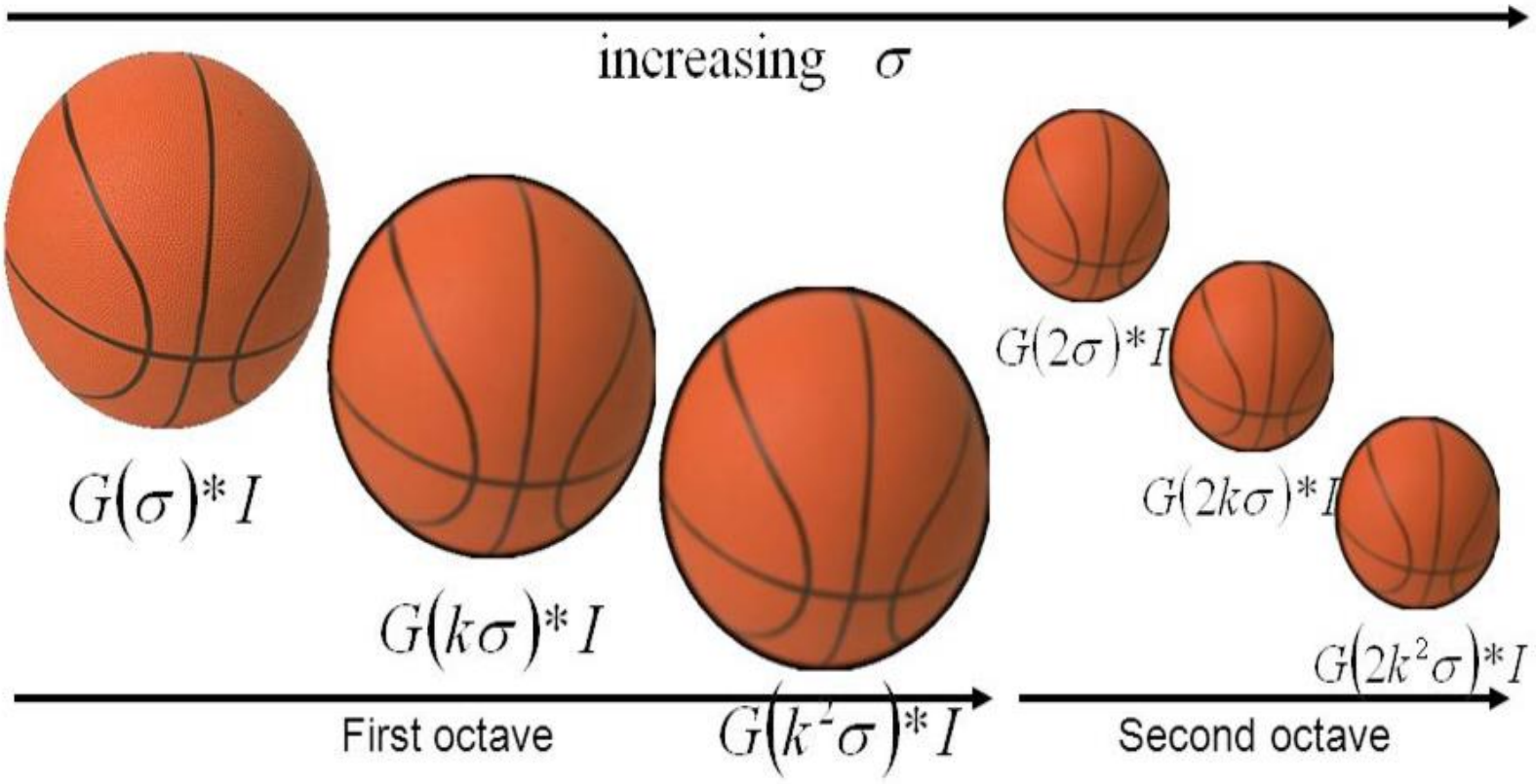
Credit: Mubarak Shah, University of Central Florida

SIFT:

- Step 1: Scale-space Extrema Detection - Detect interest points (invariant to scale and orientation) using DOG.
- Step 2: Keypoint Localization - Determine location and scale at each candidate location, and select them based on stability.
- Step 3: Orientation Estimation - Use local image gradients to assign orientation to each localized keypoint. Preserve orientation, scale and location for each feature.
- Step 4: Keypoint Descriptor - Extract local image gradients at selected scale around keypoint and form a representation invariant to local shape and illumination distortion.

SIFT: Scale Space Extrema Detection

- Constructing Scale-space



Credit: Ofir Pele

SIFT: Scale Space Extrema Detection

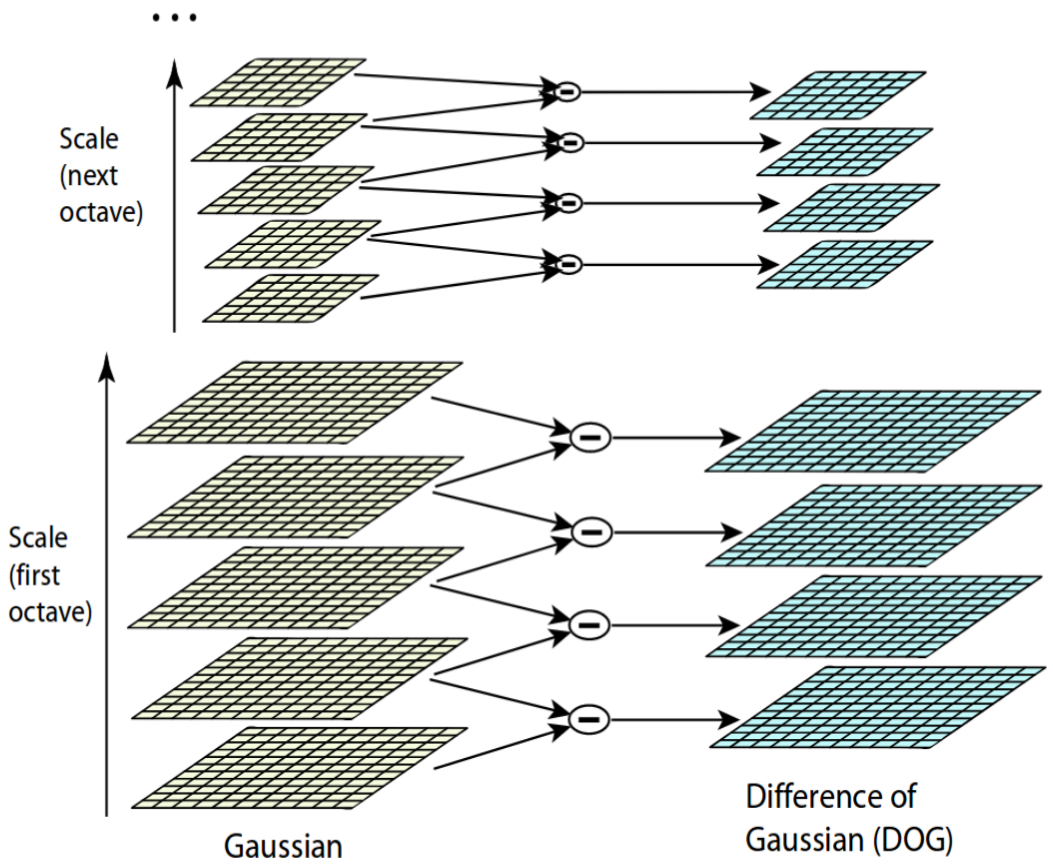
- Difference of Gaussians

  An octave corresponds to doubling the value of σ

$$D(x,y,\sigma) = \hat{I}(x,y,\sigma) - \hat{I}(x,y,k\sigma)$$

where $\hat{I}(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$

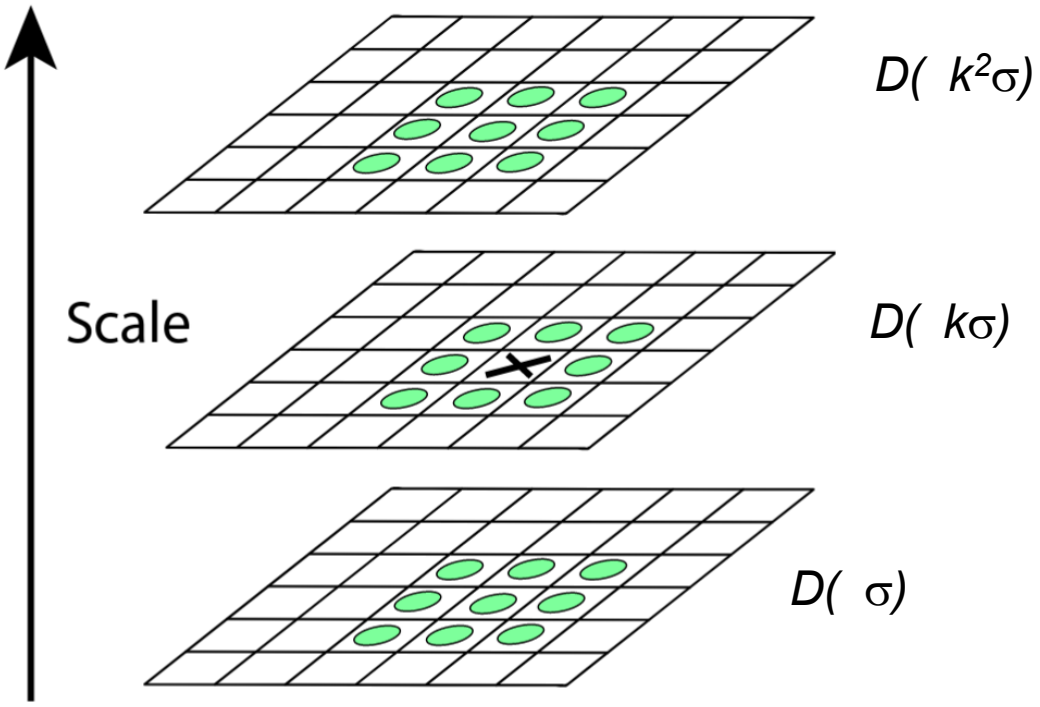
Original image



Credit: Distinctive Image Features from Scale-Invariant Points", IJCV 2004

SIFT: Scale Space Extrema Detection

- Compare a pixel (X) with 26 pixels in current and adjacent scales (Green Circles)
- Select a pixel (X) if it is larger/smaller than all 26 pixels

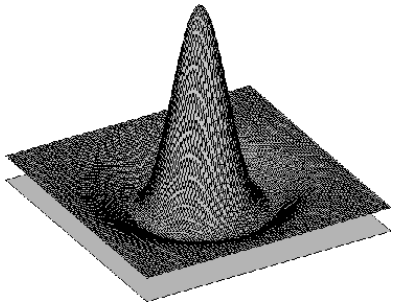


Credit: Distinctive Image Features from Scale-Invariant Points", IJCV 2004

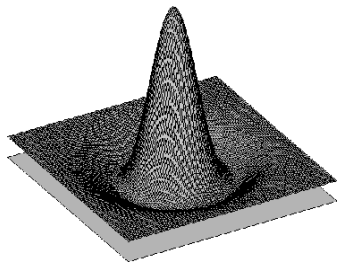
Difference-of-Gaussian (DoG)

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

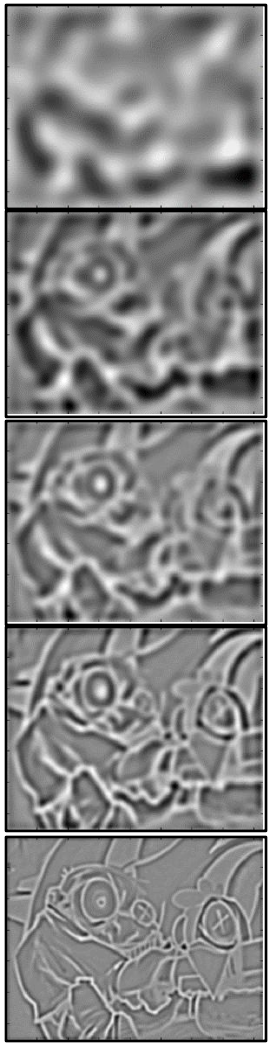
(Difference of Gaussians)

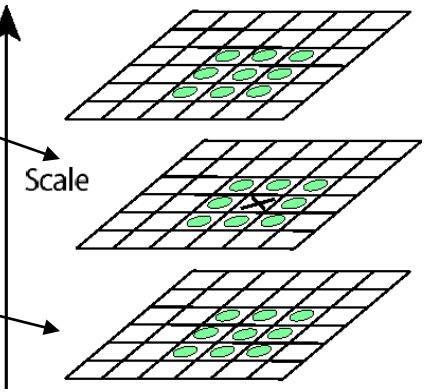


Find local maxima in position-scale space of Difference-of-Gaussian



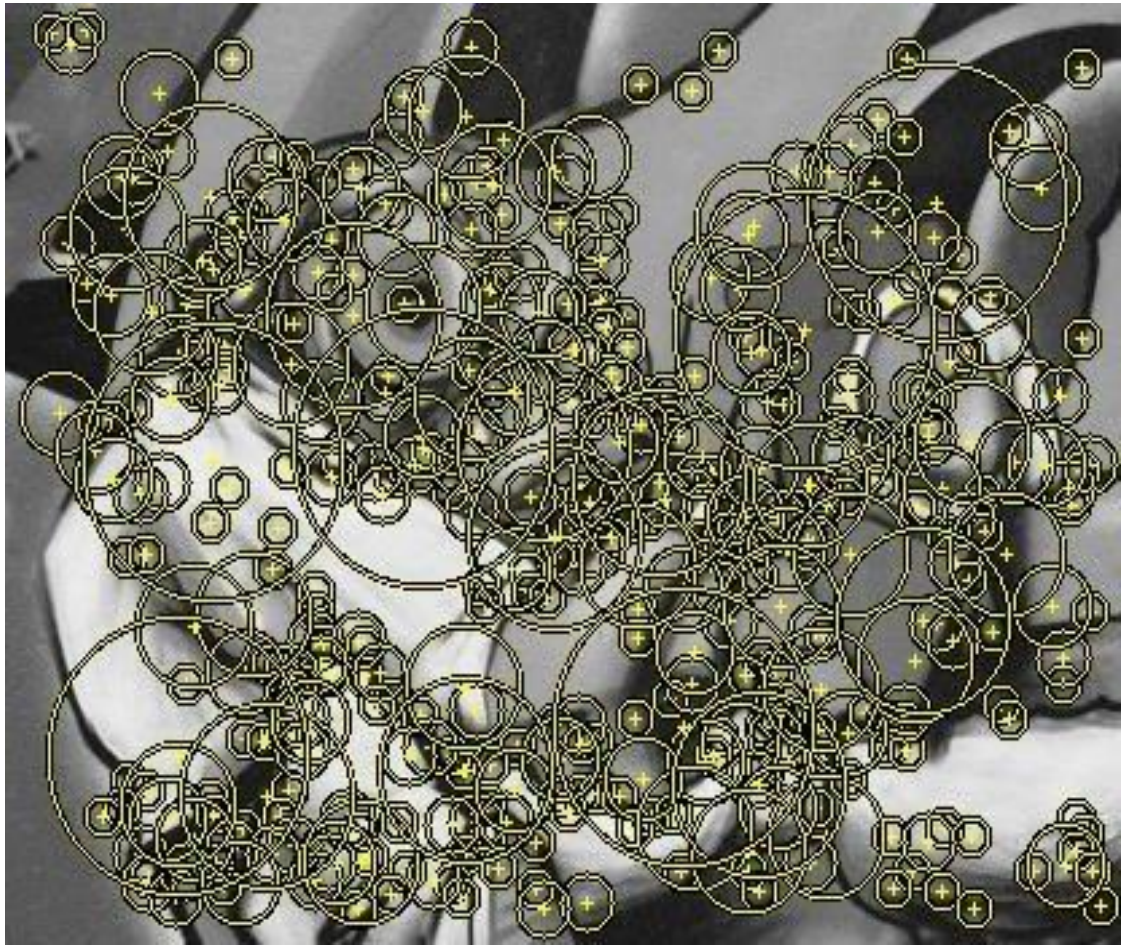
$$L_{xx}(\sigma) + L_{yy}(\sigma)$$





$\Rightarrow$ List of
(x, y, s)

Results: Difference-of-Gaussian

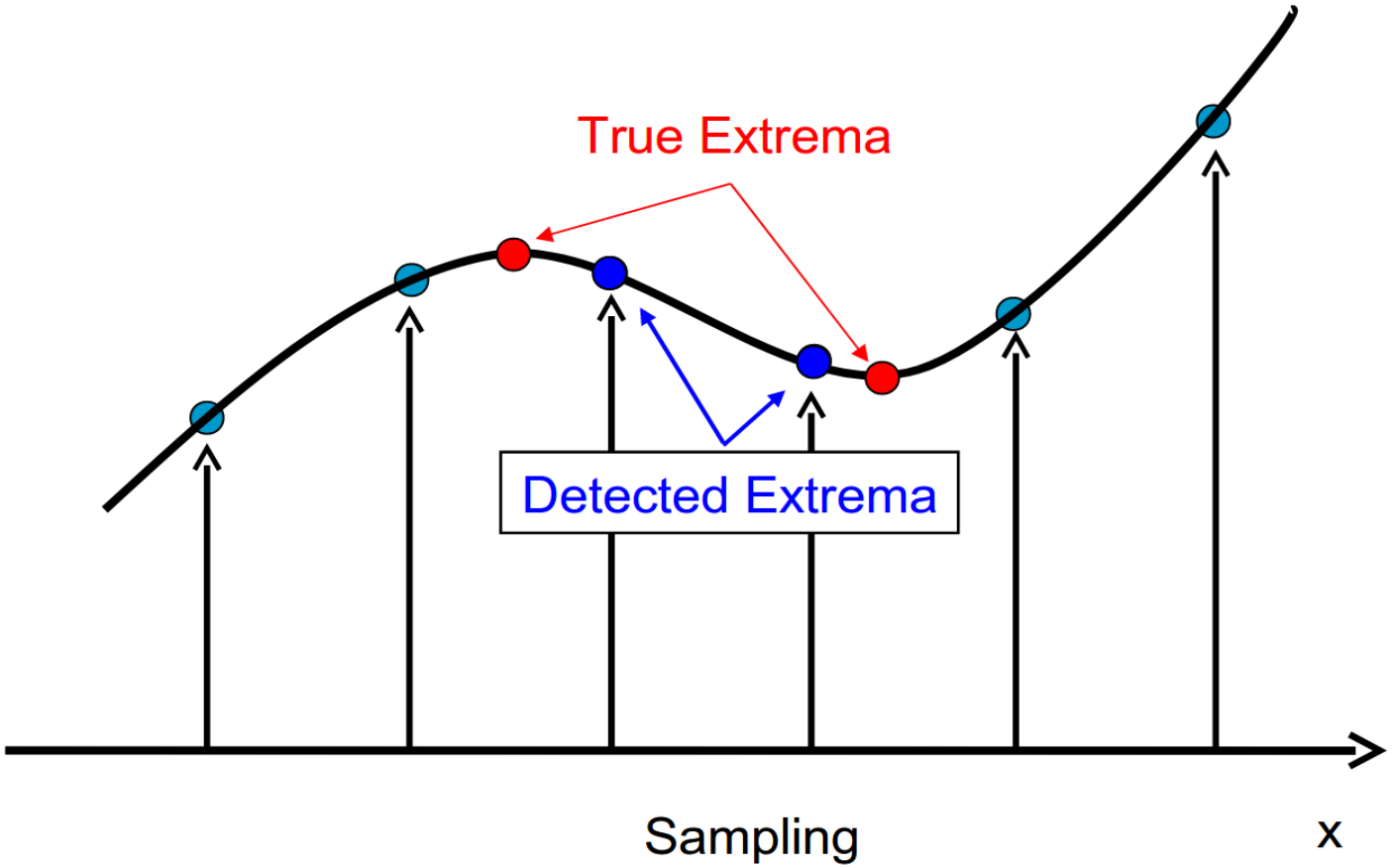


SIFT Algorithm Stages

- Step 1: Scale-space Extrema Detection - Detect interest points (invariant to scale and orientation) using DOG.
- Step 2: Keypoint Localization - Determine location and scale at each candidate location, and select them based on stability.
- Step 3: Orientation Estimation - Use local image gradients to assign orientation to each localized keypoint. Preserve orientation, scale and location for each feature.
- Step 4: Keypoint Descriptor - Extract local image gradients at selected scale around keypoint and form a representation invariant to local shape and illumination distortion.

SIFT: Keypoint Localization

- The problem:



Credit: Ofir Pele

SIFT: Keypoint Localization

- Use Taylor series expansion of the scale-space function:

$$D(\mathbf{s}_0 + \Delta\mathbf{s}) = D(\mathbf{s}_0) + \frac{\partial D}{\partial \mathbf{s}}^T \Big|_{\mathbf{s}_0} \Delta\mathbf{s} + \frac{1}{2}\Delta\mathbf{s}^T \frac{\partial^2 D}{\partial \mathbf{s}^2}\Big|_{\mathbf{s}_0} \Delta\mathbf{s}$$

where $\mathbf{s}_0 = (x_0, y_0, \sigma_0)^T$ and $\Delta\mathbf{s} = (\delta x, \delta y, \delta\sigma)^T$

- The location of the extremum, s^, is determined by taking the derivative of this function with respect to s and setting it to zero:

$$\hat{\mathbf{s}} = -\left(\frac{\partial^2 D}{\partial \mathbf{s}^2}\Big|_{\mathbf{s}_0}\right)^{-1} \frac{\partial D}{\partial \mathbf{s}}\Big|_{\mathbf{s}_0}$$

- Next, reject low contrast points and points that lie on the edges

- Low contrast points elimination: Reject keypoint if D(s^) is smaller than 0.03 (assuming image values are normalized in [0,1])

SIFT: Keypoint Localization

- Reject key points with strong edge response in one direction only
- Edge elimination: Similar to Harris corner detector. SIFT instead uses Hessian

- Compute Hessian of D (principal curvature)

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

α : largest eigenvalue(λ_{max})

β : smallest eigenvalue(λ_{min})

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

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

- Evaluate ratio

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

$$\frac{Tr(H)^2}{Det(H)} = \frac{(r+1)^2}{r} \text{ where } r = \frac{\alpha}{\beta}$$

- The ratio is minimum when r = 1 (eigen values are equal)
- Reject keypoint if the ratio is > a threshold (r = 10 for SHIFT)

SIFT Algorithm Stages

- Step 1: Scale-space Extrema Detection - Detect interest points (invariant to scale and orientation) using DOG.
- Step 2: Keypoint Localization - Determine location and scale at each candidate location, and select them based on stability.
- Step 3: Orientation Estimation - Use local image gradients to assign orientation to each localized keypoint. Preserve orientation, scale and location for each feature.
- Step 4: Keypoint Descriptor - Extract local image gradients at selected scale around keypoint and form a representation invariant to local shape and illumination distortion.

SIFT: Orientation Estimation

- Why? To achieve **rotation invariance**
- Use scale of point to choose correct image:

$$\hat{I}(x,y) = G(x,y,\sigma) * I(x,y)$$

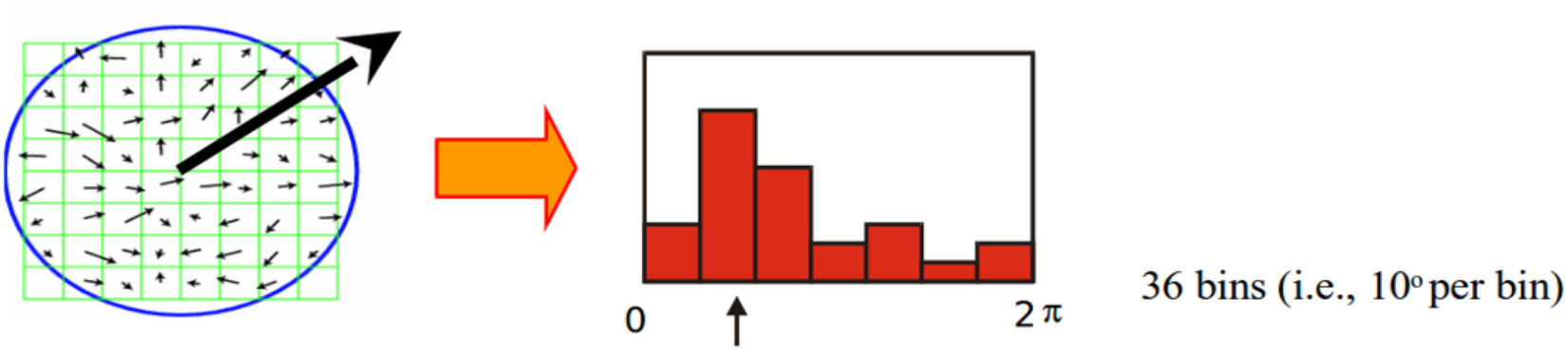
- Compute gradient magnitude and orientation using finite differences:

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

$$\theta(x,y) = \tan^{-1}\left(\frac{(\hat{I}(x,y+1) - \hat{I}(x,y-1))}{(\hat{I}(x+1,y) - \hat{I}(x-1,y))}\right)$$

SIFT: Orientation Estimation

- Create histogram of gradient directions, within a region around the keypoint, at selected scale:



- Histogram entries are weighted by:
  - gradient magnitude, and
  - a Gaussian function with equal to 1.5 times scale of the keypoint
- Select the peak as direction of keypoint
- Introduce additional key points at same location if another peak is within 80% of max peak of histogram with different direction

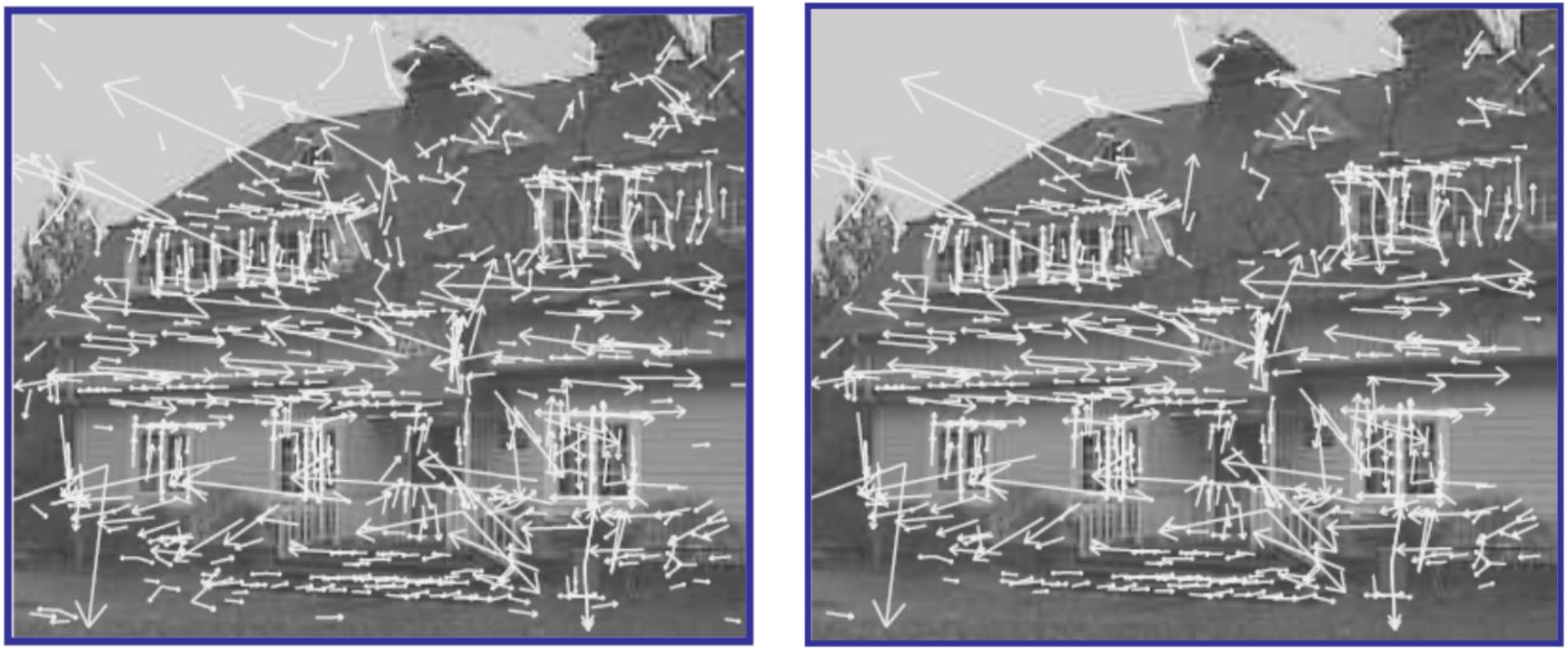
Credit: Svetlana Lazebnik, UIUC

SIFT:

From 233x189 original image to 832 DoG Extrema

Credit: Mubarak Shah, University of Central Florida

SIFT:



From 832 DoG Extrema to 729 keypoints after low contrast threshold

Credit: Mubarak Shah, University of Central Florida

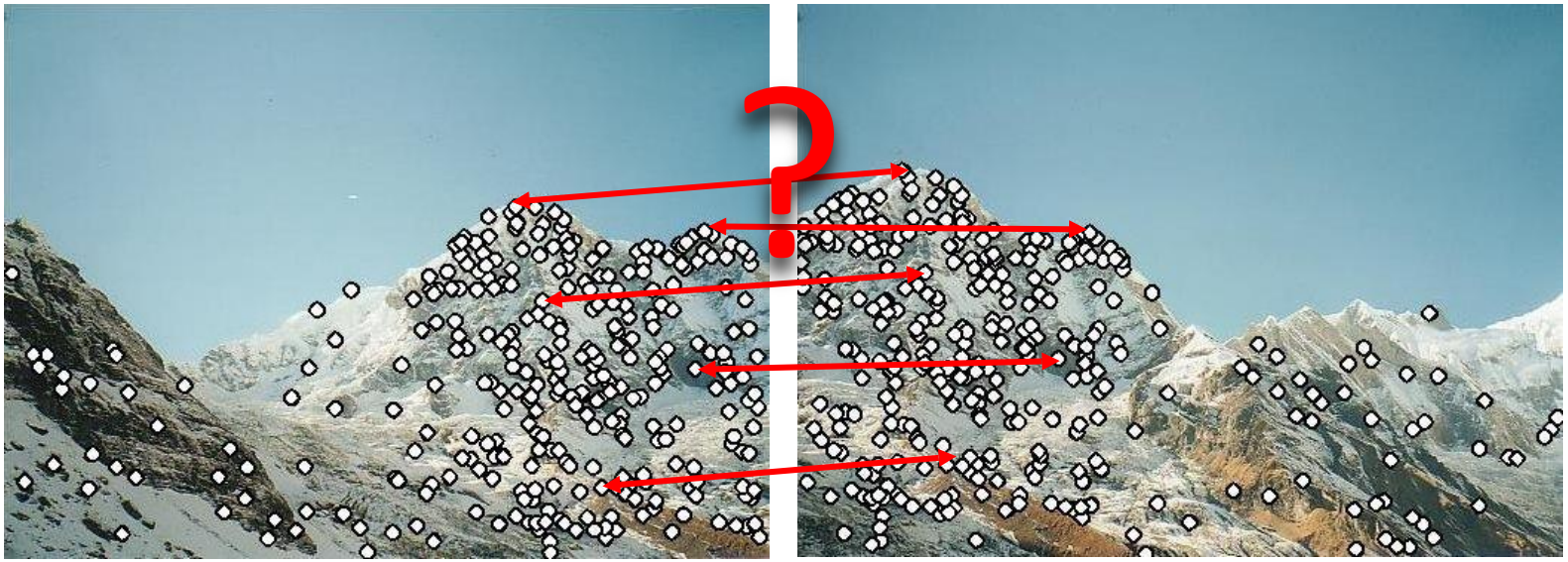
SIFT:

From 729 keypoints to 536 keypoints after testing ratio based on Hessian

Credit: Mubarak Shah, University of Central Florida

Feature descriptors

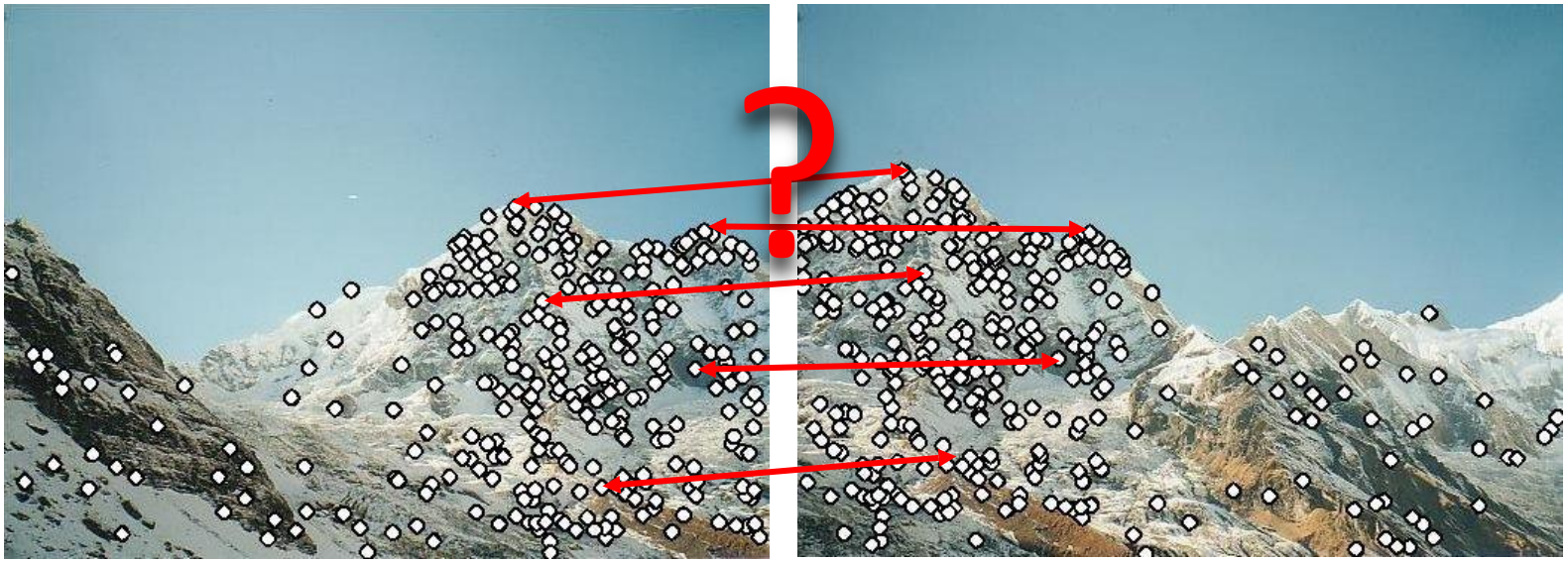
We know how to detect good points
Next question: **How to match them?**



Answer: Come up with a *descriptor* for each point, find similar descriptors between the two images

Feature descriptors

We know how to detect good points
Next question: **How to match them?**



Lots of possibilities (this is a popular research area)

- Simple option: match square windows around the point using dicriptors

SIFT Algorithm Stages

- Step 1: Scale-space Extrema Detection - Detect interest points (invariant to scale and orientation) using DOG.
- Step 2: Keypoint Localization - Determine location and scale at each candidate location, and select them based on stability.
- Step 3: Orientation Estimation - Use local image gradients to assign orientation to each localized keypoint. Preserve orientation, scale and location for each feature.
- Step 4: Keypoint Descriptor - Extract local image gradients at selected scale around keypoint and form a representation invariant to local shape and illumination distortion.

SIFT: Keypoint Descriptor

- Around the detected keypoint, compute gradient at each pixel in a 16x16 window, using the appropriate level of the Gaussian pyramid at which the keypoint was detected.

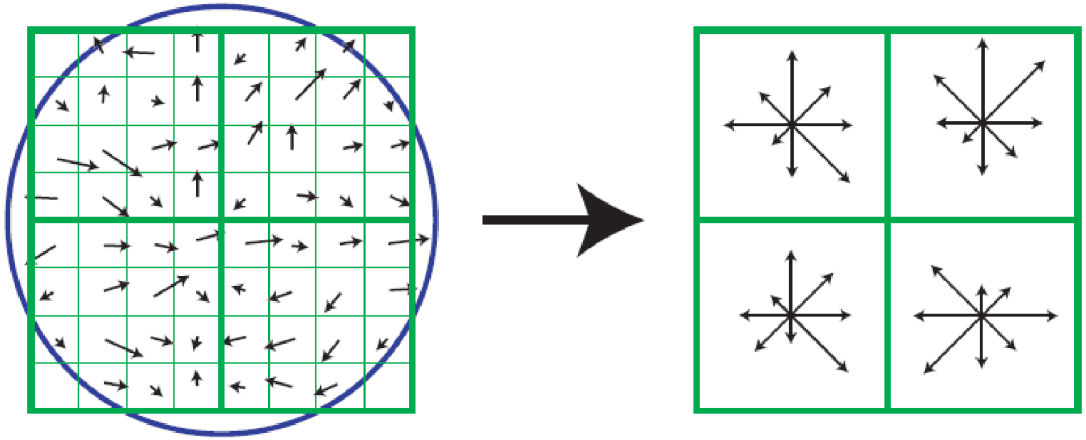


Image gradients

Keypoint descriptor

- Downweight gradients by a Gaussian fall-off function (blue circle) to reduce the in of gradients far from the center.
- In each 4x4 quadrant, compute a gradient orientation histogram using 8 orientation histogram bins.

Credit: Raquel Urtasun, Szeliski

SIFT: Keypoint Descriptor

- The resulting 128 non-negative values form a raw version of the SIFT descriptor vector.
- To reduce the effects of contrast or gain (additive variations are already removed by the gradient), the 128-D vector is normalized to unit length.
- To further make the descriptor robust to other photometric variations, values are clipped to 0.2 and the resulting vector is once again renormalized to unit length.

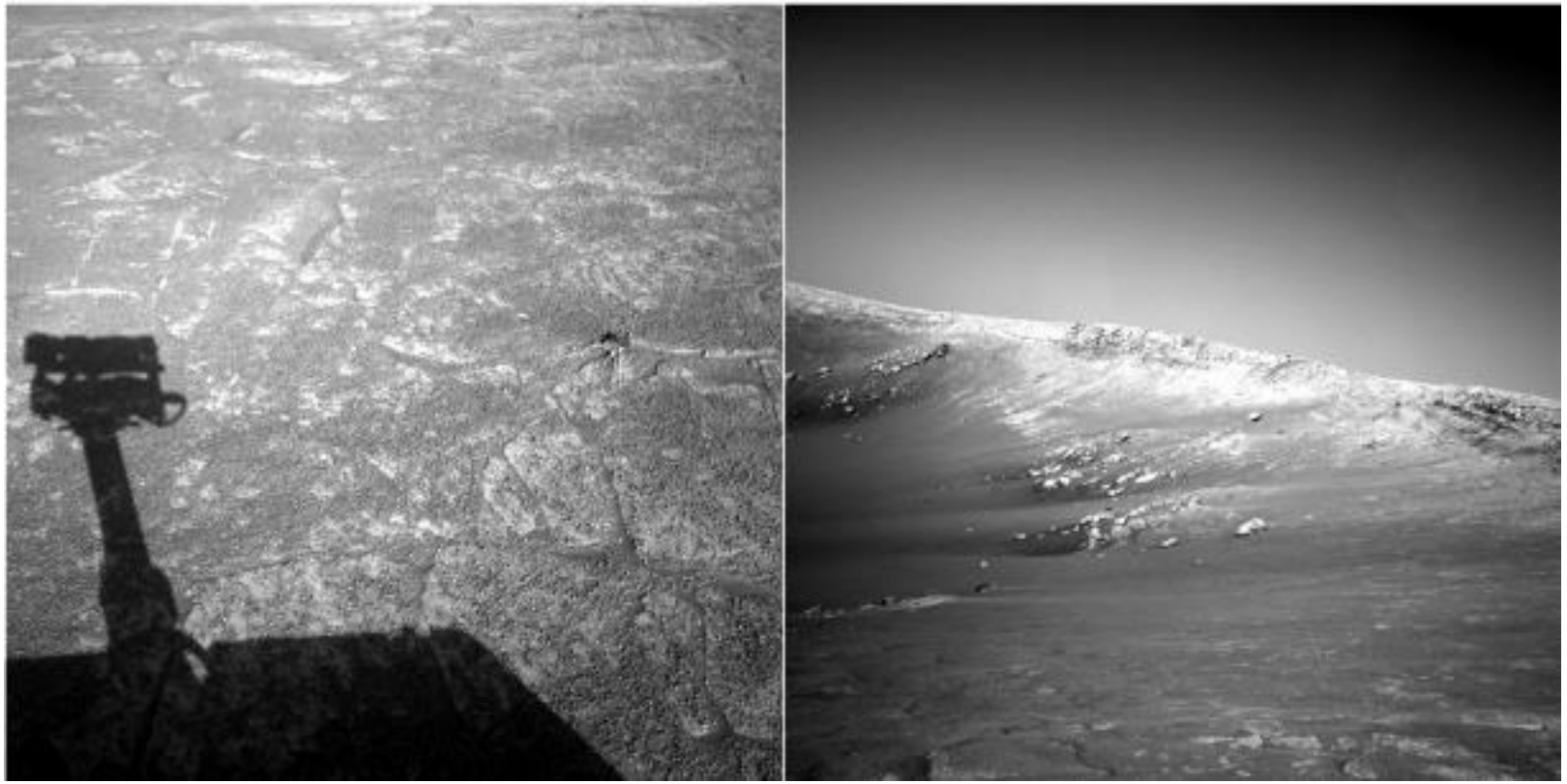
Credit: Raquel Urtasun, Szeliski

SIFT:

- Extraordinarily robust feature detection
- Changes in viewpoint: up to about 60 degree out of plane rotation
- Changes in illumination: sometimes even day vs night (below)
- Fast and efficient → can run in real-time

Credit: Raquel Urtasun, Szeliski

SIFT: Example

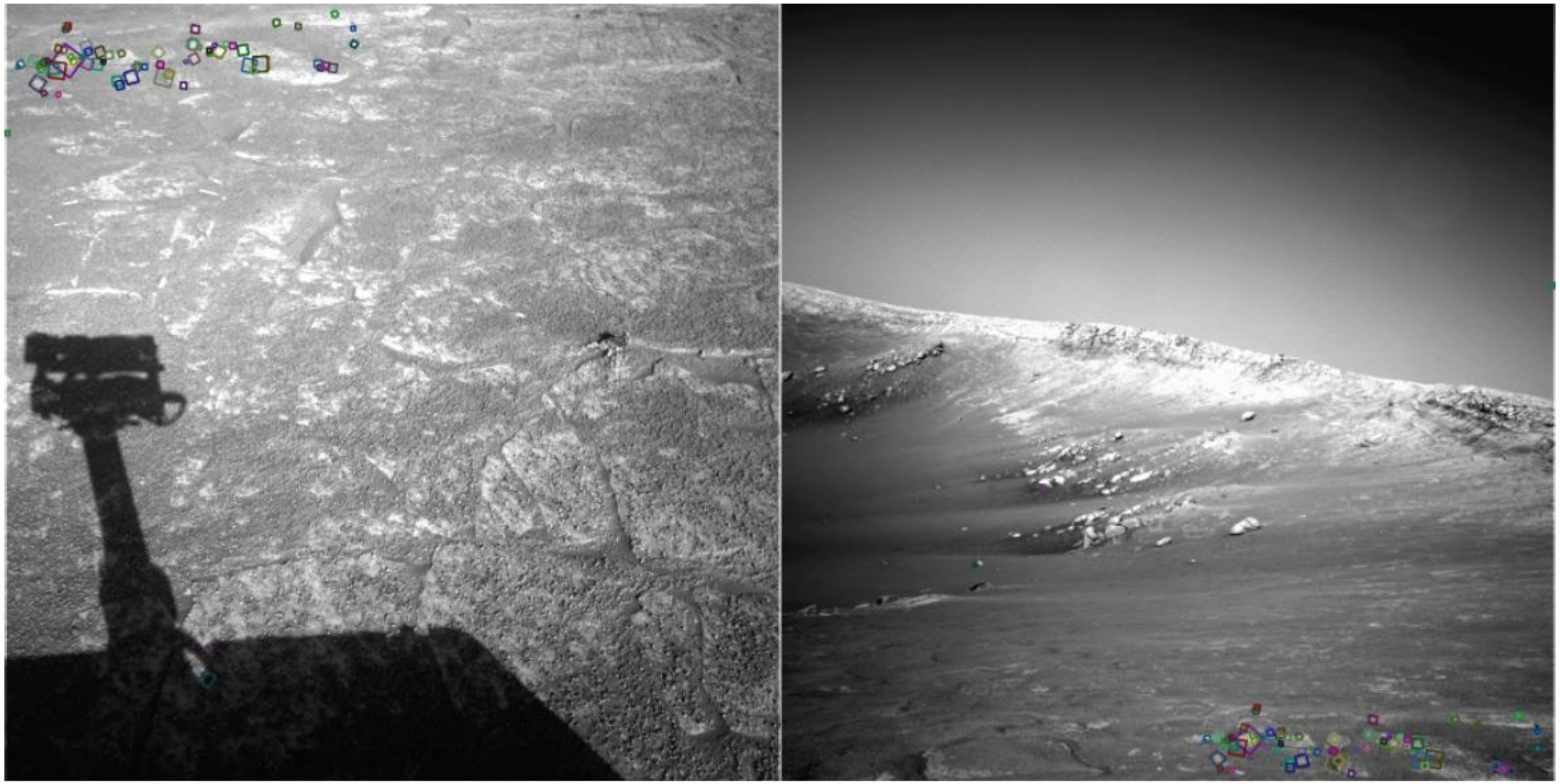


Mars Rover images

Credit: Raquel Urtasun, Szeliski

SIFT: Example

Maybe, look for tiny squares. . . ?



Mars Rover images with SIFT features

Credit: Raquel Urtasun, Szeliski

SIFT: *Invariances(Geometric Transformations)*

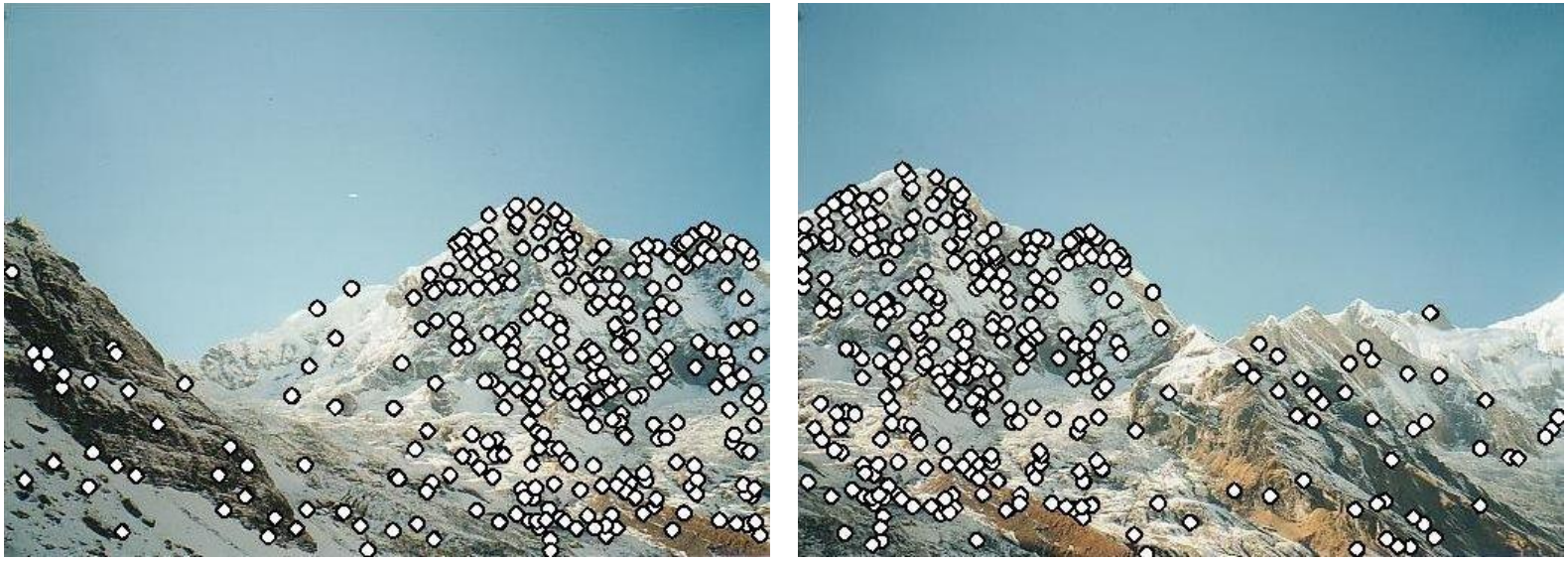
Credit: Raquel Urtasun, Szeliski

SIFT: Invariances(Photometric Transformations)

Credit: Raquel Urtasun, Szeliski

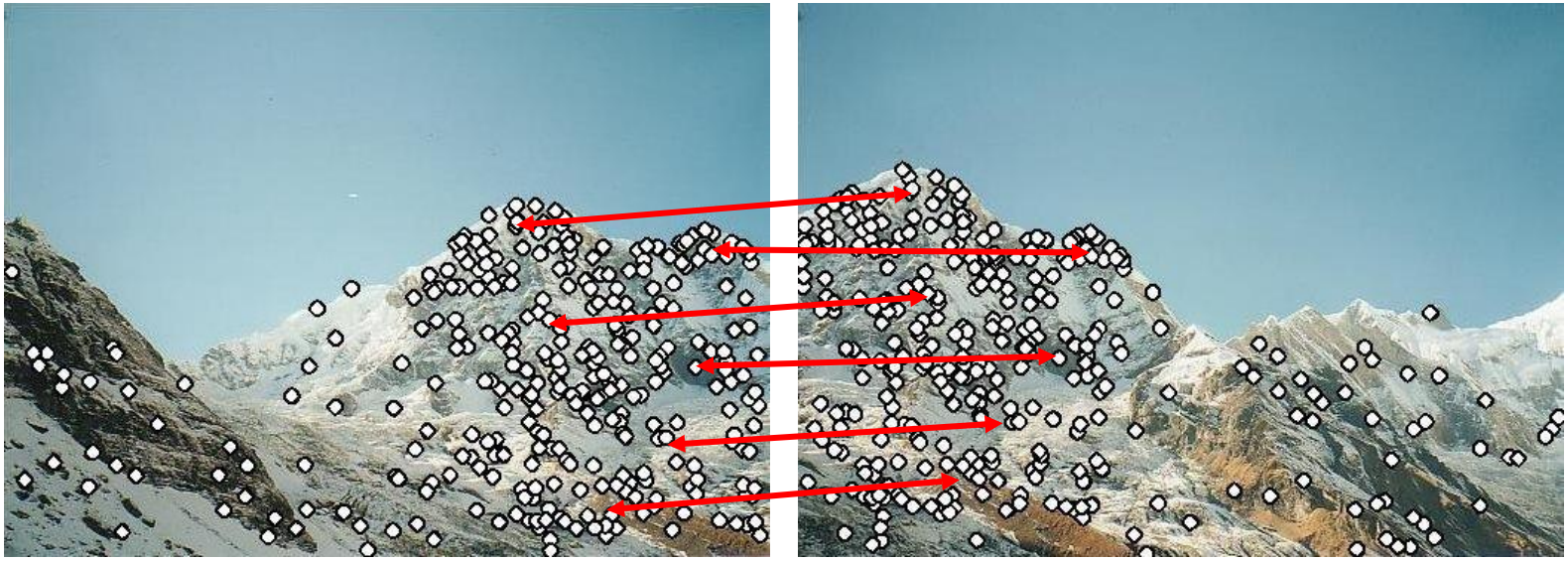
SIFT application: Image Stitching

SIFT application: Image Stitching



Step 1: extract features, describe features

SIFT application: Image Stitching



Step 1: extract features, describe features

Step 2: match features

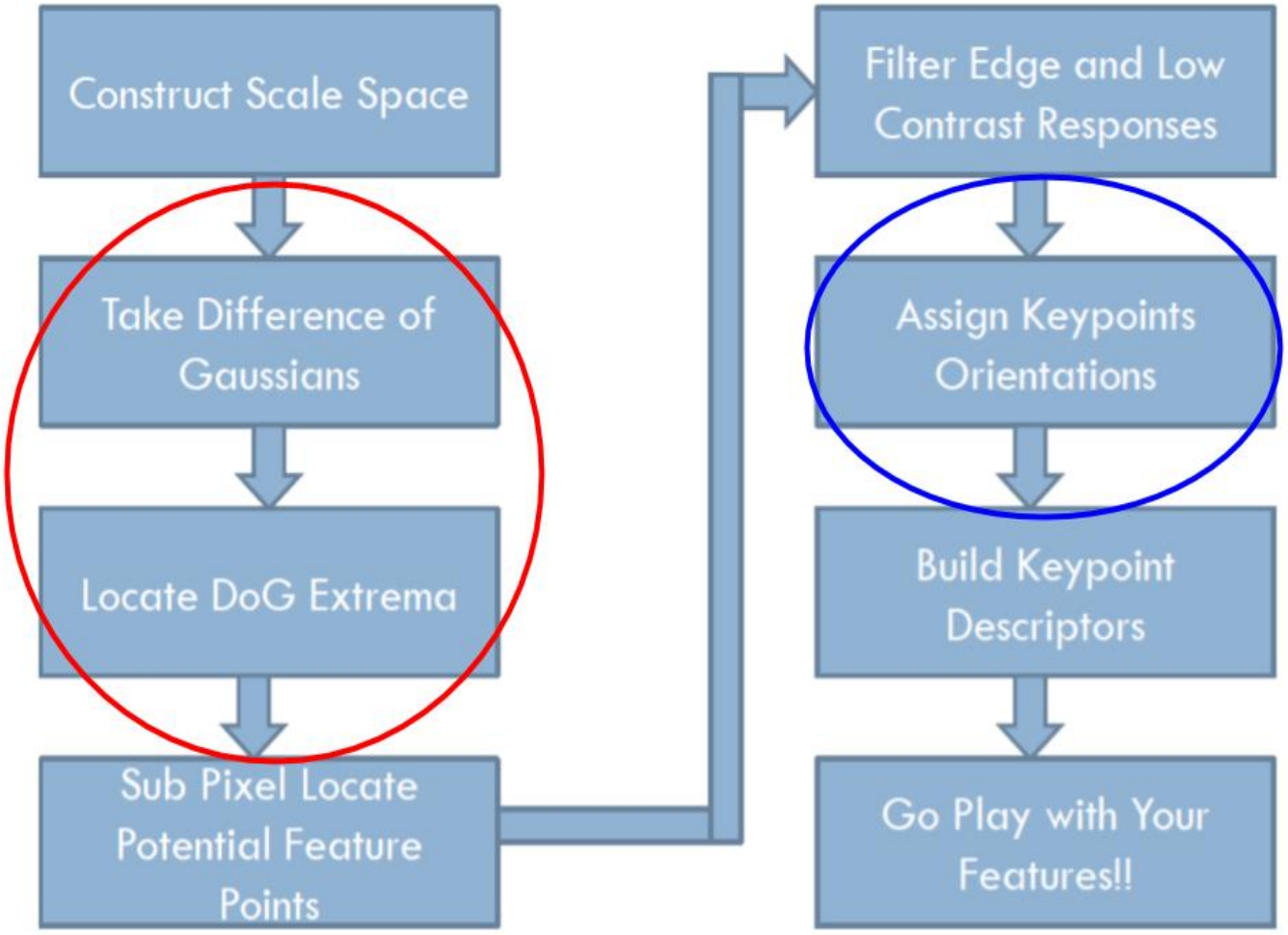
SIFT application: Image Stitching

Step 1: extract features, describe features

Step 2: match features

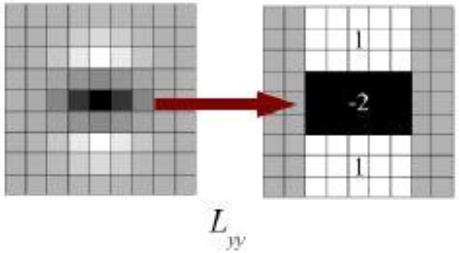
Step 3: align/fitting images

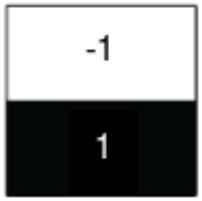
SIFT


Construct Scale Space
Take Difference of Gaussians
Locate DoG Extrema
Sub Pixel Locate Potential Feature Points
Filter Edge and Low Contrast Responses
Assign Keypoints Orientations
Build Keypoint Descriptors
Go Play with Your Features!!

SURF: Speeded Up Robust Features (ECCV 2006, 26k+ citations)

- Uses box filters instead of Gaussians to approximate Laplacians
- Uses Haar wavelets to get keypoint orientations.
- Haar wavelets are simple filters which can be used to find gradients in the x and y directions
- SURF is good at handling blur and rotation variations
- SURF is not as good as SIFT on invariance to illumination & viewpoint changes
- SURF is nearly 3 times faster than SIFT

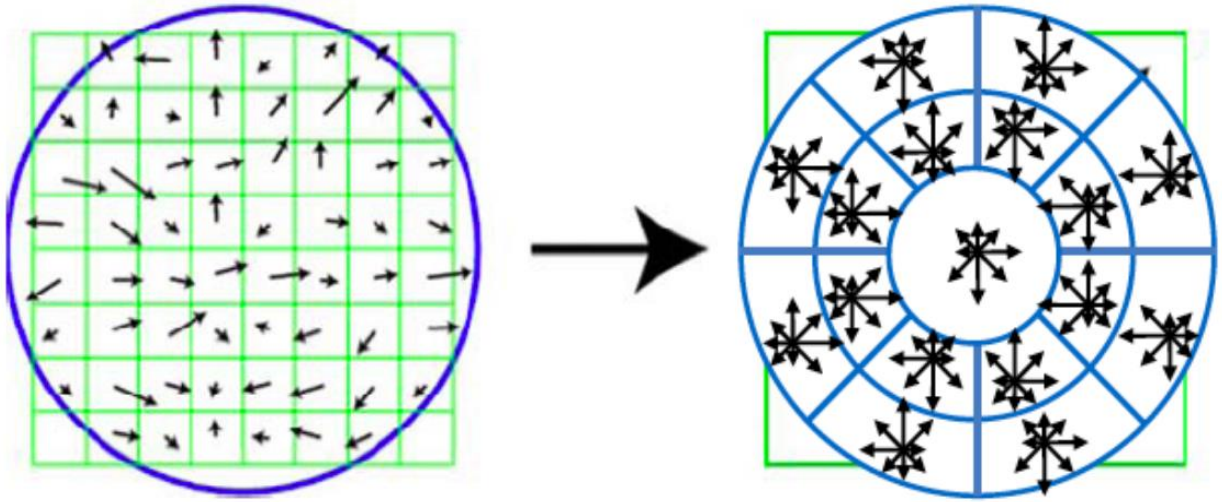




For more information:
https://medium.com/data-breach/introduction-to-surf-speeded-up-robust-features-c7396d6e7c4e
http://www.vision.ee.ethz.ch/~surf/

Gradient Location-Orientation Histogram (GLOH)

- Variant of SIFT that uses a log-polar binning structure instead of four quadrants.
- Uses 17 spatial bins and 16 orientation bins.
- The 272D histogram is then projected onto a 128D descriptor using PCA trained on a large dataset.



Credit: Matthew Brown, Kristen Grauman, Raquel Urtasun